A Scaleable Online Programming Platform for Software Engineering Education

Torge Hinrichs[1], Henri Burau[2], Jens von Pilgrim[3], Axel Schmolitzky[4]

Abstract: Programming is solving problems with computer assistance. Learning the craft of programming is a challenging task for most computer science students. It requires a high amount of training to get into the mindset of a good software engineer, and many students lack this training. A promising way to compensate this lack is the provision of an easily accessible learning platform where students can find several programming assignments that are fun to solve, ideally on the platform itself. In this paper, we present the architecture of an online programming practice platform that provides a fully featured online IDE, running in any modern browser and offering fast feedback on provided solutions. It is deployed in a scalable state-of-the-art cloud-infrastructure based on a microservice architecture to ensure a stable and extensible software system.

Keywords: Software Engineering Education; Online Programming Education Platform; Automatic Assignment Evaluation

1 Introduction

The ability to program is a fundamental skill for computer science students. But learning to program is hard. Besides struggling with the paradigms and syntax of a programming language, several other challenges await the learner: How to implement a given algorithm in a specific language? How to use all these libraries? And how to use powerful state-of-the-art IDEs without getting lost in detail? But even after these questions have been answered successfully by an individual, after several semesters, the next step lies in wait: Realistic programming problems are too complex nowadays to be solved by a single person. And learning to program in a team is hard as well: How to communicate about design decisions effectively? How to formulate source code in a way that guarantees readability and maintainability in the long run? How to decompose a software system into manageable components with clear interfaces in order to be able to work in parallel? How to handle conflicts in a version control system? How to design for better testing and modularity?

All such aspects of programming require experience, which can be achieved by solving a large number of assignments at various difficulty levels. *Online programming practice platforms* (OPPs), such as *CodinGame* and *HackerRank*, address the programming abilities of individuals, but neglect programming assignments for teams. In [HGS20], we defined an OPP as "*a web-based platform that offers several assignments of varying complexity, a*

[1] HAW Hamburg, Berliner Tor 7, 20099 Hamburg, Germany, torge.hinrichs@haw-hamburg.de
[2] HAW Hamburg, Berliner Tor 7, 20099 Hamburg, Germany, henri.burau@haw-hamburg.de
[3] HAW Hamburg, Berliner Tor 7, 20099 Hamburg, Germany, jens.vonpilgrim@haw-hamburg.de
[4] HAW Hamburg, Berliner Tor 7, 20099 Hamburg, Germany, axel.schmolitzky@haw-hamburg.de

possibility to enter code that solves an assignment, and an automatic feedback mechanism for the proposed solutions.". In that work we also assembled requirements for an OPP that further supports software engineering education at universities.

Since March 2020, we are designing and implementing OPPSEE, our "*Online Programming Practice platform for Software Engineering Education*". We aim at offering a non-mandatory option for teachers that can complement formative assessment in programming related courses, without the need or burden of summative assessment. Unlike other platforms, our approach uses a fully featured online integrated development environment (IDE) which has to be enhanced with assignment capabilities rather than using a simple (text based) assignment system which needs to be enhanced with programming capabilities. This enables more complex assignments, which may require several source files and which may even require to be solved by a team. While setting up a classical IDE for larger problems can become quite complex and distract students from the real problem, our platform can be used within any modern browser and delivers the appropriate development environment for each individual assignment with no user installation needed.

In this paper we present our approach of building OPPSEE, focusing on the architecture of our solution and how non-functional requirements such as scalability, security and modularity are handled.

2 Influential Projects

There are several projects that influenced OPPSEE. In this section we briefly discuss OPPs and "Course management systems" that offer features that we would like to incorporate into our system. We also show why these platforms do not satisfy our requirements from [HGS20].

2.1 OPPs

Several OPPs[5] are available. They provide many assignments in various difficulties, support multiple languages and an online editor without additional setup, which results in a low entrance barrier. In some systems (e.g. CodinGame) the automatic evaluation of assignments is enhanced with visualisations of the solution, which makes working with the systems more appealing. The blind spot of all these platforms is team programming, as they more or less focus on the individual ability to program.

