University of Zurich[UZH]

s.e.a.l. software evolution & architecture lab

Software Praktikum (SoPra) - FS21

Milestone 3 - Assignment

1 General Information

The assignment has to be completed as a team effort, and each member has to be able to answer content-related questions during the presentation for Milestone 3. The hand-in for Milestone 3 consists of **a written report**, **a presentation**, and **the source code**. The report and the presentation must be submitted via OLAT. The source code will be taken from the Github project's master branch. For all three deliverables, the deadline is **2.5.2021 23:59 CET**. The presentation is the next day.

1.1 Report

The report must be submitted as a single PDF file by the **project leader via OLAT**. The title page consists of the group name, group leader, and information about all group members (name, matriculation number, and **GitHub account**). Make sure the report is easily readable in printed form (figures, tables, etc.) and features a link to your running application on Heroku. Furthermore, ensure to use a consistent format (header, footer, font style, font size, page numbers, figure and table titles, etc.), page orientation (portrait), and page size (DIN A4). Use a file name of the form **FS21-Group-GROUPNUMBER-M3-Report.pdf**. The report must be written in English.

1.2 Presentation

The slides for the presentation must be submitted as a single PDF file by the **project leader via OLAT**. The title page consists of the group name and information about all group members (name, matriculation number). Use a file name of form **FS21-Group-GROUPNUMBER-M3-Slides.pdf**. The slides, as well as the presentation, have to be in English.

Advice: Each team member has to present at least at one occasion (M1+M2, M3, or M4). It is up to the team to decide how to distribute the presentation time over its members. Underline the name of the presenters on the title slide. Presentations will be **6 minutes per team and have a hard cut-off**. After the presentation, there will be a 2-4 minutes Q&A session. Make sure to prepare slides that the audience can read and understand over Zoom. Also, ensure to practice your presentation at least once beforehand, especially to make sure you stay within the given time frame. Some instructors that have not seen the report will sit in the presentation and this

will be your moment to shine.

Suggested Structure: We suggest you focus your presentation on the following items:

- The title slide should contain your names and matriculation number (presenters underlined), group number, and project name.
- Briefly re-introduce your project to the audience.
- Provide an overview of your achievements so far and what is yet to come in Milestone 4.
- Demo your app/game. Make sure you can demo the full feature set in the available time. For games, it's likely important to have a fast-forward mode or cheat to jump to the end of the game.
- Present an overview of your testing strategy. What, when, and how do you test? How much code is covered?

1.3 Source Code

The source code is submitted via GitHub. Use GitHub's release feature to tag (**M3**) the commit in the **master** branch that should be taken into consideration for grading. It is important to push the commit before the deadline **2.5.2021 23:59 CET**. Moreover, make sure the code tagged with M3 contains a stable/running version of your system. **If we cannot run your system, we cannot grade it and, therefore, must fail you**. Submissions after the deadline are not taken into account.

2 Assignment Description

Based on the previous assignment, you should have the specifications as well as the (higher level) design for your application to support the implementation and a code deployment pipeline in place. The focus of Assignment 3 is the implementation.

2.1 Implementation

At the end of Milestone 3, you are expected to have a running and usable version of your application (even if not all features are implemented yet). In the report, please include a diagram showing your latest database layout/struture and 2-3 screenshots of your user interface.

During the Milestone, each team member has to contribute to the project continuously. More specifically, every member **has to complete at least 2 tasks per week**. A single development task should have a granularity of 0.5-1 day. We will verify your progress by inspecting your GitHub issues and project board (see below).

We expect you to develop your application for a modern browser and a popular desktop display size. The target platform is **Google Chrome** version 89. The screen real estate for testing is **1366px x 768px** (this is the most common desktop resolution according to W3Counter[1]).

Advice: We strongly recommend finishing all tasks that you specified as *required* by the end of Milestone 3. Milestone 4 will be about finishing up and polishing your application for the final presentation. Depending on progress, you can also start with the optional features that you've

[1] `https://www.w3counter.com/globalstats.php`

planned. You'll also have some time to work on them in Milestone 4. Further, please notice that after Milestone 3, there will be a beta testing phase. You will get a project URL of another group and will have to provide feedback for them. Likewise, another group will test your application. Beta testing requires a certain maturity of your project by the end of Milestone 3, too.

