Faculty of Engineering, Computer Science and Psychology
Institute of Software Engineering and Programming Languages

universität ulm

Reflective Report

as part of the Software Project Software Engineering
Exam no. 13369

App for Ulm University

Winter term 2021/22

Author

Emilija Kastratovic
emilija.kastratovic@uni-ulm.de
Student no.: 1052407
Software Engineering, BSc.
7th semester

Supervisor

Alexander Raschke
alexander.raschke@uni-ulm.de

Version of December 25, 2022

1 Project

This reflection report is for a the University of Ulm's Software Engineering project.
The project involved creating an app about the university, called UniApp, with the target audience being the students at Ulm University.

The original project started many years ago, going through many iterations and changes over the course of its development. I was brought on to the project in September 2021. In the curriculum we were tasked with a total of 360 hours to complete the project. Originally, the project was supposed to be completed in one semester, however, the project was extended to January of 2023. I worked off the 360 hours in December 2022.

The app was developed using a variety of software development techniques and tools, including agile methodologies, version control systems, and testing frameworks. The team consisted of several students who worked together to design, develop, and test the app. The students were split into four teams which were responsible for different platforms of the project, including the frontend teams for Android and iOS, the backend team for the server, and a map team.

The vision for this project was to create an app that would provide students with a convenient and centralized source of information about the university. The app was intended to be a one-stop-shop for students to access important resources such as news and events, a bulletin board where students can publish advertismetns, public transportation information, the canteen menu, and a map of the university campus. Other features include an FAQ section, and a campus navigation system.

Unfortunately, the app was not completed during the initial development phase and was left in an unfinished state with many bugs. The goal of this project was to fix the bugs and get the app to a stable state that could be released in app stores.

The purpose of this report is to reflect on the process of developing the app, identify any areas for improvement, and discuss the challenges and successes encountered during the bug fixing process. Overall, the project was a valuable learning experience that allowed us to apply the concepts and skills we had learned in our coursework to a real-world project.

2 Approach

There were four teams working on the project, each responsible for a different platform, the teams being the Android team, the iOS team, the backend team, and the map team. Each team was responsible for developing and testing their own platform. I was invovled in the iOS team and later switched to the backend team.

The frontend was developed natively for Android and iOS using the Kotlin and Swift programming languages, respectively. For the iOS team, we used Swift and Xcode as our programming language and IDE. Members of the iOS team who didn't have a MacOS system were given a Mac Mini to use for development. The backend was developed using the Laravel framework and PHP as the programming language. PHPStorm, Xampp, and Postman were used as development tools. Swagger was used to document the API.

GitLab was used as the version control system for this project, and the team set up a repository for each platform (iOS, Android, Backend) in order to manage the codebase separately. This allowed us to work on the different platforms concurrently and ensured that the code was well-organized and easy to manage.
We utilized GitLab's continuous integration/continuous deployment (CI/CD) pipeline to automate the deployment process for each platform. This allowed us to easily and quickly deploy new versions of the app to production, as well as test and release other updates. Its features and capabilities made it an essential part of our workflow and contributed to the success of the project.

For this project, we decided to use the Scrum development approach as it allowed us to work quickly and iteratively. Features were already planned and documented in the Sphinx documetation located in the common repository. For every feature or bug fix, a new issue was created in GitLab, and the team would then work on the issue and close it once it was completed.

Over the course of the project, many different student were appointed as Scrum Masters, taking turns every six weeks. The Scrum Masters were responsible for organizing the team, managing the project and meetings, and ensuring that the team was working efficiently. Meetings were held weekly, or bi-weekly. Communication in the whole team was done through the project's Slack, and later on Mattermost workspace, as well as in every meeting so that everyone was up to date with the project's progress. The sub teams communicated through WhatsApp and Discord, where they could discuss issues and work on them together with pair programming. Often times this would help to solve issues faster and more efficiently.

3 Personal contribution

I started on the project as a member of the iOS team, wanting to gain experience with Swift and iOS development, as developing for iOS was only possible on a Mac.

Starting off in the team I was involved in clarifying the app's required features and functionality. I cleaned up existing but outdated issues and created new ones. I started working on existing issues which mainly involved bug fixing.

While still on the iOS team, I first became the protocol manager. This involved creating and updating the protocol for the team, as well as documenting the team's progress and decisions. Afterwards, I was appointed as the Scrum Master for the team. This involved organizing the team, managing the project and meetings, and ensuring that the team was working efficiently. To do this, I continuously communicated with the team members, asked about their progress, and helped them to resolve any issues they encountered.

One of the first tasks I wanted to complete was to update the look and feel of the app to make it more modern and user-friendly. The reason behind this was that the app looked unappeling, as many buttons and other elements were not aligned properly, and the overall design was inconsistent. I was responsible for redesigning and improving the user interface for the iOS version of the app. This involved updating the layout, design, and functionality of various screens and features to create a more user-friendly experience for users. For the rehaul, I used Figma and Photoshop to create mockups. These mockups, I then documented in the documentation repository. While doing this, I also updated missing documentation for the iOS screens and features.

During the course of the project, I also reviewed code submitted by other team members through merge requests. This process helps to ensure that only high-quality code is merged into the main branch of the project, and helps to prevent bugs and other issues from being introduced that were overseen by the other developers.

