

openHPI: 10 Years of MOOCs at the Hasso Plattner Institute

Christoph Meinel, Christian Willems, Thomas Staubitz,
Dominic Sauer, Christiane Hagedorn

Technische Berichte Nr. 148

des Hasso-Plattner-Instituts für
Digital Engineering an der Universität Potsdam



Technische Berichte des Hasso-Plattner-Instituts für
Digital Engineering an der Universität Potsdam

Technische Berichte des Hasso-Plattner-Instituts für
Digital Engineering an der Universität Potsdam | 148

Christoph Meinel | Christian Willems | Thomas Staubitz | Dominic Sauer |
Christiane Hagedorn

openHPI

10 Years of MOOCs at the Hasso Plattner Institute

Universitätsverlag Potsdam

Bibliographic information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche
Nationalbibliografie; detailed bibliographic data are available on the Internet
via <http://dnb.dnb.de/>.

Universitätsverlag Potsdam 2022
<http://verlag.ub.uni-potsdam.de/>

Am Neuen Palais 10, 14469 Potsdam
Phone: +49 (0)331 977 2533 / Fax: 2292
Email: verlag@uni-potsdam.de

The series **Technische Berichte des Hasso-Plattner-Instituts für Digital
Engineering an der Universität Potsdam** is edited by the professors of the
Hasso Plattner Institute for Digital Engineering at the University of Potsdam.

ISSN (print) 1613-5652
ISSN (online) 2191-1665

The work is protected by copyright.
Layout: Tobias Pape
Print: docupoint GmbH Magdeburg

ISBN 978-3-86956-544-6

Also published online on the publication server of the University of Potsdam:
<https://doi.org/10.25932/publishup-56020>
<https://nbn-resolving.org/urn:nbn:de:kobv:517-opus4-560208>

On the occasion of the 10th openHPI anniversary, this technical report provides information about the HPI MOOC platform, including its core features, technology, and architecture.

In an introduction, the platform family with all partner platforms is presented; these now amount to nine platforms, including openHPI. This section introduces openHPI as an advisor and research partner in various projects.

In the second chapter, the functionalities and common course formats of the platform are presented. The functionalities are divided into learner and admin features. The learner features section provides detailed information about performance records, courses, and the learning materials of which a course is composed: videos, texts, and quizzes. In addition, the learning materials can be enriched by adding external exercise tools that communicate with the HPI MOOC platform via the Learning Tools Interoperability (LTI) standard. Furthermore, the concept of peer assessments completed the possible learning materials. The section then proceeds with further information on the discussion forum, a fundamental concept of MOOCs compared to traditional e-learning offers. The section is concluded with a description of the quiz recap, learning objectives, mobile applications, gameful learning, and the help desk.

The next part of this chapter deals with the admin features. The described functionality is restricted to describing the news and announcements, dashboards and statistics, reporting capabilities, research options with A/B testing, the course feed, and the TransPipe tool to support the process of creating automated or manual subtitles. The platform supports a large variety of additional features, but a detailed description of these features goes beyond the scope of this report. The chapter then elaborates on common course formats and openHPI teaching activities at the HPI. The chapter concludes with some best practices for course design and delivery.

The third chapter provides insights into the technology and architecture behind openHPI. A special characteristic of the openHPI project is the conscious decision to operate the complete application from bare metal to platform development. Hence, the chapter starts with a section about the openHPI Cloud, including detailed information about the data center and devices, the used cloud software OpenStack and Ceph, as well as the openHPI Cloud Service provided for the HPI.

Afterward, a section on the application technology stack and development tooling describes the application infrastructure components, the used automation, the deployment pipeline, and the tools used for monitoring and alerting. The chapter is concluded with detailed information about the technology stack and concrete platform implementation details. The section describes the service-oriented Ruby on Rails application, inter-service communication, and public APIs. It also provides more information on the design system and components used in the application. The section concludes with a discussion of the original microservice architecture, where we share our insights and reasoning for migrating back to a monolithic application.

The last chapter provides a summary and an outlook on the future of digital education.

It is my great pleasure to present you with the Technical Report on the *openHPI* learning platform in the year of openHPI's 10th anniversary. I would like to use this preface to briefly outline the background of openHPI and say thank you.

The New York Times proclaimed 2012 "The Year of the MOOC". In the same year, openHPI was launched. The acronym MOOC stands for Massive Open Online Courses, i.e., online courses that are freely accessible to all interested parties. In 2011, the innovative learning format began to generate great enthusiasm and enormous numbers of participants, both at Stanford University – where the Hasso Plattner Institute of Design was founded in 2005 – and other elite US universities.

The starting signal for the new era of knowledge transfer at the *Hasso Plattner Institute* (HPI) was given by Prof. Hasso Plattner, the founder and chairman of the supervisory board of the global software corporation SAP. He presented the very first online course on openHPI starting on September 3rd, 2012, in which he introduced more than 13,000 learners over two months to the subject of a revolutionary new main memory database technology. The so-called in-memory data management for the lightning-fast processing of huge amounts of data had been researched at HPI under Prof. Plattner's leadership starting in 2007. It was successfully implemented at SAP as the *HANA* product subsequently.

Shortly before Prof. Plattner's course started, I had taken over the technical development and management of the openHPI learning platform as director of the HPI. As a computer science professor, the openHPI project was particularly appealing to me as I had already started a global e-learning project on October 29th, 2002: the *Internet Bridge*. My lecture on "Vulnerabilities and Points of Attack on the Internet" at the University of Trier was broadcasted live via the World Wide Web to a lecture hall at the Peking University of Technology. The "Internet Bridge" enabled students in China to participate in live online lectures at a foreign university. Later, instead of offering the lectures via live-streaming, we decided to record and upload them to a server at the Peking Technical University so they could be provided on demand. The lectures were recorded using a tool that emerged from a project called *tele-TASK*, whose split-screen technology is still used for recording course content for openHPI today.

The openHPI project would not be as successful as today without the many bright minds and the energetic participation of various teams. In this context, I would particularly like to thank Prof. Plattner for giving us the opportunity to establish and expand the platform. I would also like to thank all those who have worked on the platform and researched new concepts to build the platform, constantly improved it, and contributed to the reputation it has today. Naming all those involved would exceed the scope of this paragraph. Therefore, I would like to thank all current and former openHPI team members for their enduring work over the past ten years.

Best regards,

Prof. Dr. Christoph Meinel
Potsdam, Germany
September 3rd, 2022

Contents

1	Introduction	11
1.1	The openHPI MOOC Platform	11
1.2	Platform Family	16
1.3	openHPI as an Advisor and Research Partner	18
2	Functionality and Format	21
2.1	Learner Features	21
2.1.1	Performance Records	21
2.1.2	Courses	22
2.1.3	Learning Materials	24
2.1.4	Discussion Forum	43
2.1.5	Quiz Recap	45
2.1.6	Learning Objectives	45
2.1.7	Mobile Applications	48
2.1.8	Gameful Learning	49
2.1.9	Help Desk and Chatbot	51
2.2	Admin Features	54
2.2.1	News and Announcements	55
2.2.2	Dashboards and Statistics	56
2.2.3	Reporting	60
2.2.4	A/B Testing	61
2.2.5	Courses API	61
2.2.6	TransPipe	61
2.3	Common Course Formats	63
2.4	openHPI in On-Site Teaching	67
2.5	Best Practices in Course Design and Course Delivery	68
3	Technology and Architecture	72
3.1	High Availability, Scalable Infrastructure: The openHPI Cloud	73
3.1.1	Data Center and Devices	75
3.1.2	OpenStack	79
3.1.3	Ceph and radosgw	84
3.1.4	openHPI Cloud Service for the HPI	86
3.2	Application Technology Stack and Development Tooling	87
3.2.1	Application Infrastructure Components	87
3.2.2	Automation	90
3.2.3	Deployment Pipeline	95
3.2.4	Monitoring and Alerting	99

3.3	Application and Architecture	100
3.3.1	Technology Stack	100
3.3.2	Service-oriented Architecture	104
3.3.3	Design System and Components	109
3.3.4	Back to Monolith – Insights From 8 Years of Microservice Architecture	111
4	Summary and Outlook	115
4.1	Summary	115
4.2	Outlook – Digital Education in the 21st Century	116

Listings

3.1	Terraform example: deploy openHPI Testing (extract)	91
3.2	Basic salt state	93
3.3	SaltStack example: state for an openHPI web application VM (extract)	94
3.4	Integration test scenario: enroll to a course	97

1 Introduction

The online learning platform openHPI was made available to the public on September 3rd, 2012, and thus celebrates its 10th anniversary this year. Initially, the platform was only planned to be used for our own – openHPI for the *Hasso Plattner Institute* (HPI) – purposes. However, the product has not only thrived over the last ten years, but also resulted in numerous offshoots that have blossomed into a large and internationally acknowledged *learning management system* (LMS).

The Massive Open Online Courses (MOOC) format for digital education and digital awareness has proven very successful recently. Besides being instrumental in the dissemination of learning content, primarily through videos, MOOCs are characterized by a high level of activity among participants.

The following course features support this quality:

- Ease of access: Participation is free of charge. There are no enrollment restrictions.
- Large number of participants: Usually, several thousand participants enroll in a course.
- Event character: Courses are only offered in certain timeframes (have start and end dates) and contain test deadlines.
- Guided course progress: The teaching team guides learners by the targeted publication of specific course contents at set times throughout the course (as a rule, at the beginning of a new week).

Since 2012, the research group *Learning and Knowledge Engineering* at the HPI has been developing and operating its own MOOC platform – openHPI. As mentioned above, further partners have joined and now make use of our software. To differentiate between our instance of the system and the software, including its general use in all the partner platforms, we will use the terms *openHPI* and *the HPI MOOC platform*. The HPI MOOC platform has been optimized for MOOCs. Guaranteeing high scalability has always been one of the main principles when developing program functions. This focus distinguishes the platform from traditional e-learning offers, which often work with a few participants. In addition, the HPI MOOC platform is very modular and highly configurable.

1.1 The openHPI MOOC Platform

When openHPI started its operations, it was Europe's first MOOC platform. On September 3rd, 2012, the platform was launched with Prof. Hasso Plattner's course

“In-Memory Data Management.” The course and the platform were an instant success. When the course ended on November 1st, already 13,629 users were enrolled. Five thousand more learners enrolled in the following years until the second and third iterations of the course started in 2013 and 2014. About 18% of the active course participants completed the course with a certificate. This number may appear small, but it is quite the opposite, considering the context.

In the early years of the MOOC movement, the courses’ completion rates have been a big discussion topic, and a substantial amount of the MOOC-related research focused on various options for improving them. In 2015, Katy Jordan examined the completion rates of many of the MOOCs that were available back then worldwide [23]. Particularly, the larger courses did not perform too well on this metric; many of them ranged below ten percent. It has to be mentioned, however, that the whole discussion about MOOCs’ dropout rates is often based on wrong premises. MOOC learners are not comparable to regular students. Most of them are not students at all, but life-long learners. Most of them participate in these courses during their spare time, just out of interest in a course’s topic. The rest of their lives is not as dependent on their course completion as it is for most regular students. A study by the *Education Data Initiative* shows, e.g., that college dropouts in the US, on average, have about 33% less income than graduates with a bachelor’s degree. They also face an almost 20% higher risk of being unemployed [19]. Still, about 24% of the full-time and more than 50% of the part-time students drop out [19].

MOOC learners, on the contrary, have to fear hardly any consequences when they drop out, while reasons and opportunities to drop out of a MOOC are plenty. In most cases, they are entirely unrelated to the course or the platform. Moreover, many learners are not even interested in completing the course but only want to get the most relevant information. Still, they are counted as dropouts. Finally, some users have enrolled for a course but never started attending it, and those are also considered dropouts. At some point, we slightly adjusted the calculation of the course completion rate to better reflect the actual situation than just the mere numbers. Instead of using the formula *Issued Records of Achievements per Enrolled Users*, we use *Issued Records of Achievements per Shows at Course Middle*. Our definition of a *Show* is a user who has enrolled in the course and visited at least one course learning item. We work with the number at course middle, as participants who join the course later on will already have missed several graded assignments and hardly have the chance to complete the course with a good result. Long story short: first – we do not consider the low completion rates in MOOCs to be a huge issue; second – the completion rates of our MOOCs have always been comparatively good.

During the last ten years, we have experimented with different course settings, exam types, and course formats. While analyzing the results, we have made many observations that helped us improve our courses and the platform itself. Several of them will be discussed in the following chapters.

After the first course’s success, we offered more courses in a wide variety of IT and Innovation related topics in the following years. The pioneers on the platform were several HPI professors and senior researchers, particularly,

- Prof. Plattner (In-Memory Data Management),
- Prof. Meinel (Internetworking with TCP/IP),
- Prof. Sack (Semantic Web Technologies),
- Prof. Naumann (Data Management with SQL),
- Prof. Weske (Business Process Modeling and Analysis), and
- Prof. Tröger (Parallel Programming Concepts).¹

In the beginning, all our MOOCs had a fixed length of six weeks. Later, we introduced additional course formats: four weeks hands-on courses and two weeks workshop courses. Since 2019, we have offered another two weeks format – selected six-week courses have been split into three two weeks modules, which can be combined with a separate exam (see Figure 1.1).

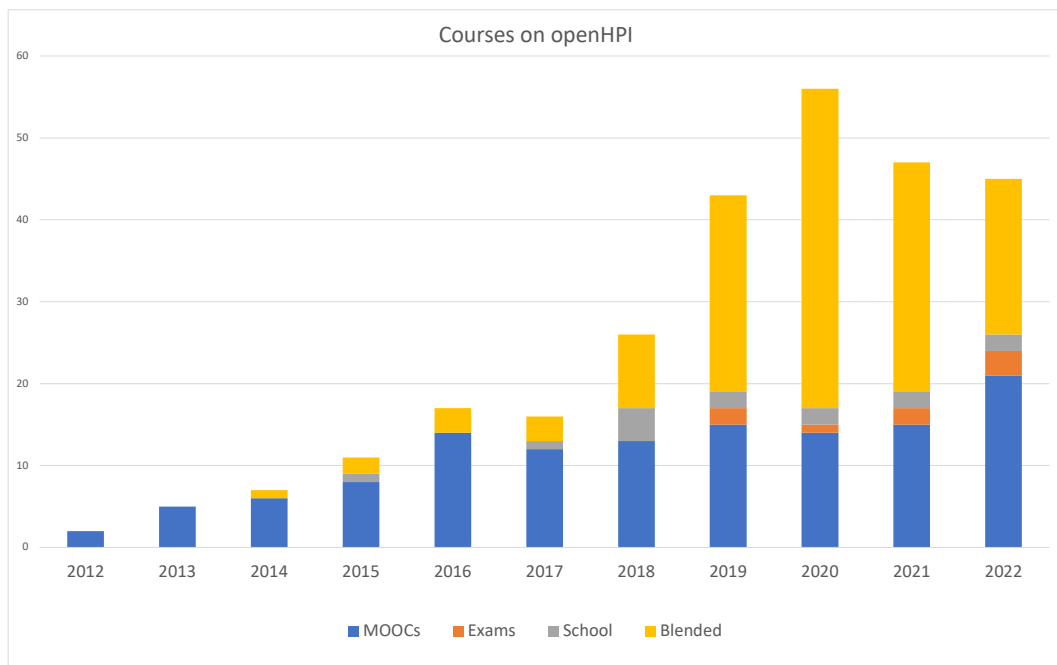


Figure 1.1: Courses on openHPI. *MOOCs* are the courses we offer for the general public. *Exams* courses offer an exam for a combination of several other courses. *School* courses have been slightly redesigned to be used by teachers in school. *Blended* courses are used in certain courses within the regular HPI on-site curriculum or by the HPI Academy.

¹The more or less complete list of our courses can be found here: <https://open.hpi.de/courses>; last access: 02.09.2022. Older course iterations might be hidden.

1 Introduction

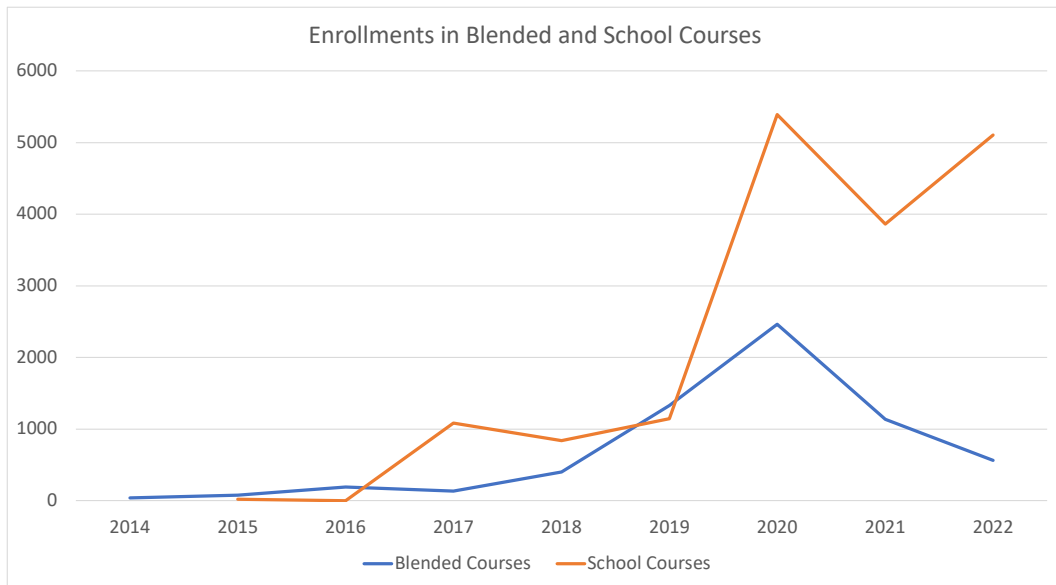


Figure 1.2: Enrollments in blended courses and school courses at course end

While originally, the platform was only used to offer MOOCs, several other use cases emerged later. Since 2014, some MOOCs have been reused for flipped classroom approaches and other blended formats within the HPI. For example, the HPI D-School uses the platform to accompany their basic and advanced tracks.² Particularly in 2020/21, due to the COVID-19 pandemic and lockdowns, we see a strong increase in the number of blended formats. The on-site face-to-face part during these years was often replaced by video calls.

Since 2015/16, we have remodeled several of our courses to be used in a school context. We started with a first pilot in one school. In 2017, we ran a broader pilot, and since 2018, we have been offering at least two programming courses (Python and Java) in a customized school version once a year. The contents of these courses are identical to the original MOOCs from which they have been derived, except for the length of each section and the resulting weekly workload. Figure 1.2 shows the enrollment numbers at the course end for the blended and school courses.³

Since 2014, we have offered a mix of course iterations (re-runs of previous courses), course translations (German and English), and newly developed courses. Next to the topics that have already been mentioned, further course topics were *Web Technologies*, *Cybersecurity*, *Python*, *Java*, *R*, and *Ruby Programming*, *Social Media and Privacy*, *How to found a Start-Up*, *Etoys*, *Embedded Smart Home*, *Homepage Design*, *Search Engines*, *Mathematics of Algorithms*, *Design Thinking*, *Big Data Analytics*, *Linux*, *IPv6*, *IT Law*, *Remote Teamwork*, *AI*, and most recently *Quantum Computing*.

²<https://hpi.de/en/school-of-design-thinking/for-students/our-programs.html>; last access: 02.09.2022.

³As the school courses stayed open after the end of the course, more learners enrolled later on.

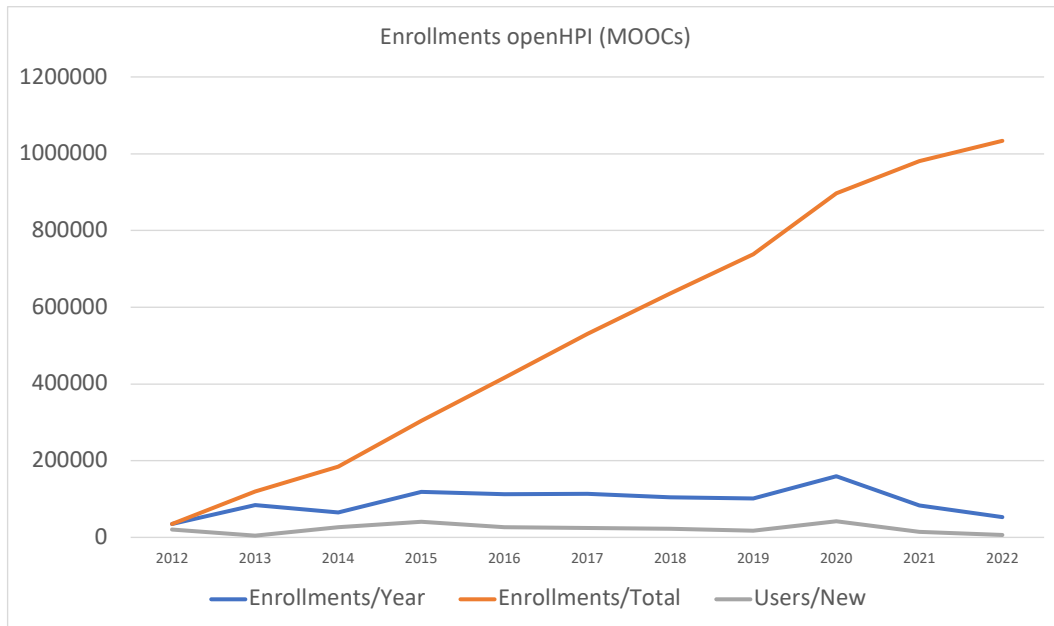


Figure 1.3: Enrollments on openHPI (MOOCs only)

Our goal is to offer half of each year's courses in German and the other half in English; often, however, the percentage is rather 60% German to 40% English courses. The contents of these courses are created by the professors and researchers of the HPI. In 2015, we also started a program to teach our students how to build MOOCs (see Section 2.4). Since then, one to three of the MOOCs per year have been produced by HPI students.

Currently, openHPI has about 1.1 million course enrollments and close to 300,000 registered users (see Figure 1.3). On average, each user has enrolled in 3.7 courses, which undoubtedly can be interpreted as a sign of quality. In total, we have issued close to 116,000 Records of Achievement and more than 220,000 Confirmations of Participation (see Section 2.1.1). The average completion rate in our courses is about 40% (min: 4%, max: 98%; see Figure 1.4). Generally, the completion rate is a neglectable indicator of the course quality. The reasons why some courses' completion rates are higher than others are manifold. Maybe the course advertisement did not attract the intended target group, or the course's final exam was scheduled during a vacation period. Many of those issues can be avoided via proper planning. Occasionally, however, the choice is between Scylla and Charybdis, as several dependencies must be considered in parallel. Naturally, the design of a course also influences its completion rate (see Section 2.5 for a list of best practices).

One important aspect in this context is the complexity of the task that has to be solved by the participants to earn a Record of Achievement. Courses that rely entirely on multiple choice exams often have much higher completion rates than courses using project-based learning approaches and other time and energy-consuming tasks. For example, in three of the four courses with a completion rate

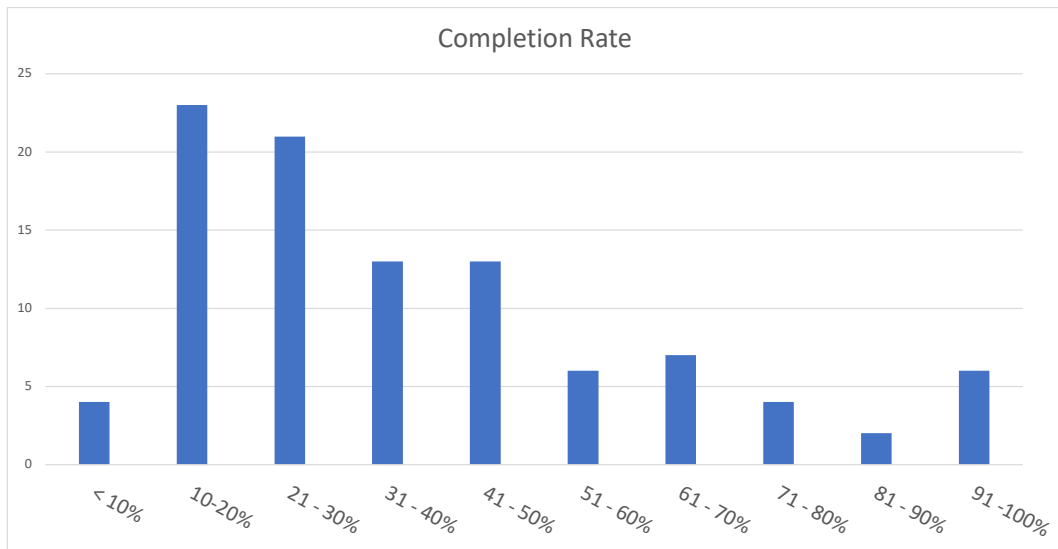


Figure 1.4: Completion rate histogram

of less than 10%, a peer-assessed project had to be completed to earn the Record of Achievement. To be completely clear on this point: not the content of the peer assessment was the reason; the method itself caused the issue.

In earlier reports, we stated a correlation between the forum activity and the completion rate of a course. With additional data from more courses we have now, this statement might need to be revised. Therefore, we are currently examining the quality of the forum discussions more closely to find out how this metric affects the completion rates.

To conclude this section: completion rates are not the only metric to consider when designing a course. Many other aspects define the quality of a course, and several factors that affect the completion rate are entirely out of the hands of the course designers and instructors.

1.2 Platform Family

In 2012 and 2013, we still relied on a third-party open source tool to deliver our courses. We soon discovered, however, that this tool did not entirely fit our needs. Hence, we started to develop a new platform from scratch, which other partners soon adopted. To date, there are nine nearly identical instances (see Figure 1.5) of the HPI MOOC platform operated on the same technical infrastructure (the openHPI Cloud, see Section 3.1):

openSAP openSAP was the first enterprise MOOC platform worldwide. Since May 2013, SAP has been providing online courses for professionals via its own platform. These courses are primarily on SAP software topics. Participation is also

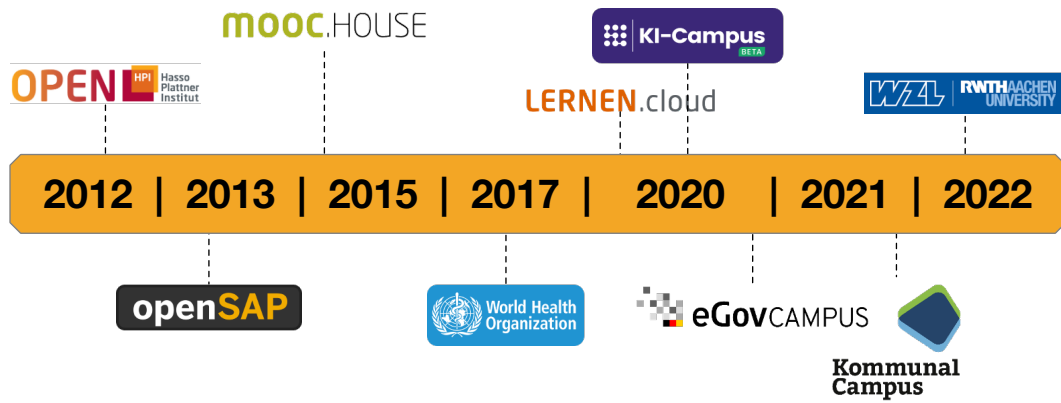


Figure 1.5: HPI MOOC platform instances

free of charge and open to everyone. Currently, openSAP lists more than 6.3 million enrollments.

mooc.house The white label platform for businesses, organizations, and individuals is called mooc.house. Everyone who wants to disseminate interesting learning offers can become a course provider and rely on the proven platform infrastructure. Among others, the Charité Berlin offers courses at mooc.house. mooc.house currently lists about 100,000 enrollments.

OpenWHO OpenWHO is the platform of the World Health Organization (WHO). The WHO uses the platform to inform and train its staff, the on-site medical teams, and the affected population in the event of an epidemic outbreak. Originally, it was developed with the upcoming Ebola outbreak in Africa in mind. When the COVID-19 pandemic started, the platform proved its enormous scalability. Within a few months, enrollment numbers skyrocketed from around 30,000 in February 2020 to more than 5 million in July 2020. In August 2022, OpenWHO surpassed 7 million enrollments.

LERNEN.cloud The LERNEN.cloud platform was set up to accompany the development of the *HPI-Schulcloud* (now *dBildungscloud*⁴). The HPI-Schulcloud started as a research project funded by the German Ministry of Education. Due to the COVID-19 pandemic, it went into productive service in three federal states before the project ended. LERNEN.cloud offers courses for teachers on how to use the *dBildungscloud* and many other topics.

KI-Campus/AI Campus The aim of this *BMBF* project is to strengthen AI competencies in Germany through digital learning opportunities at an academic level. For this purpose, the HPI transformed its existing MOOC platform, which has

⁴<https://dbildungscloud.de>; last access: 02.09.2022.

been tried and tested for years, into an open source project and expanded it by integrating AI-driven features to educate and train a broad spectrum of interested parties in Germany in the field of AI. The use of the online learning offerings on AI is open to both students and lifelong learners.

eGov-Campus The platform is funded by the IT Planning Council, which offers cross-university online modules on eGovernment and administrative informatics for education and training in the public sector. The platform also aims to impart key competencies and qualifications in the field of digitalization that will improve career opportunities in a targeted and sustainable manner. Since August 2020, the Hasso Plattner Institute has provided the technical basis for the eGov-Campus.

KommunalCampus This platform is operated by the “Metropolregion Rhein-Neckar” to offer courses to up-skill the administration personnel on the local and regional government levels. Their course format differs substantially from the other platforms as they focus on blended learning with a strong face-to-face component.

Industrial-Upskilling The Laboratory for Machine Tools and Production Engineering (WZL) at the RWTH Aachen is using the platform to offer courses in the context of a project. The target group of the MOOCs is small and medium-sized enterprises (SMEs) with their employees (in addition, if applicable, managers and works council members) in the manufacturing or production industry, as well as the skilled trades in Hesse. In particular, the existing technical knowledge in the specific task area should be deepened as well as basic knowledge in the fields of digitalization and artificial intelligence.

1.3 openHPI as an Advisor and Research Partner

Together with its partners, the Hasso Plattner Institute team investigates research questions and is constantly at work developing the platform. The dual role as a platform provider, on the one hand, and course provider, on the other hand, makes the HPI a competent advisor, regardless of whether the issue concerns the optimal creation of a course or hosting questions.

The HPI MOOC platform has proven to be a valuable research partner in national and international research projects. When *iversity* (until then the largest MOOC platform in Germany) filed for bankruptcy in 2016, the openHPI team was asked to replace *iversity* in two European research projects. The *Horizon2020* project *TraMOOC* explored options for auto-generated video transcripts and their automated translation to a variety of languages. The openHPI team organized a field test and user surveys. A few years later, the *TransPipe* (see Section 2.2.6) software was developed, building on the lessons learned during this project.

The Erasmus+ project BizMOOC⁵ explored the usage of MOOCs in the context of executive education, particularly in highlighting the topic “intrapreneurship”. In the context of this project, we produced one course that was published on the mooc.house platform,⁶ and an online book⁷ covering key questions for MOOC stakeholders from business and higher education as well as for the learner perspective. A question that has been examined was if employees are overrunning their HR departments by using MOOCs for lifelong learning [11]. The Erasmus+ project *Corship*⁸ was a follow-up on the BizMOOC project. Where BizMOOC dealt with Intrapreneurship, programs to foster innovations within large corporates, *Corship* covered the topic of “corporate entrepreneurship”, the collaboration between startups and enterprises to create new innovative products or services. In the context of this project, we developed another MOOC offered on mooc.house.⁹ Successful participants of this MOOC were invited to join a follow-up MasterClass.¹⁰ Next to the courses, the project had several other outcomes, such as a downloadable version of the MasterClass for use in university context courses,¹¹ a guidebook on how to use this course,¹² a digital toolbox¹³ including a tool to set up successful co-innovation projects,¹⁴ and publications on the results of the projects research outcomes, e.g., in the context of micro-credentials.

Further research projects in which we are involved on a national level are

- the KI-Campus and eGov-Campus projects (see page 17),
- the “Nationale Bildungsplattform”,¹⁵ a BMBF¹⁶ funded project, and
- the “Merlot”¹⁷ project that is funded by the BMWK.¹⁸

Additionally, there are further projects in which we are involved as a technology partner. We are also involved in several international informal co-operations such

⁵<https://bizmooc.eu>; last access: 02.09.2022.

⁶<https://mooc.house/courses/bizmooc2018>; last access: 02.09.2022.

⁷<https://mooc-book.eu>; last access: 02.09.2022.

⁸<https://www.corship.eu>; last access: 02.09.2022.

⁹<https://mooc.house/courses/corship2020>; last access: 02.09.2022.

¹⁰<https://mooc.house/courses/corship2021-mc>; last access: 02.09.2022.

¹¹<https://mooc.house/courses/corship-mc-template>; last access: 02.09.2022.

¹²https://www.corship.eu/wp-content/uploads/2021/12/Corship_Guidebook_FINAL.pdf;
last access: 02.09.2022.

¹³<https://www.corship.eu/digital-toolbox>; last access: 02.09.2022.

¹⁴<https://www.corship.eu/digital-toolbox/co-innovation-builder>; last access:
02.09.2022.

¹⁵<https://www.bmbf.de/bmbf/shareddocs/kurzmeldungen/de/neue-bekanntmachung-zum-aufbau-er-digitalen-bildungsplattform.html>; last access: 02.09.2022.

¹⁶Federal Ministry of Education and Research

¹⁷<https://merlot-education.eu>; last access: 02.09.2022.

¹⁸Federal Ministry of Economic Affairs and Climate Action

as the *MOOChub*,^{19,20} and a consortium²¹ that has done some ground-breaking research on differences between regional and global MOOC platforms on the learning analytics macro level [27, 36, 37]. In 2021, we had the honor to host two of the most prestigious conferences on MOOCs and scalable e-learning – the European MOOC Stakeholder Summit (EMOOCs) and the ACM LearningScale (LS) at the HPI. Unfortunately, we had to switch to a virtual event²² due to the COVID-19 situation back then. At least we were able to compensate for this a little by hosting the opening ceremony for L@S2022 at our New York office. Furthermore, we will be hosting EMOOCs 2023 – hopefully on-premise this time.

Finally, our platform offers Ph.D. students the opportunity to conduct research in their personal areas of focus. This is another way we continuously improve the feature set of our platform and enable development decisions based on cutting-edge research results. Our work and our publications are well received in the international e-learning community. While the remainder of this report focuses on the platform, features, and usage, we will add some information about this research, wherever it fits in the context.

¹⁹<https://moochub.org>; last access: 02.09.2022.

²⁰Partners: iMooX/TU Graz, on-campus/FH Lübeck, Open vhb, KI-Campus, eGov-Campus, open-SAP, LERNEN.cloud

²¹Partners: University of Murcia, Massachusetts Institute of Technology, FutureLearn, Edraak, Tsinghua University Beijing, Universitat Politècnica de València, Pontificia Universidad Católica de Chile, Universidad de Cuenca, Universidad Autónoma de Madrid, University of Colorado Boulder, HEC Paris, Université Paul Sabatier de Toulouse, Institute de Recherche en Informatique de Toulouse (IRIT).

²²<https://open.hpi.de/courses/hpi-learningatscale2021-emoocs2021>; last access: 02.09.2022.

2 Functionality and Format

In this section, we describe in detail the functionalities of the HPI MOOC platform. We will first go into the end-user features before discussing the admin functionalities. At the end of the chapter, we will look into common course formats.

2.1 Learner Features

This section provides detailed information about performance records, courses, and the learning materials of which a course is composed: videos, texts, and quizzes. In addition, the learning materials can be enriched by adding external exercise tools that communicate with the HPI MOOC platform via the Learning Tools Interoperability (LTI) standard. Furthermore, the concept of peer assessments completes the possible learning materials. Afterward, information on the discussion forum, the quiz recap, learning objectives, mobile applications, gameful learning, and the help desk follow.

2.1.1 Performance Records

Learners can enroll in a variety of courses on the platform. For participation and successful completion of a course, participants receive a performance record. Three different types of these performance records are offered on the HPI MOOC platform: *Confirmations of Participation*, *Records of Achievement*, and *Certificates* (see Figure 2.1). Which course records are offered can vary for each specific course, depending on the course content.

Confirmation of Participation (CoP) A Confirmation of Participation (CoP) is earned by participants who have been involved with at least 50% of the course material. The CoP contains the following information: Name of participant, date of birth (optional), course title and summary.

