**Software Requirements Specification**

For

Shoot’n Shield

01-05-2024

Prepared by

|  |  |  |
| --- | --- | --- |
| **Name** | **Roll Number** | **Specialization** |
| Shahan Ali | R2142211305 | Gaming and Graphics |

Project Guide

Alind Sir



Department of Systems

School Of Computer Science

UNIVERSITY OF PETROLEUM & ENERGY STUDIES,

DEHRADUN- 248007. Uttarakhand

Table of Contents

|  |  |  |
| --- | --- | --- |
| **Topic** | | **Page Number** |
| Table of Content | | 2 |
| Revision History | | 3 |
| 1 | Introduction | 4 |
|  | 1.1 Purpose of the Project | 4 |
|  | 1.2 Target Beneficiary | 4 |
|  | 1.3 Project Scope | 4 |
|  | 1.4 References | 4 |
| 2 | Project Description | 5 |
|  | 2.1 Reference Algorithm | 5 |
|  | 2.2 Data/ Data structure | 5 |
|  | 2.3 SWOT Analysis | 6 |
|  | 2.4 Project Features | 6 |
|  | 2.5 User Classes and Characteristics | 7 |
|  | 2.6 Design and Implementation Constraints | 7 |
|  | 2.7 Design diagrams | 8 |
|  | 2.8 Assumption and Dependencies | 9 |
| 3 | System Requirements | 10 |
|  | 3.1 User Interface | 10 |
|  | 3.2 Software Interface | 10 |
|  | 3.3 Database Interface | 10 |
|  | 3.4 Protocols | 10 |
| 4 | Non-functional Requirements | 11 |
|  | 4.1 Performance requirements | 11 |
|  | 4.2 Security requirements | 11 |
|  | 4.3 Software Quality Attributes | 11 |
| 5 | Other Requirements | 11 |
| Appendix A: Glossary | | 13 |
| Appendix B: Analysis Model | | 13 |
| Appendix C: Issues List | | 13 |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Change** | **Reason for Changes** | **Mentor Signature** |
| 28-02-2023 | Initial project kick-off | Initialized documentation |  |
| 01-03-2023 | Added use case and main structure | Defined project structure |  |
| 04-03-2023 | Designed Algorithm | Main part of project |  |
| 05-03-2023 | Implemented data loading from a file | Incorporating external data |  |
| 10-03-2023 | Enhanced element consecutive checking | Better user feedback |  |
| 19-03-2023 | Frontend development | Better UI |  |
| 25-03-2023 | Backend and Frontend combining | Last step |  |
| 01-04-2023 | Conducted extensive testing | Ensuring program stability |  |
| 20-04-2023 | Finalized documentation | Preparing for project completion |  |
| 30-04-2023 | Finalized Project | Project Complete |  |

1. INTRODUCTION

1.1 Purpose of the Project:

The purpose of the Shoot’n Shield project is to develop an engaging 2D game that offers players a thrilling and immersive gaming experience. By combining intuitive gameplay mechanics with captivating visuals and sound design, the project aims to provide players with hours of entertainment and enjoyment.

1.2 Target Beneficiary:

The target beneficiaries of the project are gaming enthusiasts of all ages who enjoy action-packed arcade-style games. Additionally, the project aims to appeal to casual gamers looking for a fun and accessible gaming experience that can be enjoyed on various platforms.

1.3 Project Scope:

The scope of the project includes designing and implementing gameplay mechanics, graphics, sound effects, and user interfaces for the 2D game. The project will leverage the Unity game engine and C# programming language to create a seamless gaming experience across different devices and platforms.

