**Project Plan**

***ChessNow***

*Balan Mihai-Stefan – Indivdual Assignment*

|  |
| --- |
| **Date : 08/09/2023** |
| **Version : 01** |
| **State : Incomplete** |
| **Author : Balan Mihai-Stefan** |

#### Version history

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Date** | **Author(s)** | **Changes** | **State** |
| 1 | 08/09/2023 | Balan Mihai-Stefan | Initial plan | Incomplete |
|  |  |  |  |  |
|  |  |  |  |  |

**Distribution**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Receivers** |
| 1 | 22/09/2023 | Teachers |
|  |  |  |

Contents

[1. Project assignment 4](#_Toc42673512)

[1.1 Context 4](#_Toc42673513)

[1.2 Goal of the project 4](#_Toc42673514)

[1.3 Scope and preconditions 4](#_Toc42673515)

[1.4 Strategy 4](#_Toc42673516)

[1.5 Research questions 4](#_Toc42673517)

[1.6 End products 4](#_Toc42673518)

[2. Project Organisation 6](#_Toc42673519)

[2.1 Stakeholders and team members 6](#_Toc42673520)

[2.2 Communication 6](#_Toc42673521)

[3. Activities and time plan 7](#_Toc42673522)

[3.1 Phases of the project 7](#_Toc42673523)

[3.2 Time plan and milestones 7](#_Toc42673524)

[4. Testing strategy and configuration management 8](#_Toc42673525)

[4.1 Testing strategy 8](#_Toc42673526)

[4.2 Test environment and required resources 8](#_Toc42673527)

[4.3 Configuration management 8](#_Toc42673528)

[5. Finances and Risk 9](#_Toc42673529)

[5.1 Project budget 9](#_Toc42673530)

[5.2 Risk and mitigation 9](#_Toc42673531)

# Project assignment

## Context

This project is meant as a university individual project. The stakeholders are my teachers. This project has strict requirements and will be developed by the author alone.

## Goal of the project

ChessNow is a platform for interacting and engaging with people in the mind stimulating game of chess. Users of the platform will create accounts, make friends, challenge them and improve their skill, which is reflected in their rating. Users will also be able to communicate with other parties and rate their sportsmanship. In case of a misuse of any of the platform’s features, administrators will be required to intervene and set everything to normal.

## Scope and preconditions

The platform’s inner workings are clearly divided and organized, always leaving room for expansion, as a result of end user feedback. The project is built from scratch, no pre-developed parts of the application are present.

|  |  |
| --- | --- |
| **Inside scope:** | **Outside scope:** |
| 1. Web platform with functional chess game | 1. Independent downloadable application |
| 1. User side functionalities (login, profile, email,etc.) | 1. Chat room for people who are friends on the platform and are not in-game |
| 1. Administrator view for rule enforcement | 1. Adjustable chess bot difficulty for not logged in users. |
| 1. Rating system for non-administrator users | 1. Two factor authentication for users. |
| 1. Cross user interaction: In-game chat, reporting innapropriate conduct, friend adding and removing, game invitation |  |
| 1. Statistics of matches and rating over time |  |
| 1. Chess bot gameplay for users who do not have an account. |  |

## Strategy

This project will be developed according to the AGILE method, the development period being divide in 6 total sprints.

* High Priority Tasks: Setting up the triple layered architecture, setting up the user-related functionalities, developing CRUD functionalities for the app, creating the fron-end interface in ReactJs and linking it to the Java back-end, developing a functional game, satisfying all the technical requirements for the project.
* Medium Priority Tasks: Upgrading the user interface in a visually pleasing way, adding animations to the game, adding non-mandatory user to user interactions (example: sportsmanship rating)
* Low Priority Tasks: Upgrading the chess Ai system, adding email verification and recovery of password via email.

## Research questions and methodology

1. User Needs and Preferences

RQ1.1: What are the primary needs and preferences of chess enthusiasts and players in terms of a digital chess application?

Methodology: Conduct user surveys and interviews to gather user feedback and requirements.

1. Competitive Research

RQ1.2: What features and functionalities do existing chess applications offer, and how does our project differentiate itself from them?

Methodology: Perform a competitive analysis by evaluating and comparing popular chess applications.

1. Chess Engine Development

RQ1.3: What is the most suitable chess engine algorithm for our application, and how can it be optimized for performance and accuracy?

Methodology: Research various chess engine algorithms, implement them, and conduct benchmark testing.

1. User Experience (UX)

RQ1.4: How can I ensure a seamless and enjoyable user experience within the application?