Record of Achievement (RoA) A Record of Achievement is provided to those learners who have earned at least 50% of the total course points in the course's graded and bonus assignments. In addition to the information on the CoP, the Record of Achievement also contains the points earned, the course performance (top 5%, 10%, or 20% of participants if applicable), an anti-counterfeit link and a QR code to simplify the anti-counterfeit check.

2 Functionality and Format



Figure 2.1: Left-to-Right: Confirmation of Participation, Record of Achievement, Certificate

Certificate A Certificate is provided to learners who participated in all graded exercises while being proctored and scored at least 50% of the total course points. The proctoring tool uses face recognition to ensure that the participant who is registered for the course is the one who sits in front of the computer during the exams. To achieve a Certificate, the paid proctoring option has to be booked. We recommend granting ECTS (European Credit Transfer System) credits to participants who received a Certificate. Hence, earning a Certificate is only offered for courses eligible for ECTS credit recommendation. Next to the information on the RoA, the Certificate contains the user's photo and two pages with detailed course information for the registrar's offices or HR departments. In [22] we analyzed how and for what purpose the learners used their Certificates.

Performance Records on Other Platforms Each platform offers its own performance records, which are named and styled differently and might require different achievement conditions. However, all performance records can be related to the three types described above. Not all platforms offer three levels of performance records but only one or two.

2.1.2 Courses

Learning is provided in the form of courses on the HPI MOOC platform. The platform was particularly designed for event-based, semi-synchronous courses. To support such a design, courses have dedicated start and end dates. Furthermore, the course content is separated into sections, which are published in a pre-defined time series. On openHPI, this is generally done on a weekly basis. Therefore, our sections often have titles, such as *Week 1*, *Week 2*, and the intermediate exams at the end of these sections are called *Weekly Assignments*. Typically, a course is configured in such a way that it is open for all platform users.

Active and Self-paced The time span from the start date to the end date is called the *Active state* of the course. During that time, the course’s teaching team supports the active learners. Once the end date has passed, the course automatically switches into *archive mode*. Participants can still use most of the learning materials in this *self-paced state*. However, the deadlines of the assignments will have passed, so it is no longer possible to submit a solution. Therefore, learners can no more earn *Records of Achievement* and *Certificates*. The forums will eventually be closed in archive mode, so they are available in read-only mode. Learners can still use self-tests and all video lectures. Programming exercises offered through our coding platform CodeOcean (see Section 2.1.3.4) can still be solved and tested in archive mode. However, programming exercises are usually part of the graded assignments in programming courses; thus, submitting the results to the HPI MOOC platform is no longer possible. In the self-paced mode, achieving a *Confirmation of Participation* is still possible, as this only requires visiting items and not passing exams.

Several of our partner platforms have different concepts and try to offer courses in self-paced mode only. This type of course design entails certain issues or restrictions regarding the exams/assignments and performance records. Offering a course in self-paced mode requires no deadlines on the exams, and this entails that the forum has to be closed from the course start to prevent discussions on exam content before everybody has submitted their results. Even that doesn’t guarantee that some students might obtain an illegal advantage by getting the answers to the questions from another place on the Internet. Additionally, allowing learners to help each other through the course forum is not possible, which is one of the major advantages of MOOCs over traditional e-learning offers.

Course Reactivation A new platform feature allows participants to reactivate a course that has already entered the self-paced mode. This paid feature enables learners who missed the course’s active phase to achieve a Record of Achievement. The course deadlines are set to an individual value once a learner books this feature. Generally, this individual deadline is eight weeks²³ from the point of time when the reactivation was started.

Multi-Linguality The platform’s user interface is multilingual and currently supports the following languages (as of August 2022): English, German, French, Chinese, Portuguese, Spanish, Ukrainian, Russian, and Dutch. Further languages can be provided on request. The languages in which the courses can be offered are unlimited. On openHPI, we try to achieve a fifty-fifty mix of courses offered in German and English language. OpenWHO offers courses in more than 70 languages. In addition to the actual course language, videos can be subtitled in any language (see Section 2.2.6).

²³This value is configurable per platform instance and product.

Invite-only Generally, the platform is designed to offer free, open online courses, and everybody can create an account and enroll in the courses for free. However, different possibilities for limiting access to a course exist. The simplest option is to not display the course in the list but instead to “hide it”, so only users who know the link to the course will be able to access it. Also, access to a course can be restricted by disabling users to self-enroll; hence, only the teaching team can invite learners by enrolling them to the course. There is also an option to add an external payment provider to restrict enrollment to courses for paying customers only.

2.1.2.1 **Course List**

The course list provides users with an overview of all courses offered on the platform. The list can be filtered based on certain criteria like topic, target group, or course language. The criteria can be defined for each platform instance individually.

2.1.2.2 **Channels**

Channels are a way to structure the course list and highlight certain courses by adding them to a certain channel. Originally, the channels have been added for the mooc.house platform, where they are used to separate the courses from the different course providers from each other. On openHPI the channels are used to separate rather peripheral course offers that are not part of the regular course program, such as the talks series including video recordings of HPI events or the (paid) HPI Academy courses. OpenWHO uses the channel concept to clearly distinguish the main fields of application.

2.1.2.3 **Course Sections**

A course consists of multiple sections. Sections can represent a certain period of time like a course week or a particular context like additional materials. On openHPI, the course is not entirely published at once, but the sections are gradually activated. In this way, the learners are “synchronized” and consequently deal with the same topics and questions simultaneously.

Publishing States and Start Dates Participants can access the learning materials within a section when the section is published and has no start date or the start date has already passed. If a section is published and the start date is set in the future, it is visible to learners but locked. Hence, participants can see that and when more material is coming but cannot access the content before the start date has passed. If a section is not published, it is completely hidden from the participants, and its contents are unavailable to them.

2.1.3 **Learning Materials**

The smallest units in a course are called “learning units” or “learning materials”. The unit can be a video, quiz, text, external exercise tool like CodeOcean or H5P (see Section 2.1.3.4), or a peer assessment. These units further consist of several components depending on its type. For instance, a video can have multiple video

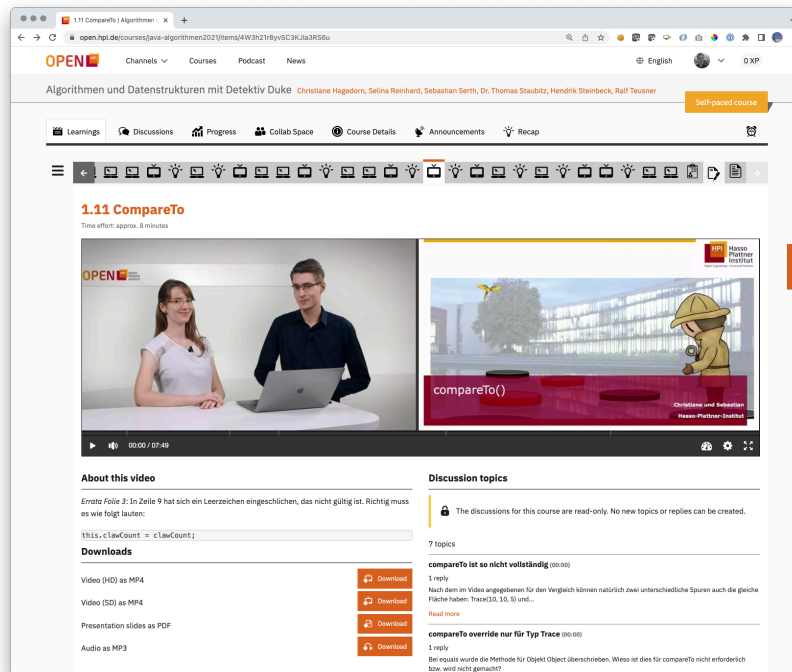


Figure 2.2: A video item in dual stream mode, including buttons for video, audio, and slide downloads

streams. Optional additions to the video can be presentation slides, extra explanatory texts, transcripts, or subtitles in different languages.

The most common learning unit used on the HPI MOOC platforms are learning videos, quizzes, and the provided reading material.

2.1.3.1 Video Lectures

One of the main learning materials on openHPI are short videos with a length of ideally ten minutes or less. We have put great effort into optimizing the learners' video experience. A video item consists of several elements, such as several video streams, transcripts and subtitles in different languages, download options, which will be explained in detail below. There are also several options to adjust the settings of the video, such as full-screen mode, the playback speed, and the language of the subtitles. Furthermore, it is possible to place additional texts (e.g., errata) and links below the videos.

Moreover, participants can start discussions directly on the video playback page and ask video-related questions, which are then automatically assigned to this video and simultaneously enter the global discussion forum of the course.

Video Player The video player supports *dual stream* (see Figure 2.2) and *Picture-in-Picture* (PiP) mode. For these, three different video streams must be attached to the video item on the platform. First, the lecturer stream contains the presen-

ter's recording and includes an audio track. Second, the slides stream contains the recording of the lecturer's computer screen. This does not have to be slides necessarily; it can be anything that lecturers want to show on their computer screen. Both streams are shown next to each other, their timing is synchronized, and the player allows each stream to be resized. The two streams are produced automatically and in sync, if the streams are recorded with the tele-TASK²⁴ recording system or a similar tool. Third, the PiP stream has to be created manually as a post-processing task. The PiP stream also serves as a fallback for devices that are not supporting the dual stream.

The video player allows learners to start, stop, pause, and jump within the video anytime. Participants can also adjust the video speed to adapt it to their individual needs and enable subtitles and video transcripts if they are provided with the video.

Subtitles and Video Transcripts The video player supports the display of subtitles within the video (see Figure 2.3) and an interactive transcript below the video (see Figure 2.4). The subtitles can be added in multiple languages, which participants can choose between. Similar to other text elements on the page, the interactive transcript can be searched for keywords and thus be used to navigate within the video. We have developed a separate tool (TransPipe, see Section 2.2.6) to simplify the process of generating subtitles, both automated and manually. Additionally, there is an option to add a video transcript in PDF format.

Slide Navigation Next to the usual navigation features, the video player supports navigation by slide previews. The tele-TASK team at the HPI has developed a tool that automatically recognizes when the lecturer proceeds to the next slide and creates small thumbnails for each slide. These are then shown below the video and allow the learner to navigate the video to a certain position quickly.

Offline Playback Per default setting, videos are available for download to the participants in *High Definition (HD)*, *Standard Definition (SD)*, and audio only format. In addition, participants can download the presentation slides. Videos might be unavailable for offline download if the course instructors decide to disable the feature, e.g., if license restrictions apply.

Video Hosting The platform itself does not host the videos to be streamed. The videos must be uploaded to an external tool, which currently can either be Vimeo or Kaltura. Both providers are well integrated into the HPI MOOC platform. The simplest and, in most cases, the cheapest solution is to work with Vimeo. However, there might be issues with Vimeo as the platform is blocked in several countries by their governments. If these countries provide important target groups, Kaltura is a valuable alternative. Kaltura also allows to set up a self-hosted instance of the

²⁴<https://www.tele-task.de>; last access: 02.09.2022.

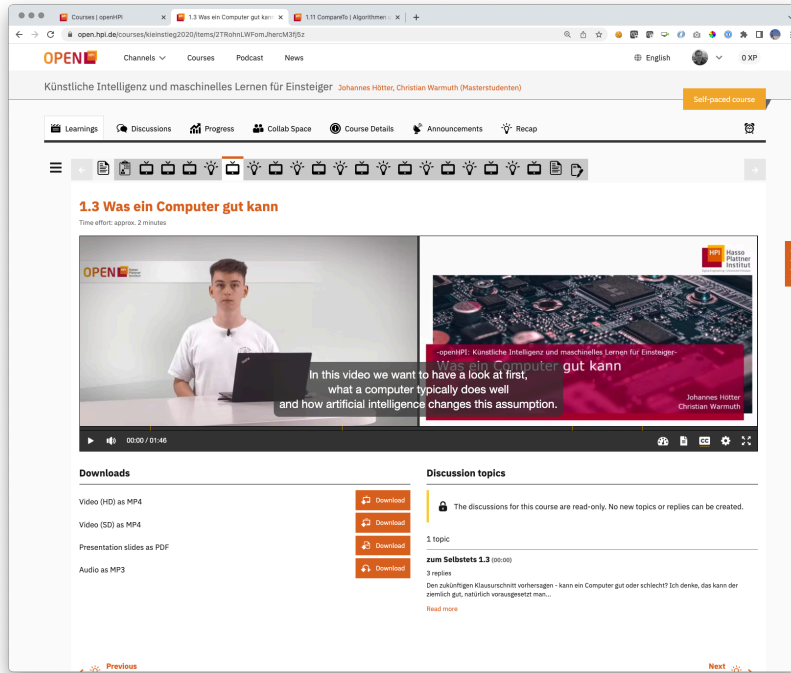


Figure 2.3: A video item with subtitles

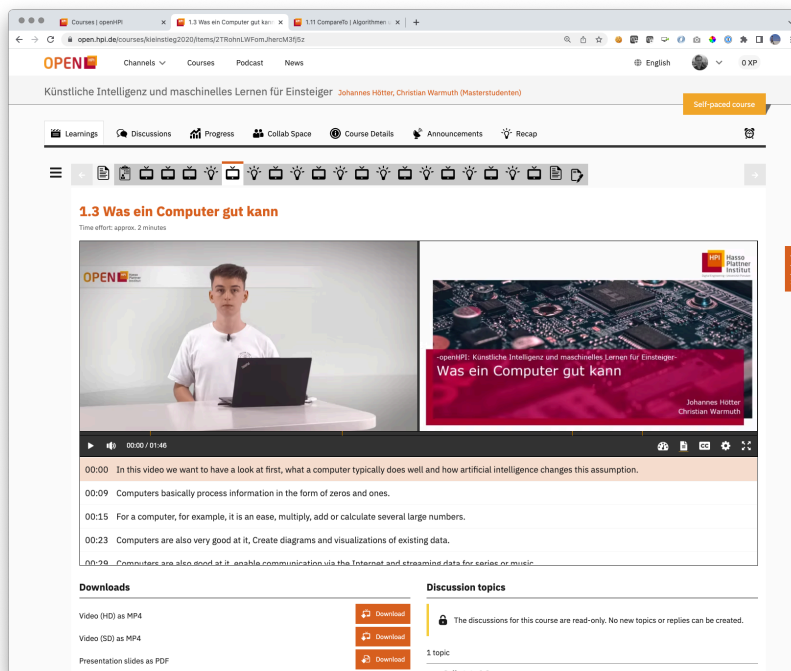


Figure 2.4: A video item with an interactive transcript

platform. This can become necessary if the content of the videos needs to be treated confidentially and should not freely be available on the Internet.

Open Mode In general, all courses on the HPI MOOC platforms require that a learner is registered on the platform and enrolled in a course to access any of the included learning materials. There is one exception to that rule for videos. Videos can be configured to be available in the so-called *open mode*. Videos in open mode can be directly indexed by Google and accessed without having an account or being enrolled in the course. Once the learner tries to access another course item, they receive a message that they need to sign up for the course.

Podcasts By extending the MOOC offering with additional, complementary podcasts, we aim to increase the learner retention rates and reveal differences in the usage of various media. We have ongoing research for podcasts which are intended to be entertaining and educating at the same time to be integrated into everyday activities [24]. First quantitative assessments of more than 900 learners showed positive feedback on podcasts. More than 80% of learners, previously being used to video-based learning only, considered the Podcasts “easy to follow”. Furthermore, course performances of learners consuming a podcast instead of the video elements did not show significant differences [25]. We hence continue to evaluate the structure and format of educational podcasts in ongoing research.

2.1.3.2 Text

In addition to videos, the platform supports text items to provide learning content (see Figure 2.5). A limited number of text formatting options via Markdown is supported. Commonly used are simple lists, headlines, bold and italic fonts, images, and links to other websites or downloadable PDFs embedded in the text. The text item also supports simple tables; however, their styling might be suboptimal on small device screens. Usually, textual learning materials are provided sparsely on openHPI and only to support preceding video lectures.

2.1.3.3 Quizzes

Next to videos, quizzes play an essential role in all openHPI courses (see Figure 2.6). Quizzes are used in different contexts as self-tests, graded or bonus assignments, and surveys. A quiz consists of questions, corresponding answers, and explanations.

So far, quizzes support four different types of questions to choose from:

- **Single Select** questions allow participants to select one of the provided answer options and have exactly one correct answer.
- **Multi Select** questions enable participants to select a flexible number of answer choices. They can have one or more possible correct answers. Incorrect answers are penalized, so participants do not receive full points for selecting all options.

- **Free Text** questions require short text answers in a single line. They can be configured with a list of possible correct answers.
- **Essay** questions allow participants to write long text spanning several lines. There is no correct answer for this type of question.

Answer options for single and multi questions can be configured to be shown in random order. Except for essay questions, the evaluation of quizzes is a fully automatic process. When creating the quizzes, the teaching team marks each answer option as correct or incorrect. The total points awarded are configured per question, and individual results are calculated depending on the question type and the given answers. In addition, explanatory texts can be added to questions and answer options. Explanatory texts are only visible to participants when the quiz results are published and the quiz solution is submitted.

Self Test Self-tests do not contribute to the learners' certification but allow participants to verify their understanding of the learning unit. In self-tests, learners receive direct feedback as soon as they have submitted the quiz by showing correct and incorrect answers along with explanation texts where they have been provided (see Figure 2.7). Usually, self-tests offer unlimited time to work on and can be repeated as often as desired by the participant.

Graded The graded exams provide the majority of the certification-relevant points. Usually, participants can take each exam only once, and the working time is limited to between one and two hours. The results of graded assignments are not shown before the deadline, or a separate publishing date has passed.

For graded exams, an additional proctoring option can be enabled. The proctoring restricts itself to automated face recognition to make sure that the registered participant is the person who solves and submits the exams. For now, we refrained from additional options that are more privacy intrusive, such as audio recordings and room checks. Before the platform's proctoring tool was integrated and published, we ran a few experiments [54]. Recently, we conducted a more general analysis on proctoring and online exams [58].

Bonus Missing points can be topped up with bonus assignments. As for graded exams, the results of bonus assignments are not shown before the deadline, or a separate publishing date has passed.

Survey Surveys are optional items that provide valuable feedback and information from the participants. Their answers might help develop the platform, future courses, or answer specific research questions. The teaching team can provide instructions to explain the purpose of this survey and how the results are used. Participants will not see any of the results after submitting their survey. Surveys can be added anywhere in the course. We recommend using standardized surveys at the beginning and the end of each course.

2 Functionality and Format

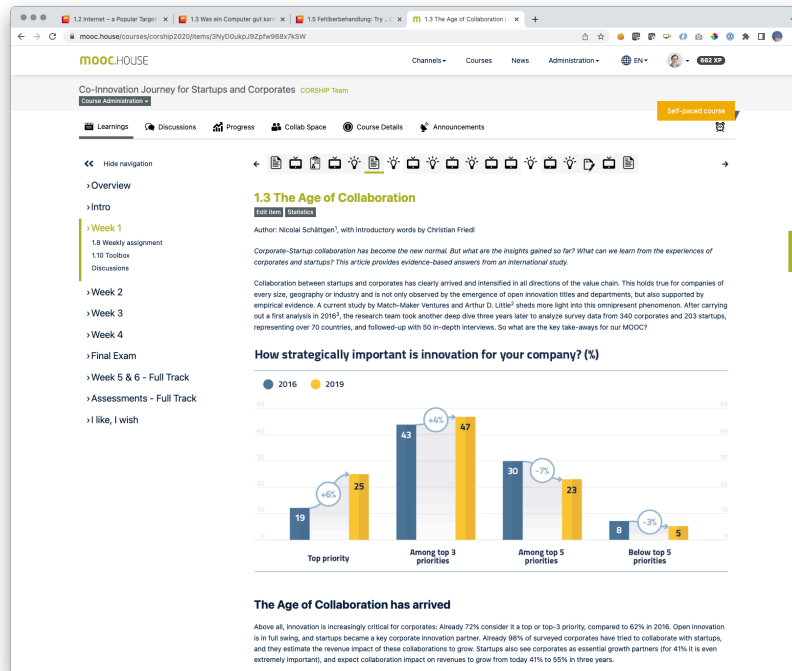


Figure 2.5: A text page on the mooc.house platform

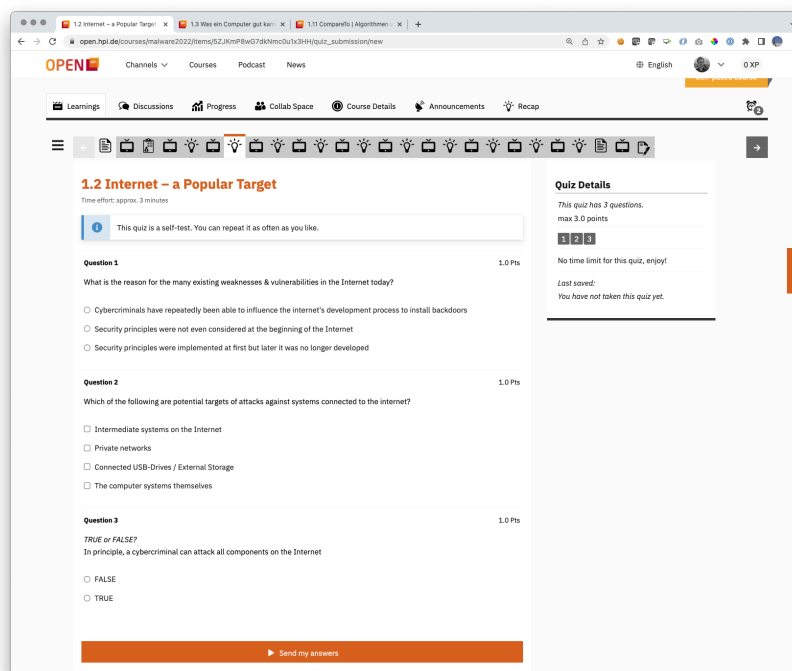


Figure 2.6: A quiz before it is submitted

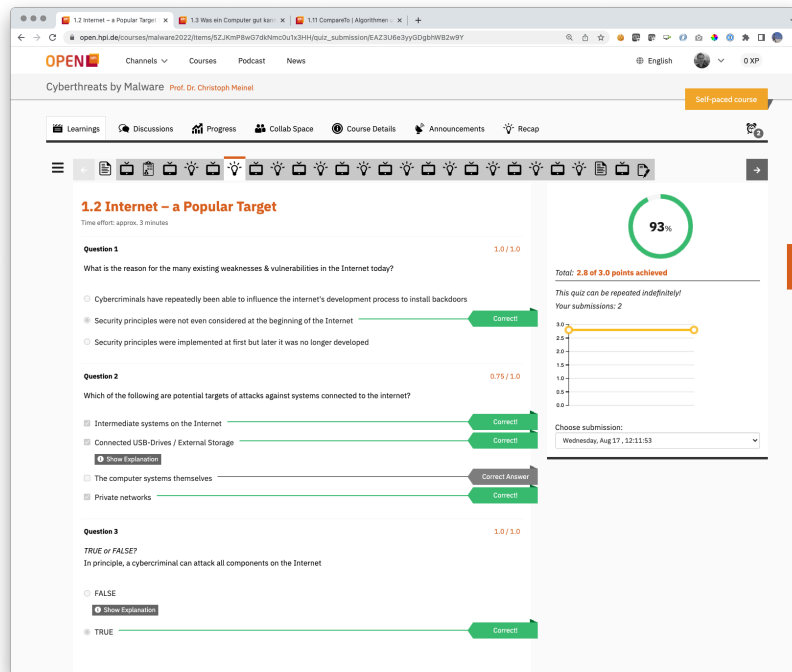


Figure 2.7: Direct feedback on self-tests

For ongoing research, we started adding surveys as self-reflection questionnaires for the learners. The results of those surveys are prepared and presented in a manual process through text items to the participants (see Figure 2.8). The purpose of the self-reflection questionnaires is to make participants reflect on their existing knowledge of the topic and the content learned [16]. In addition, the public presentation of the results tries to improve the learners' sense of community.

For more complex research surveys, it might be necessary to use an external survey tool. We provide a LimeSurvey integration for this purpose. However, we recommend using the integrated survey solution to keep participants within the platform instead of redirecting them to the external tool.

2.1.3.4 External Exercises

Like many other platforms and learning management systems, openHPI supports the Learning Tools Interoperability (LTI) interface. Via this standardized interface, it is easily possible to attach other learning tools to openHPI and integrate them smoothly. Like quizzes, external exercises can be of type self-test, graded, bonus, and survey. The points achieved within an external exercise can count towards the overall course points. If this is advisable depends on the tool used. The following paragraphs introduce three of these tools, which are commonly used to enrich our learning materials with interactive elements.

2 Functionality and Format

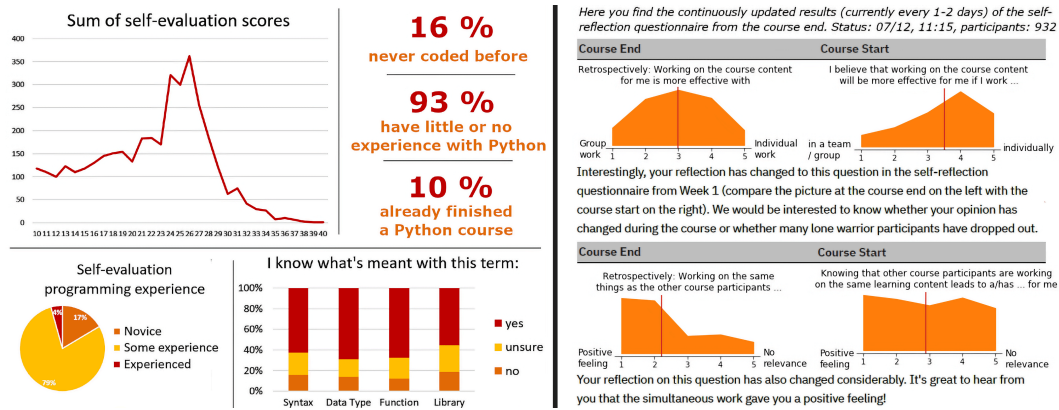


Figure 2.8: Results of different self-reflection questionnaires

CodeOcean The first external exercise tool attached to the core platform via the LTI interface is CodeOcean. CodeOcean provides programming exercises directly in the web browser, which allows a low-barrier entry point for programming novices. CodeOcean supports, at least in theory, every programming language that can be executed on a Linux operating system. In practice, we have used CodeOcean for Java, Python, Ruby, JavaScript, and R exercises in several courses. These courses range from offline seminars at our institute to full-scale MOOCs on our MOOC platforms with tens of thousands of registered participants. The participants' coding submissions are tested and automatically graded in CodeOcean. Usually, learners can execute the exercises as often as they want before transmitting the achieved score back to openHPI.

In our programming MOOCs on openHPI, most programming exercises in CodeOcean are implemented as graded assignments and provide relevant points for the course performance record. Hence, we usually limit the submission of scores until the course end date but allow learners to work on all programming exercises throughout the whole course period. In addition to the programming language courses, we have also employed CodeOcean to conduct a course on test-driven development with JUnit. The participants had to write test cases and an implementation for each exercise. While we could not guarantee that they have been working test-driven, CodeOcean at least allowed us to evaluate if they had written test cases that covered the requirements of the exercises.

CodeOcean promotes the concept of files within exercises. Multiple files can be editable per exercise, while the instructors have every possibility to restrict read and write access to each file individually. CodeOcean does not limit the number of executions before an exercise has to be submitted. Its development environment is based on widespread web standards that are natively supported by current web browsers. Hence, learners can interact with their program during execution and provide input for the command line or turtle graphics [40].

Throughout the years, CodeOcean has been one of our major research vehicles in the context of auto-graded programming exercises, and a long list of papers has

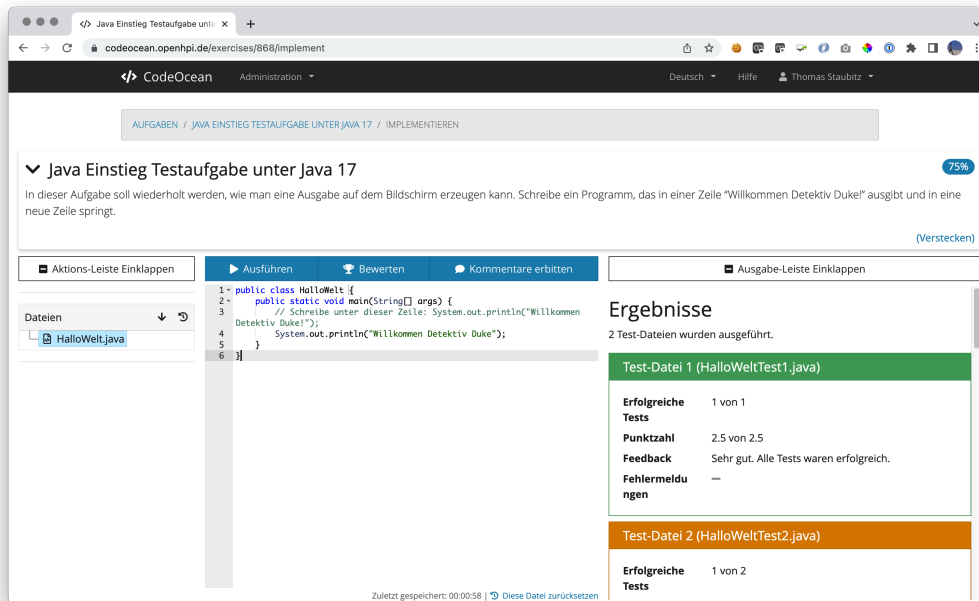


Figure 2.9: CodeOcean showing an introductory Java Exercise. The top part contains the exercise description, while the main view consists of three columns: In the left pane, a file tree with all available files is shown, the center provides an interactive code editor with dynamic syntax highlighting and optional code completion and the right sidebar features test results with automated feedback.

been published. Initially, it was developed as an element of a Master's thesis by a former HPI student. Before the actual development of CodeOcean started, we experimented with a few other tools [26, 53], further similar tools have been analyzed to examine the state-of-the-art [45], and finally, the initial design of CodeOcean was introduced in 2014 [46].

CodeOcean is developed as a web-based development environment with a client-side code editor and a server-side component for code execution. An advantage of this approach is that it allows us to provide learners with a homogeneous and novice-friendly programming environment. It simplifies the support of multiple programming languages and third-party libraries while providing a consistent code execution and assessment workflow. And very importantly, it allows us to collect insights into learners' problem-solving strategies by analyzing their code submissions. In this context, the effects of contextual tips [42] in the exercises, the impact of exercise descriptions [59], automated interventions [60], and retention rates [61] have been examined. Other research conducted explored options to allow high-school teachers to adapt the coding exercises to their needs [38, 41, 43] and analyzed security aspects of allowing total strangers to execute their code on our servers [39].

Enabling additional programming languages requires generating a Docker image containing all required components and (optionally) writing an adapter for parsing the output of the testing framework to be employed. For each programming language, we created a dedicated set of a custom Docker image containing language-specific tools and a testing adapter suitable for the employed testing framework. This allows an exercise author to use the standard testing frameworks of each language, e.g., PyUnit for Python, JUnit for Java, and RSpec for Ruby. The respective testing adapter for each framework extracts meta information (such as the number of failed tests) and test-specific error messages. So far, an additional adapter was written for C++ but has not been used yet in a course. We further created an adapter to parse the output of PyLint6 as a static code analysis tool. In addition to the test output, it provides suggestions on improving the code style based on predefined rules. It enabled us to employ a new set of exercises focusing on different aspects of software engineering (such as understanding and refactoring existing code).

CodeOcean consists of a service-oriented architecture with a user-facing Ruby on Rails application²⁵ connected to a PostgreSQL database, Poseidon²⁶ as an executor middleware written in GoLang for best performance and the container orchestrator Nomad²⁷ to be used with our custom-defined Docker images. This setup decouples the code execution from the remaining web application and allows independent scaling of the underlying components. In addition to openHPI, CodeOcean can be integrated with other Learning Management Systems (such as Moodle, Ilias, or Canvas) and also supports an import and export of exercises to other compat-

²⁵<https://github.com/openHPI/codeocean>; last access: 02.09.2022.

²⁶<https://github.com/openHPI/poseidon>; last access: 02.09.2022.

²⁷<https://www.nomadproject.io>; last access: 02.09.2022.

ible systems. As access to the instructor backend of CodeOcean is still limited to authorized teachers, we started another platform called CodeHarbor²⁸ to host all programming exercises and provide a collaboration platform for teachers to exchange exercises (see [40] for more details). All our tools (namely CodeOcean, Poseidon, and CodeHarbor) are provided as an open-source software and can be found in the respective GitHub repository.

H5P H5P is an open-source library that delivers interactive HTML content (see Figure 2.10). It can be embedded through iframes and offers integrations for different platforms (e.g., Moodle). As a result of the AI Campus project, an H5P wrapper was developed as a standalone application that adds the LTI interface to the H5P core. The interactive content library offered by the H5P tool can thus be embedded in courses offered on openHPI. H5P should not be used for graded assignments, only for ungraded exercises, as the solutions can potentially be found in the HTML code. Examples of H5P exercises are: Drag and Drop, Fill in the Blanks, Find the Hotspot, or Multiple Choice questions. We encourage the teaching teams to use only those H5P features which have no native alternative on the platform.

JupyterHub Jupyter Notebook is the most widely-used system for web-based interactive notebooks that allow literate programming. Its user-friendly design makes data analysis easier to document and reproduce. Even though Jupyter Notebooks were initially created as a tool to be used in scientific workflows for data analysis, they are quickly becoming a common choice for university courses. Many university classes use Jupyter notebooks as the preferred medium for homework, labs, projects, and lectures. When the number of users and memory requirements are low, it is easy to set up a JupyterHub on a single server. However, setup becomes more complicated when Jupyter Notebooks need to be served at scale to tens or hundreds of users. In MOOCs, where the number of learners in a given course can reach hundreds of thousands, setting up JupyterHub on a single server is almost impossible. Next to increasing the amount of hardware or outsourcing to the cloud, scheduling strategies are an example of how to solve the problem. Providing Jupyter Notebooks for our MOOC platform is something we are currently investigating in a research setup. Another problem to be addressed is the possibility of automated grading of exercises, similar to the existing process on CodeOcean [9].

2.1.3.5 Peer Assessment

Peer assessments (PAs) have proven to be an appropriate way to assess complex or creative tasks that automated tools cannot assess. As a teacher-based evaluation is not an option for MOOCs due to their large number of participants, a peer assessment is the only possibility to offer such complex or creative tasks. However, a peer assessment entails a comparably high effort for the participants. Hence, for the acceptance amongst the participants, it is essential that the tasks are well-

²⁸<https://github.com/openHPI/codeharbor>; last access: 02.09.2022.

The screenshot shows a web browser displaying an openHPI course page. The page title is "Algorithmen und Datenstrukturen mit Detektiv Duke". The main content is an H5P exercise titled "1.12 Laufzeit bestimmen (Teil 1/2)".

The exercise text reads: "Laufzeitanalyse 1 - Ordnet die Laufzeitkurven den Codebeispielen zu. Die dickeren Kurven stellen die Anzahl der Schritte da, die bei der Ausführung der Algorithmen gemessen wurden. Die anderen Kurven stellen typische Laufzeiten im Vergleich dar." (Run-time analysis 1 - Assign the run-time curves to the code examples. The thicker curves represent the number of steps measured during the execution of the algorithms. The other curves represent typical run-times for comparison.)

There are three small graphs at the top, each with a title and a y-axis:

- Graph 1: "Quadratisch gemessen" (Quadratic measured), y-axis 0-2500.
- Graph 2: "Exponentiell gemessen" (Exponential measured), y-axis 0-350000000.
- Graph 3: "Linear gemessen" (Linear measured), y-axis 0-200.

Below the graphs are two code snippets in Java:

```
public static void sieveOfEratosthenes(int n) {
    boolean prime[] = new boolean[n + 1];
    for (int i = 0; i <= n; i++) {
        prime[i] = true;
    }
    for (int p = 2; p * p <= n; p++) {
        if (prime[p] == true) {
            for (int i = p * p; i <= n; i += p) {
                prime[i] = false;
            }
        }
    }
}

public static void backAndForth(int[] array) {
    for (int i = 0; i < array.length; i++) {
        for (int j = array.length - 1; j >= 0; j--) {
            System.out.println("i: " + array[i] + " j: " + array[j]);
        }
    }
}
```

There is also a larger graph on the right side of the code snippets, titled "25000", showing a comparison between "Kubisch" (Cubic) and "quadratisch" (Quadratic) curves.