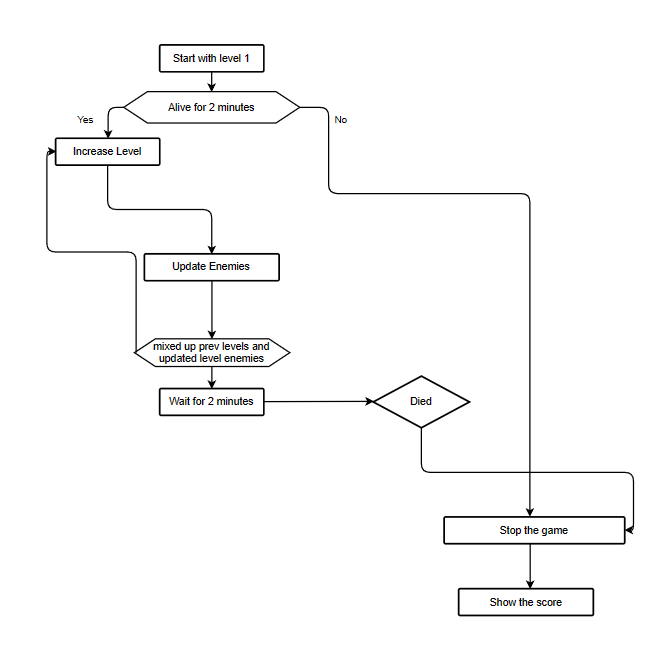
1.4 References:

No external references are included in this SRS. All information and requirements are based on internal project knowledge and objectives.

2. PROJECT DESCRIPTION

2.1 Reference Algorithm:

The game mechanics will include algorithms for enemy generation, player movement, collision detection, scoring, and fire.



2.2 Characteristic of Data:

Data structures such as arrays, lists, and dictionaries will be utilized to manage game elements like player attributes, enemy attributes, fire store, and scores.

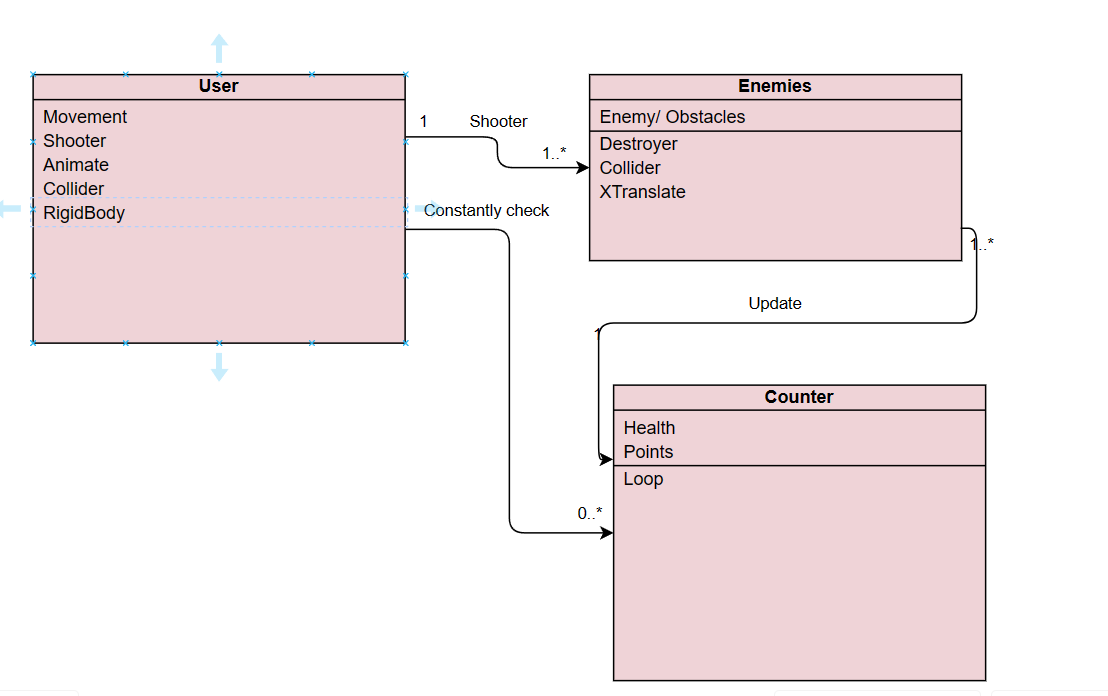
2.3 SWOT Analysis:



2.4 Project Features:

* Endless shooter gameplay
* Sprite-based visuals
* Dynamic enemy generation
* Intuitive controls
* Score-based progression
* Multi-platform deployment