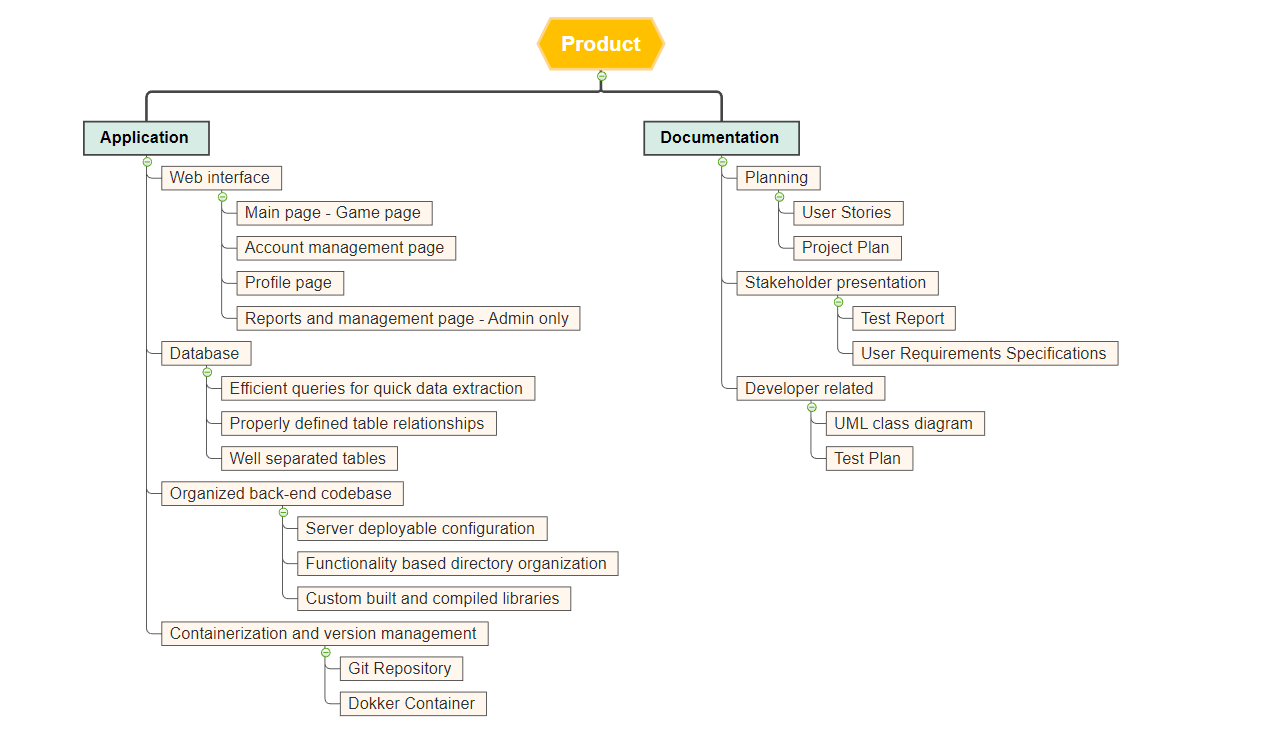
Methodology: Conduct usability testing and heuristic evaluations to identify and address UX issues.

1. Security and Fair Play

RQ1.5: How can I prevent cheating and ensure fair play in online chess matches?

Methodology: Research and implement anti-cheating measures, including engine move comparison algorithms.

## End products



# Project organisation

## Stakeholders and team members

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Abbreviation** | **Role and functions** | **Availability** |
| *Balan Mihai* | *BMS* | *Developer* | *Monday - Friday* |
| *Pesic Maja* | *M* | *Stakeholder* | *Wednesday, Friday* |
| *Rabeling Bart* | *BT* | *Stakeholder* | *Monday, Wednesday* |
| *Zuurbier-Munneke Nicole* | *N* | *Stakeholder* | *Monday, Friday* |

## Communication

* Communication will be either physical or via internet.

In case of physical communication, meetings will be held at Fontys Rachesmolen R10, with variable time slots.

In case of internet based communications, meetings will be held via Microsoft Teams, and non urgent matters will be resolved via email, using the Outlook mailing platform

* Sprint deliveries will be made via canvas, in the specific submission section. A clone of the submission will be also stored inside the attributed git repository.

# Activities and time plan

## Phases of the project

The product will be delivered at the end of each sprint. In total there are 6 sprints, of 3 weeks each, totaling at 18 weeks of development.

Sprint 1:

* Documentation – Project Plan, Initial Product Backlog, Initial User stories
* Notion board development setup
* 3-layered architecture, presentation->logic->data
* 3 working RESTful services
* Git repository
* Build and Test stages

# Testing strategy and configuration management

## 

## Testing strategy

*Testing will be done via unit tests. Each class will be tested individually before integration. The entire application will be tested during development, after each significant feature completion.*

## Test environment and required resources

*Tests will be conducted using IntelliJ Java development platform, making use of test classes. The full app tests will be run from the web interface of the application or via the Postman client.*

## Configuration management

*Version management will be done via GitLab, using a repository organized in two segments. Inside the documentation folder on the top layer, every document will be placed, as well as the pdfs for the sprint submissions. Repository can be found at* [*https://git.fhict.nl/I501909/individual-project-sem-3/-/tree/main?ref\_type=heads*](https://git.fhict.nl/I501909/individual-project-sem-3/-/tree/main?ref_type=heads) *.*

*Branching strategy: Master branch is going to be used for regular development. Main branch will have only stable versions of the application, for example a Sprint submission. Addidtional branches may be used for feature development and for testing.*

*My reference for the branching strategy can be found at* [*https://nvie.com/posts/a-successful-git-branching-model/*](https://nvie.com/posts/a-successful-git-branching-model/) *.*

# Risks and Mitigation

|  |  |  |
| --- | --- | --- |
| **Risk** | **Prevention activities** | **Mitigation activities** |
| 1. Develoment time related issues | Careful planning of each project feature in phasing | Discarding of features in case of emergencies. |
| 1. Data loss | Having backup of everything in at least two places (e.g. Git and PC, or Cloud, mirror database) | Restoration of a backup. |
| 1. Project not being satisfactory | Checking the requirements before developing any feature, making sure the basics are implement before the extras. | Replacement of features with more compliant version. |
| 1. Planned features being beyond the developer’s abilities | Research on technical features, as well as other similar working products available. | Consultation with a teacher, looking for a simpler, more or less elegant solution. |