Figure 2.10: H5P exercise embedded in openHPI

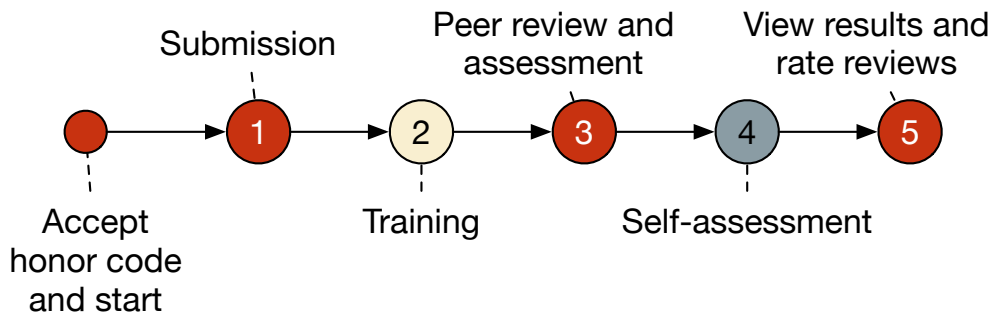


Figure 2.11: Peer assessment steps on openHPI: red-colored steps are mandatory; the other two are optional

defined and adequately communicated. This information allows learners to decide upfront if they want to invest effort in the exercise. As many of our learners participate in our courses only in their spare hours, many are just not interested in too complex tasks, and in case their course success entirely depends on such a task, they prefer not to take the course. However, a smaller number of participants strongly welcome this type of task as it lifts the learning experience to a new level. To solve this dilemma, we have two approaches: (1) We provide the peer-assessed task as a bonus, or (2) divide the course into a full track and a fast track that offer different performance records. One way or the other, it is important that the task to be solved is not too simple, as otherwise, the participants feel that the same effect could have been achieved with a multiple choice quiz that introduces much less effort.

On openHPI, a peer assessment consists of three mandatory and two optional steps (see Figure 2.11). After accepting the honor code, the participants work on their tasks and submit the result. Depending on the course and the task, the participants have about two to six weeks to complete the PA. In the second step, they must review and grade a certain number of peers. The instructors define how many peers have to be graded – the recommended minimum is three, and the optimum is five. Grading rubrics are defined by the teaching team individually for each task. In addition to the summative assessment, the participants are encouraged to provide a formative assessment of their peers' work. In the final step, the received feedback can be rated by the participants. Furthermore, optional training and self-assessment steps can be added.

If the optional steps are activated, each can be configured individually to define whether the participation is optional or mandatory. For each step, an unlock date and a deadline can be set. If the deadline of any mandatory step has been missed, the learner has failed the assignment. If the deadline of an optional step is missed, the participant is forwarded to the next step and can no longer complete the missed step.

The Training Step The teaching team has the possibility to add a simple training step to the PA workflow. The teaching team grades at least 10 samples of the participants' submissions for the current PA to ensure a more diverse training pool. A diverse set of samples benefits the training as it provides the participants with examples for each level of quality. While a sample submission is being reviewed by a teaching team member, it is blocked for all other teaching team members to ensure that it is not accidentally graded twice.

As soon as the training step has been opened for the participants, the submissions that already have been graded by the teaching team are presented to the participant one after the other. Initially, the students only can access the submission, but not the grading of the teaching team. This view is more or less identical to the view of the actual evaluation step. As soon as the participant herself has graded the sample, she/he is presented an overview, which shows the teaching team's comment and adds indicator arrows to the rubric options that have been selected by the teaching team and those that have been selected by the participant. This way, the participant can easily detect scoring differences.

Peer Evaluation—The Core Feature The system allows the teaching teams to define a summative and a formative part of the PA. The summative part is handled via grading rubrics. A rubric consists of three components: An (ideally) self-explanatory title (e.g., "Writing Style" or "Creativity"), an explanatory text, and rubric options. The explanatory text can be used to give additional hints for the grading. This serves for a better guidance of participants, which helps to understand the rubrics and aims to reduce grading bias. Rubric options represent the different levels of attainment for their respective rubric. The scoring scale of the rubric options is not required to be linear and can weight levels differently, e.g., 1, 2, 4 points, or can be an interval, e.g., 0, 5, 10 points. Rubrics and rubric options are defined by the teaching team according to their needs for each PA.

Participants grade a submission based on the available rubrics. In addition to this summative feedback, the participants are encouraged to give their peers formative feedback in the form of free text reviews. All reviews are double-blind to protect peers' identities from each other and to prevent call-outs or conflict escalations outside the peer assessment. Participants who have completed the required reviews in time, advance to the next step. Participants who do not, receive zero points for the assessment. Voluntary additional reviews up to the number of required reviews can be handed in as long as the deadline has not passed.

Distributing Submissions to Peers The distribution mechanism that maps submissions to peers is the core of the evaluation step. Our goal is that each participant receives sufficient feedback. Each participant is required to write a certain amount (n) of reviews, which means that each participant ideally receives that amount of reviews in return. The size of n is defined by the teaching team. We are using a dynamic mapping approach that assigns submissions on demand from a pool of available submissions. This allows to re-balance review counts for submissions on the fly as the current submission distribution state can be considered. With

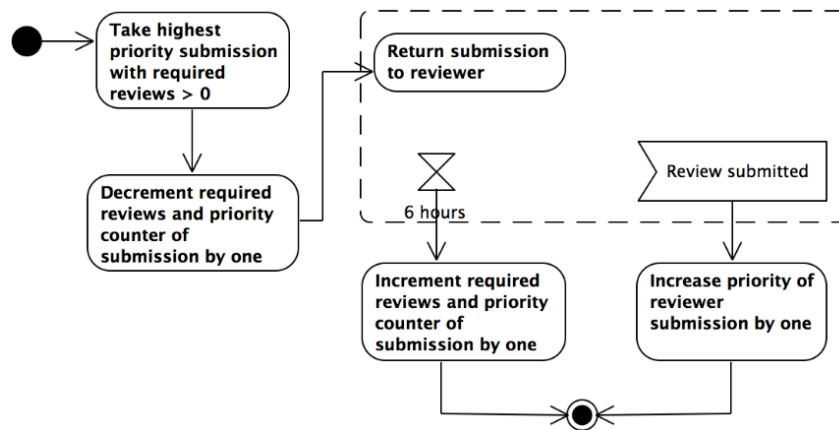


Figure 2.12: Activity diagram summarizing the distribution mechanism. The dotted line marks the time range from the start to the end of the actual review process. This time range starts when a participant requests a submission to review and ends either when the review is submitted or after six hours if the participant fails to return the review in time. A submission is the work that has been submitted to the system by a participant and needs to be reviewed.

additions such as review expiration, the problem of redistributing submissions is solved without requiring heuristics that determine when submissions are considered for redistribution. Our implementation of the pooling approach introduces a *submission priority principle* to balance submission distribution and the possibility that a submission is reviewed by several reviewers in parallel (see Figure 2.12).

Review Rating In the result step, participants can see the reviews they received and – as soon as the deadline for the evaluation step has passed – their final grade for the assessment. Our system introduces the option to report received reviews for misconduct, analog to reporting submissions in the evaluation step. A further improvement is the possibility to rate the received feedback. Learners, thus, can defend their work against rogue reviews and at the same time reward peers who write thoughtful feedback with additional points. A four-point scale has been chosen and translates into no point for zero stars, 0.3 for one star, 0.7 for two stars, and 1.0 for three stars. This bonus can accumulate to a notable amount of points, even considering that participants already can earn a significant amount of base points for their submission in the PA compared to other assessment forms on our platform. This may seem to be a rather high bonus. However, reviews are an integral part of peer assessment and its learning experience. Until the participant has rated the feedback, we only show her the overall grade but not the individual grades that have been assigned by the peers. As soon as the written part of a review is rated, we also show the individual grade given by that peer. This is an attempt to make sure that participants only rate the usefulness and quality of the written review and not the grade that they received from their peer. Furthermore, this

method uses the inherent curiosity of the participant as an incentive to rate the reviews.

Grade Computation We originally opted for an experimental weighted average approach taking multiple factors into account. Some of these were intended to serve as long-term adjustments to reduce the impact of rogue reviewers and reviewers who constantly deliver poor review quality. In the end, however, we skipped that plan as this calculation would have become absolutely nontransparent for all stakeholders. In most cases the median is good and will be only marginally improved by employing complex weighting mechanisms. In those rare cases where a submission receives less than three reviews, we use the average instead of the median.

The final grade consists of multiple components, which are transparently communicated to the participant in the result step:

- Median of review grades (base points)
- Self-assessment bonus points
- Review feedback rating bonus points
- Grading delta

The median of the review grades is calculated per rubric. We then sum up these values to the final amount of points for the participant.²⁹

Our research results on design [52] and the acceptance of the platform's peer assessment system [47] have been published. Furthermore, we contributed to a systematic literature review on peer assessment in MOOCs [12].

Team Peer Assessment To add another level of complexity, peer assessments, can also be configured to assess the work of teams. The biggest challenge here is to minimize the drop-out rates in the teams. While we do not see the drop-out rate of MOOCs as a general issue, for the team tasks it is, as every drop-out in a team is frustrating for the remaining members. See Figure 2.13 for an overview of the general workflow of a team assignment.

Several aspects of collaborative, peer assessed team work in MOOCs have been researched intensively by the openHPI team [4, 44, 48, 49, 50, 51, 55, 56].

2.1.3.6 TeamBuilder

Since learners in these courses are numerous and generally do not know each other, a tool is needed to find suitable learners and assign them to teams. This tool is the TeamBuilder, which is connected to the core platform via an *Application Programming Interface* (API). As different courses have different requirements on the matching criteria, this tool requires some flexibility. The teaching teams, therefore, can configure which parameters to employ, and, where it makes sense, to decide if the parameter should be employed homogeneously or heterogeneously. To avoid incomplete data sets, the chosen set of parameters has to be published before the

²⁹https://open.hpi.de/pages/p_a_grading; last access: 02.09.2022.

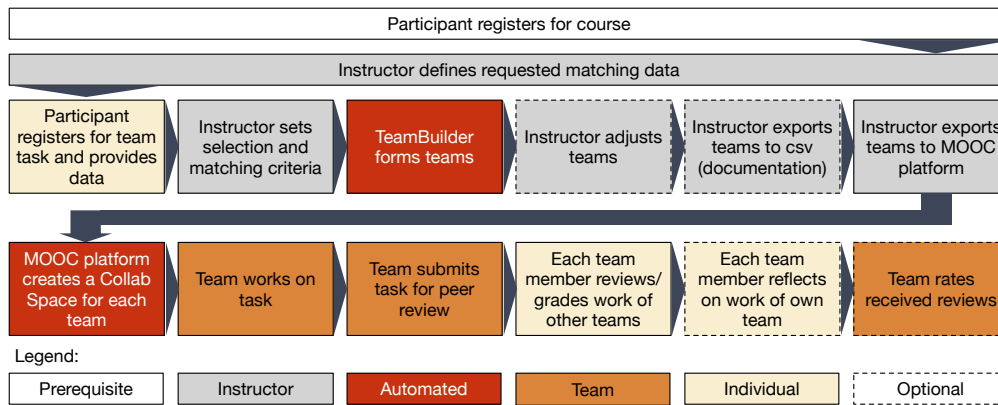


Figure 2.13: Team task workflow on openHPI

data can be collected from the users. From then on, it is no more possible to add additional parameters. Later on in the team building process, it might become obvious that it is not feasible to form teams based on the selected criteria due to the combination of available participants. The teaching teams are, therefore, enabled to exclude any of the selected parameters from the matching process. They are advised not to ask for more data than they actually intend to employ for the team building process. The following paragraphs briefly describe the possible parameters.

Language If the parameter is activated, the tool attempts to match participants who speak the same language. The choice is limited to the 20 most common languages among the users of our platforms. If a teaching team decides not to activate this parameter, the course language is assumed to be the “lingua franca” among the participants.

Location and Timezone Participants can be clustered based on their location to allow face-to-face meetings. Only homogeneous location based matching is supported. The *TeamBuilder* employs the DBScan algorithm to handle the location based clustering. In case the location parameter is deactivated, the tool automatically attempts to match the participants based on their preferred timezone. Matching based on timezone allows two different modes, homogeneous or culturally diverse. In homogeneous mode, the tool attempts to find learners in the same time zone, and only spreads to adjacent time zones if there are not enough learners in the targeted one. In culturally diverse mode, the tool picks three adjacent time zones and tries to spread the team members as wide as possible within their limits.

Area of Expertise, Age, Gender According to Design Thinking, teams work best if they are divers in area of expertise, age, and gender. Therefore, this is the default setting here. It is the teaching teams’ choice if they prefer more homogeneous settings here.

Custom Parameters The tool allows teaching teams to define custom parameters. Per default, custom parameters only allow homogeneous matching. However, they can also be used for manual reordering in particular cases. In the following are some examples how these parameters have been used.

- Assignment to different tasks: Learners were allowed to select between different tasks and then were matched according to their task preferences.
- Lead or Follow: The learners have been asked if they would be interested in leading the team or rather just tag along. In this case the matching had to be done manually as we attempted to have at least one team leader per team.
- Entrepreneur, Student, or Corporate: This was a very course specific setting. The goal was to have at least two members of each group. Manual adjustments were required.

We have deliberately decided to take a simple approach for the matching criteria and refrained from too detailed Belbin style questionnaires. We only collect the data that is absolutely necessary to build the teams based on the teaching teams' ideas. On the other hand, all data that we ask for is mandatory. Learners who refuse to provide a relevant piece of information will not be considered for the team assignments.

Next to the selection of matching criteria, the tool allows defining a range for the desired team size. It attempts to build the teams as close as possible to the given upper limit. Finally, the tool allows limiting the total number of participants for the team assignments within a course. This limiter can be employed either in first come/first serve mode, or it can be employed in connection to the results of the assignments that have been handed-in before the team building item is published in the course. First come/first serve simply allows the teaching teams to select the first N participants that have applied for the teamwork assignment(s). In case the teaching team decides to employ the results of one or more assignments to decide which participants will be admitted to take the teamwork assignments, the tool allows two different settings:

1. Set a maximum amount N of students to be admitted for teamwork – the N participants that performed best so far will be admitted.
2. Set a lower limit for the results in the exams that has to be achieved – in addition to the total maximum amount of participants, a lower limit for the results can be set. Only participants that have achieved at least $X\%$ of the points in the relevant exams – but no more participants than the total maximum – will be admitted.

Our research showed that learners who struggled in previous exams, will drop out of the team task with an almost 100% probability. While the *TeamBuilder* attempts to build teams fully automated, it still allows manual corrections. Participants can be moved from one team to another, new teams can be created, and existing ones can be removed. The tool provides meta-information about the created teams, such as

the max. local distance between team members or the team's state of heterogeneity concerning a given parameter.

Finally, there are two options to export the created teams. The *TeamBuilder* can either create an Excel sheet for further usage, or it can directly create *CollabSpaces* on the core platform and add the team members to their respective *CollabSpace* (see Section 2.1.3.7).

2.1.3.7 CollabSpaces

CollabSpaces provide private “rooms” within the general course that are only accessible to group members who have created the space or the team that was assigned to the *CollabSpace*. Within the *CollabSpace*, the teams and groups have access to synchronous and asynchronous communication tools as well as a co-creation tool to jointly work on digital artifacts.

We differentiate between two collaboration concepts: *Groups* and *Teams*. *Groups* are loosely coupled and have a self-set goal or a common interest, e.g.:

- A group of native Spanish speakers in a course offered in English, who prefer to discuss in their mother tongue.
- A school class participating in a course with their teacher.
- A group of participants discussing a special interest topic that goes beyond the communicated learning goals of the course.

Groups are self-organized. Each course participant can create *CollabSpaces*. The participant who creates the *CollabSpace* has administration privileges and decides if it is public or private (invitation-only). *Group* members can come and go as they want. *Teams* are tightly coupled and have a common task on which they are collaborating. The task is an essential element of the course and part of the grading scheme. *Teams* are formed by the teaching team using the *TeamBuilder*.

2.1.4 Discussion Forum

The platform is equipped with a discussion forum to enable communication between the course participants and the teaching teams (see Figure 2.14). All discussions belong to a specific course. All registered and enrolled course participants can post, answer, and comment as long as the forum is open. The forums usually close about a week or two after the course has ended, and the forum is then switched to read-only mode.

As MOOCs foster social learning within a peer group, we encourage learners to contribute to the social atmosphere and build a sense of community within a course. Although social collaboration is one key aspect of MOOCs, only a small fraction of learners actively participate in the course forum. On the other hand, much more learners passively use the forum by reading posts. We try to lower the barrier of the first active contact with the course forum by using welcome threads and other ice-breaking activities. Although those icebreakers are pretty successful, we cannot encourage all learners to share their stories in the forum

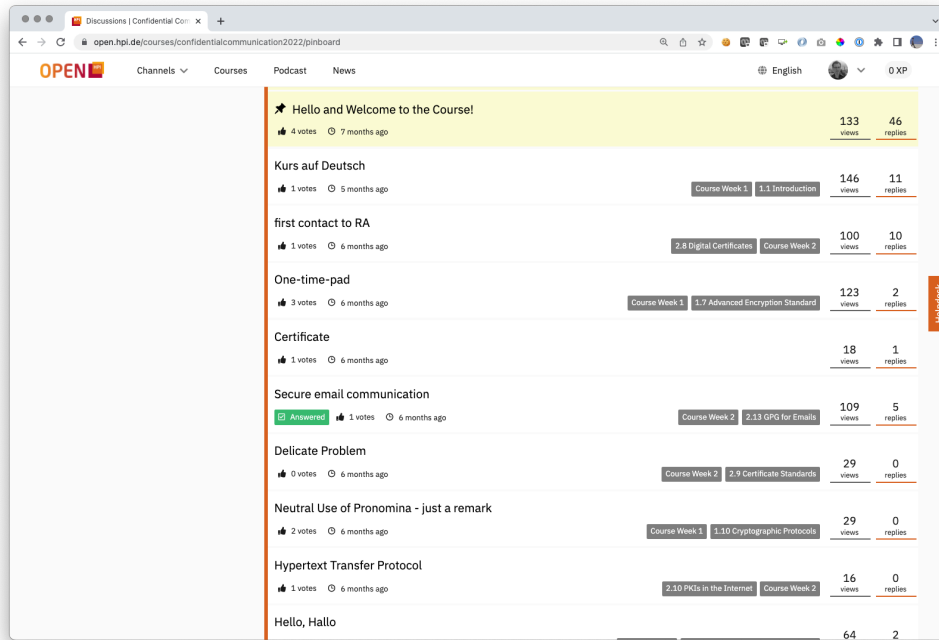


Figure 2.14: Discussion forum for a sample openHPI course

threads. Hence, we encourage learners through e-tivity quizzes granting self-test points for self-reflection on the forum usage.

The system also rewards so-called Experience Points (XP) to support active participation in the forum. Participants receive these points for high-quality contributions (answers with many positive reviews or ones marked as correct). Participants who have exceeded a certain threshold of XP receive recognition in the forum with the reward of a judo belt. The bigger the number of XP, the higher the grade of the judo belt [57].

Specific posts can be configured as sticky by the teaching team, which ensures that important information is always located at the top of the list. All participants can vote for questions and answers (in each case, one vote per participant and question/answer). The one who has posed the question can mark an answer as correct (in the sense of especially helpful).

Participants are given the option of reporting deficient forum contributions. After the post has been reported three times, it is automatically blocked and submitted to the teaching team for evaluation. The teaching team can then decide whether the post should be permanently blocked or released again. It is also possible for the teaching team to block posts independently of participants' reports. In this case, blocked posts are only visible to the teaching team.

2.1.5 Quiz Recap

This feature allows the learners to recapitulate the course content before weekly assignments and the final exams [3]. At the start of a recap session, the learner chooses the number of questions to be answered and in return receives a set of quiz cards with questions in random order. Only self-test questions of the type Single Select or Multi Select that are accessible to the learner and have not been explicitly excluded are eligible for the quiz recap. If the learner answers a question incorrectly, the question will be repeated in the recap session. At the end of the recap session, an overview of the learner's performance is displayed.

2.1.6 Learning Objectives

The terms learning goals and learning objectives are often used interchangeably as both describe the intended outcome of a learning process. In our context, a learning goal is a broad statement of what a learner will be able to do at a certain time. For example, it can refer to the outcome of a single course but also of taking several courses as a thematically consecutive series to acquire certain competences. In contrast, learning objectives have a narrow focus, describing specific and discrete units of knowledge and skills being acquired. These objectives are the results of short-time activities that can be achieved by following a certain number of steps. Consequently, they are specific enough to be observable and measurable. A learning goal thus can comprise multiple learning objectives.

Proper assessment methods need to be in place to measure and verify the attainment of objectives. Only if the outcome is measurable, a quantified decision about the level of success can be made and provided as feedback to the learner, which is desired to enable self-regulation. MOOCs typically group the content by specific topics and address different smaller thematic units. In contrast to the predominant orientation towards the completion of the course, individual objectives can be understood as completing certain parts of the course material by offering a form of optional personalized pathways. Therefore, we define the completion of these thematic units as the basis for learning objectives, since they represent the smallest unit of imparted knowledge within a course. The verification of the acquired knowledge is possible through the provided exercises. Furthermore, personalization is achieved by offering different didactically appropriate objectives per course, created by the teaching team and course instructors, from which the learner can select one if desired and follow it individually.

Three types of objectives are supported by the HPI MOOC platform: (1) receiving a graded performance record, called Record of Achievement (see Section 2.1.1), for course completion; (2) receiving an ungraded performance record, called Confirmation of Participation, for consuming a specific proportion of the learning material; and (3) completing different thematic units that can be derived and offered as learning objectives based on selected knowledge acquisition and knowledge examination items.

Objective Selection Selecting an objective is always optional to not restrict the open nature of learning in MOOCs and upset users who want to stick to the traditional learning path they are used to. Therefore, personal learning objectives were introduced as an extension and optional feature. An objective selection modal³⁰ is shown when the user accesses the learning content for the first time (see Figure 2.15). The currently selected objective is displayed on the course progress page and can be changed at any time. Previous evaluations showed that it makes sense to explicitly encourage learners to select an objective instead of relying on its discovery by learners. Additionally, offering objective selection at different places was recommended [33, 35].

Next to selecting and working on a learning objective, learners need to constantly evaluate their progress and achievement. Therefore, experiments have been conducted adapting the platform's course progress page [34] to provide progress information towards the achievement of the selected learning objective (Figure 2.16).

2.1.6.1 Time Effort (for Learning Items)

One of the most crucial factors for learning in MOOCs is the ability to manage one's time. To effectively support both time management and strategic planning, learners need to be provided with the approximate time required to complete specific learning activities. In MOOCs, these activities primarily include the consumption of the learning material in terms of watching videos, reading textual material, taking quizzes, or working on further assignments, e.g., programming tasks. Besides, there are additional activities which contribute to the required learning time. For example, the time spent in discussion forums and on repetition of learning content must be considered. Specifically, the latter activities are used very differently by learners. However, the time required for initially consuming learning materials can serve as a reference for most learners. Therefore, the approximate time effort for completing learning material is determined and provided. This allows students to compare the estimated time required for specific course content and plan their learning accordingly, e.g., schedule learning sessions. Also, students can use it to analyze their learning effectiveness in terms of their time spent on material compared to the estimated effort. The estimates can thus provide an orientation for improvement possibilities and might help learners to eliminate time wasters.

To enable students to make use of the estimated time effort, e.g., to plan their learning, it is added to the course navigation as this is the most prominent place for students to identify the learning content to work on. Moreover, the effort is displayed at the top of each learning item page, so it can be utilized when working on the material. Figure 2.17 shows the presentation of the time effort information.

In general, the time effort for a learning item could be determined manually, but this is a tedious task for courses lasting several weeks, with typically ten up to thirty items per week. Automated tool support is required to limit the effort.

³⁰A graphical overlay window, also called dialog or pop-up.

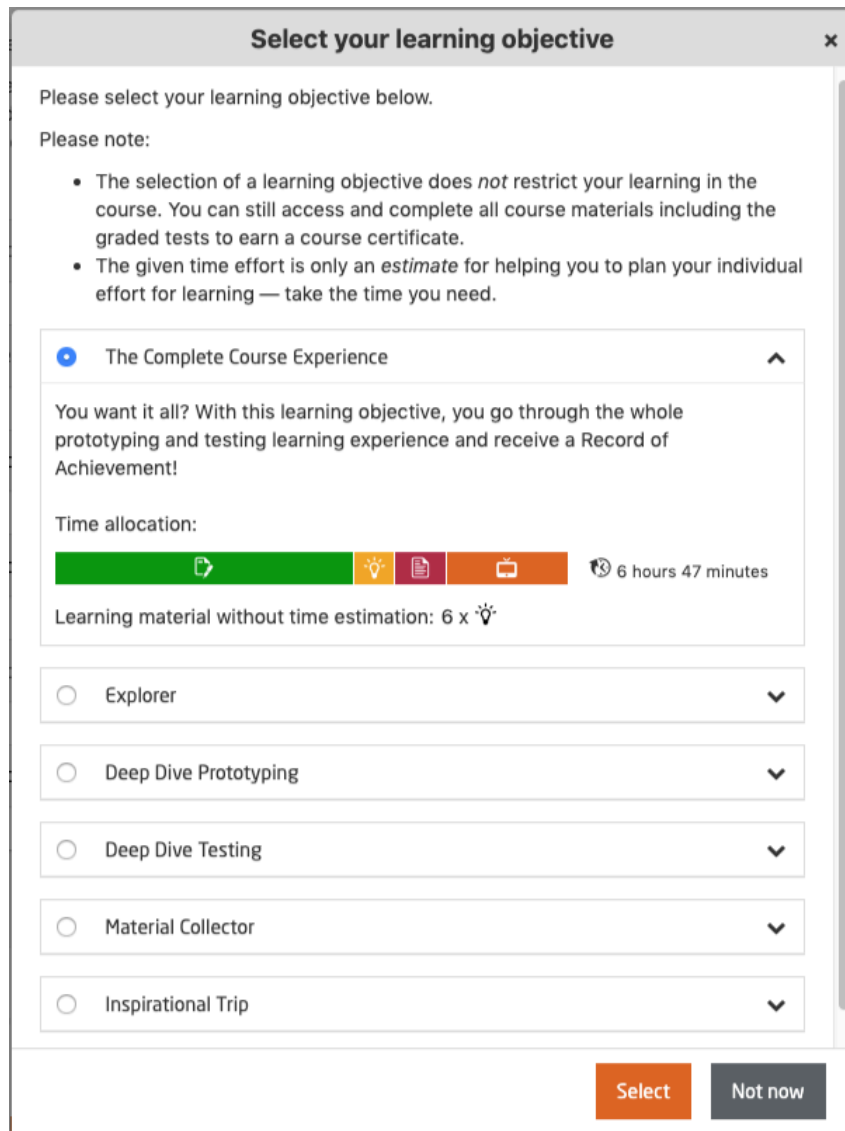


Figure 2.15: Modal for the selection of a learning objective

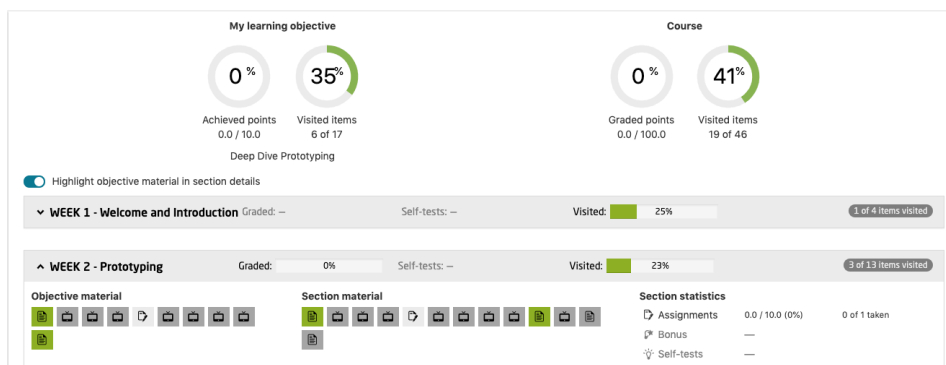
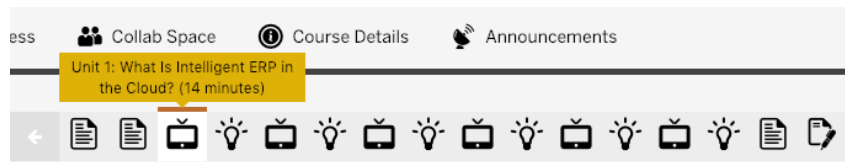


Figure 2.16: Dashboard visualizing the progress for a selected learning objective and the entire course



Unit 1: What Is Intelligent ERP in the Cloud? Time effort approx. 14 minutes

Figure 2.17: Screenshot of the displayed time effort information in the course navigation and on the item page for a sample learning item

2.1.6.2 Time Effort Calculation

The HPI MOOC platform offers videos, reading material, quizzes, peer assessments, and programming tasks. A basic approach to determine the estimated time effort is to consider the actual content and structural characteristics of the items. However, the time needed for consuming material can vary considerably between learners. It depends on various factors, such as the students' educational background, their familiarity with the course subject, the course design, or the online learning environment in general. Also, the students' intentions for a course mainly influence their interaction with course material. Many diverse strategies are applied to master the learning content. For example, students might watch a video at an increased playback speed as they are familiar with the subject, while others often pause a video to reflect on certain aspects, jump back if they did not understand a part, or research additional material. Similarly, some learners may only skim over the content, whereas others work diligently on quizzes to test their acquired knowledge. Therefore, the calculation solely based on the learning content can only be a coarse approximation.

A more accurate calculation can be achieved with the incorporation of historical learner data. For instance, the average duration for quiz submissions can be used to identify general tendencies for the time effort of quizzes. One of the implemented metrics is the average time that learners spend on a specific item. In addition to the incorporation of data of all learners, the estimation could be further personalized by considering a student's learning styles and characteristics. A learner's preferences and history, (user-specific) learning analytics data, or other techniques, such as machine learning, could be utilized for this purpose. The downside of these approaches is that prior course or user data, i.e., a previous run of the course or a specific amount of collected learner data, is required. Until then, basic approaches that provide initial estimates must be used. As a rigorous calculation bears the risk to put students under pressure and upset them, the estimation therefore is rather generous.

2.1.7 Mobile Applications

Complementary to the web platform, native mobile applications are developed for iOS and Android, which users can install on their smartphones and tablets alike.

The mobile applications of the HPI MOOC platforms focus on providing the essential features of a platform – the interactions with course content [32]. This focus was selected because implementation efforts increase when providing multiple clients for accessing a platform. Instead of optimizing each platform feature for the mobile applications, many non-essential features are available through embedded web views within the mobile application. By doing so, fewer implementation efforts are introduced while still providing all core features of the HPI MOOC platform in a well-designed way for mobile devices.

Our mobile applications aim to improve usability and streamline interaction patterns for learners on mobile devices. At the same time, they are more conservative in their network consumption. They allow deeper integration with the mobile device's operating system, e.g., via the notification system or by accessing the local storage. As a result, our mobile applications offer network-independent learning activities by downloading and storing the learning material on the mobile device [2]. In this way, the primary web platform is not just replicated by mobile applications but enhanced for mobile learning.

2.1.8 Gameful Learning

Gameful learning is an umbrella term for game-based learning and gamification and describes the use of games, game elements, or game mechanics in learning environments [14]. Thus, gameful learning can be applied in manifold ways to platforms, courses, and learning units. The most commonly used method to apply gameful learning is gamification on a platform level. Typical platform gamification elements are the display of a distinct experience status per user, leaderboards, or certain badges. The use of gamification elements on the platform has been researched quite thoroughly. In 2015, a master project of several students investigated gamification options and recommended a set of elements to be implemented. This work was further elaborated in several master's theses and follow-up work with an external student and a professional game designer. Although gamification might have its benefits, it also comes with a certain amount of drawbacks. Basically, gamification is rooted in B.F. Skinner's highly controversial behaviorist learning theory, which already was rebutted by Noam Chomsky decades ago [57]. Hence, structural platform gamification, as provided with points, badges, and leaderboards, can reduce intrinsic learner motivation and should be used sparsely [15].

Therefore, the actual use of gamification on the platform has been significantly shortened compared to the original plans. The most important leftovers are *experience points* (XP), used to determine a learner's "credibility" and displayed to other learners in the form of status symbols based on judo belts next to a learner's avatar in the discussion forum. The most important source to earn XP is the discussion forum. All learners can upvote questions and answers; the learner who has asked a question can accept an answer as the one that solved the problem. Our goal is to reward quality, not quantity. Therefore, only very few XP can be earned for posting questions or answers, while learners can earn much more XP by receiving upvotes or getting an answer accepted as the solution:



Figure 2.18: Detective Duke and his robot working on Java code

- User answers a forum question: 1 XP
- User receives an upvote on a question: 5 XP
- User receives an upvote on an answer: 10 XP
- User's answer is accepted by question author: 30 XP

Instead of further focussing on structural gamification, we concentrated our research on topics such as content gamification, serious games, and story-based learning [18]. For this, we applied a detective story to our Java programming courses, giving the courses a simple structure to follow. Such an approach is particularly useful for course topics that are difficult to master. When the learning goal is hardly achievable through videos and quizzes, learners need more hands-on exercises to deepen their knowledge by practically applying what they have learned. Even if the course content is easy to understand, applying the newly acquired knowledge directly in one global course context is more accessible than focusing on a new problem for each assignment. Research results also indicate that implemented story approaches improve the perceived quality of the learning materials [17]. Hence, using gameful learning for course materials is advisable.

By now, the story about Detective Duke (see Figure 2.18) spans three different detective cases in three of our Java courses. The story was implemented by adding story videos, texts, quizzes, and exercises related to the story to the course (see Figure 2.19). In addition, one of the courses offered a Serious Game [14] in the same universe.

2.1.8.1 Open Badges

Open badges are issued as an equivalent to the RoA to simplify sharing the successful completion of courses in portfolios, such as LinkedIn or Degreed or on other social media channels.

As already mentioned, the required percentage of points or visited items on openHPI is 50%. This value is configurable per platform. The default value of the platform can be overwritten per course. We strongly encourage, however, to stick to the defined platform-wide values.

As mentioned above, Records of Achievement and Certificates, feature a verification URL and a QR code. As the issued performance records are just downloadable PDFs, learners with a sufficient amount of criminal energy, might try to fiddle with the results or the participant's name. The verification URL provides the original information, which then can easily be compared with the presented performance record. The QR code simplifies this verification as it simply can be scanned with a mobile device to open the verification URL.

2.1.9 Help Desk and Chatbot

The help desk that can be seen in Figure 2.21 is used for support requests from our MOOC platforms. When creating a request to the help desk, the user must select whether it is a general or course-specific question. If a learner directly opens the help desk form when being within a course, this course is pre-selected as a category. In addition, a title and a description need to be added. For every support request, a ticket is created in an external ticket system, which then needs to be answered manually. In the last ten years, about 156,000 tickets have been created over the help desk on the three largest platform instances (openHPI, openSAP, and OpenWHO).

In 2020, we analyzed the received tickets to identify the most common languages used and requested topics. For openHPI, about 11,5k support requests³¹ had been analyzed. Nearly 80% of the analyzed tickets were written in the German language. The keyword-based search on frequently used subjects (visualized in Figure 2.22) showed that most tickets were created containing questions about the topic *Exercises*, followed by questions concerning either the topic *Email* or *Certificate* (about 7% each).

Since the COVID-19 pandemic started, there has been a rapid increase in the usage numbers of OpenWHO and, consequently, in the number of support requests. Accordingly, OpenWHO accounts for the most support requests of all platforms and the largest share of our analyzed tickets. Approximately 25,5k tickets³² had been analyzed, of which 83% were created in English. As with openHPI, a keyword-based search was performed on the English tickets. With 6k appearances, the keyword *Certificate* is the most frequently asked topic. It is followed by the topics *Exam*, *Email* and *Login*. The distribution of the most common subjects is visualized in Figure 2.23.

Many of the most common questions are already answered in the platforms' FAQ sections. Nevertheless, barely half of the tickets dealt with these – comparably simple – questions and requests, which resulted in time and cost-inefficient, manual

³¹Data period: 11.2014–01.2020.

³²Data period: 02.2017–05.2020.