2.2 Course Management Systems

Course management systems are widely used at universities. They allow submitting assignments, deadline monitoring, automatic feedback, and plagiarism detection. In this

[5] `https://www.codingame.com/`, `https://www.codewars.com/`, `https://coderbyte.com/`, `https://www.hackerrank.com/`

section we will focus on systems that also allow the submission of programming assignments.

2.2.1 Moodle Plugins For Programming Exercises

The Modular Object-Oriented Dynamic Learning Environment (Moodle) is a free and open-source learning management system. Its default configuration does not include programming assignments, however this feature can be added via plugins. We developed such a plugin that can be used as an additional learning resource [GS19], its JUnit-Grader is able to generate feedback for an assignment in less than a second. During the development and usage of this plugin we observed some fundamental problems: Adding IDE-like capabilities such as code completion or working on multiple files were hard to include. Due to its bad user experience, only few students used the online editor. Instead they developed the source code in their preferred IDE and copied the solution into the Moodle editor only for submission. This turned out to be a major drawback since the provided assignments where optional and the poor usability resulted in few submissions.

2.2.2 ArTEMiS

ArTEMiS is an open-source AuTomated assEssment Management System for interactive learning, written in Java and developed at TU Munich. It combines version control and continuous integration with automated assessment of programming exercises and immediate feedback [KS18]. Students can submit their solutions both via the online editor or a local IDE. The architecture of ArTEMiS is very flexible and language independent: It allows every programming language which can be build in a CI-Pipeline. ArTEMiS offers more advanced features such as UML diagram analysis. It also supports the management of and assignments for teams. It is proven to be a robust and flexible system that can be used in large courses. Although the approach is very useful it does not offer students a fully featured IDE without a local installation and the knowledge of git. Students might not be willing to overcome this barrier if ArTEMiS would be used for non-mandatory additional assignments.

2.2.3 JACK

JACK is a framework for modular grading and feedback generation, developed at the University of Duisburg-Essen. JACK's focus lies on useful feedback. Solutions can be submitted by uploading the file directly to JACK or by using a plugin for Eclipse. For testing the submitted code, a wide range of “checkers” is offered that can analyze the code with different metrics. This includes dynamic tests as well as static code analysis [St16]. JACK can be used without the knowledge of git but it requires a local IDE installation. Setting up and tailoring a course in JACK is a fairly complex task, even with a large number of available assignments. This raises the entry level for (casual) teachers significantly.

3 OPPSEE

The OPPSEE project aims to provide a programming platform as an additional (non-mandatory) offer for all computer science students, on which they can practice and improve their programming skills in as many ways as possible, as individuals or in teams. One basic requirement states the need for updating the platform without user interaction. This is obviously fulfilled by web applications in general, using container based technology makes this even easier. A lot of requirements describe features already fulfilled by modern source code management systems, such as collaboration features, or branching. Git and GitLab in particular supports these features directly and can be accessed through the platform. Most importantly, we learned that a programming platform must provide state-of-the-art IDE support, such as content assist. It is also necessary to support projects with multiple files in order to allow for realistic settings and assignments. This requirement, combined with the need for a web-application-based solution, leads to a Web-IDE. There are not that many Web-IDEs which are open source and vendor neutral, *Gitpod* is one of them. This is why we selected this product in combination with Gitlab (already used at our university) as the base technology for our solution. In this section the overall architecture of the platform will be discussed.

3.1 System Overview

Figure 1 provides a simplified overview of the currently implemented components of the platform.

