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 26, 2022

1 Project

This reflection report is for the University of Ulm's Software Engineering project.

The project involved creating an app about the university, UniApp, with the target audience being the students at Ulm University.

UniApp started many years ago, going through many iterations and changes throughout its development. I joined the project in September 2021. We were tasked with a total of 360 hours in the curriculum to contribute to the project. Initially, the assignment was supposed to be completed in one semester; however, our time was extended to January 2023. I finished the 360 hours in December 2022.

The app was developed using various software development techniques and tools, including agile methodologies, version control systems, and testing frameworks. The development team consisted of several students who worked together to design, develop, and test the app. The students were split into four teams responsible for different project platforms: the frontend teams for Android and iOS, the backend team for the API, 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 service for students to access essential resources such as news and events, a bulletin board where students can publish advertisements, public transportation information, and the canteen menu. 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 unfinished 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.

2 Approach

In the following section, I will explain the approaches that were used to develop the app.

As mentioned above, four teams were working on the project, each responsible for a different platform: the Android team, the iOS team, the backend team, and the map team. Each group was responsible for developing and testing their platform.

I was involved 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. Members of the iOS team who didn't have a MacOS system were given a Mac Mini, as developing for iOS was only possible on a Mac with Xcode installed.
The backend used the Laravel framework and PHP as the programming language. The team used PHPStorm, Xampp, and Postman as development tool and Swagger to document the API.

We used GitLab as the version control system for this project, with repository for each platform (iOS, Android, Backend) to manage the codebase separately. It allowed us to work on the differ-

ent platforms concurrently and ensured that the code was well-organized and easy to manage. GitLab's continuous integration/continuous deployment (CI/CD) pipeline helped us automate the deployment process for each platform. It allowed us to easily and quickly deploy new versions of the app to production and test and release other updates. Its features and capabilities made it an essential part of our workflow.

The team used the Scrum development approach for this project, allowing us to work quickly and iteratively. Features were already planned and documented in the Sphinx documentation 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.

Throughout the project, many different students 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 worked 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 updated 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 this would help to solve problems 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.
TO get familiar with the UniApp, I was involved in clarifying the app's required features and functionality. I cleaned up existing but outdated issues and created new ones.

Afterwards, I started working on available issues which mainly involved bug fixing. One of the first tasks I wanted to complete was updating the app's look and feel to make it more modern and user-friendly. The reason behind this was that the app looked unappealing, as many buttons and other elements were not aligned properly. The overall design was inconsistent. I was responsible for redesigning and improving the user interface for the iOS version of the app. The rehaul involved updating the layout, design, and functionality of various screens and features to create a more user-friendly user experience. For that, I used Figma and Photoshop to create mockups, which I then documented together with updating the missing documentation for the iOS screens and features.

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. Consequently, 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 worked efficiently. To do this, I continuously communicated with the team members, asked about their progress, and helped them to resolve any issues they encountered.
After changing to the backend team, I, once again, needed to get familiar with this new codebase, reading through the code and documentation, as well as asking other team members for help. 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. THe task involved translating text and other content and integrating it into the codebase.

Afterwards, I started work on existing issues. 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 noticed that the test cases were always passing, giving a false positive. While the the idea was correct, the implementation was slightly wrong, as there was only one question in the database. I fixed this by adding more FAQs and improving them. This type of test was present in many other places, so I decided to fix them as well.

A crucial change I made was to change how the app handles user authorization. I changed it from using a query parameter to an authorization header, which improved the security of the app. Even though the Laravel documentation offered both types of authorization, I decided to use the authorization header as it was more secure. The alteration required the frontend teams to update their code, making it a breaking change.

I was also involved in reviewing the PHP 8 upgrade, working woth other reviewers to get the upgrade done. As it turned out, 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, leading to us being unable to fix the issues in time.

One of these was to change how the app handles advertisement requests. When posting a new advertisement, the needed parameters could be sent as a query parameter or 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 how the app handles 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.

During the whole 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.

Both on the backend and frontend team, I made many 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. Besides the iOS rehaul, another big task was to document how to import all HTTP requests into Postman. I wrote a detailed guide on how to do this, and experiencing difficulties with updating the local PHP version, I modified the instructions so that people who encounter these issues can easily fix them.
While coding, I noticed that the code was inconsistent with the Laravel code style guide. I updated the code to follow the Laravel code style guide, documented this change and added a code-style template to the documentation so other developers could use 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 my first and best opportunity to work with a Mac, Swift, and SwiftUI. After many smaller university assignments, this was the perfect opportunity to work on a more significant, meaningful project.

However, finding a way into the UniApp took work. 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 app, were not really available for help and left shortly after. We had to bear the brunt of mistakes we didn't make, making it very difficult to get started on the project. In contrast, when many new members joined, we could help the newcomers and be of better help.

This rough introduction to the UniApp lowered my motivation. In addition, Swift and SwiftUI were new to me and challenging to learn, as the syntax is very different from Java and other programming languages I already knew.
This struggle was amplified when the only available tasks were bug fixing. Starting fresh with a new project or programming language is vastly different from fixing bugs in an existing project in a language you are unfamiliar with. Tasks like the iOS rehaul, which was 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 struggled 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 find my motivation again.

However, I realized that my journey with Swift should end, and I decided to switch to the backend team. It was the best decision I made during the project. The introduction was very smooth, as the member introducing me to the backend was already familiar with the codebase. While PHP and Laravel were as new to me as Swift, getting started was much more manageable.

I began to enjoy the UniApp again and contributed 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, find bugs and fix them, and optimize the code. In addition, pair programming was much more successful on the backend team and working with others simultaneously in a call benefited the whole project. I only noticed how efficient it is to pair program when I was working on the backend. We could work more efficiently and find bugs much faster, and encountering bugs was much more fun.

In general, I learned much more during the backend period, not only for the app but also for my understanding of software development and programming. The UniApp, a big question mark at the beginning, turned out to be an exciting assignment. Unfortunately, this period came too short, as I had already spent most hours, and I would have liked to work on the backend for longer.

Nevertheless, I am very grateful and happy for the opportunity to have worked on the UniApp. It is being left in a much better state than at the beginning, especially for the new team members.