2 Functionality and Format

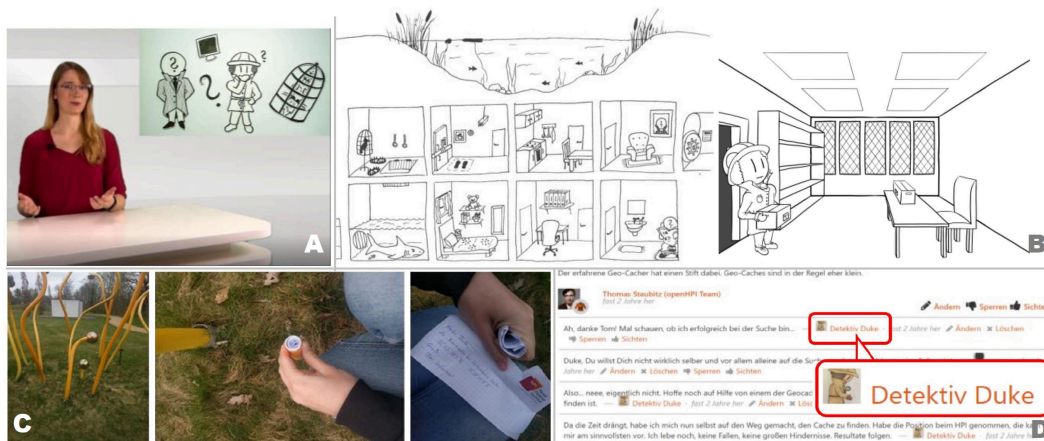


Figure 2.19: Story elements used to advance the story for Duke and his detective cases

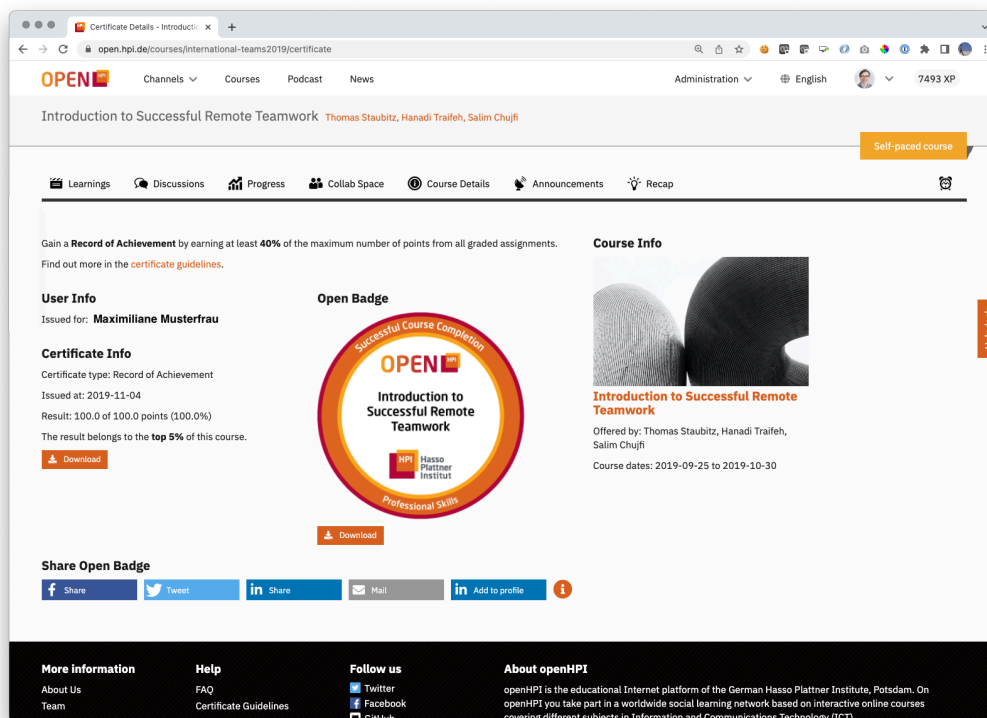


Figure 2.20: Performance record verification and OpenBadge

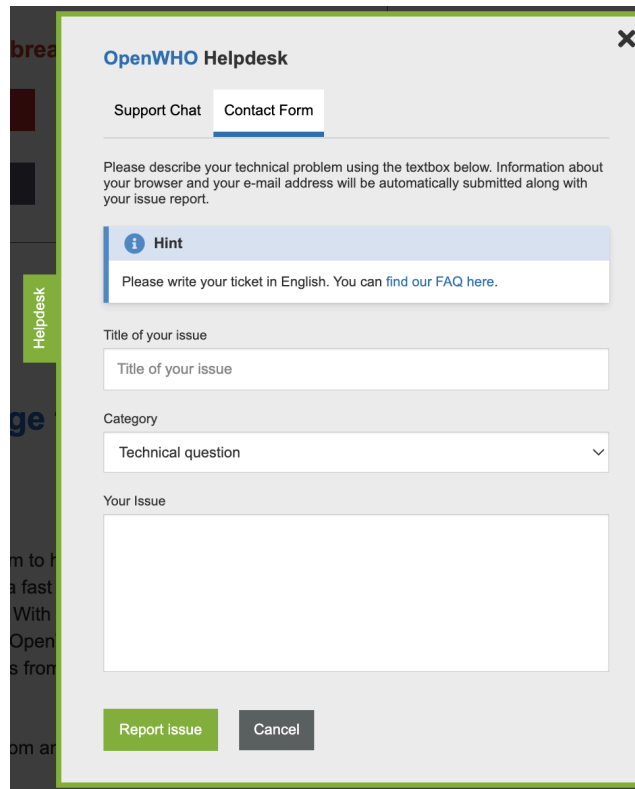


Figure 2.21: openHPI help desk widget

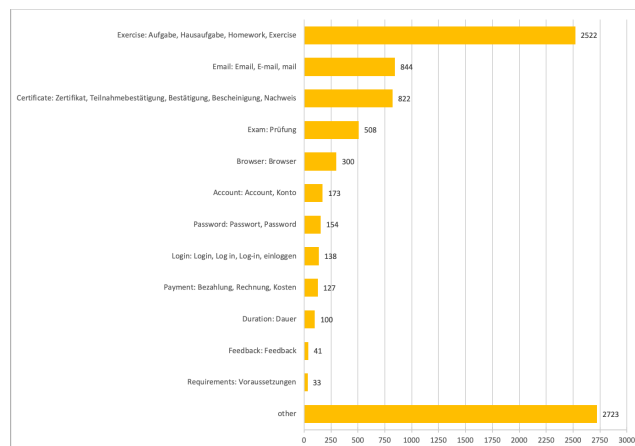


Figure 2.22: openHPI: Most common subjects of support requests by keyword-based search. Data period: 11.2014–01.2020.

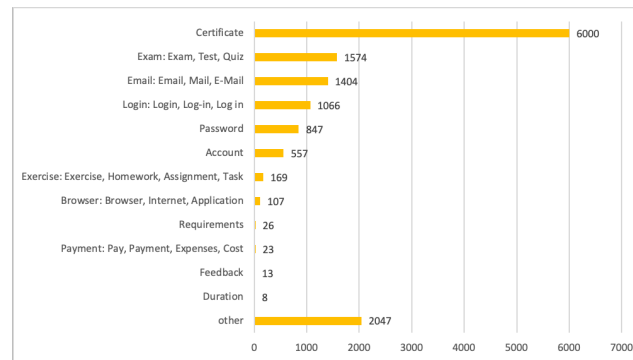


Figure 2.23: OpenWHO: Most common subjects of support requests by keyword-based search. Data period: 02.2017–05.2020.

feedback for the help desk team. The problem increases when users are unsatisfied with the first answer and ask again. Additionally, with the repeatedly large number of tickets created, the time-to-answer metric for each ticket increases, which results in frustrated learners. To counteract this development, the development of conversational agents was initiated. As a first step, the chatbot is designed to help users with simple requests. As the chatbot domain is regularly extended with new intents, even newly emerging FAQs can quickly be answered by the chatbot. Whenever the chatbot cannot help, support from the help desk team can still be requested by creating a “traditional” ticket.

The current version represents a dialogue-based system in the traditional conversational way. The user interface of OpenWHO is modified as shown in Figure 2.24. The user can select between the original help desk view or the chatbot view. The default view shows the chatbot encouraging users to explore and test the dialogue-based system.

Users can evaluate the helpfulness of the chatbot’s answers and give feedback. For this, a simple 5-star rating mechanism is shown at the end of each conversation. As part of the chatbot’s self-learning ability, the feedback is evaluated, and, on that basis, the domain is updated. The chatbot suggests several topics if a question is not understood. The user can then select one of the topics or try to rephrase the question. The initial question of the user is then manually added to the selected topic’s set of variations to further enhance the chatbot’s knowledge.

Currently, the chatbot is only activated for the general public on the OpenWHO platform. On KI-Campus and openHPI, it is being tested and only available for a selected group of users.

2.2 Admin Features

In this section, we restrict ourselves to describing the announcement and reporting possibilities, the platform and course dashboards, research options, our course feed,

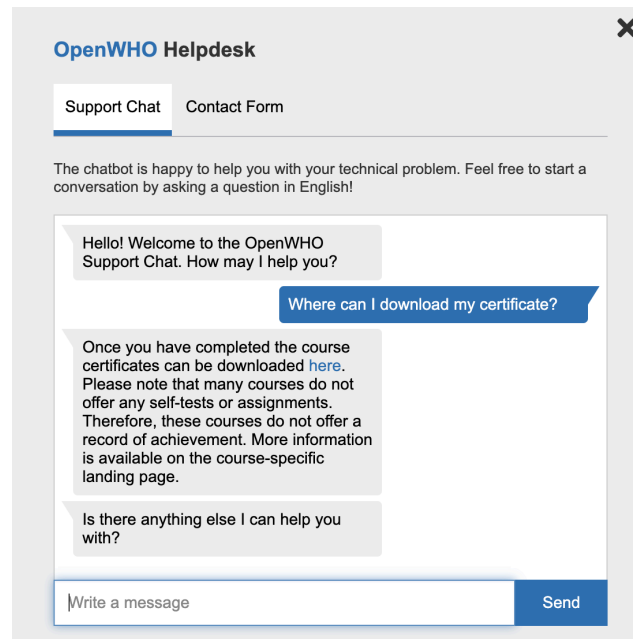


Figure 2.24: Sample conversation with the chatbot on OpenWHO

and the TransPipe tool to support the process of creating automated or manual subtitles. As indicated in Section 2.1 describing the learner features, the platform supports a large variety of features that require configuring the course settings and metadata, creating course structures and adding contents, managing learners, submissions, performance records, and much more. We'll leave the detailed description of these features to our user manual³³ as this goes beyond the scope of this report.

2.2.1 News and Announcements

The platform supports sending course-specific announcements and global, course-independent news. Both types are sent to the receivers via e-mail and displayed on the platform – either on the platform's news section or within a course.

2.2.1.1 Course Announcements

Within a course, learners should be provided with information about the course at various points in time. First, an initial message is sent immediately after course enrollment. Depending on the enrollment date, the mail should include relevant information about the start of the course and the general course agenda. For example, information about the course start is no longer applicable if the course has switched into self-paced mode; instead, the mail sent should highlight the self-paced mode of that course.

³³<https://teachingteamguidelines.readthedocs.io/>; last access: 02.09.2022.

During the course, the teaching team should inform learners once a new course week has started or new materials have been added. Highlighting approaching deadlines with a separate announcement has also proven to help enhance the learning experience. We like to close our courses with a final thank-you message that includes course statistics and asks for learner feedback, so the teaching team can improve their methods and materials.

For conducted research, it might also be helpful to point out specific learning items or highlight relevant surveys in those course announcements. For the teaching team, course announcements are an essential tool for interacting with learners and improving learner satisfaction.

2.2.1.2 Global News

The news function exists to send a message to all registered users of the platform. The newsletter is usually used to provide important information about new and upcoming courses, platform features, and temporary special offers. In contrast to the course-internal announcements, which are written exclusively in the primary course language, the global news generally supports all offered platform languages. Accordingly, the form for creating news contains not only one title and text in the platforms' default language (usually English) but also title and text for all possible translations. When sending out the news, the platform automatically identifies the user-selected platform language and sends the news in the respective translation. The mail is sent in the default platform language if the translation is unavailable.

2.2.1.3 Targeted Announcements

In addition to sending news to all registered platform users, platform administrators can send targeted announcements to a predefined user group. This feature allows defining a user group that meets certain requirements to be contacted via mail. For example, only those users who have previously achieved a performance record in at least one course out of a course selection on a particular topic can be notified of a new course or some other important information. The feature is useful for both marketing and research purposes.

So far, a developer must create the group for the targeted announcement manually in most cases. In the future, a platform administrator self-service for configuring the group of a targeted announcement from a standard repertoire of metrics will be available.

2.2.2 Dashboards and Statistics

The platform provides separate dashboards for platform-wide data and more detailed statistics for course-specific data.

2.2.2.1 Platform Dashboard and Statistics

The platform dashboard gives a quick overview of the most relevant platform-wide KPIs. These KPIs include total course enrollment numbers, average course enrollments per learner, total registered learners on the platform, and active learners. The

dashboard also includes statistics regarding help desk requests and the number of issued performance records and badges. Additionally, more insights on the learner metrics can be presented, like age distribution and other profile data, user locations, an activity heatmap, and the clients used to access the platform (desktop browser, mobile browser, native app). Most of these KPIs are listed not only as total numbers but also filtered by different time ranges, such as the last 24 hours or the last seven days (see Figure 2.25 and Figure 2.26).

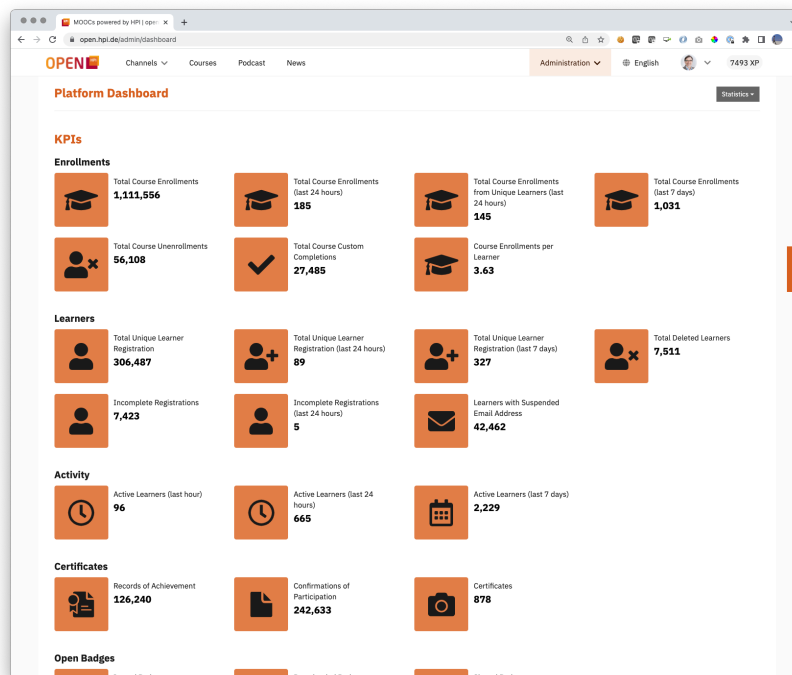


Figure 2.25: Platform KPIs with different numeric metrics

The dashboard offers an additional menu that provides platform administrators with statistics on the status of global announcements, quality alerts, social shares, and referrers.

2.2.2.2 Course Dashboard and Statistics

While the general platform dashboard is only available for platform administrators, course instructors and teaching teams can get a quick overview of the KPIs within their course on the course dashboard. The dashboard includes general enrollment, activity and performance record information along with more detailed usage information (see Figure 2.27).

Along with the total course enrollments, this dashboard shows enrollment snapshots at specific points, such as the last 24 hours, course start, middle, and end. Next to the enrollments, it also shows the number of active users at the current time,

2 Functionality and Format

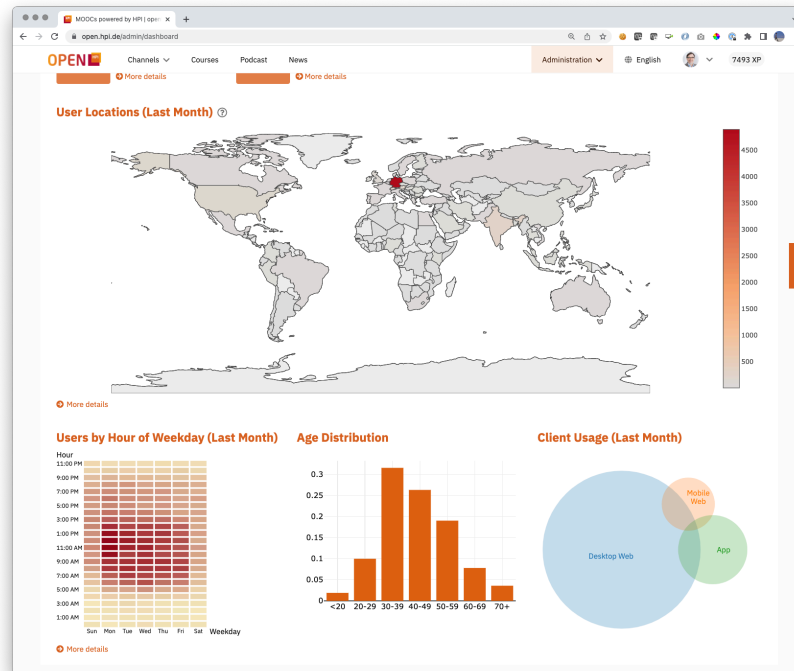


Figure 2.26: Platform dashboard including user location, activity heatmap, age distribution, and client usage

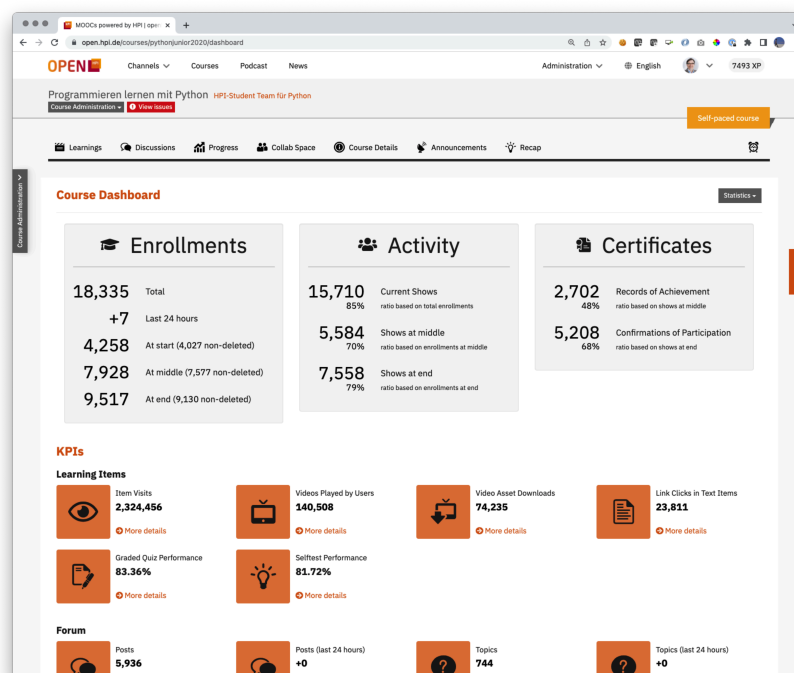


Figure 2.27: Course enrollments and completion rates

course middle, and course end. In this context, active users are defined as users that have at least visited one course item. We often call them “Shows” in contrast to so-called “No-Shows” – users who have enrolled in a course but never started attending it. The most interesting number here is the “Shows at middle” which serves as the basis to calculate the course completion rate. The rationale behind using this number instead of the total number of enrollments to calculate course completion is that, most likely, only users who joined the course before course middle will be able to earn a Record of Achievement. In most cases, late-comers will already have missed half of the performance record-relevant assignments and thus are likely unable to achieve the required minimum number of points for the performance record.

The course “middle” is calculated automatically by the course start and end dates; however, there is also an option to overwrite this date manually. Overwriting the course middle date could be required if particular circumstances in a course differ from our recommended standards, e.g., if a course does not contain any assignments except for the final exam.

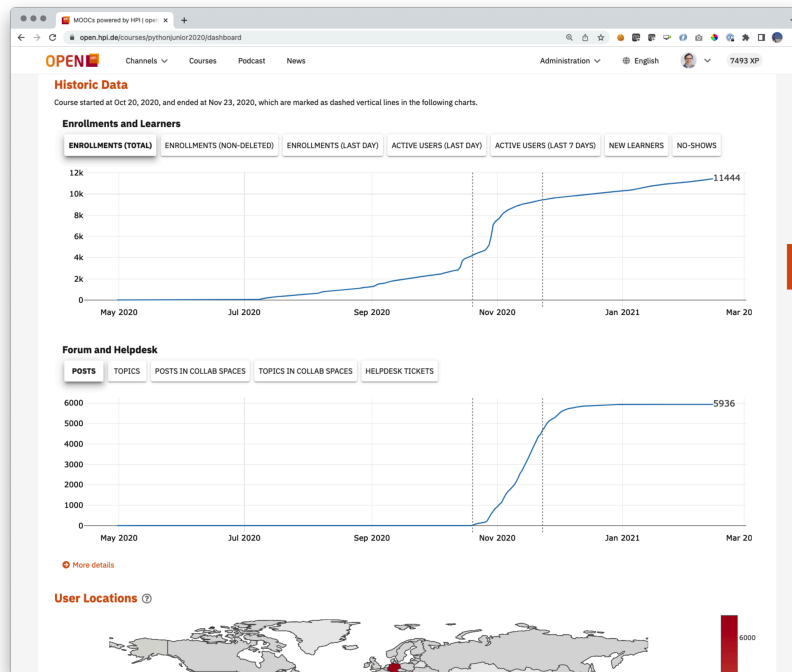


Figure 2.28: Course enrollments, forum interaction, and help desk tickets over time

The course dashboard also includes KPIs for detailed user interaction events, such as the number of item visits, video plays, or video downloads. Course teachers can get an overview of the graded quiz and the self-test performance. Next, the course dashboard lists active and passive forum activity metrics, listed for the

whole course period and during the last 24 hours. Furthermore, two interactive line graphs show the development of course enrollments over time (see Figure 2.28), which is a good indicator to see which marketing campaigns were successful, and the development of forum interaction over time.

Like in the platform dashboard, there is information about the geographic location of the learners, the age distribution, and the client usage with data only for this course. Different dashboard subpages can provide more detailed information on item visits, videos, downloads, text items, quizzes, section conversions, forum interaction, geographic background, announcements, and social shares.

2.2.3 Reporting

While the dashboards provide a quick overview of the fundamental platform and course data, these presets are often insufficient to answer more specific research questions or reporting details. For this purpose, the platform provides several reporting options for course, user, quiz submission, forum, and other course data. For privacy reasons, all reports are generally pseudonymized. Furthermore, downloading reports requires additional permissions beyond the powers of plain platform administrators, ensuring that only users who are adequately instructed in handling user data have access.

All reports contain more or less raw data and are designed to be further processed by data analysis tools. A wide variety of such tools exists, and every researcher is used to working with a different one. Hence, we decided to keep the export format simple and offer compatibility with as many of these tools as possible. Therefore, reports are delivered in the form of .csv files.

Course Report Course reports contain all information on the learners' interaction within a course. Interaction data includes forum activity, number and percentage of visited items, results in self-tests and exams, and more. Each of the report's rows represents one learner. The platform provides options to configure the included information, e.g., profile data can be added.

User Report User reports contain information about each registered user on the platform, including profile information, enrollments, and more.

Quiz Submissions Report A quiz submission report contains all information about the submissions in a quiz. Each row represents a learner's submission; the columns represent questions and answers. Selected answers are marked with a 1, unselected answers are marked with a zero, and an empty string is included if the learner gave no answer.

Course Pinboard Report The pinboard reports contain the forum discussion posts and their relation to each other (thread, answer, comment).

Statistic Reports Further reports such as the course events, the course contents, or course summaries are available.

2.2.4 A/B Testing

Randomized controlled trials (RCTs) are an essential tool of scientific research to examine the effects of treatment on test subjects. Along with pre- and post-tests, RCTs are a powerful method for quantitative analysis in education science. Usually, an RCT shows different treatments to different user groups. In computer science and general web development, this process is often called A/B testing. Our platform has provided A/B testing on the feature level [31] since 2015, i.e., replacing web page elements. In 2021, an experimental feature to exchange the learning materials on a course content level has been implemented. This experimental feature that currently requires manual setup in the production database via console has successfully been used for multiple research questions in the past year. Hence, we plan to improve this feature and implement a self-service for course administrators. The feature is not only planned to assign users to test groups automatically (i.e., silently) but also to provide an option letting learners select between available treatments. This way, we introduce the highest flexibility for using this feature for research and different learning paths within a course.

2.2.5 Courses API

The platform provides an API containing all courses' public metadata so external library tools, course aggregators, or other consumers can embed the course catalog. The API documentation is publicly available.³⁴ The format of the API builds on the schema.org format. By now, all German-speaking MOOC providers are using this catalog format. Several MOOC aggregators and library tools in Germany have implemented interfaces that can read this format. Discussions with further European MOOC providers about the format are ongoing and are starting to gain traction.

2.2.6 TransPipe

Videos allow not only transcriptions in the course language but also translated subtitles. We started using existing tools to extract transcriptions (speech to text) and automatically translate these transcripts to add subtitles in multiple languages to our course videos. Soon after, we realized this procedure was insufficient for our purposes as it involved too many manual steps:

1. Retrieve the video ID from Vimeo.
2. Sometimes extend the video's visibility.

³⁴<https://openmpi.stoplight.io/docs/bridges/48d65d8141c0f-moo-chub-api>; last access: 02.09.2022.

2 Functionality and Format

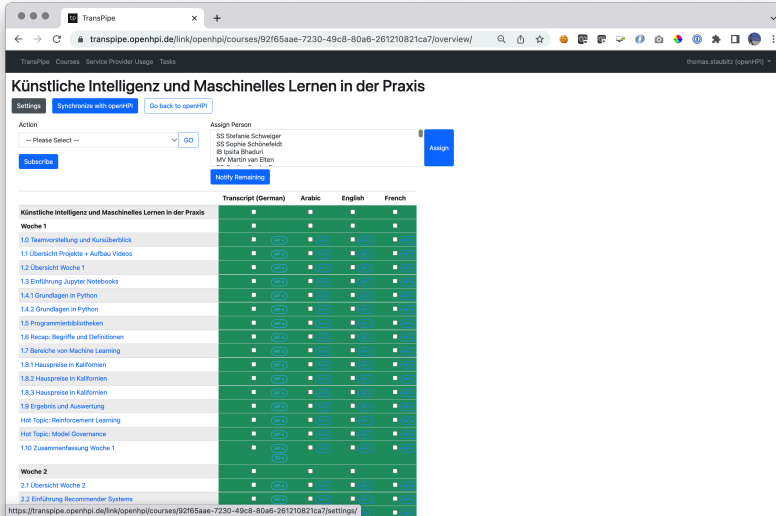
3. Enter the video ID in the transcription UI.
4. Upload the finished transcripts via the command line to the auto-translator.
5. Finally, download the translations via the command line again.

The processing effort even increased when processing subtitles for partners, as subtitles were often sent via email with the wrong character encoding or format.

Furthermore, the quality of the auto-generated transcripts and translations is often very low. Therefore, manual adjustments must be integrated at several workflow stages. Thus, we decided to implement our own transcription and translation process, giving us more control over the whole process.

During a master project in 2020, a group of three students delivered the initial requirement analysis and a prototypical implementation for TransPipe (Transcription and Translation Pipeline). Since then, the tool has constantly been developed further.

The TransPipe tool is directly integrated with the HPI MOOC platform. With one click, all videos from a course are imported from the platform to TransPipe. TransPipe then allows selecting the tool for transcriptions and translations in different languages (see Figure 2.29). A few clicks trigger a semi-automated process that creates transcripts and translations in the selected languages (see Figure 2.30). Employees can be assigned to quality control and fix the auto-generated results. Finally, the completed subtitles can be exported to the course on the HPI MOOC platform with a single click.



The screenshot shows the TransPipe web interface. At the top, there's a navigation bar with 'TransPipe', 'Courses', 'Service Provider Usage', and 'Tasks'. The main heading is 'Künstliche Intelligenz und Maschinelles Lernen in der Praxis'. Below this, there are buttons for 'Settings', 'Synchronize with openAI', and 'Go back to openAI'. An 'Action' section includes a dropdown menu for 'Please Select...', a 'GO' button, and an 'Assign' button. Below this, there's a 'Notify Remarking' button. The main content is a table with columns for 'Transcript (German)', 'Arabic', 'English', and 'French'. The table lists various video topics and their status in each language. The status is indicated by green checkmarks or red 'X' marks.

	Transcript (German)	Arabic	English	French
Künstliche Intelligenz und Maschinelles Lernen in der Praxis	✓	✓	✓	✓
Woche 1	✓	✓	✓	✓
1.0 Teamvorstellung und Kursüberblick	✓	✓	✓	✓
1.1 Übersicht Projekte + Aufbau Videos	✓	✓	✓	✓
1.2 Übersicht Woche 1	✓	✓	✓	✓
1.3 Einführung Jupyter Notebooks	✓	✓	✓	✓
1.4.1 Grundlagen in Python	✓	✓	✓	✓
1.4.2 Grundlagen in Python	✓	✓	✓	✓
1.5 Programmierbibliotheken	✓	✓	✓	✓
1.6 Recap: Begriffe und Definitionen	✓	✓	✓	✓
1.7 Bereiche von Machine Learning	✓	✓	✓	✓
1.8.1 Hausprize in Kalifornien	✓	✓	✓	✓
1.8.2 Hausprize in Kalifornien	✓	✓	✓	✓
1.8.3 Hausprize in Kalifornien	✓	✓	✓	✓
1.9 Ergebnisse und Auswertung	✓	✓	✓	✓
Hot Topic: Reinforcement Learning	✓	✓	✓	✓
Hot Topic: Model Governance	✓	✓	✓	✓
1.10 Zusammenfassung Woche 1	✓	✓	✓	✓
Woche 2	✓	✓	✓	✓
2.1 Übersicht Woche 2	✓	✓	✓	✓
2.2 Einführung Recommender Systems	✓	✓	✓	✓

Figure 2.29: List of imported, transcribed, and translated videos in TransPipe

The TransPipe software allows course instructors to make their MOOCs more accessible to the interested public and remove access barriers. Subtitles in the video

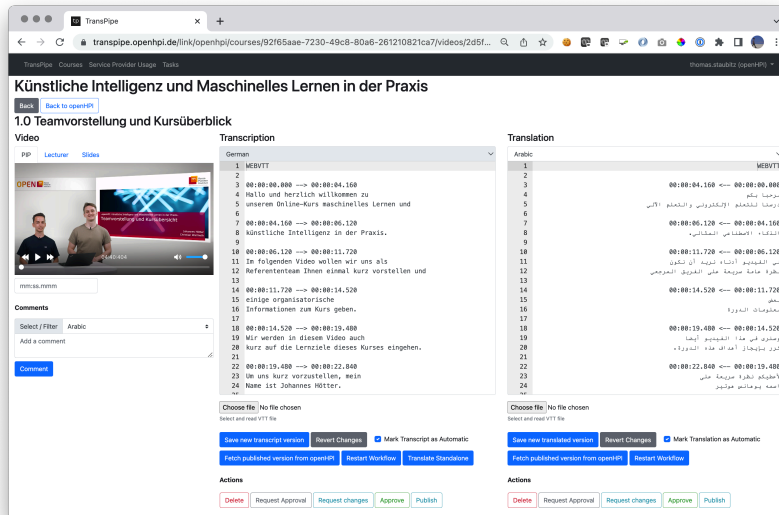


Figure 2.30: TransPipe transcript and translation editor

player can not only be used by deaf and hard-of-hearing learners but also support the learning process for all learners. It also eases navigation within the courses. Further, the machine translations allow learners to participate in a course, even when not speaking the main course language fluently [1].

2.3 Common Course Formats

Courses on openHPI are generally offered as event-based courses with a dedicated start and end date and several deadlines. However, the courses are still operated asynchronously. The learners do not have to attend particular sessions but can watch the videos and work on the quizzes whenever they want to within the given course boundaries.

All courses offer a course forum that allows learners to discuss with each other and the teaching team. The teaching team is present in the forum and directs discussion in the right direction if necessary. The general policy is not to intervene too early to allow the learners to find the answers to their questions within their peers. However, if discussions take a wrong turn or questions are not answered within a reasonable time, the team itself posts actively. Being present as a teaching team in the course forum supports the overall course atmosphere and creates higher interaction rates. For the teaching team, reacting to reports of technical problems or fundamental errors in the course material is especially important to develop trust. As a rule of thumb, we can say that smaller courses require more teaching team writing effort, while larger courses require more reading effort.

After the course’s official end, it is automatically switched to archive mode. Once a course enters archive mode, the discussion forums are changed to read-only, and the graded exams and exercises are no longer available or do not grant any course points for performance records.

As not all potential participants have the possibility to join a course during the offered time frame, selected self-paced courses can be reactivated with a voucher (see Section 2.1.2). When using a reactivation voucher, the course end date and all submission deadlines are individually reset for the redeeming user to eight weeks from the voucher redemption date.

Knowledge Essentials Knowledge essential courses are openHPI’s equivalent to lectures at a regular University. These courses’ main learning item types are videos and multiple choice quizzes. Learning content is provided by short videos of about ten minutes in length. The videos are designed and recorded in the openHPI studio. Video recordings of lectures in the lecture hall turned out to be inappropriate for use in the courses, as they are too long, require too much post-editing effort, and in the end, still are not as good as custom-designed videos.

The majority of the quizzes in the Knowledge Essential courses are self-tests. As a rule of thumb, we add one short quiz of two or three questions after each video. These self-tests do not provide points for the course’s performance records; their only purpose is to allow the learner to self-evaluate their understanding of the previous video. Furthermore, there are exams or bonus exercises, which provide the points for the performance records. Knowledge Essentials are either offered as six-week courses with one exam per week and one final exam at the end of the course (see Figure 2.31); or they are offered as a combination of three two-week courses plus a standalone final exam (see Figure 2.32). Technically, this exam is also a course that contains precisely one learning item – the exam. Only those participants who have completed the three two-week courses with at least a Confirmation of Participation are admitted to submitting the exam.

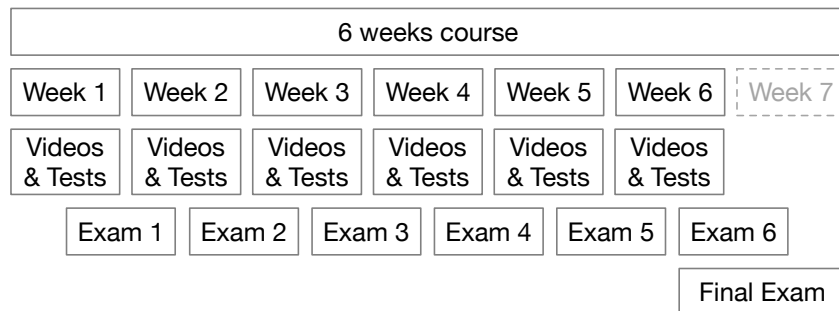


Figure 2.31: Knowledge Essential, one six-weeks course

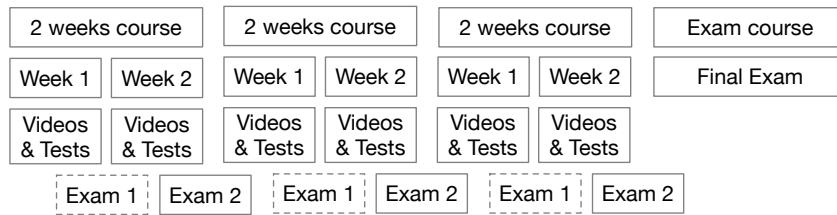


Figure 2.32: Knowledge Essential, three two-weeks courses

Hands-on Courses All Hands-on courses on openHPI are programming courses. Regarding production mode and length, the videos in these courses are similar to those in the Knowledge Essential courses. Content-wise, they usually explain a certain programming construct, such as variables or loops. Another commonality with the Knowledge Essentials courses is that a self-test follows most videos. The difference is that next to the videos and the self-tests, there is a certain amount of programming exercises per video that allow the participants to apply the newly learned concepts. The best practice is to add three programming exercises per video. The first one requires the participants to follow the steps shown in the video lecture. The second exercise requires some adaptation of the concept to a different problem related to the example in the video. In the third exercise, the participants have to apply the learned concept in a different context. In most cases, the programming exercises are relatively small, and the complexity is manageable.

Most programming exercises in the Hands-on courses are offered through CodeOcean (see Section 2.1.3.4). This tool allows learners to work on the exercises in the browser and course instructors to add unit tests for each task and thus make the exercises auto-gradable. Therefore, the points earned in these exercises often count towards the performance records. Furthermore, most hands-on courses offer weekly multiple-choice assignments and one final exam. The graded exams often resemble debugging exercises, where a certain erroneous code snippet is provided, and the learners must select all correct answers to fix the problem. The general length of the Hands-on courses is four weeks (see Figure 2.33).

