**Project Plan**

*A white background with pink text

Description automatically generated*

|  |
| --- |
| **Date : 06/09/2023** |
| **Version : 1.0** |
| **State : Sprint 1** |
| **Author : Danko** |

#### Version history

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Date** | **Author(s)** | **Changes** | **State** |
| 1.0 | **06/09/2023** | Danko | First attempt | Sprint 1 |
|  |  |  |  |  |
|  |  |  |  |  |

Contents

[1. Project assignment 4](#_Toc145330416)

[1.1 Context 4](#_Toc145330417)

[1.2 Goal of the project 4](#_Toc145330418)

[1.3 Scope and preconditions 4](#_Toc145330419)

[1.4 Strategy 4](#_Toc145330420)

[1.5 End products 4](#_Toc145330421)

[2. Project organization 5](#_Toc145330422)

[2.1 Stakeholders and team members 5](#_Toc145330423)

[2.2 Communication 5](#_Toc145330424)

[3. Activities and time plan 6](#_Toc145330425)

[3.1 Phases of the project 6](#_Toc145330426)

[3.2 Time plan and milestones 6](#_Toc145330427)

[4. Testing strategy and configuration management 7](#_Toc145330428)

[4.1 Testing strategy 7](#_Toc145330429)

[4.2 Test environment and required resources 7](#_Toc145330430)

[4.3 Configuration management 7](#_Toc145330431)

[5. Finances and risk 8](#_Toc145330432)

[5.1 Risk and mitigation 8](#_Toc145330433)

# Project assignment

## Context

The clients are looking for a software system to enhance their babysitting services.

## Goal of the project

The aim of the project is to create a software application that provides fast and secure babysitting services through a website for the customers. The user-friendly interface and the minimal risk when hiring a babysitter are a big advantage compared to the “old” ways of settling between parents and unknown babysitters.

## Scope and preconditions

|  |  |
| --- | --- |
| **Inside scope:** | **Outside scope:** |
| 1. Babysitter can register | 1. Customers can call each other |
| 1. Parent can register | 1. A way to ban accounts |
| 1. Parent can see the available babysitters (sorting) |  |
| 1. Customers can manage their accounts |  |
| 1. A way of communication between a parent and a babysitter |  |
| 1. Admin can manage the users and posters |  |
| 1. Parent can manage the poster of their child |  |
| 1. Point system for babysitters so parents can contact “trusted” source |  |

***Preconditions***

We have some pre-determined limitations:

* It should comprise an Javascript framework based front-end part, and a RESTful API with relational database based back-end. An ORM should be used for database access.
* Preferably the tech stack should be: React frontend, Java Spring Boot REST API, JPA/Hibernate as the ORM and MySQL as the database.

## Strategy

The approach for the project is Kanban. The software will be developed in several iterations (sprints) with a deadline on 19.01.2024. The main reason behind the choice is the fact that the skills needed to complete the project are acquired during the semester.

## End products

* Source code – all the required code to run the website and the Unit tests.
* Project documentation

# Project organization

## Stakeholders and team members

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Abbreviation** | **Role and functions** | **Availability** |
| **Chung Kuah** | CK | Product owner | Tuesdays and Thursdays at Fontys |
| **Jacco Snoeren** | JS | Product owner | Tuesdays and Thursdays at Fontys |
| Danko Kralski | DK | Software developer | Every working day from 9am to 5pm. |

## Communication

The communication will be done either in person or online in Teams/via mail.

The contact emails are:

* **Chung Kuah -** [c.kuah@fontys.nl](mailto:c.kuah@fontys.nl)
* **Jacco Snoeren -** [j.snoeren@fontys.nl](mailto:j.snoeren@fontys.nl)
* Danko Kralski – [d.kralski@student.fontys.nl](mailto:d.kralski@student.fontys.nl)

# Activities and time plan

## Phases of the project

The planning will be done in Jira ([KAN board - Agile board - Jira (atlassian.net)](https://babysita.atlassian.net/jira/software/projects/KAN/boards/1)).There will be columns that represent the various stages/phases of the workflow:

* To Do: All the tasks or work items that need to be done.
* Sprint: The tasks needed to be finished in the current sprint.
* In Progress: Tasks that are actively being worked on (current focus).
* Testing/Review: Tasks that are completed but need validation or review.
* Done: Once a task is completed and has passed all necessary checks.

## Time plan and milestones

|  |  |  |
| --- | --- | --- |
| **Phasing** | **Start date** | **Finish date** |
| 1. Java introduction, Project plan, RESTfull service, CI | 04/09/2023 | 22/09/2023 |
| 1. C4 Model diagrams, CORS configuration, Initial Frontend setup | 23/09/2023 | 13/10/2023 |
| 1. Design documentation V2, Initial Backend to Database setup, SonarQube | 14/10/2023 | 10/11/2023 |
| 1. Design document V3, Authentication and authorization implementation, Continuous Integration and Sonarqube | 11/11/2023 | 01/12/2023 |
| 1. Final design document, Security report, Websockets feature, MVP, Continuous Integration and Sonarqube | 02/12/2023 | 22/12/2023 |
| 1. Final UX feedback report, Final product, Continuous Delivery, Continuous Integration and Sonarqube | 23/12/2023 | 19/01/2024 |

# Testing strategy and configuration management

## 

## Testing strategy

***Unit Testing:***

* JUnit and Unit to test the CRUD functionality of the program.

***Integration Testing:***

* Integration test using Jest and Cypress.

***Error Handling:***

* Unit test to check for incorrect input and show an error message explaining the reason behind it.

## Test environment and required resources

**Frontend:**

Jest will be used to test the frontend part of the product and in later stages E2E testing will be done using Cypress.

**Backend:**

Unit and JUnit test will be used locally to test the CRUD functionality of the product. (Mockito)

Testing controllers on the API level (HTTP requests and responses) using Spring WebMvcTest.

## Configuration management

The source code will be located in GitLab and will be published by using branching.

# Finances and risk

## Risk and mitigation

|  |  |  |
| --- | --- | --- |
| **Risk** | **Prevention activities** | **Mitigation activities** |
| 1. Not fulfilling the expected requirements | Weekly contact with client | Contact client online/in person |
| 1. Too much work overflow | Careful planning for execution | Ask for help |
| 1. Sick team member |  | Contact teachers/clients |
| 1. Change of Google Maps API | Check for updates on their API | Find a Long-supported API |