FH University of Applied Sciences TECHNIKUM WIEN

Documentation & Project Diary

Innovation Lab 1/2/3
Year 2021

Project: **Git-Game**

Team: **Group 01**

1. General Information

Project name: Git-Game

Supervisor: Lukas Aichbauer

Innovation Lab < *1/2/3, summer term/winter term 2021/23* >

Projectteam:
Tomanovic Pavle, if20b162@technikum-wien.at
Awan Malik, if20b501@technikum-wien.at
Hitalani Tarek, if20b101@technikum-wien.at
Saifee Farhan, if20b505@technikum-wien.at
Yang Li Fu, if20b171@technikum-wien.at

Management Summary of the Project

This project will help users to learn more about git functions in an interactive and interesting way. The Github web game is built from low to high level difficulties. Users with zero knowledge of Github functionalities will be able to master them through our web game.

Framework Conditions and Project Environment

- **PHP:** Data Handling und Validation
- **JavaScript/Typescript:** Interactive Frontend Features
- **HTML/CSS:** Base layout of the Frontend and responsiveness
- **SQL:** Database and DB-queries

Semester-Roadmap

In the first semester we will start with backend. We will create database with all the tables. We will specify all the tables and the ERM diagram in Sprint 2. After that, we will start with registration and login as the first pages to be done. And after that we will plan and develop the game logic.
Focus in the second and third semester will be on design and developing sharing functions, so that more people can hear about our game. But this will be discussed in the beginning of the second semester.

Collaboration & Tooling

GitHub: https://github.com/farhansaifee/Git-Game
Azure DevOps: https://dev.azure.com/saifeefarhan/

Remarks

No other remarks.

2. Brief Description of the Project

This project will help users to learn more about git functions in an interactive and interesting way. The Github web game is built from low to high level difficulties. Users with zero knowledge of Github functionalities will be able to master them through our web game.

Our main priority will be to create a game logic using all functions, so users can experience all of them through the game.

The challenges for the whole team could be the bulid-process/implementation of the game logic, as well as connecting frontend with backend. Responsive designs could take a few weeks to build.

Our first main task on this project will be database creation and project creation. We will create the pages in our application. After that we will start planning and developing the game logic.

Our idea is to create Web-Application where the user should register himself. This will be created as a normal registration form. We will also require the password with one big letter and one number.

When the user logs in, he will be able to start the new game or to continue where he left last time.

We will create levels in the game and every new level will be harder than the previous one. Our idea is to have and to develop new levels all the time, so that particularly game never ends. If somebody is really motivated to play and learn and he finish all the levels, he will get the popup message that we are working on the new levels and that he will get the email when we finish them. In meantime, he will be able to start the new game.

Our goal is not to create multiple choice or fill in the blank questions, on which user should answer. Because we want to create more interactive and interesting way of learning more about git.

3. Specification of the Solution

< Once the order has been clarified (pre-project phase), you start the project implementation. Create a specification of your solution parallel to the implementation of your project across the sprints!

Before each sprint, at least those details must be specified that you will implement in the next sprint. Use techniques such as writing epics & user stories and build a product backlog (use the course content from the course Agile Project Management).

For the specification, generally use visualization techniques that fit the task at hand. For example, in addition to the mockups and user stories, database diagrams, class diagrams, or sequence diagrams (representation of temporal processes) can also be useful.

Usually, you go from rough to detail. The structure of this section can be as follows:

- *System environment: Describe the delimitation of the solution to be implemented (system boundaries)*
- *Features (functional requirements): All required solution properties - in the case of software usually the features or a description of these as user stories or similar)*
    - *Create screen mockups of all essential UI views!*
- *Interfaces: All relevant interfaces of your solution.*
- *Quality characteristics, technical requirements (non-functional requirements): performance, scalability, availability, usability, information on architecture and expandability, etc.*
- *Other "not clear at first glance" but essential solution features!*

Agree with your supervisor how the specification should be structured!

Ask whenever you feel that there may be a misunderstanding, different expectations, or if you did not fully understand a requirement! >

4. Delivery

< In this section you describe the scope of delivery of your solution and everything you need to pass it on to a customer or another software team (in practice this is often referred to as "hand-over to operations" when the solution enters the operational phase).

- *Final solution or solution components including source code*
- *System architecture and data storage*
- *List of any required licenses and information about copyrights (e.g. if third-party software / frameworks or similar were used).*
- *Any hardware specifications*
- *Description of how to install your solution including a list of all components to be installed, installation procedures, migration of databases, etc.*

The content of this section is mostly project-specific. Agree with your supervisor what exactly this section should contain! >

5. Our Project Diary

< This section should be a kind of diary in which you record “what happened in our team in the project”. Use photos from your meetings, take photos of any reflections from whiteboards. Take screenshots.

Describe in short text sections which problems there were, which challenges were solved, what was "cool" in the project, etc.

ATTENTION: Create this section continuously (!) Parallel to the project and not only at the end on the last evening before the project is submitted! This enables your supervisor to understand why something worked particularly well or not so well, why there was great progress or delays, etc.

In practice, such a diary is used as the basis for a project retrospective and team feedback rounds.

Tip: Meet each other at the end of the semester and let your project "pass in review" over a good project closing meal: This is a good opportunity to discuss what you have experienced again and for the future or what you have learned in the next semester and take the Innovation Lab with you! >