Workshops and Capstone Courses Both Workshop and Capstone courses have a length of two weeks of content delivery, and they often extend a Knowledge Essential or a Hands-on course (see Figure 2.34). Workshops usually focus on a specific topic that can be covered in the rather short course span of two weeks, e.g., *Data Structures and Algorithms* or *Test-driven Development*. The main difference between a Workshop and a Capstone course is the type of practical tasks. While Workshops include multiple exercise-like activities, the Capstone courses often consist of a larger project that can be solved individually or as a team. Capstone courses often employ Peer Assessment as a scalable grading tool for open-ended tasks.

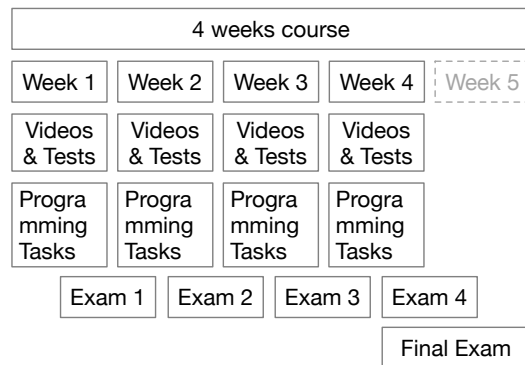


Figure 2.33: Hands-on course format

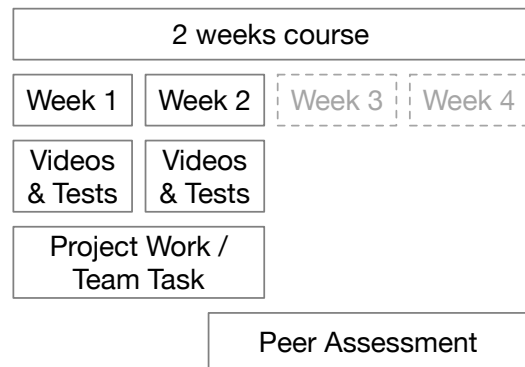


Figure 2.34: Capstone course format

Mixed Forms In real life, the course types mentioned above should be seen as flexible templates rather than as a strict corset. For example, Knowledge Essentials or Hands-on courses might contain peer-assessed tasks. Depending on the topic, a Knowledge Essential might include Hands-on tasks, while a Capstone course might offer self-tests or instructional videos. In summary, the difference is mostly a question of focus.

Other The above-listed formats are not a restriction from the platform technology but rather conventions that we agreed to follow. Other platform partners are often offering quite different formats. KI-Campus, e.g., primarily uses a self-paced course format rather than our event-based format, and OpenWHO mainly uses the platform as a material distribution tool. On the other end of the spectrum, KommunalCampus, as well as the courses of the HPI Academy on openHPI, are synchronous blended courses or webinars.

2.4 openHPI in On-Site Teaching

From the very first day, the openHPI platform served four purposes:

1. The obvious one: we want to share our knowledge with the world and contribute to the digital enlightenment of society in general.
2. The more or less hidden agenda: we need a vehicle to answer our research questions.
3. The welcome side effect: The HPI is always looking to attract the best students. Many applying students have had their first encounter with the HPI via openHPI.
4. The teaching aspect: Student education at the HPI is often achieved by hands-on seminars. The platform and several side projects have served as a context for bachelor projects, master's projects, master's theses, and many project-based learning seminars.

In the context of openHPI in HPI on-site teaching, we have supervised 28 master's theses by now. Seven of these students stayed with the openHPI team as Ph.D. students or full-time developers. Many theses built the foundation of platform features still used today, such as the CollabSpaces, CodeOcean, gamification elements, learning analytics, the TeamBuilder, mobile learning support, and platform monitoring systems. Furthermore, we supervised four bachelor projects³⁵ with partners,

³⁵The bachelor's project is an important element of the HPI bachelor program in terms of its practical orientation. It covers the fifth and sixth semesters and involves working on concrete problems from a partner in industry or society. In teams of six to eight persons, students learn to develop software solutions and master complex IT systems using structured, goal-oriented methods, and role-playing.

such as SAP, the WHO, or openHPI itself, and three master's projects. Finally, we supervised a plethora of bachelor's and master's seminars. Next to many seminars with technical tasks, such as developing some feature or a new tool, we particularly want to highlight one seminar where students learn how to produce MOOCs themselves. The seminar was offered for the first time in 2015. Since then, HPI students have produced more than 20 courses for the general public.

2.5 Best Practices in Course Design and Course Delivery

During the last ten years, we have designed, implemented, and conducted about 120 courses on openHPI. Furthermore, we have offered several courses on our partner platforms mooc.house, openSAP, KI-Campus, eGov-Campus, and LERNEN.cloud. In this process, we have learned a few things about designing successful courses, which we would like to share in this section.

First, it is essential to clearly define the course's target group, address them in a way that appeals to them, and state the target group in the course details. When designing content, course instructors should consider how the target group will react to this and if they can fully understand the lecture with their current knowledge. Before designing the course materials, creating a couple of personas expected in the course is helpful to visualize the learners' needs, previous skills, and knowledge. These personas should include both the standard learner and the expected extremes. Giving personas names, ages, personal interests, and skills helps visualize why they enroll in the course and what they expect to take away as learning after completing the course. Checking each learning unit with those personas ensures that the content created positively adds to the learning experience of the course's primary target group.

Using a specific course setting around which most examples evolve or even adding an entertaining story has also proven to enhance the quality of the learning content. On the one hand, this helps learners by providing fewer context switches they need to adapt to when working on the subsequent learning units. On the other hand, it also helps the course instructor to follow the main course theme when searching for suitable examples. Last but not least, a story can add to the fun both for learners when taking and for teachers when creating the course. From our experience, learners realize whether the course instructors had fun designing the course content; they are more inspired by such an atmosphere and often find the courses more enjoyable. Imparting a positive attitude adds more to a positive learning experience than perfectly prepared learning materials – as long as the provided materials are of good quality.

We recommend that at least one other person reviews each course item for quality, comprehensibility, consistency, and spelling. For videos, the slides must be checked before recording as it is hard to adjust them in the video later. When finding a mistake after the recording, we recommend updating the slides and adding an erratum to the video description. While self-tests do not always have to lead to one correct answer but instead can also make learners reflect, thoughtfully checking

the exams required for a performance record is crucial. Typically, learners handle material mistakes relaxed, as they see that the teaching team corrects them after reporting. It is helpful to have one learner review the complete course materials for consistency once they have been produced before releasing the course.

One of the challenges in designing MOOCs is that it is often not predictable who else will join the course along with the planned target group. Hence, no matter who the intended target group was, everybody should be welcomed with open arms. However, when complaints emerge that a course is not as a particular participant has expected, the instructors can voice why the course is designed in one way and not another if there was a clearly defined target group during the design and implementation phase. For example, our first Python course targeted a younger audience of school children aged 12 to 17. Although a large group of participants did fit with the intended target group, the majority of the participants were way older. The oldest school kid was more than 80 years old.

As all the videos, texts, exercises, and more are produced before the course start, there is no chance to re-adjust the material to better fit the actual audience than the targeted audience. Furthermore, even if this was possible, we do not want to alienate the original target group by adjusting the course's tonality, complexity, and more. The target group information must be clearly communicated before the course start to ensure that everybody understands the intended target group. This transparent communication can define the difference between *the course is badly designed* and *the participant is in the wrong course*. Such information is particularly important if previous knowledge is required to succeed in the course; otherwise, learners will quickly get frustrated when lacking presumed skills.

On the other hand, participants complaining that the course is too easy or on a lower level of expertise than they expected often can be directed towards bringing in their knowledge, skills, and experience to help others with less experience in the course's subject. Generally, we think participants who bring in personal experience should always be seen as an enrichment to the course, not some competitor to the instructors. On the one hand, they can highly reduce the workload for the teaching team by supporting learners who struggle. They might provide help when the teaching team is temporarily unavailable or introduce other examples and reasoning the course instructors did not think about, which might be the solution to unblock an individual participant's learning process. On the other hand, they will deepen their knowledge by helping others, ensuring they also benefit from participating in the course.

Welcoming the help and feedback from learners experienced in a course topic also applies to discussions about the correctness of certain exam answers. Even the most diligent quality controls can never eliminate all ambiguities. Although instructors often tend to think that some people intentionally misinterpret certain questions and answers, such concerns should always be taken seriously and, if justified, be accepted. All feedback helps improve the course content and the instructors' didactic experience, although sometimes it might be hard to embrace at first. Course instructors are only human, and unfiltered criticism might hurt. The teaching team should always respond professionally to a complaint; however, it might be

helpful to remind the users about a productive feedback culture, especially when targeting junior course instructors. Remembering the community about appropriate communication also helps maintain a positive atmosphere in the course forum, which is crucial to motivating participants to join a forum discussion.

As already mentioned above, communication is key. This applies not only to the target group definition or the discussion and feedback culture but concerns every aspect of the course. Particularly, course deadlines are better checked and communicated too often than not often enough to avoid confusion and frustration. From our experience, the teaching team should send at least one course announcement (see Section 2.2.1.1) per week to establish a bond between the learners, the teaching team, and the course. For us, it turned out to be a good practice if a teaching team member signed the announcement on behalf of the whole team rather than by the less personal signature of the teaching team as a whole.

Next to communication via email, it is essential to monitor the discussion forum actively. However, it is counterproductive if the teaching team answers questions too soon. In general, it is a good idea to give the other participants room and time so that a discussion amongst them can develop. The teaching team should only intervene if such a discussion gets out of control, moves in an unwanted direction, or if no one responds to a question after a while. Sometimes a course is haunted by a troll. In this case, like with the positive feedback culture, we warn the trolling participants and remind them of our forum rules. If they ignore several warnings, we do not hesitate to ban a user from a course or, in the worst case, delete their account after repeated violations. Fortunately, we did not have to do this more than once or twice in all the years. Nonetheless, we argue that maintaining a positive forum culture for all participants is key to learning success, and we want to create a safe and positive learning environment for everyone.

Apart from taking a passive role in the forum by monitoring and answering questions, the teaching team can also take a more active part by triggering wanted discussions on topics the learners should investigate or deepen their understanding of a specific topic. An excellent way of doing this is to combine a forum discussion with a multiple-choice quiz. We often use the term *e-tivity*³⁶ for this combination of a quiz with a triggered discussion. The quiz question is basically a prompt to discuss a certain topic in a specified forum thread. The possible answers are *I actively posted in the discussion*, *I passively followed the discussion*, *I don't like this kind of activity*, and *I did not have the time to participate*.

Generally, a course should offer both active and passive elements to cater different interests of the learners. Depending on the instructors' intentions and the context of the course offer, it is important to consider the effects of certain exam types on the completion rates. Project-based and active learning is often preferred over more passive approaches. However, the learning context of many MOOC participants rather affords an edutainment approach with easily consumable learning snippets

³⁶The term originally has been introduced by Gilly Salmon, <https://www.gillysalmon.com/e-tivities.html>; last access: 02.09.2022.

and not too time and resource-intensive examinations. Although not in the majority, there still is a significant number of learners, who do appreciate more active, project-based approaches. Providing them with peer-assessed bonus tasks to solve alone or in teams is rewarding (see Section 2.1.3.5). However, assuming that team tasks are a way to better integrate struggling learners into the course community is wrong. In our experience, only those learners who are already performing well in a course will also thrive in team assignments, and the others will drop out very soon. It is recommended to restrict access to such tasks to the successful learners only to keep the frustration level low.

3 Technology and Architecture

This section will present technical insights and platform architecture concepts for the HPI MOOC platforms. A particular characteristic of the openHPI project is the conscious decision to operate as much as possible of the entire application in-house – from the data center over the software development for the HPI MOOC platforms to the courses offered by the HPI teaching teams on openHPI. While this approach comes with some overhead in terms of staffing compared to operations based on a public cloud, it offers many advantages, including:

- the complete control over each component of the full technology stack,
- the ability to implement fine-grained performance tweaks at all levels,
- supporting experiments and research questions even below the application level,
- the GDPR-compliant, ISO 27001-certified [63] operation of the application at the Hasso Plattner Institute.

This decision was already taken in 2012 when openHPI was built from scratch for the first course on *In-Memory Data Management* [28, p. 7]. As openHPI was the first European MOOC platform, there was no empirical data available on enrollment numbers to be expected: user registrations within a range of a few hundred up to 100,000 were to be considered. This uncertainty of specific needs prior to the launch of openHPI resulted in the requirement for scalable infrastructure. For this reason, the technology team decided for an infrastructure based on a private cloud framework – specifically *OpenNebula*³⁷ – at that time. The setup was limited to two physical cloud compute hosts with 64 CPU cores and 64 GB RAM each (upgraded to 128 GB later), as well as “high-speed” hard drives. Additional servers had been provided for the databases and external services. A dedicated appliance was deployed for load balancing and Transport Layer Security (TLS) termination [28, p. 10].

Within ten years, not only the HPI MOOC platform has been entirely rebuilt and is constantly extended, refactored, and rebrushed – but also the infrastructure and technology stack are subject to permanent improvement. Today, the *openHPI Cloud* fills three full-size server racks. It provides a wide range of services, extensively employs redundancy, and is moving to a modern HPI data center recently put into operation.

³⁷OpenNebula Systems, <https://opennebula.io>; last access: 02.09.2022.

3.1 High Availability, Scalable Infrastructure: The openHPI Cloud

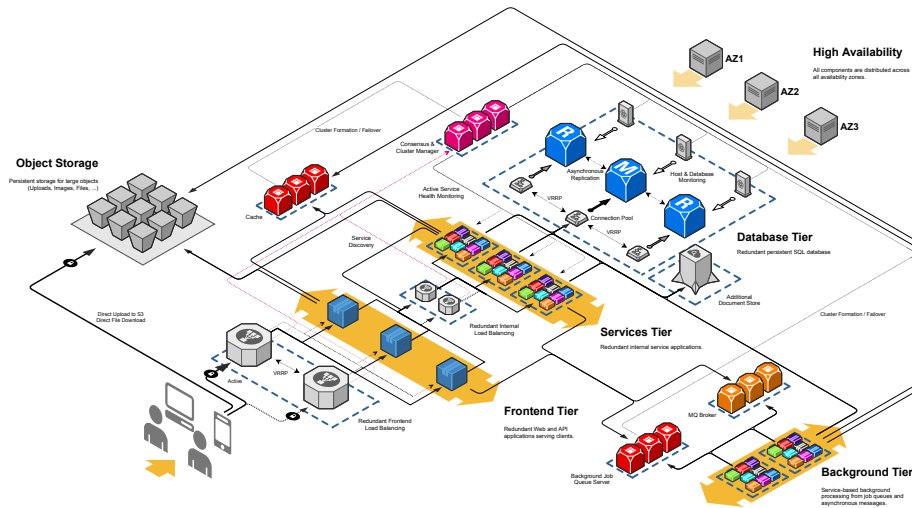


Figure 3.1: openHPI reference architecture

The *openHPI reference architecture* illustrated in Figure 3.1 allows a first glance at the complexity of the matured infrastructure of the openHPI Cloud, the basic services, and applications. The following chapters provide more insights into the components, starting from the bare metal in the data center, through cloud architecture (Section 3.1) and basic service layer (Section 3.2), up to the actual application layer (Section 3.3).

openHPI in Numbers The project started with three research associates and a handful of students that built the first platform prototype and ran it on the OpenNebula cloud sketched above. Since then, the platform, infrastructure, and team have grown up. The numbers in Table 3.1 bear witness to the successful journey of the project so far.

EU GDPR Compliance As mentioned above, all our servers are located in Potsdam, Germany. The platform complies with the *General Data Protection Regulation* (GDPR/DSGVO). All users are fully informed about which data is collected. Except for data required for the platform’s operation, no additional data is collected from the user without their consent. Additional consents can be configured separately for each platform. The user can change the consent settings at any time in their profile.

3.1 High Availability, Scalable Infrastructure: The openHPI Cloud

As already outlined, openHPI and all other HPI MOOC platforms Tier run on a private cloud on the premises of the HPI in Potsdam. The infrastructure’s cloud charac-

Overall Team Size	28
Developers (current/target)	8/12
DevOps/Cloud Engineers	4
PhD Candidates	8
Lines of Code	300,000+
Git Commits	35,000+
Test Cases in CI Pipeline	11,000+
Servers	30+
Cores	1,000+
RAM	4+ TB
Storage Capacity (gross)	85+ TB
Requests/minute (avg. rush hour)	6,000
Traffic/month	3–4 TB

Table 3.1: openHPI in numbers

teristics bring a solid base for scalability – it is easy to apply vertical scaling by using more resources for a machine or service and horizontal scaling with more machines or service instances in the cloud. Upgrading CPU count, memory, or storage requires only configuration and a reboot. Adding additional instances can be done via mouse clicks or – preferably – scripting.

The openHPI Cloud fundamentally builds on automation (see Section 3.2.2). This automation allows providing additional physical resources through setting up more compute nodes virtually in almost no time by installing the server to the rack, connecting power and network, configuring an initial IP address, and registering it in the configuration management. System setup, configuration, and integration as a new compute node will run without requiring manual operator interaction.

The rationale behind this high focus on automation is the relatively small team size in the Cloud Engineering and Operations team (see Table 3.1). Due to this fact – and to provide a failure-resistant service – the openHPI infrastructure is also designed for high availability: redundancy is applied at many levels using various concepts and technologies. For example, object and block storage are provided by *Ceph*³⁸ distributing storage over many physical hosts. Additional disaster recovery backup is being performed, particularly for databases and configuration data.

All essential services are run as clusters of three, deployed to three availability zones in the cloud. This allows virtually a third of the infrastructure to fail without consequence for the user since there is always a quorum of nodes left being able to vote for a new lead for any service cluster. This principle applies to all layers of the infrastructure, from the cloud-based services over the database and background tiers up to the application (see Figure 3.1).

³⁸Ceph is a software-defined storage platform for object storage on a single distributed computer cluster, implementing fault-tolerance through redundancy, self-healing, and self-management to minimize administration effort, see <https://ceph.io>; last access: 02.09.2022.

3.1.1 Data Center and Devices

For historical reasons, servers at the Hasso Plattner Institute were deployed to decentralized server rooms distributed over several buildings on the HPI campus. This decentralization is a result of individual chairs managing their own servers. Only a share of the IT infrastructure is operated by central IT – essential services like the directory service and user management, file service, network infrastructure and Internet connectivity, mail and groupware, DNS, the (central) website, and printers. Many research groups run individual servers for research projects and special laboratories, e.g., the HPI Future SOC Lab³⁹ or the HPI Data Engineering Lab.⁴⁰ Like these labs, openHPI operates its own infrastructure, relying solely on a few centralized services, e.g.,

- power supply including battery- and generator-driven uninterruptible power supply,
- cooling,
- domain name service,
- e-mail delivery, and
- 2nd (hard drive) and 3rd tier (tape library) backup.

At present, openHPI runs three sites corresponding to availability zones and clusters in two different buildings. Both buildings are separate fire protection compartments, so the operated infrastructure offers a certain degree of fire resistance. The project's servers are mounted in three locked racks. All server rooms have transponder-based access control with two-factor authentication and are equipped with intrusion alarm systems.

The HPI recently constructed a new building that hosts a modern data center. The new facility, including the data center, was first put into operation in January 2022.⁴¹ The data center comes with a wide range of features and security considerations, making it superior to the existing server room infrastructure, such as

- a redundant cooling system with six recirculating air cooling units (100 kW heat dissipation each); drained heat from the servers is being recovered into the heating system for sustainability reasons,
- uninterruptible power supply with 2x 500 kW battery packs and a diesel-driven emergency power system with fuel supply for several days of self-sufficient operation,

³⁹<https://hpi.de/forschung/future-soc-lab.html>; last access: 02.09.2022.

⁴⁰<https://hpi.de/forschung/hpi-data-engineering-lab.html>; last access: 02.09.2022.

⁴¹<https://hpi.de/news/jahrgaenge/2021/hasso-plattner-foundation-uebergibt-neues-institutsgaebaeude-ans-hpi.html>; last access: 02.09.2022.

- redundant power grid connection with two transformers, allowing interruption-free maintenance,
- facilities for redundant telecommunication and Internet connectivity,
- a separation sluice with two-factor authentication,
- multiple access control sections (cages) for different projects and protection requirements,
- lockable server racks with transponder-based authentication, managed by the central access control system, and
- a separate maintenance corridor for cooling and power supply systems with no access to computing devices.

Thus – despite losing one fire protection compartment – the entire openHPI infrastructure will move to the data center in the near future to benefit from the higher overall failure resilience and more elaborate physical access control.

Servers The openHPI Cloud computing hardware, primarily used for hypervisors – *compute nodes* – or databases, is mainly built from Dell’s PowerEdge series, e.g., R730 and R740 with Intel Xeon CPUs (various generations). The project recently deployed servers with AMD EPYC CPUs, i.e., Dell PowerEdge R7515. Several smaller R6515 are in use as utility hosts, e.g., for management services for OpenStack and Ceph.

The Intel-based compute nodes usually have two CPUs with 32 cores/64 threads and 768 GB of RAM, resulting in 24 GB of memory per real CPU. This allows going for a 1:24 CPU over-allocation in the cloud at 1 GB per CPU – which is a fair approximation that reflects the virtual machine instance flavor distribution (see Section 3.1.2). The newer AMD machines are equipped with 32 cores at 512 GB RAM (1:16 CPU over-allocation).

Many machines – particularly the 2U compute nodes – are equipped with SATA SSD storage; recent models come with an NVMe backplane, allowing much higher data transfer rates. These servers contribute to the physical foundation of the distributed storage system (see Section 3.1.3).

Recently, the server fleet was extended with GPU servers. Each comes with two 24-core AMD EPYC CPUs, 1 TB main memory, 10 TB (gross) fast SSD storage, and four NVIDIA A40 GPUs. These machines are used for research and teaching activities in the context of artificial intelligence.

Network Every site of the openHPI server infrastructure currently has a Brocade ICX7450-48 switch with 48 Ethernet ports at 1 Gbps and four additional 10 Gbps fiber ports. The Ethernet ports connect the servers for data transfer and management. The fiber ports are used to build a ring between the switches, always leaving two sites connected even if one switch fails (see Figure 3.2).

The servers – in particular those used for hypervisors (the compute nodes) – make use of bonding via the *Link Aggregation Control Protocol* (LACP), connecting

four copper ports to each server for the actual data traffic. This allows a theoretical upper limit of 4 Gbps throughput for parallel connections, while each connection is still limited to 1 Gbps. With management connections (in-band and out-of-band via iDRAC⁴²), single servers can occupy up to six Ethernet ports on a site switch.

There are additional switches for transition networks to the HPI DMZ (to access utility services managed by central IT) and to the Internet – openHPI has its own dedicated fiber connection to an Internet Service Provider (Versatel; currently at 1 Gbps) but can fall back to the HPI’s connection to the DFN⁴³ backbone in case of a severe disruption of the service.

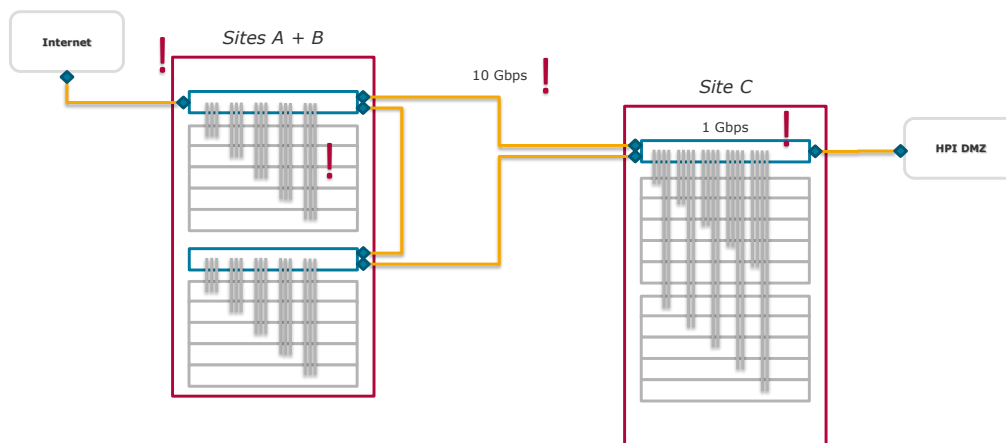


Figure 3.2: openHPI Cloud: current network setup

With a growing number of servers per site as well as growing utilization of the server capacity, the current network setup is reaching its limitations in multiple ways:

- **Capacity:** The switch ports are already fully occupied, since link aggregation requires many cables and ports. This could, however, be solved by adding more switches.
- **Bandwidth:** The low throughput of 1 Gbps between switches and servers severely limits maintainability. Every time a host needs maintenance, all virtual machines need to be live migrated to other servers. As most servers

⁴²Integrated Dell Remote Access Controller, <https://www.dell.com/en-us/dt/solutions/openmanage/idrac.htm>; last access: 02.09.2022.

⁴³Deutsches Forschungsnetz, the German national research and education network.

are utilized to their full capacity, in some case this can take up to 16 hours in our setup. Only after a server has been evacuated, it can be upgraded and rebooted. Additionally, the bandwidth of 10 Gbps between sites can already be saturated in peak load situations. Both bandwidth limitations account for a limited data throughput when using distributed storage.

- **Complexity:** Different server groups need different configurations, while such switches only come with limited automation functionality and thus require an undesired share of manual configuration effort.
- **Availability:** Site switches are single points of failure in the current networking setup. When a switch drops out, a complete site – a third of the available computing and storage capacity – is down. While this is fine for emergency situations – the architecture is built that way – it is undesirable for regular switch maintenance.

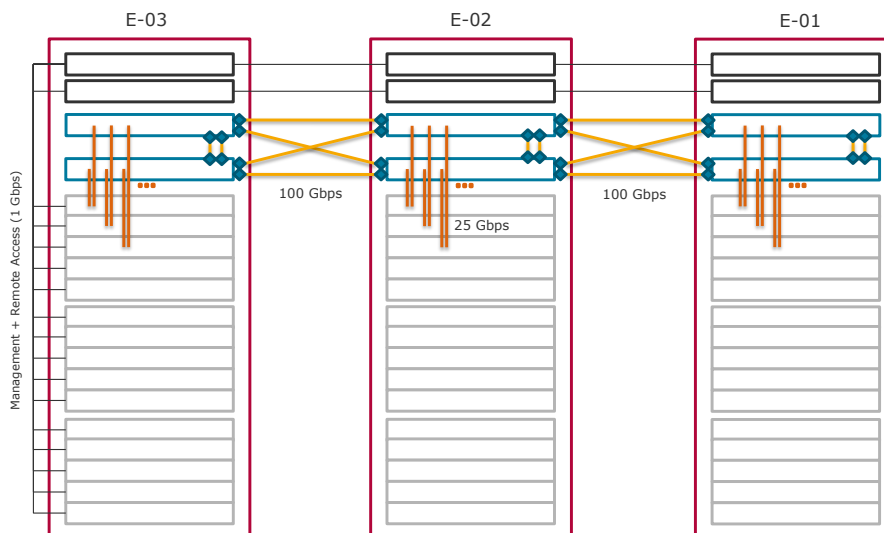


Figure 3.3: openHPI Cloud: planned network setup

When relocating the servers to the new data center, the networking infrastructure will be completely rebuilt (see Figure 3.3). Management and data networks will be uncoupled and implemented on independent switch installations. For the *management network*, six Dell S3048-ON switches will be deployed, two switches per site. The management network doesn't need high throughput; thus, the limitation to 1 Gbps is not an issue. The switch model was chosen for its relatively low-energy consumption and *Open Network Install Environment* (ONIE) support. ONIE is an installation environment for alternative *network operating systems* (NOS), including Linux-based systems with extensive automation capability.

The *data network* will be set up similarly: six NVIDIA/Mellanox MSN3420-CB2FC switches, coming with 48 25 Gbps access ports and twelve 100 Gbps trunk ports. All servers are connected to two data switches, allowing immediate fail-over. The sites are crosswise linked with each other to sustain the continuous link between all sites even when one switch fails or is under maintenance. The data switches will be run with *Cumulus Linux*⁴⁴ as NOS.

The new networking infrastructure is planned to become operative in 2023.

3.1.2 OpenStack

OpenStack is an open cloud computing platform that has become an industry standard. It is mainly used for providing *infrastructure-as-a-service (IaaS)* in both public and private clouds where services, such as computing resources, cloud storage, virtual networking, and load balancing, are made available to users. The software platform consists of interrelated components (see Figure 3.4) that control heterogeneous hardware pools of computing, storage, and networking devices. Cloud service users can manage their resources on a web-based dashboard, through command-line tools, or via an API.

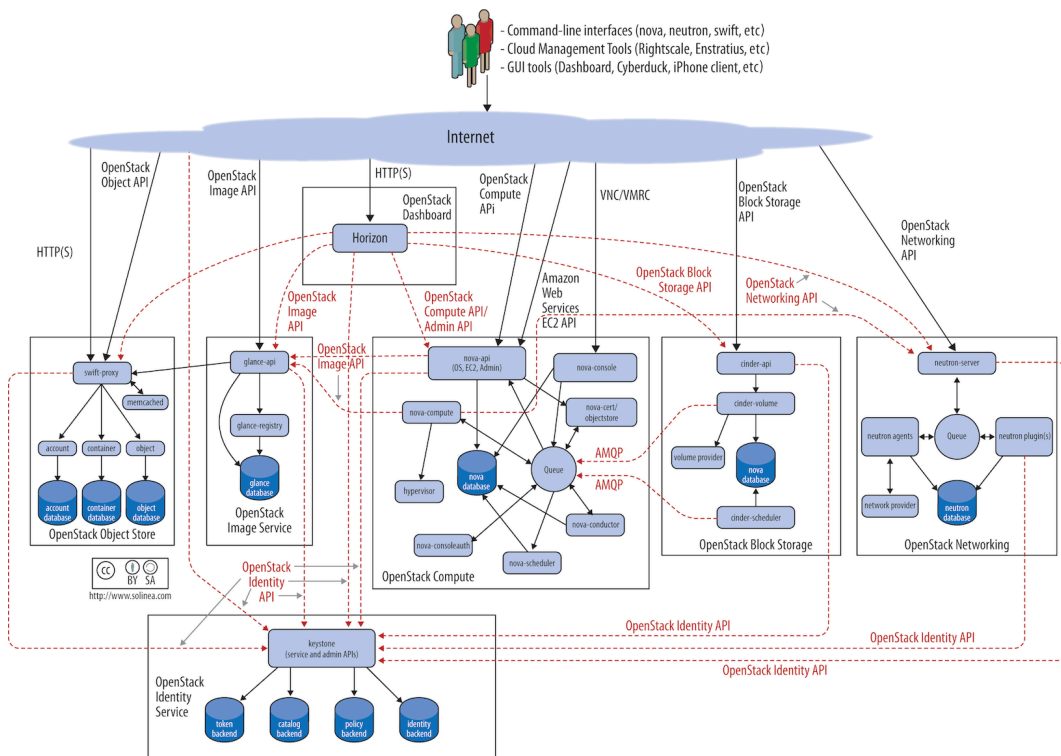


Figure 3.4: OpenStack logical architecture [29]

⁴⁴<https://docs.nvidia.com/networking-ethernet-software/cumulus-linux-51/>
last access: 02.09.2022.

Especially in Europe, several public cloud service providers employ OpenStack to offer IaaS for customers who cannot use the services of US-based companies (e.g., Amazon, Google, Microsoft) due to requirements in the context of privacy or information security. For example, the largest clouds are run by Deutsche Telekom or the French OHV. Also, Chinese telecommunication providers build their public cloud infrastructure based on OpenStack (e.g., China Telecom and China Unicom).

The openHPI Cloud deploys a subset of the OpenStack components to provide its services.

Compute *Nova* is the OpenStack component for provisioning compute instances, i.e., virtual machines or containers. The openHPI Cloud uses Nova with *libvirt*⁴⁵ to manage *QEMU*⁴⁶ virtual machines. The QEMU virtual machines are used to run the HPI MOOC platform, as well as the development tooling, e.g., a *Version Control System* (VCS), an issue tracker, or *Continuous Integration* (CI) tools and agents. The cloud also provides Linux containers with *LXD*.⁴⁷ These containers are mainly used for databases and are meant to isolate those of different platform instances from each other. Thus, *nova-compute* agents run on all compute nodes, *nova-compute-lxd* on the database hosts. Other Nova services, e.g., for scheduling or internal API, run on a cluster of three OpenStack control plane servers (*control nodes*).

Networking *Neutron* provides “network connectivity as a service” between interface devices, e.g., physical or virtual *Network Interface Cards* (NICs), managed by other OpenStack services – in the openHPI Cloud, mostly Nova. Neutron manages all networking aspects for the *Virtual Networking Infrastructure* (VNI) and the access layer (L2) aspects of the physical network. Neutron enables projects to create virtual network topologies, which may also include services such as routing and firewalls.

For each HPI MOOC platform, there is a dedicated OpenStack project, and additional projects exist for development tooling and other purposes. All these OpenStack projects run routers connected to the project’s internal network and – depending on the use case – to gateway networks, e.g., to cloud services outside of OpenStack, the HPI network, or the public Internet.

Neutron is run with a *Linux Bridge Agent*⁴⁸ setup that creates routers, firewalls, and interfaces using Linux network namespaces, bridges, and *iptables*.⁴⁹ Each Neutron router is deployed in a *highly available* (HA) fashion. Each virtual router instance utilizes a Linux networking namespace to manage its own routing table, firewall, and virtual network interfaces. These connect the project’s virtual net-

⁴⁵libvirt is a toolkit to manage virtualization platforms, <https://libvirt.org>; last access: 02.09.2022.

⁴⁶QEMU is a generic and open source machine emulator and virtualizer, <https://www.qemu.org>; last access: 02.09.2022.

⁴⁷LXD is a system container and virtual machine manager for Linux, <https://linuxcontainers.org/lxd>; last access: 02.09.2022.

⁴⁸https://docs.openstack.org/neutron/latest/contributor/internals/linuxbridge_agent.html; last access: 02.09.2022.

⁴⁹<https://netfilter.org/projects/iptables/>; last access: 02.09.2022.

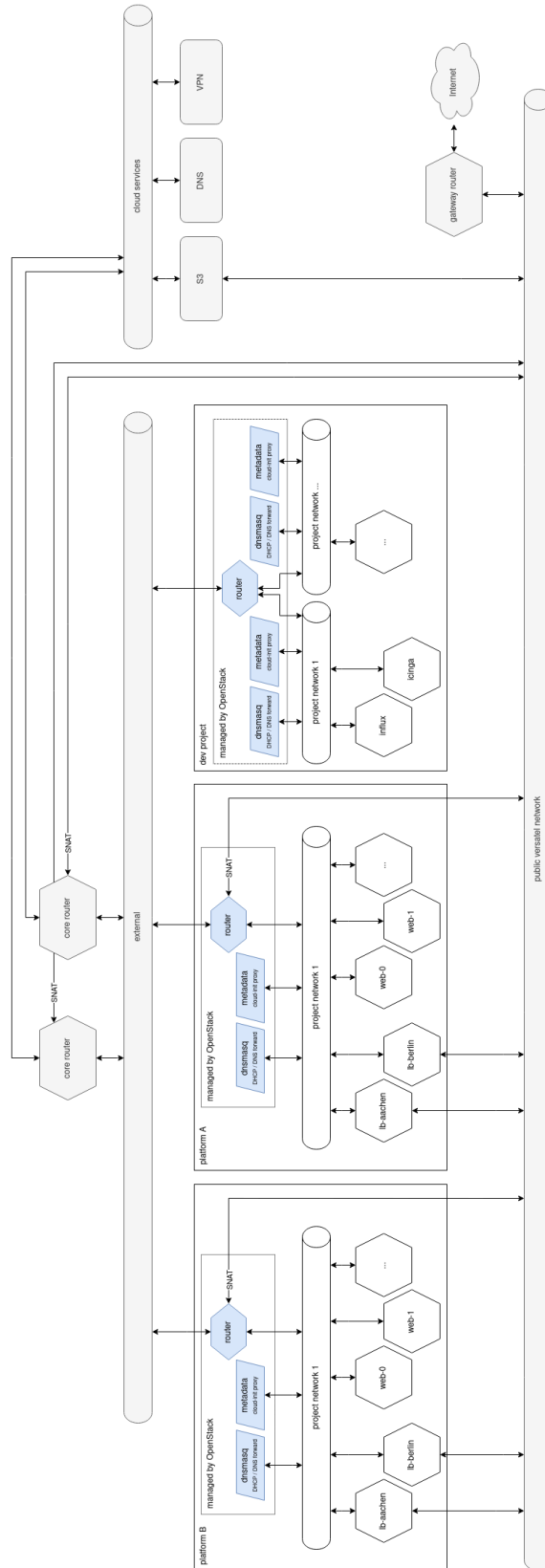


Figure 3.5: Neutron: project and gateway networks in the openHPI Cloud

works with gateway nodes and core routers, to provide connectivity to the Internet and other projects networks. Besides routing and firewall, a *keepalived*⁵⁰ process is run in each namespace to provide IP failover.