2.2 Project Management on GitHub

At the beginning of this Milestone you are expected to create a *GitHub Project Board* for the 5-week sprint.

1. Navigate to your GitHub organization. Open the "Projects" Tab on an **organizational** level.
2. Create a new project board for the upcoming sprint (e.g., "Sprint 1"). Select "Automated Kanban" from the templates list and link your server and client repositories.
3. Move development tasks you want to work on in the upcoming weeks to the "To Do" column.
4. Assign team members to the tasks.
5. Add an additional column "User Stories". Assign all user stories your selected development tasks originate from to it. This step will help provide a quick overview of your current sprint to outsiders.

While you are programming, you are required to ensure traceability of your progress. Your GitHub issues and project boards should always be up to date. There are two ways to achieve consistency and traceability between your source code and your development tasks.

- **Manual:** After testing and committing the implementation, locate the GitHub issue and type the SHA[2] of the commit as an issue comment. You can then proceed to close the GitHub issue manually.
- **Automated:** Reference and close the GitHub issue in your git commit message[3]. The issue in your project board should automatically move to the "Done" column, too.

Note: we expect every completed development task to link to at least one commit (this excludes user stories).

2.3 Sprint Planning

Towards the end of this Milestone, you should start planning for the second and final sprint. Go through the remaining user stories and development tasks and decide which ones to implement for Milestone 4. Set a GitHub milestone tag (e.g., "Sprint 2") to mark selected tasks. Follow the guidelines provided in the previous assignment sheet: tasks should have a time estimate, acceptance criteria, and an assignee.

[2] `https://docs.github.com/en/github/writing-on-github/autolinked-references-and-urls#commit-shas`

[3] `https://github.com/gitbucket/gitbucket/wiki/How-to-Close-Reference-issues-and-pull-request`

2.4 Testing

Proper software implementations include rigorous and extensive testing. You must have a high degree of test coverage for the server in terms of unit, integration, and REST interface tests. We will check whether the tests are meaningful, i.e., they serve the purpose of identifying bugs. A test that covers getter/setter is not valuable. Aim for tests that deal with the core logic of your application. As an indication of how much code you should test, we require the test coverage to be at least 50%. Check SonarQube for test metrics. We expect that every task (from the back-end) has at least 1 test associated with it.

Moreover, include one *complex* unit, one integration, and one REST interface test with description of the use case in the report. Explain why these particular tests are well-written for the tested functionality and how they are able to capture future regressions (e.g. oracle selection[4], code coverage, etc.). Moreover, elaborate on why your examples (1x unit test, 1x integration test, and 1x REST interface test) are representative of their category.

3 Grading

As SoPra is a pass/fail course, the grade for M3 will be pass or fail as well. Overall, you have to pass 3 out of 4 milestones, where M1 and M4 have to be passed. You need to hand-in reasonable reports for all the milestones.

In order to pass M3, the report, the presentation, and your application (as provided by the source code) have to be assessed positively as described below. The report and the presentation are assessed based on their completeness and quality. The application/coding part is positive if 70% or more of the user stories are completed, it is deployed to Heroku, the user stories implemented are functional, there is a running (and reasonable) prototype with no major bugs, and there are meaningful tests. Further, in every weekly meeting, your TA will assess your group's performance by checking the status of your product backlog. *Every* team member has to implement at least 2 tasks per iteration (i.e., per week). If a team member fails to do so, she/he will fail the milestone (unless there is a valid reason). Note: individuals can fail even if the (rest of the) group succeeds. You will receive feedback on your report and presentation including an assessment (either pass, borderline pass, or fail).

In addition to the report assessment, we require you to award your teammates with "brownie points". This way, you can tell us who you think was the driver and contributed most to the project. Equally important, this is an opportunity for every student to be aware of her/his contribution level. Every student has 10 points per *other* team member (e.g., groups of 5, 4 other team members, hence 40 points); she/he can distribute the points on the other team members to her/his own liking. For borderline submissions, the brownie points can decide whether individual group members pass or fail. **Please submit your brownie points via OLAT by 2.5.2021 23:59 CET**.

[4] `https://en.wikipedia.org/wiki/Oracle_(software_testing)`