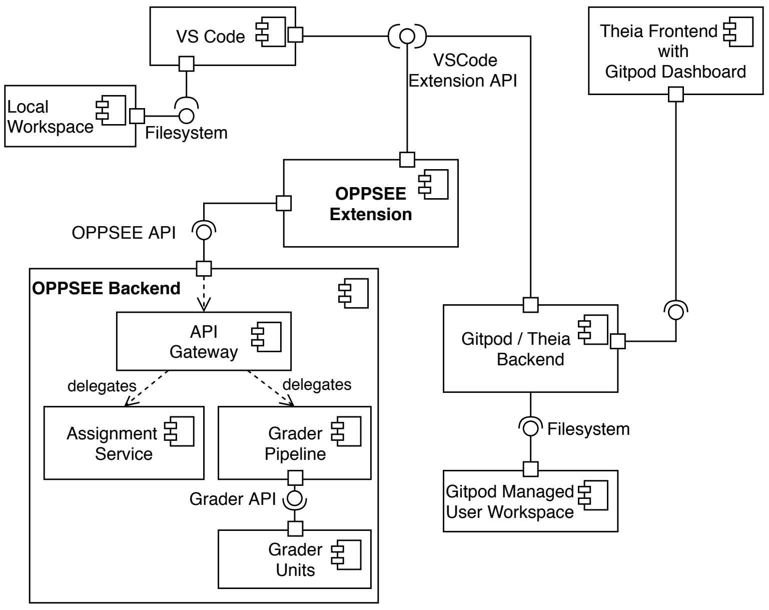


Fig. 1: Simplified UML Component Diagram of the OPPSEE Platform

OPPSEE is an online platform and therefore has to run at least one online service. This service is provided by the OPPSEE Backend. The backend serves an "API Gateway" that provides all needed services for working on different assignments as well as grading the solutions. This service is used by the client, which can be either a standalone IDE or a browser-based IDE. If the browser-based IDE is used, opening a link is sufficient to open the IDE and create an individual workspace inside the cloud. This workspace contains all needed software or run time environments (compiler, assignment files, source files, user tests, etc.) needed to solve the assignment. The workspace is managed by Gitpod [6]. Gitpod is a development environment specifically build as a cloud application and was released as an open source project in August 2020. It allows for configuring individually built environments and loading Git repositories into the setting. Gitpod also comprises a full featured Web-IDE called *Theia*[7]. One of its main features is its full compatibility with Microsoft's *Visual Studio Code* (VS Code) Extensions. VS Code [8] is an Open-Source source code editor that supports development operations such as debugging, task execution and version control. In order to extend Gitpod and the Web-IDE with OPPSEE specific features (including communication with the OPPSEE backend), the *OPPSEE Extension* is used. It is an VS Code extensions which can be plugged into the Gitpod backend on the server side. Alternatively, students can set up their own development environment on their client machine and use VS Code for editing their source code. In this scenario, the OPPSEE Extension is plugged in VS Code on client-side.

This is one of the advantages of this architecture: We can use the very same OPPSEE Extension either on server side or on client side. The latter case is in particular interesting for platform developers, as they can interact with the system, develop new features and test them without the need of deploying the cloud infrastructure on their local machines.

3.2 Frontend

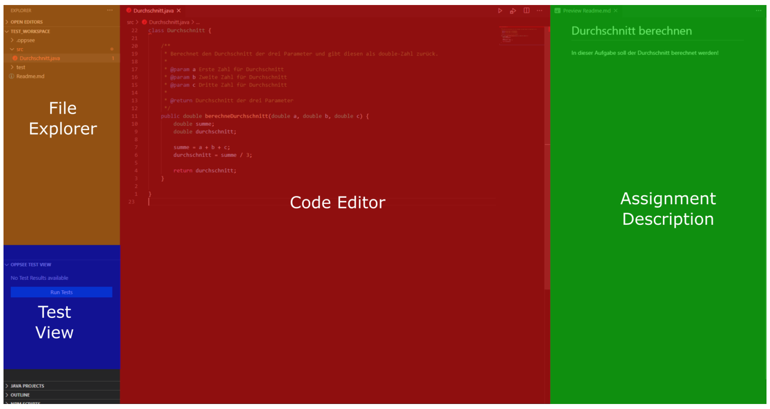
In this section, the GUI components are described. Other meta features, such as an advanced assignment selection, progress tracking or badges for completed assignments, will be discussed and implemented in future work.