These network namespaces are distributed over our three network nodes. For example, Neutron does *Source Network Address Translation* (SNAT) for all VMs that are connecting to the Internet to translate their internal IP to a common public one, usually from the *versatel* network, i.e., the public Internet (see Figure 3.5). In this way, virtual machines can connect to the Internet.

The virtual networks are realized using VXLAN to transport the overlay network traffic between compute hosts and routers. It utilizes IP multicast to efficiently send traffic only to compute nodes and routers involved with a specific virtual network. Therefore, all kind of traffic is supported between virtual machines.

Block Storage The block storage service of OpenStack is *Cinder*.⁵¹ Cinder provides volumes – virtual hard disks – to the virtual machines managed by Nova. The openHPI Cloud uses Ceph (see Section 3.1.3) as the storage backend. Therefore, volumes are highly available and replicated across multiple servers. Virtual machines transparently connect to attached volumes, providing standard block devices to the system.

Images Virtual machines are created from operating system images, i.e., a minimal base system of a Linux distribution. In OpenStack, the *Glance*⁵² service provides a catalog of these images, allowing registering, discovery, and retrieval of these VM images. The cloud operators keep these images up-to-date, while users can easily create new instances, for example, with the latest Ubuntu operating system.

Object Storage While file systems manage data in a file hierarchy, object storage manages data as objects with metadata and a globally unique identifier. Such systems are widely adopted as cloud storage services, e.g., Amazon's *S3* (Simple Storage Service). Their main characteristics are

- programming interfaces to use the object stores directly in (web) applications,
- the ability to be spanned over many physical devices, servers, or locations including the potential to grow as needed, and
- data replication at object-level granularity for the prevention of data loss.

Even though OpenStack comes with an own object store service – *Swift* – the openHPI Cloud uses *Ceph radosgw* (see Section 3.1.3) as object store, which comes with a Swift-compatible API.⁵³

⁵⁰<https://www.keepalived.org>; last access: 02.09.2022.

⁵¹<https://docs.openstack.org/cinder/latest>; last access: 02.09.2022.

⁵²<https://docs.openstack.org/glance/latest>; last access: 02.09.2022.

⁵³<https://docs.ceph.com/en/quincy/radosgw/swift>; last access: 02.09.2022.

Identity Management *Keystone*⁵⁴ is the OpenStack service for API client authentication and service discovery. It is used to authenticate and authorize users and software across the different services and common clients, e.g., the OpenStack *Horizon* dashboard, the command line client tools, and custom third-party management applications.

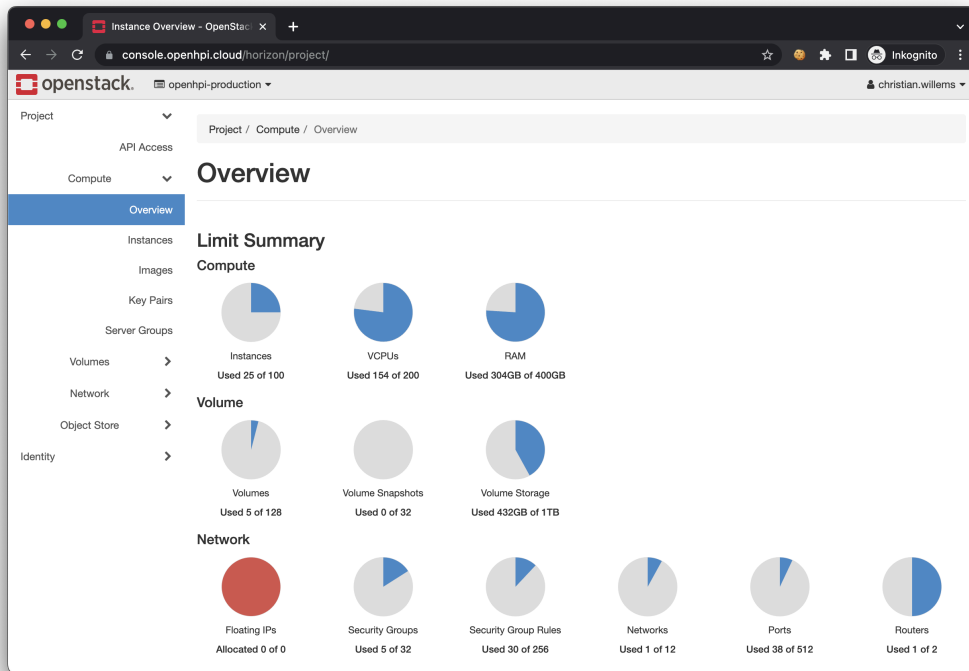


Figure 3.6: OpenStack Horizon dashboard for openHPI production

Dashboard The web-based client for managing an OpenStack cloud is *Horizon*⁵⁵ which provides an interface to the OpenStack services, including Nova, Swift, Keystone, Neutron, and Glance. Horizon can be used to manage images, virtual machine instances, virtual networks, volumes, security groups, quotas, users and projects, and flavors for VM instances. It also provides dashboards for resource utilization in projects (see Figure 3.6).

Domain Name Service (planned) Domain Name Service (DNS) records for applications hosted in the openHPI Cloud (i.e., the HPI MOOC platform instances) are currently managed outside of OpenStack. When a DevOps engineer creates a new

⁵⁴<https://docs.openstack.org/keystone/latest>; last access: 02.09.2022.

⁵⁵<https://docs.openstack.org/horizon/latest>; last access: 02.09.2022.

platform instance, she would provide the domain name(s) for this platform with the Terraform configuration (see Section 3.2.2.1) that already specifies the virtual machine orchestration for this instance. When this configuration is applied, the DNS records are added to the zone files of openHPI's BIND DNS servers. However, there are plans to facilitate *Designate* for this purpose in the future. *Designate* provides DNS as a service and is compatible with BIND as backend technology. *Designate* actually does not include DNS servers, but is meant to manage DNS zones on a project basis.

Virtual Machine Flavors In OpenStack, flavors define the compute, memory, and storage capacity of nova computing instances. In simple terms, a flavor is an available hardware configuration for a machine. It defines the size of a virtual server that can be launched.

A flavor definition comes with specific values for the number of vCPUs, memory size, attached disks (and their types and sizes), and attached GPUs (if available).

The openHPI Cloud currently provides four types of flavors, each in different sizes:

General Purpose The so called *t2 flavors* are instances for general-purpose workload in different sizes. Storage can only be provided through explicitly attached (persistent) volumes. Typical use cases for these flavors are databases or applications from the developer tool chain, such as GitLab, CI, or the issue tracker. The flavor names and sizes are therefore close to those in AWS EC2.

Ephemeral Workload The *e2 flavors* (see Table 3.2) are basically equipped like the *t2* ones, with the difference of having ephemeral storage already attached. These volumes only exist for the lifetime of the virtual machine itself, so it does not persist data. *e2* VMs are usually used for high-availability, ephemeral workload, e.g., the actual application servers (*e2.2xlarge*), routers, gateways (*e2.tiny*, *e2.micro*), or CI agents (*e2.medium*).

GPU Servers The *p3a flavors* come with GPU accelerators attached. *p3a.large* has 24 vCPUs, 240 GB RAM, 50 GB instance storage, 900 GB ephemeral storage, and 1 NVIDIA A40 GPU. *p3a.xlarge* doubles all these resources. Such instances are used for AI applications.

Classic Servers The (planned) *m flavors* will be provided on dedicated physical servers to provide virtual machines with (most) characteristics of a bare-metal server, such as fixed (pinned) CPUs, no over-allocation, and fast local storage. Those will mainly host database applications.

3.1.3 Ceph and radosgw

The openHPI Cloud provides an S3-compatible object store based on a Ceph software-defined storage (*SDS*) cluster. It utilizes this cluster as a high-available,

Instance Size	vCPU	Memory (GB)	Instance Storage
e2.nano	2	0.5	10 GB
e2.tiny	2	1	20 GB
e2.micro	2	2	25 GB
e2.small	2	4	25 GB
e2.medium	4	8	25 GB
e2.large	4	12	40 GB
e2.xlarge	4	16	50 GB
e2.2xlarge	8	32	80 GB

Table 3.2: openHPI Cloud flavors for ephemeral virtual machines

redundant backing store for virtual machines and volumes in OpenStack, and also as a shared file system (*CephFS*) and volume store for control services.

Ceph is an open source SDS system, implementing object storage on a server cluster. Ceph provides interfaces for object storage, as well as block and file storage (and also expose a POSIX-compatible file system), mapping all three storage paradigms to its internal object store. In recent versions, Ceph is not built on top of traditional file systems, but can manage disks and flash devices directly.

The main characteristics and features of Ceph are

- comprehensive distributed operation for failure safety (no single point of failure),
- scalability up to exabyte level,
- fault-tolerance through data replication and erasure coding,
- disaster recovery tooling with snapshots and storage cloning, and
- minimized administration effort with a self-healing and self-managing design.

A basic Ceph setup requires a number of services to be available. In the present deployment, three instances of each service are run for high availability.

Monitor *ceph-mon* maintains so-called maps of the cluster state. These maps allow the other services to coordinate with each others. The monitor also handles authentication.

Manager *ceph-mgr* provides runtime information on the cluster, e.g., storage utilization, current performance metrics, and system load. It also exposes this information via web UI (see Figure 3.7) and REST API.

Metadata Service *ceph-mds* stores metadata for the Ceph File System (CephFS), allowing the execution of basic POSIX file system operations (e.g., `ls`).

OSDs Object Storage Daemons (*ceph-osd*) store data and handle replication, recovery, and rebalancing. Basically, all drives in a Ceph cluster are managed by a dedicated OSD.

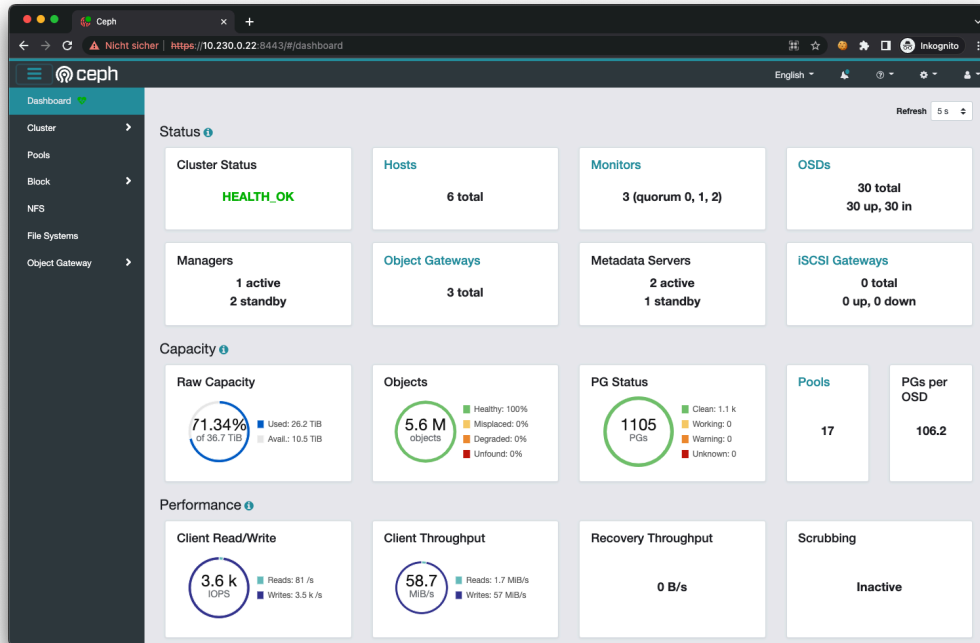


Figure 3.7: Ceph dashboard for the openHPI Cloud

Currently, all Ceph management services still run on three compute nodes. This includes *ceph-mgr*, *ceph-mon*, *ceph-mds*, and *radosgw*. The introduction of a cluster of three dedicated Ceph management nodes is planned. Additionally, multiple *ceph-osd* run on all compute nodes. Some disks on each server are used for OSDs to provide the actual storage space.

The *RADOS gateway* (*radosgw*) is a Ceph client that provides an S3- and Swift-compatible API to provide object storage. Especially, S3 is used by the HPI MOOC platform to host and serve files, also directly to users. This is the Object Store as depicted in the reference architecture in Figure 3.1 (page 73).

3.1.4 openHPI Cloud Service for the HPI

With growing maturity and recognition inside the Hasso Plattner Institute, other research and student groups as well as other departments of the HPI requested computing resources in the openHPI Cloud. The team already provides a number of Bachelor and Master projects, the *Deep Learning* research group, and the research project *Smart4Health* with their own project spaces in the openHPI Cloud.

Furthermore, web applications like a *LimeSurvey* instance, EU-project websites (for BizMOOC⁵⁶ and the MOOC Book⁵⁷), and the HPI Podcast (“Neuland”⁵⁸) run in the openHPI Cloud.

Recently, there is an incipient cooperation of the openHPI Cloud and the *HPI Digital Health Center* (DHC), evaluating a consolidation of the DHC computing infrastructure. Our vision is that the openHPI Cloud, which originally started out with four servers and one switch, will become a cloud service available for the whole institute.

3.2 Application Technology Stack and Development Tooling

This section gives a brief overview of the infrastructure parts around the *frontend tier* and *services tier* in the openHPI reference architecture (see Figure 3.1, p. 73). It will also provide insights into the automation tooling and the deployment pipeline set up to bring the HPI MOOC platform code into production.

3.2.1 Application Infrastructure Components

The most important building blocks around the HPI MOOC platform are databases for persistence as well as message queue and HTTP servers for (inter-service) communication.

3.2.1.1 Databases

The platform uses a set of different databases for specific usage scenarios.

PostgreSQL The main data storage for the application is the open-source relational database management system *PostgreSQL*.⁵⁹ Postgres is the persistence layer for the actual HPI MOOC platform, including all services (see Section 3.3.2). The application also uses Postgres for background jobs that are only allowed to run once at a time, employing database locking features. Additionally, *pg_bouncer*⁶⁰ is used for connection pooling. The most important database backend is run as a high-availability (HA) cluster, based on *Patroni*.⁶¹ Patroni is a cluster manager framework used to customize and automate PostgreSQL HA clusters. The tool prevents data loss by deciding what actions to perform in the cluster based on consensus algorithms. It takes care of synchronicity requirements and manages planned switchovers as well as unplanned failovers. The execution of these tasks is largely automated. Furthermore, certain conditions can be defined that must

⁵⁶<https://bizmooc.eu>; last access: 02.09.2022.

⁵⁷<https://mooc-book.eu>; last access: 02.09.2022.

⁵⁸<https://podcast.hpi.de>; last access: 02.09.2022.

⁵⁹<https://www.postgresql.org>; last access: 02.09.2022.

⁶⁰<https://www.pgboncer.org>; last access: 02.09.2022.

⁶¹<https://github.com/zalando/patroni>; last access: 02.09.2022.

always be met in order to completely exclude irrevocable damage to the data. The architecture of Patroni is such that each PostgreSQL instance is accompanied by a Patroni instance that monitors and controls the PostgreSQL instance. *Consul*⁶² is used as distributed key-value store for cluster state and building consensus.

Elasticsearch The HPI MOOC platform generates a huge amount of analytics events, reflecting learner interaction with platform and learning content. These events are stored in the document-oriented (schema-free JSON documents) database *Elasticsearch*.⁶³ Elasticsearch actually is a (full-text) search engine, providing a search index backend and an HTTP-based query endpoint. It works well for processing large amounts of data and building aggregate functions – therefore, the HPI MOOC platform mostly uses Elasticsearch for the compilation of reports or the calculation of KPIs and other statistics. Further, it mainly serves as a backend for admin-facing features (compare Section 2.2.2 and Section 2.2.3), and is not deployed as a fail-safe cluster setup. Event data is written asynchronously through a message queue (see Section 3.2.1.2), which stores the events until they can be passed on to the database. Thus, a temporary unavailability of Elasticsearch due to maintenance or failure is acceptable since data won't be lost and the majority of users (i.e., the learners) are not affected.

Redis The HPI MOOC platform uses *Redis*,⁶⁴ an open-source, distributed, in-memory key-value database, 1) for various caching purposes, e.g., server-side application caching of rendered HTML pages or data structures, and 2) as backend for the Sidekiq⁶⁵ background job scheduling. Both usage scenarios require an own instance of Redis, since the cache data is volatile and does not need to be kept over re-starts of the service while the background job queue must be persisted. Redis allows the latter, for example, by creating regular snapshots of the memory state or by appending new entries to a file when writing those to memory. Redis is run as a high-availability failover cluster managed by *resec*⁶⁶ – again using Consul for building consensus in the cluster. Since *resec*-based Redis clusters turn out to be a fragile construct, and Redis is the only backend for the Sidekiq job scheduler, an alternative approach with a relational database as job queuing backend is currently evaluated (see Section 3.3.2).

Others In peripheral systems around the actual application, there are also a number of other databases being used. Application metrics (e.g., for Grafana Dashboards, see Section 3.2.4) are stored in the open-source time series database *In-*

⁶²<https://www.consul.io>; last access: 02.09.2022.

⁶³<https://github.com/elastic/elasticsearch>; last access: 02.09.2022.

⁶⁴<https://github.com/redis/redis>; last access: 02.09.2022.

⁶⁵<https://github.com/mperham/sidekiq>; last access: 02.09.2022.

⁶⁶Consul-based highly available Redis replication agent, <https://github.com/YotpoLtd/resec>; last access: 02.09.2022.

fluxDB.⁶⁷ The query language of InfluxDB comes with time-centric functions for querying a data structure composed of measurements, series, and points, making the database a preferred backend for operations monitoring and application metrics. Other peripheral services, such as *Icinga*⁶⁸ (host and network monitoring tool) or *Matomo*⁶⁹ (web analytics, formerly known as Piwik) run their local instances of *MySQL*.⁷⁰

3.2.1.2 Messaging

The HPI MOOC platform makes heavy use of asynchronous communication to not block or delay responses to user interaction with tasks that might be long-running or need to process shared data on limited resources. Messaging is performed by *RabbitMQ*,⁷¹ an open-source message broker service implementing the *Advanced Message Queuing Protocol* (AMQP). A message broker listens for incoming messages from connected applications (*publishers*), which are sorted into the specified queue(s). Other applications (*consumers*) can subscribe to queues. The message broker ensures that all relevant incoming messages are delivered to all subscribing consumers. The message broker is responsible for guaranteed message delivery, thus also handling errors and repeated deliveries of messages that are not acknowledged by all subscribers. Message brokers are fast – they can easily process thousands of messages per second.

The infrastructure at hand was built as a high-availability cluster of three nodes each, currently using a *mirrored queues* approach: all data and state is replicated over all nodes. There is no leader/follower concept, but all nodes are treated equally. This comes with the downside that all incoming and outgoing messages must be synchronized among all nodes, which impacts performance. This becomes an issue when a node fails and needs to be re-synchronized: bringing a cluster node back in sync can take a long time for large queues, and while syncing the cluster cannot process new messages.

For this reason, the cluster is being refactored to use *quorum queues* [64], which outperform mirrored queues for both performance and the synchronization as they ensure consistency by design. A quorum queue is a replicated queue with a leader and followers. In our case, with three cluster nodes, this results in two replicas per queue. Clients (publishers and consumers) always interact with the leader, which then replicates all the commands (e.g., write, read and acknowledge) to the followers. Those do not interact with the clients at all, they exist only for redundancy. When a broker goes offline, a follower replica will be elected leader and service will continue.

⁶⁷<https://github.com/influxdata/influxdb>; last access: 02.09.2022.

⁶⁸<https://icinga.com>; last access: 02.09.2022.

⁶⁹<https://matomo.org>; last access: 02.09.2022.

⁷⁰<https://www.mysql.com>; last access: 02.09.2022.

⁷¹<https://www.rabbitmq.com>; last access: 02.09.2022.

3.2.1.3 HTTP Servers and Load Balancing

Since the HPI MOOC platform is a web application, the HTTP servers are a crucial component of the setup. When revisiting the reference architecture (see Figure 3.1, p. 73), there are two places where load balancing is applied: 1) the frontend load balancers between client apps and the frontend tier, and 2) the internal load balancers between frontend tier and backend tier.

All load balancers are realized using *HAProxy*,⁷² an open-source high-availability load balancer and reverse proxy. The load balancer setup is designed redundantly, failover is realized by the *Virtual Router Redundancy Protocol (VRRP)*.

The frontend load balancers also take care of TLS offloading. All HTTP traffic between the services in the openHPI Cloud and clients are encrypted with TLS ≥ 1.2 and a modern cipher suite, resulting in an *A+* rating in the Qualys SSL Server Test.⁷³ TLS certificates are dynamically created with *Let's Encrypt*,⁷⁴ supporting automation. Application-internal communication is not encrypted but isolated on the physical network layer.

The actual application servers running in the frontend and backend tiers are prepended with *nginx*⁷⁵ HTTP servers, which can easily be configured to deliver static content or to proxy for a web application. They serve static assets bundled with the applications, such as JavaScript, Cascading Style Sheets (CSS), images, or error pages.

3.2.2 Automation

As already mentioned, the openHPI Cloud and operations teams heavily rely on automation, for a variety of reasons:

Reduction of Manual Effort The current (and growing) numbers of more than 550 virtual machines and several dozen containers running on more than 30 physical servers call for an efficient allocation of engineering resources. To run this machinery park with the available staff, manual intervention in setup and configuration of systems must be reduced to a minimum. Therefore, all systems – besides a handful of legacy systems currently being refactored – are deployed and configured only using *Infrastructure-as-Code (IaC)* and configuration management tools, namely *Hashicorp Terraform*⁷⁶ and *Saltstack*.⁷⁷

Technical Documentation Proper use of the IaC paradigm obsoletes the need for extensive technical documentation prose in wikis or other documents. All necessary information about the state of the running systems is already

⁷²<https://github.com/haproxy/haproxy>; last access: 02.09.2022.

⁷³<https://www.ssllabs.com/ssltest/analyze.html?d=open.hpi.de&s=82.140.0.52&latest>; last access: 02.09.2022.

⁷⁴<https://letsencrypt.org>; last access: 02.09.2022.

⁷⁵<https://nginx.org>; last access: 02.09.2022.

⁷⁶<https://www.terraform.io>; last access: 02.09.2022.

⁷⁷<https://saltproject.io>; last access: 02.09.2022.

documented in the IaC scripts, which are stored in a version control system, allowing to track changes. This also reduces the workload when setting up new or modifying existing systems.

Business Continuity IaC and declarative configuration stored in a distributed manner allows for quick recovery in cases of disaster. When systems fail completely and must be re-built from scratch, extensive automation can create whole infrastructure landscapes on a keystroke. In the openHPI Cloud, also the system on lower levels, e.g., OpenStack compute or control nodes are completely built from Salt states.

Extendability of the Cloud Automation for the deployment of physical hosts allows to easily extend the cloud resources. New compute nodes are mounted to the server rack, connected to power and network, configured with an initial IP address, provided with a base operating system, and finally configured and added to the cloud via Saltstack.

Deploying New Platform Instances An instance for only one environment of the HPI MOOC platform (e.g., a production system) consists of at least 23 virtual machines: 3x Consul, 3x PostgreSQL, 3x RabbitMQ, 3x Redis, 1x Elasticsearch, 1x PostgreSQL for analytics, 2x internal load balancing, 1x application tasks, at least 2x of each application web, services, and background job. All these machines share service, application, and network configuration and must also be registered in other systems, such as DNS or frontend load balancers. Larger platforms have separate instances for the production, staging and testing environments. Since this is a complex, recurring task, it is extensively automated.

3.2.2.1 Infrastructure-as-Code – Terraform

Terraform is an open source Infrastructure-as-Code that allows to define and provide data center infrastructure using a declarative configuration language. A Terraform definition describes the desired state of an infrastructure. When invoking Terraform with such a definition, the tool outlays a *plan* of changes – which resources will have to be created, changed, or destroyed in order to form the desired state.

When executing the plan, Terraform calls the appropriate set of so-called *providers*, adapters for software systems used to create the desired infrastructure by, e.g., calling APIs or invoking command line tools. Such providers include public cloud infrastructure such as Amazon Web Services, Microsoft Azure, and Google Cloud Platform, but also private cloud environments like VMware vSphere and OpenStack. In the openHPI Cloud, Terraform is mostly used to manage OpenStack resources – even though the Terraform Provider Registry holds more than 2,000 projects for a multitude of tools, including databases, CI/CD tools, monitoring, and many others.

Listing 3.1: Terraform example: deploy openHPI Testing (extract)

```
1 module "site" {
```

3 Technology and Architecture

```
2 source = "../modules/site"
3
4 id      = "openhpi-testing"
5 label   = "openHPI Testing"
6 key     = "[REDACTED]"
7 minion_prefix = "de.xopic.xi.openhpi.testing"
8 subzone = "testing.openhpi.xi"
9 subzone_x = "testing.openhpi"
10 }
11
12 # core instances
13
14 module "db" {
15   source = "../modules/db"
16   site   = module.site
17
18   flavor_name = "p2.small"
19 }
20
21 module "elasticsearch" {
22   source = "../modules/elasticsearch"
23   site   = module.site
24
25   data_size = 10
26 }
27
28 [...]
29
30 # application instances
31
32 module "srvs" {
33   source = "../modules/worker"
34   site   = module.site
35
36   number      = 2
37   name        = "srvs"
38   flavor_name = "e2.small"
39   image_id    = module.site.ubuntu2004_image_id
40   roles       = ["services"]
41   secgroups   = [module.site.secgroup.name]
42 }
43
44 module "web" {
45   source = "../modules/worker"
46   site   = module.site
47
48   number      = 2
49   name        = "web"
50   flavor_name = "e2.small"
51   image_id    = module.site.ubuntu2004_image_id
52   roles       = ["frontend"]
```

```

53   secgroups   = [module.site.secgroup.name]
54 }
55
56 [...]
57
58 # Standalone monitoring
59
60 module "monitoring" {
61   source      = "../modules/icinga"
62   site        = module.site
63   ingress_name = "ingress"
64   rabbitmq_pwd = "[REDACTED]"
65   postgres_pwd = "[REDACTED]"
66
67   services = [
68     "account",
69     "certificate",
70     "collabspace",
71     "course",
72     [...]
73     "web",
74   ]
75
76   require_service_count = 1
77   expected_service_count = 2
78 }

```

Listing 3.1 shows an extract of the description necessary for deploying the openHPI Testing platform. It composes the necessary backend tier services (e.g., databases, messaging, and load balancers) and the application VMs for tasks, jobs, services and the web interface. It makes use of a set of *modules* (*db*, *elasticsearch*, *worker*) that contain reusable descriptions of the actual VM instances to be spawned. Some attributes are explicitly overwritten: `flavor_name = "e2.small"` declares that we only need small application server instances for the testing environment, or `data_size = 10` attaches a rather small disk to the Elasticsearch instance.

It also uses the *icinga* provider (from line 60, see also Section 3.2.4.1) to configure monitoring for the availability of all application services: the monitoring tool would show a warning when the service count is less than expected, i.e., only one instance, and a critical alert if no instance of a service is available (through `require_service_count = 1`).

3.2.2.2 Configuration Management – SaltStack

SaltStack (short: *Salt*) is a Python-based, open source software for event-driven IT automation, remote task execution, and configuration management. It allows applying commands or system configuration to a group of systems, the *salt minions*. Those minions register to a *salt master*, which applies *salt states* on registered and authenticated minions.

Listing 3.2: Basic salt state

```

1 xikolo-web:
2   pkg.installed

```

The very basic salt state in Listing 3.2 for an openHPI web application VM would ensure that the Debian package `xikolo-web`⁷⁸ is installed on a system where the state is applied.

Listing 3.3: SaltStack example: state for an openHPI web application VM (extract)

```

1 {% include 'xikolo/mod/application.sls' %}
2 {% include 'xikolo/mod/config.sls' %}
3 {% include 'xikolo/mod/database.sls' %}
4 {% include 'xikolo/mod/delayed.sls' %}
5 {% include 'xikolo/mod/mnemosyne.sls' %}
6 {% include 'xikolo/mod/rabbitmq.sls' %}
7 {% include 'xikolo/mod/secrets.sls' %}
8 {% include 'xikolo/mod/services.sls' %}
9 {% include 'xikolo/mod/sidekiq.sls' %}
10
11 # Only web needs to migrate the database.
12 # All services will use the same one.
13 {{ map.name }}/task/migrate:
14   cmd.run:
15     - name: '{{ map.name }} rake db:migrate'
16     - env:
17       - DATABASE_PORT: '5432'
18     - onchanges:
19       - pkg: {{ map.package.name }}
20     - require:
21       - test: {{ map.name }}/config
22
23 [...]

```

Listing 3.3 illustrates the complexity of nested salt states used for the configuration of an openHPI learning platform. It makes heavy use of reusable modules and variables, e.g., for configuring the actual application, shared configuration, database connection, background jobs, monitoring integration, or application secrets. The example shows the state for the openHPI web application, which also includes the database definition for all services. The definition of `task/migrate` (from line 11) states, that the command `xikolo-web rake db:migrate` (line 15) with the environment variable `DATABASE_PORT=5432` (line 17) is to be executed, when the Debian package `xikolo-web` changes (line 19) – the config directory for the application must be present (line 21). A change of the package could also include a

⁷⁸*Xikolo* is the internal name for the HPI MOOC platform software.

change of the database schema, so the database migration task must be executed on this condition.

3.2.3 Deployment Pipeline

The openHPI developer team follows the principle of agile software development and has implemented Scrum, with a sprint cycle of two weeks. The team has committed to the *Release often, release early* principle [30]. Code increments are deployed as early and often as possible, allowing to get early customer feedback. A useful side effect of this release cycle is that increments are *small* – this facilitates error diagnosis and bug fixing when code doesn't work as expected in production.

In practice, the openHPI team releases five to ten times a week on average, but even multiple rollouts on a single day are common. This requires a highly automated and tool-supported deployment process.

3.2.3.1 Source Code Management

The openHPI project uses *git* as version control system with *GitLab* as web frontend for all source code management, including the actual application code, libraries and components (such as the Video Player), and all Infrastructure-as-Code (Terraform and Salt files).

New code is committed to git repositories following the *ship/show/ask* branching strategy [10]: trivial changes, implemented by an experienced developer with profound domain knowledge can be pushed to the mainline (master branch) immediately (*ship*). Non-trivial changes in areas with an excellent test coverage shall be pushed in a feature branch, but do not necessarily need approval by other developers (*show*). All other changes must be pushed to feature branches and need the approval of at least two other developers (*ask*, 4-eyes-principle).

Historically, the HPI MOOC platform code was dispersed over several git repositories – one for each involved service (see Section 3.3.2). A first step towards going back from a service-oriented architecture to a monolithic application was merging all these repositories into a single *monorepo* (see Section 3.3.4 for discussion). This is the way the code is organized today.

3.2.3.2 Continuous Integration

In software engineering, *Continuous Integration* means that code changes in developers' working copies of a code repository are merged with the mainline/master several times a day. All new versions of the mainline are then automatically tested. Additionally, it is a common practice to already test feature branches before merging to the mainline, to be able to already exterminate errors beforehand.

This is realized with an automated CI pipeline, using a Continuous Integration service. In the openHPI project, the on-premise GitLab instance performs this task. The pipeline looks as follows:

1. *Linting*: The first step of a CI pipeline is linting. Linting is a method for static code analysis aiming to find (syntax) error, bad code style, or bad

coding practice. Linting in the CI pipeline enforces a quality standard for the whole codebase. Linters can correct many formatting issues or syntax errors automatically. The linting steps in the CI pipeline use

- *ESLint*⁷⁹ for JavaScript code, which also performs formatting tasks,
- *Stylelint*⁸⁰ for Cascading Stylesheets (more precise: SCSS, see Section 3.3.1), and
- *Rubocop*⁸¹ for the Ruby on Rails code.

The linter configurations are consistent over the whole monorepo.

2. *Gem Version Comparison*: A second step towards moving to a monolithic application was the unification of dependencies in all existing services of the application. Dependencies of a Rails application are defined in a Gemfile. This task ensures that all of these are in sync.
3. *Vulnerability Scanning*: selected parts of the application are scanned for vulnerabilities using *Brakeman*.⁸² Brakeman statically analyzes Rails application code for security issues using the CWE vulnerability database.
4. *Tests for Local Gems*: The monorepo also includes a few gems, packaged Ruby libraries. The gems implemented for the HPI MOOC platform (e.g., an S3 connector, shared configuration, helper for tests and user handling) all come with unit tests, which are part of the pipeline.
5. *Open Source License Management*: The HPI MOOC platform includes many dependencies, either as Ruby Gems or as npm packages for JavaScript. These packages should all be open source, and most of them come with a license. Since there are many different open source licenses around and the manual effort for checking those for thousands of software packages is not feasible, the CI pipeline has a task for this. The job integrates FOSSA,⁸³ an external service that checks the dependency files for license compliance violations, i.e., licenses that do not match the configured characteristics.
6. *Documentation Release*: The monorepo includes several documentation projects, i.e., a development guide and documentation for public APIs. The development guide is built with MkDocs⁸⁴ and pushed to a web server, the API docs are published to Stoplight.⁸⁵
7. *Web Application Assets*: When running the HPI MOOC platform in a production-like environment – like the environment for integration testing – the

⁷⁹<https://eslint.org>; last access: 02.09.2022.

⁸⁰<https://stylelint.io>; last access: 02.09.2022.

⁸¹<https://rubocop.org>; last access: 02.09.2022.

⁸²<https://brakemanscanner.org>; last access: 02.09.2022.

⁸³<https://fossa.com>; last access: 02.09.2022.

⁸⁴<https://www.mkdocs.org>; last access: 02.09.2022.

⁸⁵Public API documentation: <https://openhpi.stoplight.io>; last access: 02.09.2022.

Rails assets must be compiled. This step currently takes approx. 6 minutes and would be performed several times in a naive setup (once for each task in step Item 9, as well as for the unit tests in the web application).

8. *Unit Tests*: Every service as well as the web application of openHPI comes with an extensive test suite, written in *RSpec*,⁸⁶ e.g., 2,680 test cases in the web app, 1,080 in the account service, or 3,300 in the course service. These tests can range from low level model, controller or presenter specs to request specs (testing an API endpoint) or even feature specs (using a remote controlled, headless browser).
9. *Integration Tests*: An essential test suite, especially for a service-based, composed application, are the integration tests or end-to-end tests. For these tests, an environment is built that mimics the production environment as close as possible. All services from the background tier (e.g., databases and message queue) and all services of the HPI MOOC platform are spawned on a CI agent virtual machine. Then, the agent performs a sequence of scenarios, each reflecting a use case. Again, a remote controlled browser is used to perform “real” clicks on the actual user interface of the application. The test scenarios are written with Cucumber, and in a formal language that can be read and understood by non-developers (see Listing 3.4). Currently, there are 275 use cases being tested. For performance, the integration test suite has been split into four jobs, which can then run in parallel on different agents.

Even though the above list reflects the logical flow of the jobs in a pipeline, they must not necessarily run in strict sequence, e.g., the linting tasks, (most) unit tests, or asset precompilation can run in parallel, while the unit tests for the web application and the integration tests cannot start before asset precompilation is finished. A complete build currently takes 20 to 50 minutes, depending on the size of the build queue.

A build in the CI pipeline fails, if one of the jobs fails.

Listing 3.4: Integration test scenario: enroll to a course

```

1 Feature: Enroll in Course
2   In order to fully participate (read and write) in a course
3   As a user
4   I want to enroll as a student in a course
5
6   Background:
7     Given an active course was created
8
9   Scenario: Enroll as logged in user
10    Given I am a confirmed user
11    And I am logged in

```

⁸⁶<https://rspec.info> ; last access: 02.09.2022.

3 Technology and Architecture

```
12     And I am on the course list
13     When I enroll in the course
14     Then I am enrolled in the course
15     And I am on the course detail page
16     And I receive a course welcome mail
17
18     Scenario: Enroll as not logged in user
19     Given I am a confirmed user
20     And I am on the course list
21     When I enroll in the course
22     And I submit my login credentials
23     Then I am enrolled in the course
24     And I am on the course detail page
25     And I receive a course welcome mail
```

3.2.3.3 Packaging

The components of the HPI MOOC platform (web app and services) are built as Debian packages. The package description is part of each component, including

- package dependencies,
- target directories for application, config, or binaries, and
- systemd service definitions.

