**Project Plan**

***Wired Space***

|  |
| --- |
| **Date : 28.02.2025** |
| **Version : 1** |
| **State : In progress** |
| **Author : A. Matviienko** |

#### Version history

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Date** | **Author(s)** | **Changes** | **State** |
| 0.1 | February 28 | Andrii Matviienko | Setting up | Finished |
|  |  |  |  |  |
|  |  |  |  |  |

**Distribution**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Receivers** |
| 0.1 | February 28 |  |
|  |  |  |

Contents

[1. Project assignment 4](#_Toc191663719)

[1.1 Context 4](#_Toc191663720)

[1.2 Goal of the project 4](#_Toc191663721)

[1.3 Scope and preconditions 4](#_Toc191663722)

[1.4 Strategy 4](#_Toc191663723)

[1.5 Research questions and methodology 4](#_Toc191663724)

[1.6 End products 4](#_Toc191663725)

[2. Project organisation 5](#_Toc191663726)

[2.1 Stakeholders and team members 5](#_Toc191663727)

[2.2 Communication 5](#_Toc191663728)

[3. Activities and time plan 6](#_Toc191663729)

[3.1 Phases of the project 6](#_Toc191663730)

[3.2 Time plan and milestones 6](#_Toc191663731)

[4. Testing strategy and configuration management 7](#_Toc191663732)

[4.1 Testing strategy 7](#_Toc191663733)

[4.2 Test environment and required resources 7](#_Toc191663734)

[5. Finances and risk 8](#_Toc191663735)

# Project assignment

## Context

*EvilCorp Inc. requires a social network with the following features: profile customization, absence of censorship (with exceptions), the ability to interact with friends online, and exchange messages. The client needs a reliable and secure system, flexible for integration with external services, and with a high level of data security.*

## Goal of the project

*The goal of the project is to develop a comprehensive and expandable social network system with different roles (user and admin), the ability to authorize/register for users, the ability to add and delete friends, communicate with them via messages. Also, customization of a personal page, and CRUD posts on your own or someone else's personal profile. The admin functionality consists of checking complaints, and resolving them, and blocking and unblocking users.*

## Scope and preconditions

|  |  |
| --- | --- |
| **Inside scope:** | **Outside scope:** |
| 1. User registration and authentication. | 1. Advanced moderation and content filtering (will be considered for future updates). |
| 1. Friends and messaging system. | 1. Third-party integrations |

## Strategy

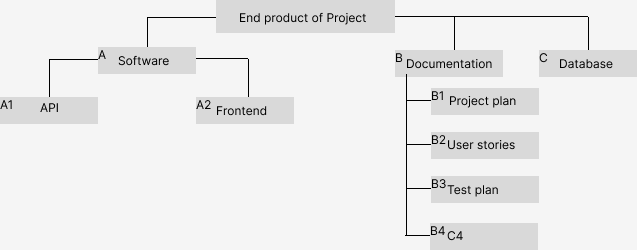
The project follows an Agile methodology with Scrum practices. The development process is managed using Jira, ensuring iterative development and continuous feedback. Each sprint will include planning, development, testing, and review phases.

## Research questions and methodology

* How to implement safe and restricted HTML/CSS customization to prevent security vulnerabilities?
* What are the best practices for database security in a social networking platform?
* How to optimize performance for real-time interactions (e.g., messaging, posts)?

## End products

* Functional social networking platform.
* System documentation (architecture, APIs, security guidelines).
* Test reports and security audit.
* Deployment and hosting strategy.



# Project organisation

## Stakeholders and team members

|  |  |  |
| --- | --- | --- |
| **Name** | **Role and functions** | **Availability** |
| Andrii Matviienko  Email: a.matviienko@student.fontys.nl | Project Owner & Developer | Full-time |
| Erik Schriek Email: e.vanderschriek@fontys.nl | Technical teacher | Every Tuesday from 09:00 until 13:00.  Every Friday from 13:00 until 16:00 |
| Maja Pesic  Email:  m.pesic@fontys.nl | Technical teacher | Every Wednesday from 09:00 until 13:00.  Every Friday from 09:00 until 13:00 |
| Jessie Cheung  Email: j.chua@fontys.nl | Project teacher | Every Tuesday From 13:00 untill 16:00  Every Wednesday from 13:00 until 16:00. |

## Communication

* **Development discussions:** Jira
* **Progress updates:** Feedback sessions
* **Code collaboration:** [GitLab repository.](https://git.fhict.nl/I531789/WiredSpace)

# Activities and time plan

## Phases of the project

1. **Research & Planning** (Security, Customization, Architecture).
2. **Development** (Backend, Frontend, Database Setup, Authentication, Customization Features).
3. **Testing & Optimization** (Security testing, performance tuning, user feedback).
4. **Deployment Strategy & Documentation**.

## Time plan and milestones

|  |  |  |
| --- | --- | --- |
| **Phasing** | **Start date** | **Finish date** |
| Sprint 1 | 20-02-2025 | 28-02-2025 |
| Sprint 2 | 01-03-2025 | 28-03-2025 |
| Sprint 3 | 29-03-2025 | 18-04-2025 |
| Sprint 4 | 19-04-2025 | 16-05-2025 |
| Sprint 5 | 17-05-2025 | 06-06-2025 |

# Testing strategy and configuration management

## 

## Testing strategy

* **Unit Testing:** Each component of the software, such as main logic, algorithm ect., will be tested in isolation to ensure that individual functions work correctly. Aim for a code coverage of at least 80% with these tests

* **User Acceptance Testing:** After the solution is developed, it will be thoroughly tested along with test plan. The test plan and solution will be sent to the test team, and they will star tests according to the test plan. Based on their analysis and Test Report, any identified issues will be mitigated.

* **Integration testing:** Integration testing will focus on verifying, that different components of the REST API, such as the database, external services, and third-party integrations, interact correctly. We will test API endpoints to ensure they return the expected responses and handle errors properly. Automated integration tests will be used to simulate real-world interactions and identify potential issues with data consistency, authentication, and request handling. Any issues discovered during testing will be addressed before moving to user acceptance testing.

## Test environment and required resources

Our testing environment will be set up using **Gradle with Git Runner**, ensuring automated and efficient testing throughout development. This will help maintain code quality and detect issues early in the process.

For version control and collaboration, we will follow the **GitHub Flow branching model,** which allows for streamlined development with feature branches, pull requests, and continuous integration. This approach will enable smooth code reviews, quick iterations, and seamless integration of new features while maintaining a stable main branch.

# Finances and risk

|  |  |  |
| --- | --- | --- |
| **Risk** | **Prevention activities** | **Mitigation activities** |
| 1. **Sudden illness of a developer** | **It cannot be prevented. There should always be a person ready to take over the tasks of the missing teammate.** | **As soon as developer feels bad, he should immediately inform other teachers** |
| 1. **Missing a deadline** | **Our time management is done in time, and the tasks are distributed evenly and must be done beforehand.** | **If our team missed a deadline, first of all, the client must get informed about the circumstances. Second, the team must properly edit the agenda to make sure that the missed deadline will not affect other tasks.** |
| 1. **Database malfunction** | **Prepare MOCK data to be able to showcase the functionality of the software.** | **In this case the team should have a copy of the existing database on a local server.** |
| **4.Hardware malfunction** | **Cannot be prevented. Extra hardware can be borrowed from the ISSD department.** | **Be ready to switch to different laptops and computers, so the best solution is to keep all files in Git repository.** |