Using a VS Code based UI is beneficial for students, because they work in a well known environment. Figure 2 shows a structure overview of VS Code with an installed OPPSSEE Extension. Note that this view is identical to the Gitpod version in the browser. In either case, the editor consists of 4 basic views. Each of them can be resized, moved and closed or reopened if needed.The "**File Explorer**" can be used to structure assignments in multiple files. Whereas most other platforms shown in 2.1, only provide the ability to work in a single file or to enter a single method. Furthermore, additional information can be stored and linked in the "Assignment Description". The assignment description is displayed in the "**Assignment Description**" view. It is possible to link files in the text that will

[6] `https://gitpod.io`

[7] `https://theia-ide.org`

[8] `https://code.visualstudio.com`



```
class Durchschnitt {

    /**
     * Berechnet den Durchschnitt der drei Parameter und gibt diesen als double-Zahl zurück.
     *
     * @param a Erste Zahl für Durchschnitt
     * @param b Zweite Zahl für Durchschnitt
     * @param c Dritte Zahl für Durchschnitt
     *
     * @return Durchschnitt der drei Parameter
     */
    public double berechneDurchschnitt(double a, double b, double c) {
        double summe;
        double durchschnitt;

        summe = a + b + c;
        durchschnitt = summe / 3;

        return durchschnitt;
    }

}
```

Durchschnitt berechnen

In dieser Aufgabe soll der Durchschnitt berechnet werden!

Fig. 2: Basic component overview of the OPPSEE Extension

be automatically opened in the "Code Editor" by clicking on them. The "**Code Editor**" view uses the full range of features VS Code provides such as syntax highlighting, code completion, peeks on documentation and even linting the written source code.

After working on the assignment, testing its correctness is the next logical step. Therefore, the "**Test View**" is used. By running the tests, the solution will be transmitted to the OPPSEE-Backend and processed by the necessary grading units. The corresponding test results will be received and can be opened by clicking on a result in the test view. It is worth mentioning that the test reports are generated by the backend and, at the moment, simply displayed via the extension as an HTML document. In the future, much more flexible reports can be created based on the grading unit, even graphical components for visualizing the report are possible. Further, additional grading units or updated versions can be used without updating the client software. Since we can use any VS code extensions, a wide variety of languages is supported. We also benefit from planned features of Gitpod, such as providing collaboration features in the editor (a feature already provided by commercial extensions).One typical task for students in earlier semesters is to develop simple GUI applications, e.g., Java Swing or JavaFX apps. The Gitpod technology stack allows for creating a virtual display (also known as virtual-frame-buffer) that can be viewed in the frontend.

3.3 Cloud Infrastructure

This section focuses on the cloud components shown in Fig. 1, especially properties such as scale-ability, security and modularity are investigated.

Gitpod

The first major component is Gitpod. As already described in section 3.1, Gitpod is a development environment build for the cloud. Therefore, each workspace created and managed by Gitpod is contained in a separate pod (a collection of containers that can communicate among each other) with its own disk space. For the sake of performance, the disk space is only used to create snapshots of the pod that can be used to pause and resume working in the pod. The file system itself is only loaded in memory which increases the performance drastically, due to the absence of IO-operations. The images used in the containers are pre-built and stored in an external container registry. In our case, we use “harbor.io”. It is an open-source container registry that “secures artifacts with policies and role-based access control, ensures images are scanned and free from vulnerabilities, and signs images as trusted”[9]. Harbor automatically performs security and vulnerability analysis of the stored images and reacts based on provided rules. This can range from notifying admins up to directly removing the image. Furthermore, each workspace has only restricted access CPU and memory usage and are also monitored by the system. This prevents exploiting or slowing down the infrastructure, workspaces with exceptionally high resource demands are reported and can be stopped by an admin.The encapsulated characteristics of the individual pods ensures a flexible and scalable deployment. The superordinated orchestration provided by Kubernetes ensures that the workload can be balanced throughout the cluster.