This allows also to define different systemd services that should be started for each component, e.g., the application server and one or more processes for background job processing.

Packaging is also part of the Continuous Integration pipeline.

3.2.3.4 Deployment

When a CI pipeline results in a “green build”, i.e., all jobs of the pipeline have succeeded, the build can be deployed – either to *all* production platforms, or to selected staging or testing instances. For this purpose, there is a *Release* job on the CI server, which basically pushes the Debian packages from a build to an internal Debian repository server.

The salt master for the platform instances then runs periodical deployment jobs. If those find new package versions in the repository, they will install those on the respective instance and run all defined post-processing tasks defined in the deployment states, e.g., the database migration as described in Section 3.2.2.2. When deployment on a virtual machine is finished, this is reported to a Grafana dashboard, as well as in the team chat.

For user/customer facing changes, release notes are sent out by the developer who took responsibility for the deployment. To streamline this process, there is a convention (supported by a template for merge request descriptions in GitLab) to already note down the release notes entry for each merge request (MR). The developer scans the MR descriptions for release note lines, compiles those in a text file and tags them with interested parties. A job on the CI server cares for sending out the notes to the configured recipients (see Figure 3.8).

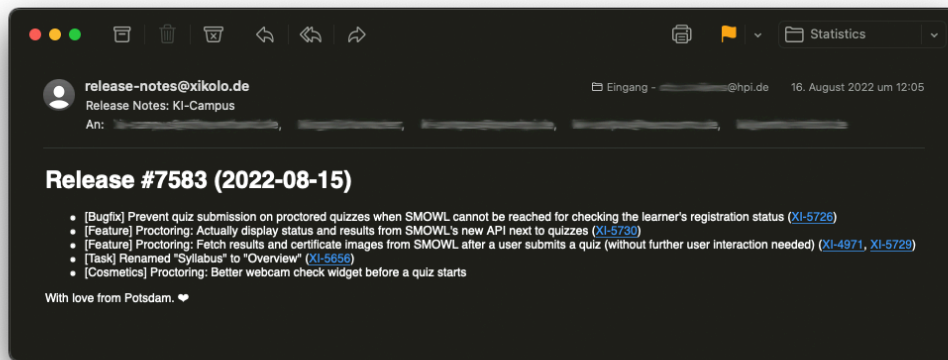


Figure 3.8: Release notes sent after deployment

3.2.4 Monitoring and Alerting

To monitor the infrastructure services, the availability of the applications, and receive alerts in case of issues or application errors, additional tools are used, some of them self-hosted and developed by the openHPI team.

3.2.4.1 Icinga

Icinga is a monitoring tool for infrastructure components, such as the PostgreSQL, Redis, and RabbitMQ clusters or load balancers described in Section 3.2.1 and the availability of peripheral services, e.g., CodeOcean (see Section 2.1.3.4). It allows defining custom health checks that are run regularly and notify specific user groups based on certain states and possible failures. Icinga is also integrated with the team chat to receive timely alerts.

3.2.4.2 Grafana

With Grafana, a wide variety of metrics can be queried and visualized, e.g., on custom dashboards, which in turn enables exploration of data and logs of the application and infrastructure components. For example, PostgreSQL connections, CPU usage, platform load and performance, RabbitMQ queue processing, and background job execution can be monitored. Grafana builds upon InfluxDB (see Section 3.2.1) and *Telegraf*,⁸⁷ an agent for collecting and aggregating metrics sent by the application, for which a telegraf extension⁸⁸ developed by the openHPI team is used. It collects request events from different sources like the application itself but also integrates with Sidekiq.

⁸⁷<https://github.com/influxdata/telegraf>; last access: 02.09.2022.

⁸⁸<https://github.com/jgraichen/telegraf-ruby>; last access: 02.09.2022.

3.2.4.3 Sentry

On application level, *Sentry*⁸⁹ is a tool for tracking errors occurring in the application, both visible to the user and, for example, for background job or message queue processing. Every error, e.g., HTTP errors like 404 and 500, is reported with its stack trace, grouped with similar occurrences and enriched with other metadata, such as the respective platform where the error was raised or the user client used. With this information, efficient debugging of issues for the production applications is possible.

3.2.4.4 Mnemosyne

To further monitor the application and specifically the interaction between different distributed services (see Section 3.3.2), *Mnemosyne* has been developed by the openHPI team. It enables detailed exploration and analyses of each individual request made to the application. For this, the tool collects full application traces, including cross-application requests for distributed applications, enhanced with relevant metadata to analyze request execution (e.g., the request duration) as shown in Figure 3.9. These traces can be application requests, database queries, job processing, or rendering of pages. A full list of supported so-called *probes* can be found in the documentation of the tool. Additionally, *Mnemosyne* shows stack traces for errors (see Figure 3.10).

3.2.4.5 Updown.io

Platform availability in general is monitored with *Updown.io*,⁹⁰ which frequently checks whether the platforms can be reached from the internet. It provides uptime statistics as shown in Figure 3.11. This tool is also integrated with the team chat to receive timely alerts in case of (partial) platform unavailability.

3.3 Application and Architecture

In the following, the technical foundation of the HPI MOOC platform, its architecture, and design decisions are presented.

3.3.1 Technology Stack

The platform is being developed using Ruby on Rails.⁹¹ Known for its ease of use and rapid prototyping capabilities, the Ruby framework allows for less development times and quicker growth. According to its MVC pattern, which focuses on separating application logic from view logic, data is typically requested in controllers, decorated with presenters, and displayed in views, using the lightweight

⁸⁹<https://sentry.io>; last access: 02.09.2022.

⁹⁰<https://updown.io>; last access: 02.09.2022.

⁹¹<https://rubyonrails.org>; last access: 02.09.2022.

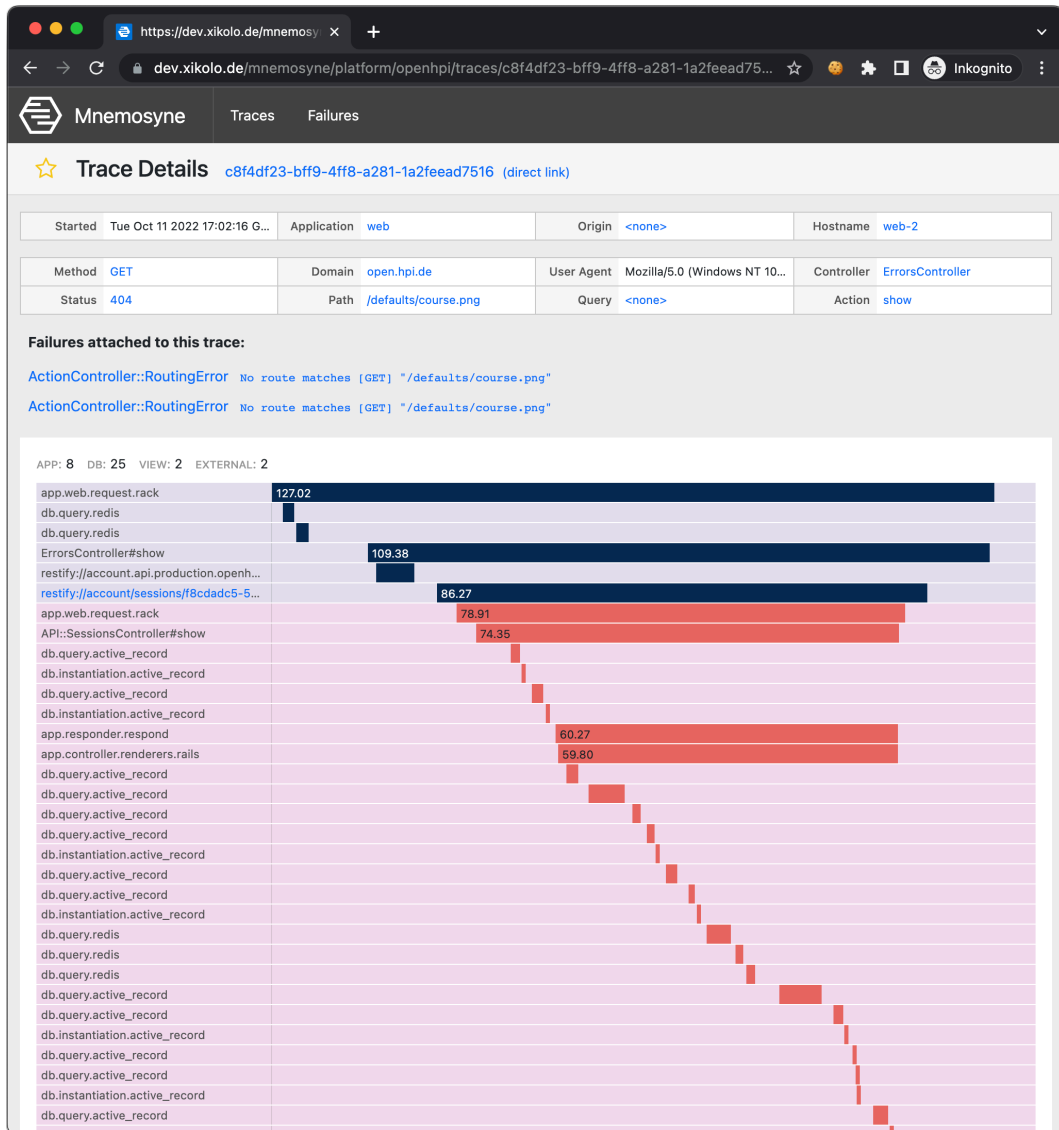


Figure 3.9: Sample Mnemosyne trace for an application request

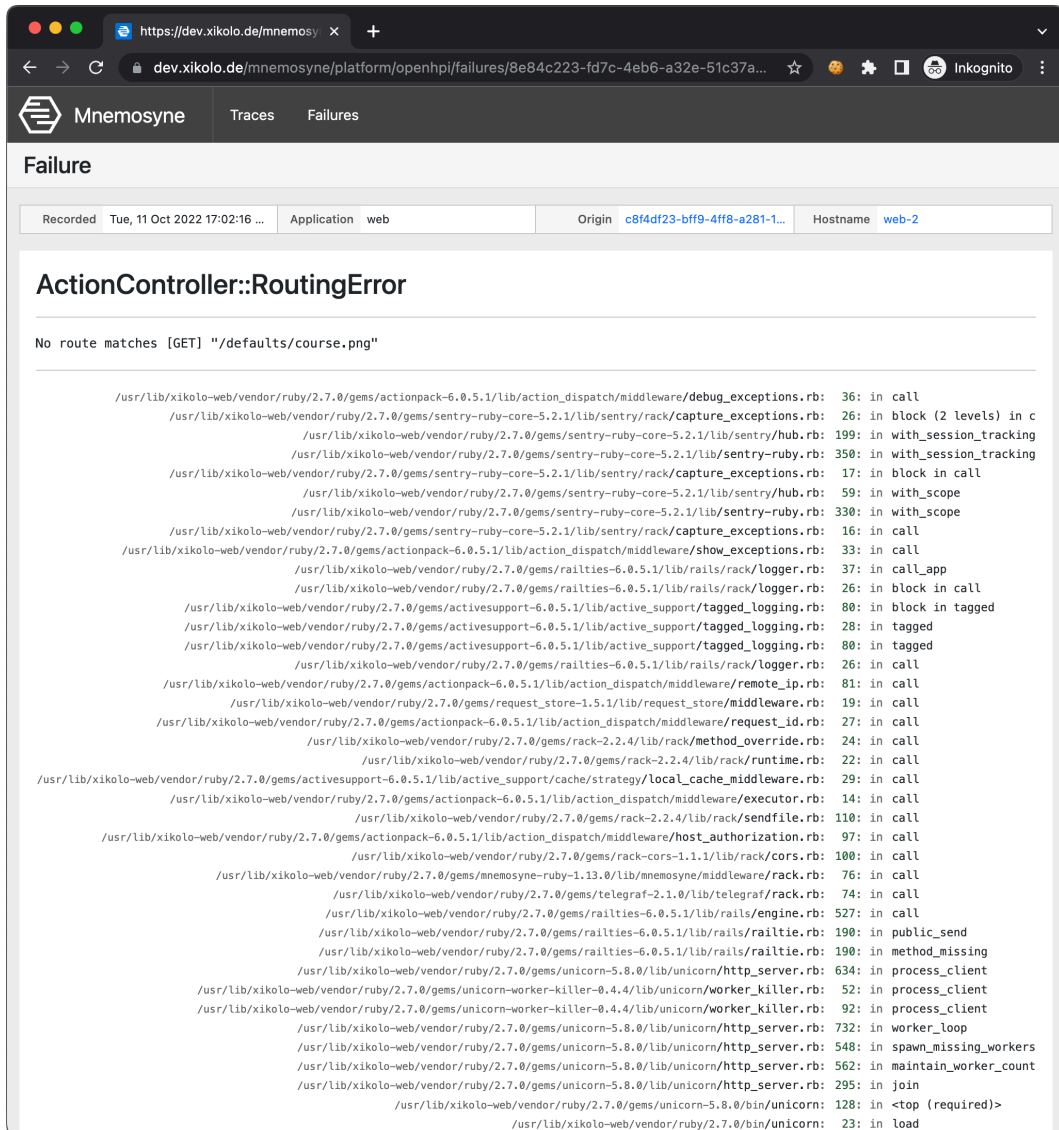


Figure 3.10: Sample Mnemosyne failure with stack trace

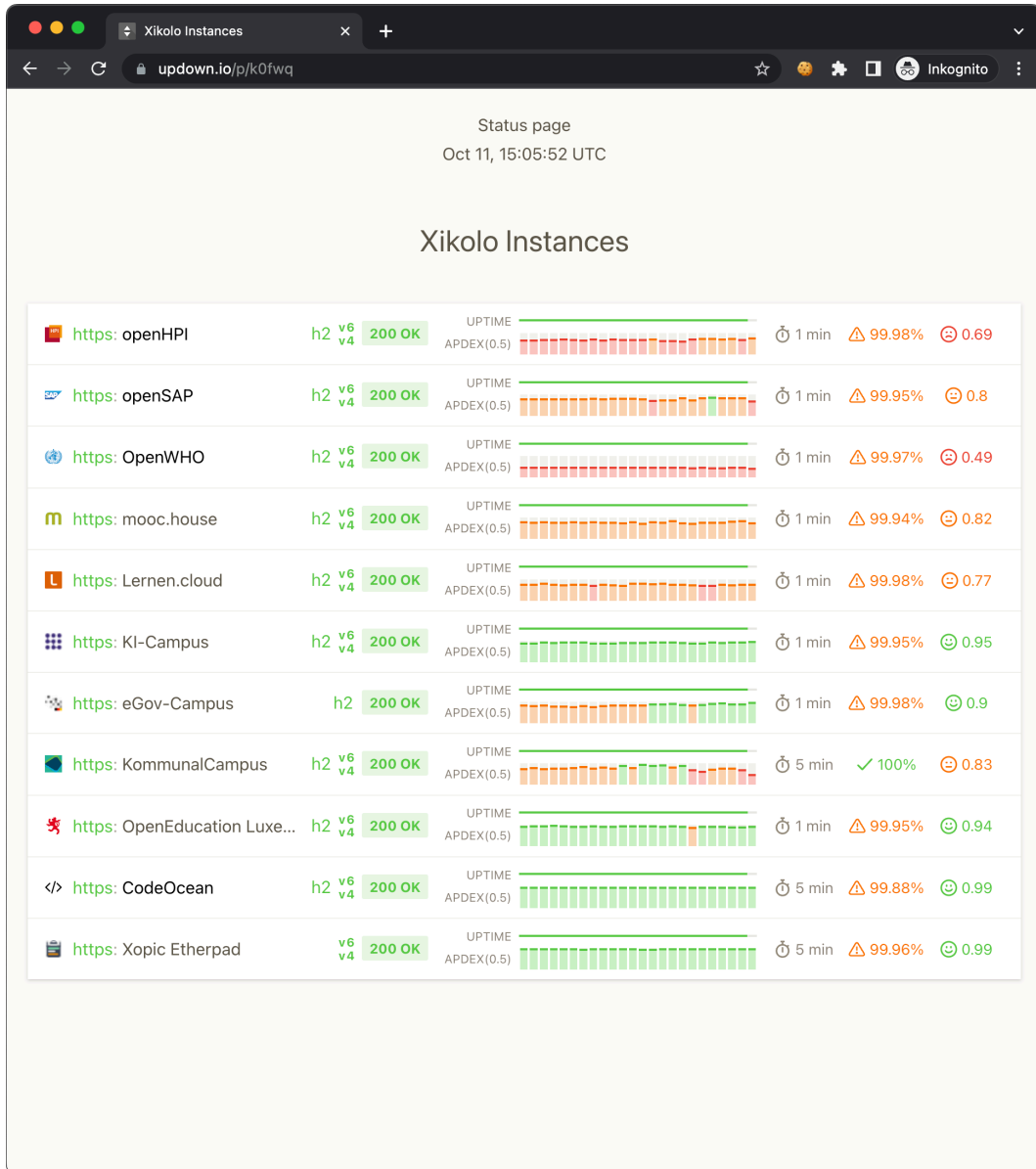


Figure 3.11: Updown.io dashboard showing the uptime statistics for the different platforms

Slim templating engine.⁹² To manage assets in the application (asset pipeline, e.g., to minify or compress JavaScript and CSS assets), both Sprockets⁹³ and webpack⁹⁴ have been integrated. For example, this allows using the CSS extension language SCSS and SASS preprocessing capabilities,⁹⁵ e.g., enabling advanced CSS features, such as mixins and functions, and better maintainability. An important use case in this context is branding of the application.

3.3.2 Service-oriented Architecture

The initial version of the HPI MOOC platform was based on the open-source LMS Canvas to quickly experiment and test the platform with first courses in 2012, which was a pioneering work in Europe[28].

Based on these first insights, a custom-tailored platform has been developed from scratch, which better fits the paradigm of MOOCs, with thousands of learners in a single course and social activity, as well as a better scalability and performance. Therefore, the current platform has been implemented based on the principles of a service-oriented architecture (SOA) with logically separated functionality in individual services[62] as shown in Figure 3.12. All services encapsulate a part of the business logic of the overall application and care for the persistence of their data in own databases, typically a PostgreSQL database. This enables loose coupling between services and facilitates scalability as individual services can be deployed and scaled independently by running multiple instances of a service (depending on the system workload).

While the web service provides the UI components for the platform and is the interface for the clients, dedicated services implement the logic for certain parts of the domain. For example, the account service is responsible for managing user accounts whereas the course service contains all information regarding courses, their structure, and course enrollments. The services can communicate with each other synchronously through RESTful Hypertext Transfer Protocol (HTTP) interfaces, or asynchronously by publishing events on a shared message queue.

Currently, three clients are available for the platform: a web client served by the web service and two native mobile clients for Android and iOS, which use the platform's application programming interface (API).

In total, there are 17 services providing the backend for the application:

Account The account service is responsible for handling user accounts, authentication credentials and sessions. It provides several authentication mechanisms to the clients. The service further manages permissions, feature flags and groups, including group membership and grants. Permissions, groups, memberships and grants can be assigned within a hierarchical context tree, allowing to give permissions, e.g., for a specific course only.

⁹²<https://github.com/slim-template/slim>; last access: 02.09.2022.

⁹³<https://github.com/rails/sprockets>; last access: 02.09.2022.

⁹⁴<https://webpack.js.org>; last access: 02.09.2022.

⁹⁵<https://sass-lang.com>; last access: 02.09.2022.

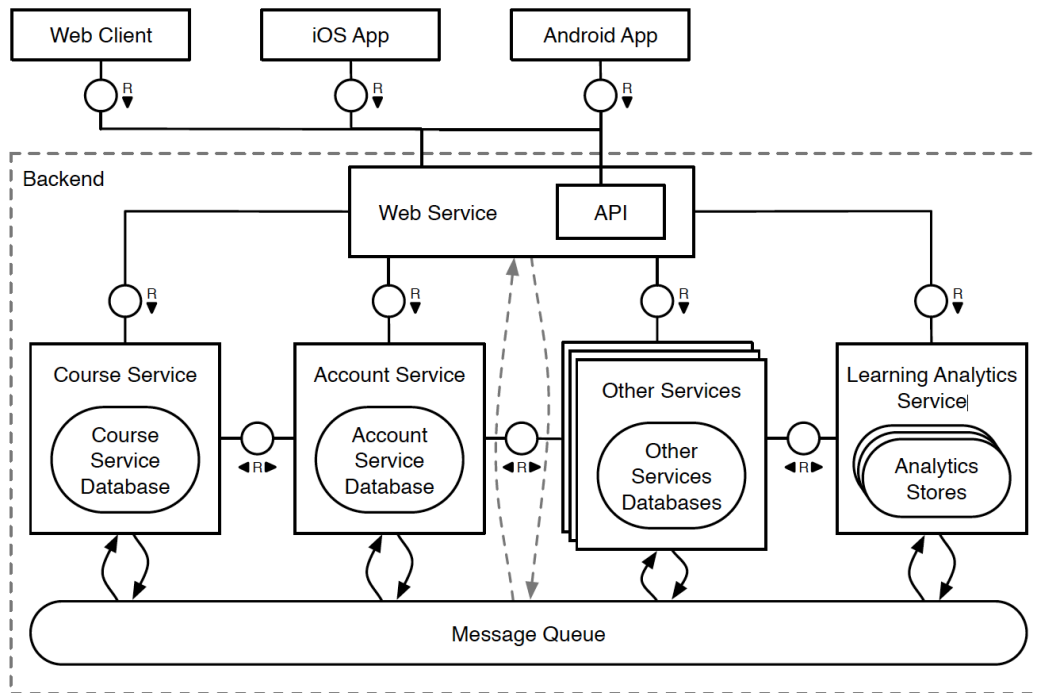


Figure 3.12: Service-oriented platform architecture

Course The course service holds information and logic related to courses and their metadata and structure (sections, items, in particular: also the content of rich-text items), channels, course classification. It also manages learner enrollments, progress (i.e., item visits) in courses and the learning evaluation, containing the overall score of a learner in a particular course. The course service also manages the visibility of learning content based on user roles and course state, delivers filtered course collections for all course listings (e.g., on the overall course list, channel landing pages, or the learner dashboard).

Quiz The quiz service is responsible for storing the quiz data – meta-data, such as quiz type, deadlines, and the actual questions, answers, explanations, and points – and the learners submissions.

Video The video service stores video (meta) data, including stream information, subtitles, and thumbnails, and implements the processes for syncing videos and related subtitles with video provider platforms (Kaltura, Vimeo) and extracting audio to be offered for downloads. It also uses other services for subtitle translation and thumbnail generation.

LTI The LTI service integrates external providers adhering to the LTI standard. It further manages the LTI exercises and the users' submissions in order to report achieved scores to the course service.

Peer Assessment The peer assessment service handles all peer assessment-related data: configuration (e.g., deadlines, solo or team assessments), task description, grading rubrics and also the learners submissions.

Pinboard The pinboard service persists all data required for the discussion forums in a course. This includes the posts in forums, tags that can be added to categorize posts, the watched status and votes by users, the users' subscriptions to posts, and also admin-specific information, such as abuse reports and forum statistics.

CollabSpace The collabSpace service keeps track of the members, the attached collaboration tools (only references to the instances of the tools that have been used within a CollabSpace, but not the actual data that has been created within these tools). In case of a Team Peer Assessment, the collabSpace service passes the member information on to the peer assessment service.

Certificate The certificate service is responsible for creating performance records (CoP, RoA, Certificate) that can be downloaded by the user. In addition, it provides functionality for baking Open Badges and allows their verification. For both performance records and badges, templates are stored and managed per course.

Gamification The gamification service contains logic for awarding experience points, badges, and user levels upon user interaction with platform content, e.g., activity in the pinboard or for attending or completing courses, sections, and quizzes.

Learning Objectives The learning objectives service stores a representation of course items and their assignment to (content-based and manually created) objectives to be achieved within a course. The objectives can focus on course completion or mastering specific topics, for which items can be categorized as knowledge acquisition (typically videos and texts) or examination (typically quizzes or other exercises). It also stores the user's selection of these objectives for a particular course and the corresponding progress towards achieving them.

Time Effort To allow estimating the required time effort to complete a set of items or an entire course, the time effort service contains a representation of course items and their expected time effort. To determine the time effort, different processors apply methods based on the item's content type and combine metrics, such as the word count for texts, to generate an overall estimation for an item. Additional parameters, e.g., historical user, data can be incorporated for these estimates.

Helpdesk The helpdesk service tracks reported customer support tickets and forwards them to external helpdesk systems. Additionally, it handles and persists data for polls, static content pages and platform-wide alerts.

News The news service stores platform-wide (global) as well as course announcements that can be published in a blog and sent out to users via email. It also manages so-called targeted announcements that can be sent out to a specific user group, e.g., individual users, course students, pre-configured groups with access restrictions or other manually created groups based on custom criteria. With this, it is possible to publish announcements to a group of users across multiple courses, e.g., as part of a marketing campaign.

Notification The notification service takes care of sending out any kind of emails to the user. This includes course or global announcements (see the news service), account-specific emails, such as welcome emails, account confirmation and password reset emails, pinboard notifications, or course statistics for course admins. It also stores a mail log, including statistics for emails opened by the users.

Grouping This service provides the basis for A/B tests, taking care of the user assignment to test groups and evaluating the results based on metrics. For the latter, it makes use of the Learning Analytics service.

Voucher The voucher service is responsible for activating vouchers that can be redeemed by users, enabling specific course or platform features, such as proctored exams or course reactivation. For example, vouchers can be distributed via an external shop, for which an API is provided to generate vouchers.

3.3.2.1 Learning Analytics Service

An additional service, decoupled from the main application and serving a different purpose, is the learning analytics service. It is responsible for processing and storing events, e.g., triggered by user events in the web service, which can then be used to perform analyses of the learners' behavior on the platform. The events are published by the individual services to dedicated message queues to which the learning analytics service subscribes. Both an Elasticsearch and PostgreSQL are available as analytics stores and can be queried. Therefore, different metrics were implemented, which gather the interaction data and aggregate it for different purposes, such as the use in dashboards or for course reports.

3.3.2.2 Synchronous Inter-service Communication

As briefly described before, the communication between services is realized on the basis of HTTP, with each service providing an API based on the Representational State Transfer (REST) paradigm. With frequent inter-service communication between many services in the SOA architecture, large parts of the core service interaction had been implemented with the Acfs⁹⁶ API client library, developed by the openHPI team. This client is optimized for service abstraction, automatic request queuing, and parallel processing, e.g., for many different resources loaded from

⁹⁶<https://github.com/jgraichen/acfs>; last access: 02.09.2022.

the different services. To reduce the complexity of the inter-service communication and in an effort to replace Acfs, the Restify⁹⁷ gem was introduced.

3.3.2.3 Asynchronous Inter-service Communication

Besides, asynchronous communication between services is possible using message queues. For this, RabbitMQ is used as message broker, implementing the AMQP (see Section 3.2). To simplify the configuration of queues and processing of messages, the Messaging Framework Msgr,⁹⁸ which is developed by the openHPI team as well, is utilized as an abstraction layer. Instead of accessing the message queues directly, a service configures routes that define the queues from which messages are consumed. For receiving messages, so-called consumers can be assigned to these routes, which are then invoked by Msgr to process the incoming messages.

3.3.2.4 Background Job Processing

Often, more complex and potentially resource- or time-expensive tasks need to be performed. To minimize the time spent responding to a request and not strain the regular service execution, this can be realized using asynchronous background job processing so that such work happens outside the immediate web request. This may be sending out emails, executing scheduled consistency checks or other regular tasks, or potentially long-running calculations, such as report generation. Another frequent use case is fetching and processing data from an external API, as this relies on unreliable network. Many services utilize background job processing.

The applications make use of the Active Job framework,⁹⁹ which provides a standardized interface that allows declaring jobs and running them on different queuing systems. For this, Sidekiq is used as the queuing system, which in turn relies on Redis to store the queued events. While using Active Job works well for scheduling a small number of short-running jobs, in other scenarios, Sidekiq is also used directly, e.g., for using its bulk enqueueing functionality. For business-critical tasks requiring a high level of resilience and reliability, where it needs to be ensured that jobs are not lost and need to be executed at least once, delayed¹⁰⁰ has recently been introduced. It uses the PostgreSQL database as a persistent queuing storage.

3.3.2.5 Ember Frontend and Public API

In the course of improving the user experience with the rise of single page application (SPA), some parts of the platform have been ported to Ember.js,¹⁰¹ which is a client-side JavaScript web framework. In its core, it utilizes a version of the templating engine Handlebars.js¹⁰² that allows updating the rendered HTML when changes occur to the underlying data model without reloading the page.

⁹⁷<https://github.com/jgraichen/restify>; last access: 02.09.2022.

⁹⁸<https://github.com/jgraichen/msgr>; last access: 02.09.2022.

⁹⁹https://edgeguides.rubyonrails.org/active_job_basics.html; last access: 02.09.2022.

¹⁰⁰<https://github.com/Betterment/delayed>; last access: 02.09.2022.

¹⁰¹<https://www.emberjs.com>; last access: 02.09.2022.

¹⁰²<http://handlebarsjs.com>; last access: 02.09.2022.

For retrieving data, the Ember application uses the API provided by the web service (see Figure 3.12). This is required since the different services are not publicly accessible from a client. The API is built in Ruby using the Grape¹⁰³ framework and follows the JSON:API specification.¹⁰⁴ Thus, it seamlessly integrates with Ember, which relies on JSON:API by default.

However, this experiment was abandoned, and the application widely adopts server-side rendering. The remaining parts of the Ember application are converted accordingly. The public API is still in use for the native mobile apps.

3.3.3 Design System and Components

The openHPI platform is currently undergoing a complete rebrush of the landing page¹⁰⁵ (see Figure 3.13 on page 110) and the course list.¹⁰⁶ The development team uses this opportunity to start with the introduction of a comprehensive design system for the platform family that shall grow to a modular toolbox allowing to construct the complete user interface out of components from this modular system.

3.3.3.1 Components

Component based systems are widely used and accepted as the “go-to” approach for modern frontends. Client-side web frameworks like React, Vue and Angular have popularized it to build encapsulated components and compose them to form a more complex user interface. Components can be build independently as well as act independently on the UI by managing their own state. Re-usability of components lowers maintenance costs and lead to better scalability of the frontend.

After starting with a home-grown solution, we later switched to the matured and now widespread *ViewComponent*¹⁰⁷ library by GitHub. By adapting this framework, we kept the server side rendered approach while gaining benefits from a component based approach.

The previewing tool *LookBook*¹⁰⁸ (see Figure 3.14, p. 112) contributes to a more speedy development and enhances developer experience.

Standardizing the user interface is an ongoing project. Almost all existing components have already been converted into ViewComponents, additional components are built and refined along the way, e.g., during the rebrush of the landing page and course list.

There are currently 76 components, including the following, which are described in more detail:

Navigation The main navigation on top of every page, containing the logo, link items and dropdowns.

¹⁰³<http://www.ruby-grape.org/>; last access: 02.09.2022.

¹⁰⁴<https://jsonapi.org/>; last access: 02.09.2022.

¹⁰⁵<https://open.hpi.de/>; last access: 02.09.2022.

¹⁰⁶<https://open.hpi.de/courses/>; last access: 02.09.2022.

¹⁰⁷https://github.com/github/view_component; last access: 02.09.2022.

¹⁰⁸<https://github.com/allmarkedup/lookbook>; last access: 02.09.2022.

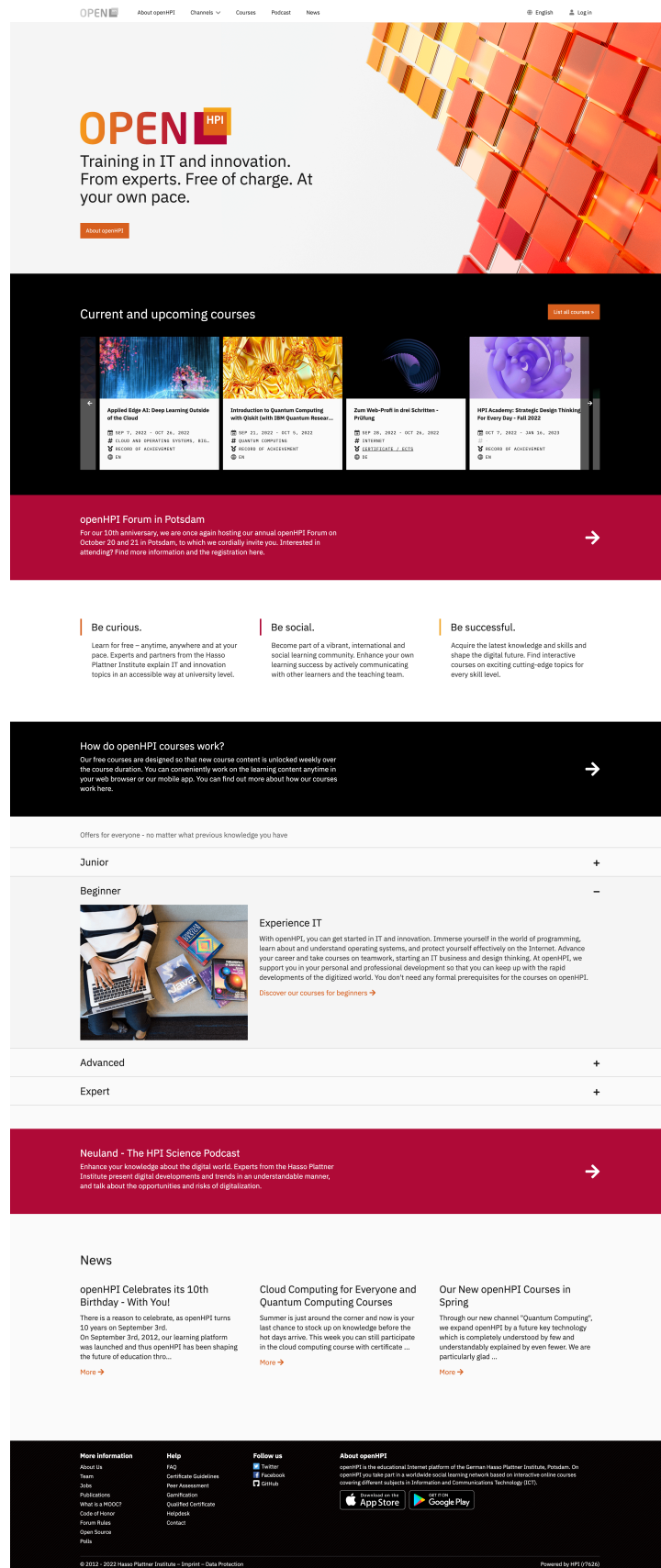


Figure 3.13: New openHPI landing page; more components will be added in the future

Footer The bottom part of every page, containing utility links, the platform information, and social media or newsletter links if enabled.

Callout An eye-catching text box that can be configured with different types to emphasize the content, typically used for success notices, hints, or error messages.

CourseCard This card contains information about a course, such as title, visual, dates, keywords, and action buttons. It supports different sizes, e.g., a compact version with only selected information to be presented.

Accordion A component that contains arbitrary items inside expandable containers.

HeadHero A full width banner, typically with a background image and promoting content.

Tile A container that can hold text and image content with an optional decoration.

TilesGroup Wrapper component containing multiple Tile components.

Slider The component can hold any items arranged in a row with scroll indication and buttons to scroll. On the homepage, this is used for a seamless experience for exploring offered (e.g., featured) courses.

Poll Widget that presents a poll to the user, allowing to vote and present the poll results afterward.

For example, the new landing page depicted in Figure 3.13 is completely built based on components, including *Navigation*, *HeadHero*, a *Slider* holding *CourseCard* components for the list of current and upcoming courses, *TilesGroup* components with differently styled *Tiles* for the claims and news elements, and the *Footer*.

3.3.4 Back to Monolith – Insights From 8 Years of Microservice Architecture

As described in Section 3.3.2, the HPI MOOC platform has been developed following the SOA paradigm. Still, separate services encapsulate domain knowledge and are deployed individually (see Section 3.2.3.4). Besides the advantages, e.g., of loose coupling, this approach comes with disadvantages that after eight years of a growing platform and code base outweigh the benefits. Therefore, the platform is currently migrated back to a monolithic application.