After changing to the backend team, I needed to get to know the backend codebase. I did this by reading through the code and documentation. For my first backend task, I implemented support for the French language on the website, allowing users to browse the site in their preferred language. This involved translating text and other content and integrating it into the codebase.

Afterwards, I started work on existing bugs. One bug caused FAQ reactions to be given to the wrong question. This was confusing, as the tests were passing, but the bug was still present in the app. I first fixed the bug by changing the used function. Then I took a look at the test and noticed that, while the test's idea was correct, the implementation was wrong, as there was only one question in the database, so the test was always passing. I fixed this by adding more FAQs and improved the tests.

I also worked on improving the security of the app. I modified the way that the app handles user authorization, changing it from using a query parameter to using an authorization header. This change improved the security of the app by making it more difficult for unauthorized users to gain access. Even though the Laravel documetation offered both types of authorization, I decided to use the authorization header as it was more secure. This was a breaking change, as it required the frontend teams to update their code.

While coding, I noticed that the code was inconsistent with the Laravel code style guide. I decided to fix this by updating the code to follow the Laravel code style guide. Addidtionally, I documented this change and added a code style template to the documentation repository, so that other developers could use it.

I was also involed in reviweing the PHP 8 upgrade. This was done with pair programming, where I worked with another team member to fix the issues. There were many complications with the pipeline that prevented the app from being successfully deployed to production. I was working on fixing these issues with the continuous integration/continuous deployment (CI/CD) pipeline. The pipeline failed at many different points, and every small progression brought up new issues. Fixing the pipeline was a very time-consuming process. Unfortunately, we were unable to fix the issues in time.

During the development, many wishes were made by the frontend teams. One of these was to change the way that the app handles the advertisement requests. When posting a new advertisement, the needed parameters could be sent as a query parameter or as a JSON object. However, updating the advertisement required the parameters to be sent as a query parameter. This was inconsistent and confusing, so I wanted to change the way that the app handles the advertisement requests. However, the code was not well-documented, and I was unable to find out why this feature was implemented in this way. Unfortunately, I was unable to fix this issue.

Both on the backend and frontend team, I made many changes and improvements to the project's documentation to ensure that it was accurate and up-to-date. This included correcting errors, adding missing information, and clarifying confusing sections. One big task was to document how to import all HTTP requests into Postman and I wrote a detailed guide on how to do this.

Experiencing difficulties with updating the local PHP version, I also modified the instructions so that people who also encounter these issues can easily fix them.

Important to note is that most of the time spent on a bug was not spent on fixing the bug itself, but on finding the bug and researching how to fix it.

4 Evaluation

Starting off, I was very excited to work on the project. I was looking forward to learning new technologies since this was the first and best opportunities to work with a Mac and Swift and SwiftUI. After having a lot of smaller university projets, this was the perfect opportunity to work on a bigger, more meaningful project.

However, finding a way into the project was not easy. The project was left in a very unapproachable state by the previous teams. The code was not well-documented, and the documentation was outdated. Former members of the team, while giving us a brief introduction to the project, were not really available for help and left shorty after. This left a huge impact and made it very difficult to get started on the project. In contrast, at the end of the project, where many new members joined, I feel like we were able to help the newcomers much better and be of better help.

This rough introduction to the UniApp really lowered the motivation for the project and made it hard to get started. In addidiotn, Swift and SwiftUI were very new to me, and honestly very difficult to learn, as the syntax is very different from Java and other programming languages that I already knew. This struggle was amplified when the only available tasks were bug fixing. Starting freshly with a new project or programming language is vastly different from fixing bugs in an existing project in a language you are not familiar with. This made it very difficult to stay motivated. In addition, tasks like the iOS rehaul, which was very crucial and exciting to me as I loved designing, were not the priority for the application. The focus was on bug fixing and getting the app into a stable state and I was struggling to contribute to the project in a meaningful way. However, I was told that doing something is better than doing nothing, so I started on the iOS rehaul. This contribution helped me to find motivation again and to get into the project.

Nevertheless, I realized that my journey with Swift should come to an end and I decided to switch to the backend team. This is probably the best decision I have made during the project. The start was very smooth and I was able to get started quickly, as members introducing me to the backend were already familiar with the codebase. While PHP and Laravel were as new to me as Swift, getting started was much easier. I started to enjoy the project again and was able to contribute to in a meaningful way.

This period is in stark contrast to the beginning of the project. I was very excited to work on the backend, try to find bugs and fix them, and optimize the code. In addition, pair programming was much more successful on the backend team. Pair programming, and working with other people at the same time in a call was very beneficial for the whole project. I only noticed how efficient it is to pair program when I was working on the backend. We were able to work more efficiently and find bugs much faster. Encountering bugs was much more fun.

In general, I feel like I learned a lot more during the backend period; not only for the project but for my understanding of software development and programming. The UniApp, while a big

question mark at the beginning, turned out to be a very interesting project. Unfortunately, this period came too short, as most hours had already been spent and I would definitely have liked to work on the backend for longer.

To summarize; I was very excited to work on the project at the beginning, but the rough start made it hard to get started. I was able to find motivation and purpose again when I switched to the backend team, but this period was too short. I learned a lot during the project. Not only was I able to gain understanding for the technologies that I was working with, but I also learned how important it is to have a team that works well together. I am very grateful and happy for the opportunity to have worked on the UniApp and I feel like it is being left in a state that is much better than it was at the beginning, especially for the new members of the team.