OPPSEE Backend

The “OPPSEE backend” is a REST-based backend system that handles all services for providing assignments, evaluating solutions and managing the results. The backend serves a scalable entry point called **“API Gateway”**. This separates the underlying micro-service architecture from the other parts of the system. However, this increases the scalability of the backend, because individual services can be replicated or removed depending on of the load that has to be handled. Scaling individual services can be automatized by the orchestration framework, our approach is based on recommendations noted in Song et al. [SZH18]. Due to the smaller architecture, a service discovery is not implemented, but will be considered in the future.

We still need to measure performance in real life scenarios. For that we plan to introduce OPPSEE in selected courses with limited participants (e.g. 60 students). This is part of future work. The architecture also has advantages concerning its modularity. Components can be added, changed or removed without changing the other parts of the platform. Taking security into consideration, each component of the OPPSEE backend is stored in an container image managed by harbor and are also monitored. If an exceptional behavior is detected, the incidence will be reported and further actions can be taken.

Another component included in the OPPSEE backend is the **“Assignment Service”**. This instance is the interface to the Assignment Database and manages all needed files and meta information a student needs to work on an assignment. Note that an assignment can contain different types of exercises, including “programming language independent assignments”

[9] `https://goharbor.io/`

such as multiple choice tests, or “language specific assignments” that might add special requirements to the solution, e.g. a recursive implementation or only use the list datatype. These requirements are checked by the **“Grader Pipeline”**. This component selects and manages different types of grading units to review whether the solution matches the specific requirements of the assignment. Therefore, various “Grading Units” are provided and registered in the Grader Pipeline. Each unit registers itself by notifying the pipeline of its capabilities and its service URL. This enables the Grader Pipeline to schedule tasks to the corresponding grading unit. Furthermore, it keeps the system dynamic and scalable. Updates to individual grading services can be deployed easily. In addition, balancing high loads on single Grading Units can be handled automatically by kubernetes. If the cluster detects a high frequency of requests to a specific URL, kubernetes takes care of replicating the service and due to the any cast characteristic of the service infrastructure the load is split between the replicas.

4 Conclusion

In this paper we described a flexible, scalable, microservice-based architecture of an online platform for programming education. Unlike others our approach is centered around a fully featured online IDE that is enhanced by assignment and testing functionalities. The platform uses Gitpod to serve a ready to use development environment. Further, we added a flexible backend that is autoscaling depending on the load of the individual modules. We also built a VS Code extension that can be deployed easily to different client environments and most importantly, automatically set up in our online IDE. Our feedback mechanism is capable of returning individual reports for different test and verification methods.

Bibliography

[GS19] Gandraß, Niels; Schmolitzky, Axel: Automatisierte Bewertung von Java-Programmieraufgaben im Rahmen einer Moodle E-Learning Plattform. In: Proceedings of the Fourth Workshop “Automatische Bewertung von Programmieraufgaben” (ABP 2019), Essen, Germany, October 8-9, 2019. Gesellschaft für Informatik e.V., pp. 3–10, 10 2019.

[HGS20] Hinrichs, Torge; Gandraß, Niels; Schmolitzky, Axel: Towards an Online Programming Platform Complementing Software Engineering Education. In (Krusche, Stephan; Wagner, Stefan, eds): Tagungsband des 17. Workshops “Software Engineering im Unterricht der Hochschulen” 2020, Innsbruck, Österreich, 26. - 27.02.2020. volume 2531 of CEUR Workshop Proceedings. CEUR-WS.org, pp. 27–35, 02 2020.

[KS18] Krusche, Stephan; Seitz, Andreas: ArTEMiS: An Automatic Assessment Management System for Interactive Learning. In: Proceedings of the 49th ACM Technical Symposium on Computer Science Education. SIGCSE ’18, Association for Computing Machinery, New York, NY, USA, p. 284–289, 2018.

[St16] Striewe, Michael: An Architecture for Modular Grading and Feedback Generation for Complex Exercises. Sci. Comput. Program., 129(C):35–47, November 2016.

[SZH18] Song, M.; Zhang, C.; Haihong, E.: An Auto Scaling System for API Gateway Based on Kubernetes. In: 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS). pp. 109–112, 2018.