3.3.4.1 The Decision for Microservices

For the platform architecture, not only the technical perspective with SOA being the current trend at that time, but also the context in which the system is developed, was relevant. When the openHPI Team started developing an own implementation of the platform from scratch, the focus was on the research aspects of the

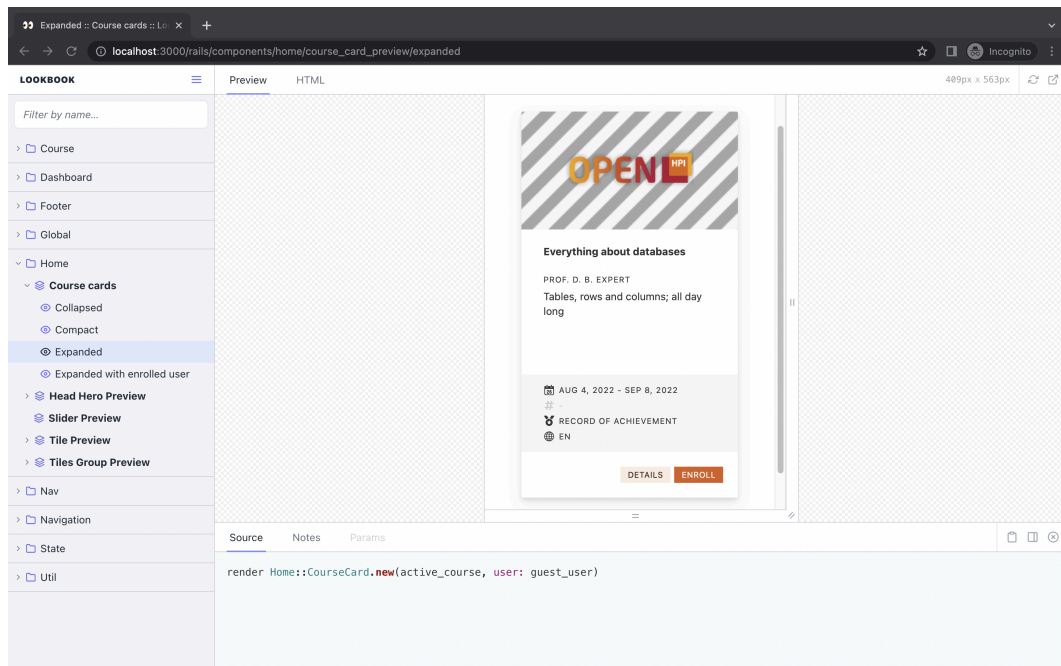


Figure 3.14: LookBook preview of a *CourseCard* component

project. Different people and teams with different backgrounds were working on it, incrementally extending the implementation.

Only few full-time developers have been working on the core platform functionality, supported by researchers as well as bachelor's and master's projects and student assistants due to the university context. To limit the scope of what contributors have to work with, reducing both technical and domain complexity, individual services were easy to set up and could be extended with clearly defined interfaces based on REST APIs.

From a technical perspective, dedicated service by design facilitate separation of concerns. It also enables flexibility and scalability as services can be deployed as well as scaled independently, which was specifically important in the beginning when the number of users was growing rapidly.

In terms of customization, individual services can also be deployed and enabled only when needed. This modular deployment can spare resources for platforms not requiring a specific feature implemented by one of the services. With a small number of platforms with a custom setup, these customization capabilities have a benefit and can be managed adequately.

3.3.4.2 Insights from Operating a Microservice-driven Application

From a development and product perspective, the effort for developing new features within a single service works smoothly. However, developing features that require multiple services to interact, and even the integration with the web service, often increases complexity as indirection is needed through inter-service commu-

nication and exposing data via APIs. This to some extent contradicts the Ruby on Rails way, which focuses on monolithic applications. Also, data migrations and extensions for services need to be planned more carefully as the services are deployed independently.

In addition, communication via network is unreliable and requires diligent error-handling. This increases the effort for mechanisms in terms of application resilience to avoid user-facing errors, and even more importantly, data loss. While most requests between services can be handled quickly, also the performance for loading data needs to be considered particularly. The problem of data synchronization exists alike.

To increase communication performance, requests to backend services should be parallelized as much as possible. This requires to work with (shared) promises, which again introduces complexity. Also, the sequence of requests or the question of which resources might have been loaded (or not) increases complexity and must be carefully considered.

As for data consistency, it's much more effort if application data is split across different databases as typical solutions that ensure consistency, such as foreign key constraints, cannot be applied across service boundaries. Moreover, joining data from different source, which is for example needed for report generation or other more complex features, is not possible – unless data is duplicated in different services which adds more application complexity. This heavily impacts execution and response times or requires additional effort to (asynchronously) aggregate such data.

In the times of a research project with a limited scope and only few platforms, the cost for providing a performant infrastructure and deployment process was manageable. With the platform becoming more mature and being operated in a standardized way as software-as-a-service (SaaS) for many partners, the effort for customization on deployment and infrastructure level increases significantly. At this point, a common setup, configuration, and deployment, e.g., deploying all services for every platform but disabling features at the application level, reduces complexity. However, with individual services, i.e., VMs and even processes, the problem of deployment order dependencies remains. Atomic deployment or rollback mechanisms cannot be applied.

One last aspect is the maintainability of the code base: since all applications (web and services) use the Ruby on Rails framework, they all must be upgraded separately. While this is usually not an issue for patches or minor upgrades, major version upgrades always require a fair amount of refactoring. When the project started, Rails was at version 4. In the meantime, Rails 7 is the latest stable release – which means 3 major version upgrades, and a lot of effort.

3.3.4.3 Migration

The roadmap for the migration back to a monolithic application contains five main steps, which have partially been completed and some of them currently being in progress:

Monorepo Complexity can be reduced for development by integrating all services into a single monorepo (a repository that contains multiple logic projects). With this, developers always check out the entire application, but the individual services are still individual Rails applications. Most importantly, the overhead of synchronizing changes between different repositories is reduced.

Shared application database As shown in Figure 3.12, each service has its own persistence layer. To enable access to required data, streamline data migrations, and introduce consistency constraints on database level, all services require access to a shared database. Also, the web service can be granted access to the database to load data directly without indirection via the services.

Unify service configuration and deployment All service configurations and the deployment process can be aligned. Next to reducing the complexity and the benefit of better understanding and maintainability of the applications, this allows integrating the code bases of the services.

Integrate services The chosen migration path designates the web application as the monolith. All services are moved into the monolithic application one by one. For each service, models are re-created in the monolith, controller actions are refactored as necessary. Existing controllers or workers in the web application that belong to the same business domain as the service being integrated, can now directly access the database (instead of the service via REST) and make full use of the framework capabilities (e.g., the *ActiveModel/ActiveRecord* persistence layer). Even though the integration mainly serves complexity reduction, the task itself is a challenge for each service as it involves domain expertise and raises technical questions. It might be difficult to find all places in the code base where a service is called – but there is tool support for this. Grafana dashboards can visualize the usage for single endpoints or whole services. Possible side effects of (model) callbacks also need to be considered carefully, as well as the sequence of deployments when applying slices of a service integration.

This migration is an ongoing refactoring project and will probably take until the end of 2023. The consideration of the effort of this refactoring and the other option, keeping the services and put effort into infrastructure and failure resilience, was covered in numerous discussions, and might look as “a step back”. However, the openHPI development team learned based on the insights previously shown, discussed and evaluated options, took a conscious decision, and is convinced that returning to a monolithic application will be worth the effort in the long term.

4 Summary and Outlook

We conclude our report with a summary, a short discussion on MOOC business models, and an outlook on digital education in the 21st century.

4.1 Summary

openHPI is both a successful MOOC platform, contributing to the digital enlightenment of the masses by spreading HPI's knowledge worldwide, and a state-of-the-art learning management system, which many partners use in various contexts. In both aspects, openHPI is a pioneer in the German e-learning landscape. The platform has been preceding other major players in Germany and Europe for years, outlasting its main competitor in Germany, and – not only in terms of numbers – is hard to catch up with for other German players.

The platform is optimized for a specific use case (MOOCs); however, it can be used for a wide variety of use cases, from small face-to-face courses to MOOCs. In many conversations, we are fluttered with compliments about the platform's look and feel. However, it is not the platform's nice appearance that make the difference; it is its scalability, the many features optimized for this use case, and the constant ongoing research of our development team and our students. If scalability is not a concern, other choices might be a better fit, but within the targeted use case, the platform is cutting edge and doesn't have to hide behind competitors with much larger team sizes.

We have presented some cutting-edge features, such as our work in learning analytics, social elements including peer assessment and teamwork, auto-graded programming exercises, dual-stream videos, chatbots, mobile learning, individual learning paths, and many more. As far as our business model allows this, we have given very detailed insights on the set-up of the system and its technological components.

Technology-wise, the HPI MOOC platform can keep up with its international competitors. On the other hand, we have to be careful not to get caught in the trap of complacency. The world of e-learning is constantly moving, as research and the world of technology are in general. openHPI is facing the same challenges as many other companies: finding properly trained experts and developers to improve the platform and create new high-quality courses is challenging.

Although the number of MOOCs and learners is continuously increasing on openHPI and even more on many of the larger HPI MOOC platforms, we also see several danger signals. The enthusiasm of the early days, when many professors wanted to offer MOOCs as this was an international hype, has faded. Nowadays, it

takes more effort to convince professors to invest the effort that creating a MOOC requires. Particularly those who have learned that developing a good MOOC requires significant time and effort often are reluctant to reinvest in a new course. Therewith, the platform's requirements to offer a MOOC have become less strict over the last decade. If a course concept fits the general program, external professors can offer courses on openHPI. However, supporting external course instructors also requires more effort from the course production team.

In addition, there is still no sound business model for MOOC platforms except for government or foundation funding. The other large German MOOC platform filed for bankruptcy in 2016 and later was sold to *Holtzbrinck* and then *Springer publishers* [20, 21]. *Coursera* and *FutureLearn* – two of the largest MOOC platforms worldwide – have been sold to *SEEK*, an Australian job marketplace, but still aren't working profitable, even though their revenue numbers sound impressive [7]. Since Coursera went public, "their stock price has been steadily falling even though revenue has been increasing" [5]. FutureLearn's financial situation is also tricky [8]. *edX* recently was sold to *2U*, a commercial online university – shortly after, *2U*'s stock price lost value [5]. *MiradaX* has been for sale since 2020 when *Telefónica* decided to focus on their main business again [6]. Our business model is to operate the platform as a service for our partners, which does not generate fortunes but covers the costs.

4.2 Outlook – Digital Education in the 21st Century

Online learning platforms have seen a tremendous increase in enrollments for the past two years in light of the COVID-19 pandemic but also due to the continuously increasing demand for scalable learning formats – to build and retain an agile workforce for businesses or stay competitive in the labor market for individuals. However, a continuous gradual growth could also be observed before 2020, underlining the general trend of moving education more and more online.

Approximately ten years ago, higher education institutions – especially those in Anglo-Saxon countries – started to expand their offers and, with that, have enlarged their target group. European universities are catching up by offering full-time and part-time blended or online programs and certificates. Still, scalable, student-centered, engaging online programs embracing the latest digital learning technology with university-level teaching are rare – not only in Europe.

Nowadays, not only the delivery format of education is transforming, but also the approaches to transferring required skills for the 21st century differ from classical forms of education. The student-centered approach to learning is one way that underlines these differences. It is composed of four main areas (see Figure 4.1):

1. personalized learning, including the customization of content and formats,
2. student-owned learning, including self-paced learning activities,

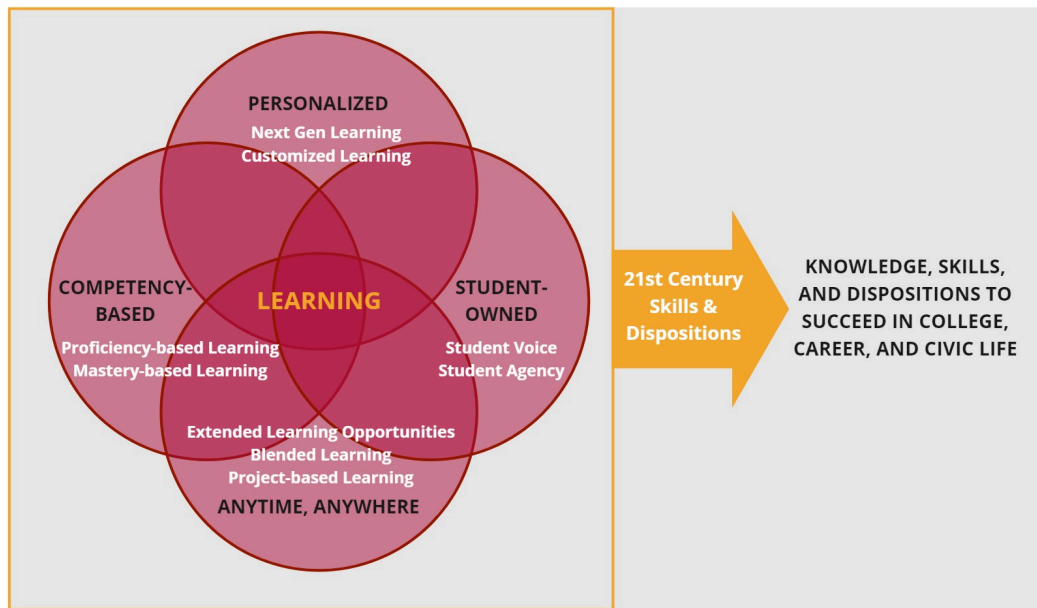


Figure 4.1: The student-centered learning approach in reference to Glowa & Goodell (2016) [13]

3. anytime-anywhere-learning, including the increased flexibility and device independence as well as
4. competency-based learning, focusing on specific skills and capabilities rather than pure knowledge transfer.

These trends show that standard university education curricula are no longer suitable to prepare students for (future) challenges in business and society. Therefore, the next step in developing higher education is to fully integrate learning with scalable online platforms using the latest technology to make it accessible for as many students as possible – independent of their location or cultural background. Online education platforms play a crucial role in this development. The e-learning market is expected to grow by more than 20% in the upcoming years. Coming from a market size of around \$ 250 billion in 2020, the projected market size will be \$ 1 trillion in 2027.

Especially, the combination of MOOCs, short courses, and certificates with degree programs is an emerging trend on the market. Millions of users get attracted to online education platforms through open courses. Building on positive experience, there are possibilities of “upselling” from free courses to specific certificates and degree programs. Education Platforms with a combination of MOOCs and other offerings, including degree programs, seem to be the most prominent model in the future. Therefore, our vision is to revolutionize and democratize university education, making it accessible to everyone, everywhere, and at any time. Supported by state-of-the-art technology and a dedication to student-centered challenge-based online learning, a new type of online university with the openHPI platform as the

4 Summary and Outlook

basis provides the knowledge and skills that empower people to understand digital transformation and digital technologies. These people will shape the digital age around the world.

Bibliography

- [1] J. Bethge, S. Serth, T. Staubitz, T. Wuttke, O. Nordemann, P.-P. Das, and C. Meinel. “TransPipe – A Pipeline for Automated Transcription and Translation of Videos”. In: *Proceedings of the 7th European MOOC Stakeholder Summit (EMOOCs 2021)*. Potsdam, Germany: Universitätsverlag Potsdam, June 2021, pages 79–94. DOI: 10.25932/publishup-51694.
- [2] M. Bothe and C. Meinel. “On the Potential of Automated Downloads for MOOC Content on Mobile Devices”. In: *2020 IEEE Learning With MOOCs (LWMOOCs)*. Antigua Guatemala, Guatemala: IEEE, Sept. 2020, pages 58–63. ISBN: 978-1-72819-728-9. DOI: 10.1109/LWMOOCs50143.2020.9234338.
- [3] M. Bothe, J. Renz, and C. Meinel. “On the Acceptance and Effects of Recapping Self-Test Questions in MOOCs”. In: *2020 IEEE Global Engineering Education Conference (EDUCON)*. Porto, Portugal: IEEE, Apr. 2020, pages 264–272. ISBN: 978-1-72810-930-5. DOI: 10.1109/EDUCON45650.2020.9125145.
- [4] S. Chujfi, H. Traifeh, T. Staubitz, R. Refaie, and C. Meinel. “Exploring Collaboration and Assessment of Digital Remote Teams in Online Training Environments”. In: *Workgroups eAssessment: Planning, Implementing and Analysing Frameworks*. Edited by R. Babo, N. Dey, and A. S. Ashour. Singapore: Springer Singapore, 2021, pages 27–53. DOI: 10.1007/978-981-15-9908-8_2.
- [5] Dhawal Shah. *A Decade of MOOCs: A Review of MOOC Stats and Trends in 2021*. <https://www.classcentral.com/report/moocs-stats-and-trends-2021/>. [Online; accessed 24-August-2022]. 2021.
- [6] Dhawal Shah. *Latin America’s Largest MOOC Provider is Up for Sale*. <https://www.classcentral.com/report/miriadax-up-for-sale/>. [Online; accessed 24-August-2022]. 2021.
- [7] Dhawal Shah. *Online Degrees Slowdown: A Review of MOOC Stats and Trends in 2019*. <https://www.classcentral.com/report/moocs-stats-and-trends-2019/>. [Online; accessed 24-August-2022]. 2019.
- [8] Donald Clark. *Futurelearn may not be a going concern*. <http://donaldclarkplanb.blogspot.com/2022/07/futurelearn-may-not-be-going-concern.html>. [Online; accessed 24-August-2022]. 2022.
- [9] M. Elhayany, R.-R. Nair, T. Staubitz, and C. Meinel. “A Study about Future Prospects of JupyterHub in MOOCs”. In: *Proceedings of the Ninth ACM Conference on Learning @ Scale. L@S ’22*. New York City, NY, USA: Association for Computing Machinery, 2022, pages 275–279. DOI: 10.1145/3491140.3529537.

Bibliography

- [10] M. Fowler. *Ship / Show / Ask – a modern branching strategy*. <https://martinfowler.com/articles/ship-show-ask.html>. [Online; accessed 23-August-2022]. 2021.
- [11] C. Friedl, T. Staubitz, and D. Jansen. “Flexible, Self-Directed and Bottom-Up: Are Employees Overtaking Their Human Resource Departments with MOOCs?” In: *2018 Learning With MOOCS (LWMOOCS)*. Sept. 2018, pages 66–69. DOI: 10.1109/LWMOOCS.2018.8534616.
- [12] D. Gamage, T. Staubitz, and M. Whiting. “Peer assessment in MOOCs: Systematic literature review”. In: *Distance Education* 42.2 (2021), pages 268–289. DOI: 10.1080/01587919.2021.1911626. eprint: <https://doi.org/10.1080/01587919.2021.1911626>.
- [13] L. Glowa and J. Goodell. *Student-Centered Learning: Functional Requirements for Integrated Systems to Optimize Learning*. <https://files.eric.ed.gov/fulltext/ED567875.pdf>. [Online; accessed 31-August-2022]. Vienna, VA., USA, 2016.
- [14] C. Hagedorn, M. Haubold, J. Renz, and C. Meinel. “Identifying Requirements for a Tool to Support Collaborative Gameful Learning in Scalable E-Learning Environments”. In: *Proceedings of European Conference on Games Based Learning (ECGBL2018)*. 2018.
- [15] C. Hagedorn, J. Renz, and C. Meinel. “Introducing Digital Game-Based Learning in MOOCs: What do the learners want and need?” In: *Proceedings of IEEE Global Engineering Education Conference (Educon2017)*. 2017. DOI: 10.1109/EDUCON.2017.7942987.
- [16] C. Hagedorn, S. Serth, and C. Meinel. “Breaking the Ice? How to Foster the Sense of Community in MOOCs”. In: *2022 International Conference on Advanced Learning Technologies (ICALT)*. Bucharest, Romania: IEEE, July 2022, pages 22–26. ISBN: 978-1-66549-519-6. DOI: 10.1109/ICALT55010.2022.00013.
- [17] C. Hagedorn, S. Serth, and C. Meinel. “The Mysterious Adventures of Detective Duke: How Storified Programming MOOCs Support Learners in Achieving their Learning Goals”. In: *Frontiers in Education: Digital Learning Innovations (Manuscript in review)*. 2022.
- [18] C. Hagedorn, T. Staubitz, R. Teusner, and C. Meinel. “Design and First Insights of a Case Study on Storified Programming MOOCs”. In: *2019 IEEE International Conference on Engineering, Technology and Education (TALE)*. 2019, pages 1–8. DOI: 10.1109/TALE48000.2019.9225861.
- [19] M. Hanson. *College Dropout Rates*. <https://educationdata.org/college-dropout-rates>. [Online; accessed 31-August-2022]. 2022.
- [20] Iversity. *iversity reboots with Holtzbrinck Digital*. <https://iversity.org/en/pages/iversity-reboots-with-holtzbrinck-digital>. [Online; accessed 11-August-2022]. 2016.

- [21] Iversity. *Nach Antrag auf Eröffnung des Insolvenzverfahrens: iversity plant Neustart mit neuen Gesellschaftern*. <https://iversity.org/en/pages/insolvenz>. [Online; accessed 24-August-2022]. 2016.
- [22] C. John, T. Staubitz, and C. Meinel. "Took a MOOC. Got a Certificate. What now?" In: *Proceedings of FiE 2019*. 2019 IEEE Frontiers in Education Conference (FIE). Cincinatti, OH, US: IEEE, 2019.
- [23] K. Jordan. *MOOC Completion Rates: The Data, The Katy Jordan MOOC Project*. <http://www.katyjordan.com/MOOCproject.html>. [Online; accessed 31-August-2022]. 2015.
- [24] D. Koehler, S. Serth, and C. Meinel. "Consuming Security: Evaluating Podcasts to Promote Online Learning Integrated with Everyday Life". In: *2021 World Engineering Education Forum/Global Engineering Deans Council (WEEF/GEDC)*. Madrid, Spain: IEEE, Nov. 2021, pages 476–481. ISBN: 978-1-66542-488-2. DOI: 10.1109/WEEF/GEDC53299.2021.9657464.
- [25] D. Koehler, S. Serth, H. Steinbeck, and C. Meinel. "Integrating Podcasts into MOOCs: Comparing Effects of Audio- and Video-Based Education for Secondary Content". In: *Educating for a New Future: Making Sense of Technology-Enhanced Learning Adoption (EC-TEL 2022)*. Toulouse, France: Springer, Sept. 2022. DOI: 10.1007/978-3-031-16290-9_10.
- [26] M. v. Löwis, T. Staubitz, R. Teusner, J. Renz, C. Meinel, and S. Tannert. "Scaling youth development training in IT using an xMOOC platform". In: *Frontiers in Education Conference (FIE), 2015*. 32614 2015. IEEE. Oct. 2015, pages 1–9. DOI: 10.1109/FIE.2015.7344145.
- [27] J. Maldonado-Mahauad, J. A. Ruipérez-Valiente, M. Pérez-Sanagustín, M. Jenner, I. Despujol, C. Turró, T. Staubitz, T. Rohloff, G. Montoro, and J. Reich. "Participation of Latin America in MOOCs: Exploring Trends Across Providers". In: *2020 IEEE Learning With MOOCs (LWMOOCs)*. 2020, pages 25–30. DOI: 10.1109/LWMOOCs50143.2020.9234376.
- [28] C. Meinel and C. Willems. *openHPI - The MOOC Offer at Hasso Plattner Institute*. Technical report 80. Potsdam, Germany: Hasso Plattner Institute for Digital Engineering, 2013.
- [29] OpenStack. *OpenStack Architecture Design – OpenStack Logical Architecture*. <https://docs.openstack.org/arch-design/design.html>. [Online; accessed 18-August-2022]. 2018.
- [30] E. S. Raymond. *The cathedral and the bazaar: musings on Linux and open source by an accidental revolutionary*. eng. With a foreword by Bob Young. Beijing; Cambridge; Farnham; Köln; Paris; Sebastopol; Taip: O'Reilly Media, 2001, page 241. ISBN: 0-596-00108-8.

- [31] J. Renz, D. Hoffmann, T. Staubitz, and C. Meinel. "Using A/B Testing in MOOC Environments". In: *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge*. LAK '16. Edinburgh, United Kingdom: Association for Computing Machinery, 2016, pages 304–313. DOI: 10.1145/2883851.2883876.
- [32] T. Rohloff, M. Bothe, J. Renz, and C. Meinel. "Towards a Better Understanding of Mobile Learning in MOOCs". In: *2018 Learning With MOOCS (LWMOOCS)*. IEEE. Madrid: IEEE, Sept. 2018, pages 1–4. ISBN: 978-1-5386-6533-6. DOI: 10.1109/LWMOOCS.2018.8534685.
- [33] T. Rohloff, D. Sauer, and C. Meinel. "On the Acceptance and Usefulness of Personalized Learning Objectives in MOOCs". In: *Proceedings of the Sixth (2019) ACM Conference on Learning @ Scale*. L@S '19. ACM, 2019. DOI: 10.1145/3330430.3333624.
- [34] T. Rohloff, D. Sauer, and C. Meinel. "Student Perception of a Learner Dashboard in MOOCs to Encourage Self-Regulated Learning". In: *2019 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*. 2019. DOI: 10.1109/TALE48000.2019.9225939.
- [35] T. Rohloff, D. Sauer, and C. Meinel. "Students' Achievement of Personalized Learning Objectives in MOOCs". In: *Proceedings of the Seventh ACM Conference on Learning @ Scale*. L@S '20. ACM, 2020. DOI: 10.1145/3386527.3405918.
- [36] J. A. Ruipérez-Valiente, M. Jenner, T. Staubitz, X. Li, T. Rohloff, S. Halawa, C. Turró, Y. Cheng, J. Zhang, I. Despujol, and J. Reich. "Macro MOOC Learning Analytics: Exploring Trends across Global and Regional Providers". In: *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*. LAK '20. Frankfurt, Germany: Association for Computing Machinery, 2020, pages 518–523. DOI: 10.1145/3375462.3375482.
- [37] J. A. Ruipérez-Valiente, T. Staubitz, M. Jenner, S. Halawa, J. Zhang, I. Despujol, J. Maldonado-Mahauad, G. Montoro, M. Pfeffer, T. Rohloff, J. Lane, C. Turro, X. Li, M. Pérez-Sanagustín, and J. Reich. "Large scale analytics of global and regional MOOC providers: Differences in learners & demographics, preferences, and perceptions". In: *Computers & Education* 180 (2022), page 104426. ISSN: 0360-1315. DOI: <https://doi.org/10.1016/j.compedu.2021.104426>.
- [38] S. Serth. "Individual Worksheets with Interactive Programming Exercises within the HPI Schul-Cloud". Master's thesis. Potsdam, Germany: Hasso Plattner Institute, University of Potsdam, May 2019.
- [39] S. Serth, D. Köhler, L. Marschke, F. Auringer, K. Hanff, J.-E. Hellenberg, T. Kantusch, M. Paß, and C. Meinel. "Improving the Scalability and Security of Execution Environments for Auto-Graders in the Context of MOOCs". In: *Proceedings of the Fifth Workshop "Automatische Bewertung von Programmieraufgaben" (ABP 2021)*. Edited by A. Greubel, S. Strickroth, and M. Striewe. Volume 5. Virtual Event, Germany: Gesellschaft für Informatik e.V. (GI), Oct. 2021, pages 3–10. DOI: 10.18420/abp2021-1.

- [40] S. Serth, T. Staubitz, R. Teusner, and C. Meinel. "CodeOcean and CodeHarbor: Auto-Grader and Code Repository". In: *SPLICE 2021 Workshop CS Education Infrastructure for All III: From Ideas to Practice*. Virtual Event, Mar. 2021, page 5.
- [41] S. Serth, R. Teusner, and C. Meinel. "Digitale Arbeitsblätter mit interaktiven Programmieraufgaben im Informatik-Unterricht". In: *Lecture Notes in Informatics (LNI) - Proceedings: DELFI 2020 – Die 18. Fachtagung Bildungstechnologien der Gesellschaft für Informatik e.V.* Edited by R. Zender, D. Ifenthaler, T. Leonhardt, and C. Schumacher. Volume P-308. Lecture Notes in Informatics. Bonn, Germany: Gesellschaft für Informatik e.V. (GI), Sept. 2020, pages 235–246. ISBN: 978-3-88579-702-9.
- [42] S. Serth, R. Teusner, and C. Meinel. "Impact of Contextual Tips for Auto-Gradable Programming Exercises in MOOCs". In: *Proceedings of the Eighth ACM Conference on Learning @ Scale*. Virtual Event, Germany: ACM, June 2021, pages 307–310. ISBN: 978-1-4503-8215-1. DOI: 10.1145/3430895.3460166.
- [43] S. Serth, R. Teusner, J. Renz, and M. Uflacker. "Evaluating Digital Worksheets with Interactive Programming Exercises for K-12 Education". In: *2019 IEEE Frontiers in Education Conference (FIE)*. Cincinnati, OH, USA: IEEE, Oct. 2019, pages 1–9. ISBN: 978-1-72811-746-1. DOI: 10.1109/FIE43999.2019.9028680.
- [44] T. Staubitz. "Gradable team assignments in large scale learning environments". Doctoral Thesis. Universität Potsdam, 2020, page 122. DOI: 10.25932/publishup-47183.
- [45] T. Staubitz, H. Klement, J. Renz, R. Teusner, and C. Meinel. "Towards practical programming exercises and automated assessment in Massive Open Online Courses". In: *2015 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*. Dec. 2015, pages 23–30. DOI: 10.1109/TALE.2015.7386010.
- [46] T. Staubitz, H. Klement, R. Teusner, J. Renz, and C. Meinel. "CodeOcean – A versatile platform for practical programming exercises in online environments". In: *2016 IEEE Global Engineering Education Conference (EDUCON)*. Apr. 2016, pages 314–323. DOI: 10.1109/EDUCON.2016.7474573.
- [47] T. Staubitz and C. Meinel. "A Systematic Quantitative and Qualitative Analysis of Participants – Opinions on Peer Assessment in Surveys and Course Forum Discussions of MOOCs". In: *2020 IEEE Global Engineering Education Conference (EDUCON)*. 2020, pages 962–971. DOI: 10.1109/EDUCON45650.2020.9125089.
- [48] T. Staubitz and C. Meinel. "Collaboration and Teamwork on a MOOC Platform: A Toolset". In: *Proceedings of the Fourth (2017) ACM Conference on Learning @ Scale. L@S '17*. Cambridge, Massachusetts, USA: ACM, 2017, pages 165–168. ISBN: 978-1-4503-4450-0.
- [49] T. Staubitz and C. Meinel. "Collaborative Learning in MOOCs Approaches and Experiments". In: *2018 IEEE Frontiers in Education Conference (FIE)*. Oct. 2018, pages 1–9. DOI: 10.1109/FIE.2018.8659340.

- [50] T. Staubitz and C. Meinel. "Graded Team Assignments in MOOCs: Effects of Team Composition and Further Factors on Team Dropout Rates and Performance". In: *Proceedings of the Sixth (2019) ACM Conference on Learning @ Scale*. L@S '19. Chicago, IL, USA: ACM, 2019, 5:1–5:10. ISBN: 978-1-4503-6804-9. DOI: 10.1145/3330430.3333619.
- [51] T. Staubitz and C. Meinel. "Team Based Assignments in MOOCs: Results and Observations". In: *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*. L@S '18. London, United Kingdom: ACM, 2018, 47:1–47:4. ISBN: 978-1-4503-5886-6. DOI: 10.1145/3231644.3231705.
- [52] T. Staubitz, D. Petrick, M. Bauer, J. Renz, and C. Meinel. "Improving the Peer Assessment Experience on MOOC Platforms". In: *Proceedings of the Third (2016) ACM Conference on Learning @ Scale*. L@S '16. Edinburgh, Scotland, UK: ACM, 2016, pages 389–398.
- [53] T. Staubitz, J. Renz, C. Willems, J. Jasper, and C. Meinel. "Lightweight Ad Hoc Assessment of Practical Programming Skills at Scale". In: *Global Engineering Education Conference (EDUCON), 2014 IEEE*. IEEE, 2014, pages 475–483.
- [54] T. Staubitz, R. Teusner, J. Renz, and C. Meinel. "An Experiment in Automated Proctoring". In: *European MOOC Stakeholder Conference (EMOOCs)* (Graz, Austria). P.A.U. Education, 2016, pages 41–53.
- [55] T. Staubitz, H. Traifeh, S. Chujfi, and C. Meinel. "Have Your Tickets Ready! Impede Free Riding in Large Scale Team Assignments". In: *L@S '20*. Virtual Event, USA: Association for Computing Machinery, 2020, pages 349–352. DOI: 10.1145/3386527.3406744.
- [56] T. Staubitz, H. Traifeh, and C. Meinel. "Team-Based Assignments in MOOCs – User Feedback". In: *2018 Learning With MOOCs (LWMOOCs)*. Sept. 2018, pages 39–42. DOI: 10.1109/LWMOOCs.2018.8534607.
- [57] T. Staubitz, C. Willems, C. Hagedorn, and C. Meinel. "The gamification of a MOOC platform". In: *2017 IEEE Global Engineering Education Conference (EDUCON)*. Apr. 2017, pages 883–892. DOI: 10.1109/EDUCON.2017.7942952.
- [58] H. Steinbeck, T. Staubitz, and C. Meinel. "Proctoring und digitale Prüfungen – Durchführungsbeispiele und Gestaltungselemente für die digitale Lehre". In: *DELFI 2021*. Edited by A. Kienle, A. Harrer, J. M. Haake, and A. Lingnau. Bonn: Gesellschaft für Informatik e.V., 2021, pages 253–264.
- [59] R. Teusner and T. Hille. "On the Impact of Programming Exercise Descriptions". In: *2018 Learning With MOOCs (LWMOOCs)*. Sept. 2018, pages 51–54. DOI: 10.1109/LWMOOCs.2018.8534676.
- [60] R. Teusner, T. Hille, and T. Staubitz. "Effects of Automated Interventions in Programming Assignments: Evidence from a Field Experiment". In: *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*. L@S '18. London, United Kingdom: ACM, 2018, 60:1–60:10. ISBN: 978-1-4503-5886-6. DOI: 10.1145/3231644.3231650.

- [61] R. Teusner, C. Matthies, and T. Staubitz. "What Stays in Mind? – Retention Rates in Programming MOOCs". In: *2018 IEEE Frontiers in Education Conference (FIE)*. Oct. 2018, pages 1–9. DOI: 10.1109/FIE.2018.8658890.
- [62] M. Totschnig, C. Willems, and C. Meinel. "openHPI: Evolution of a MOOC Platform from LMS to SOA." In: *CSEDU*. Edited by O. Foley, M. T. Restivo, J. O. Uhomoibhi, and M. Helfert. SciTePress, 2013, pages 593–598. ISBN: 978-989-8565-53-2.
- [63] TÜV Rheinland. *Online Verification for Certificate No. 01 153 1900724*. <https://www.certipedia.com/certificates/01+153+1900724>. [Online; accessed 11-August-2022]. 2022.
- [64] VMware, Inc. *RabbitMQ – Quorum Queues*. <https://www.rabbitmq.com/quorum-queues.html>. [Online; accessed 22-August-2022]. 2022.

Current Technische Berichte des Hasso-Plattner-Instituts

Vol.	ISBN	Title	Authors/Editors
147	978-3-86956-533-0	Modeling and formal analysis of meta-ecosystems with dynamic structure using graph transformation	Boris Flotterer, Maria Maximova, Sven Schneider, Johannes Dyck, Christian Zöllner, Holger Giese, Christelle Hély, Cédric Gaucherel
146	978-3-86956-532-3	Probabilistic metric temporal graph logic	Sven Schneider, Maria Maximova, Holger Giese
145	978-3-86956-528-6	Learning from failure : a history-based, lightweight test prioritization technique connecting software changes to test failures	Falco Dürsch, Patrick Rein, Toni Mattis, Robert Hirschfeld
144	978-3-86956-526-2	Die HPI Schul-Cloud – Von der Vision zur digitalen Infrastruktur für deutsche Schulen	Christoph Meinel, Catrina John, Tobias Wollowski, HPI Schul-Cloud Team
143	978-3-86956-531-6	Invariant analysis for multi-agent graph transformation systems using k-induction	Sven Schneider, Maria Maximova, Holger Giese
142	978-3-86956-524-8	Quantum computing from a software developers perspective	Marcel Garus, Rohan Sawahn, Jonas Wanke, Clemens Tiedt, Clara Granzow, Tim Kuffner, Jannis Rosenbaum, Linus Hagemann, Tom Wollnik, Lorenz Woth, Felix Auringer, Tobias Kantusch, Felix Roth, Konrad Hanff, Niklas Schilli, Leonard Seibold, Marc Fabian Lindner, Selina Raschack
141	978-3-86956-521-7	Tool support for collaborative creation of interactive storytelling media	Paula Klinke, Silvan Verhoeven, Felix Roth, Linus Hagemann, Tarik Alnawa, Jens Lincke, Patrick Rein, Robert Hirschfeld
140	978-3-86956-517-0	Probabilistic metric temporal graph logic	Sven Schneider, Maria Maximova, Holger Giese

ISBN 978-3-86956-544-6
ISSN 1613-5652