2.5 User Classes and Characteristics:

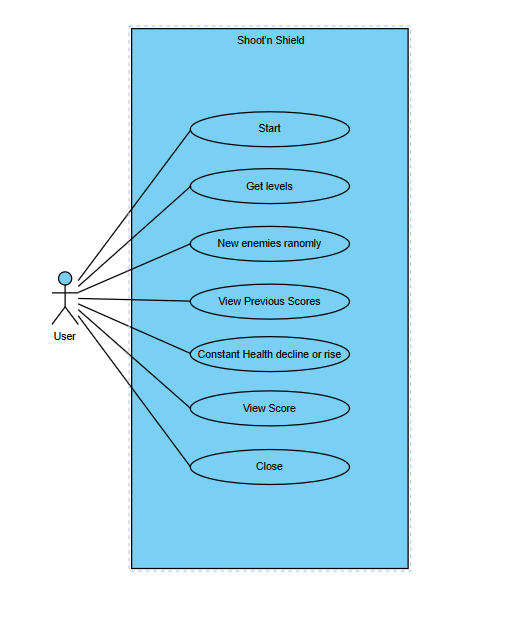


2.6 Design and Implementation Constraints:

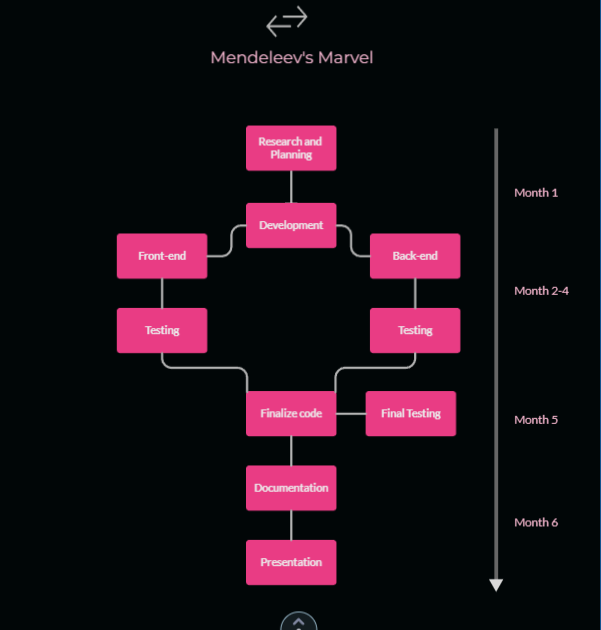
Design and implementation constraints may include limitations in hardware capabilities, compatibility issues across platforms, and resource constraints such as memory and processing power.

2.7 Design Diagrams:

* USE-Case Diagram:



* Pert Chart



2.8 Assumptions and Dependencies:

* Assumptions:
  + Players have basic familiarity with 2D game mechanics and controls.
  + The target platform supports Unity game engine and C# programming language.
* Dependencies:
  + Availability of resources such as graphics, sound effects, and music assets.
  + Integration of third-party plugins or libraries for specific functionalities (e.g., ads, analytics).

3. SYSTEM REQUIREMENTS

3.1 User Interface:

The user interface should be intuitive and user-friendly, with controls for player movement, shooting. Visual elements should be clear and engaging, with smooth transitions and animations.

3.2 Software Interface:

The game should be compatible with operating systems such as Linux and Windows, utilizing appropriate APIs and libraries for platform-specific functionalities.

3.3 Database Interface:

While Shoot’n Shield may not require a traditional database interface, it may utilize local storage or cloud-based services for storing player data, scores, and settings.

3.4 Protocols:

The game may utilize standard communication protocols for online features such as leaderboards, multiplayer modes, or social integration.

4. NON-FUNCTIONAL REQUIREMENTS

4.1 Performance Requirements:

* Smooth gameplay with minimal lag or stuttering.
* Fast loading times for levels, assets, and menus.
* Efficient use of system resources to ensure compatibility with a wide range of devices.

4.2 Security Requirements:

* Implementation of secure data transmission protocols for online features.
* Protection against cheating or exploitation of game mechanics.

4.3 Software Quality Attributes:

* Reliability: The game should run smoothly without frequent crashes or glitches.
* Usability: The user interface should be intuitive and easy to navigate.
* Maintainability: The codebase should be well-structured and documented for ease of maintenance and future updates.

5. OTHER REQUIREMENTS

5.1 Educational Content:

* Accuracy: Ensure that any educational content, such as information about enemies or power-ups, is factually accurate and aligned with the game's theme.
* Learning Element: The game should have an educational focus, encouraging players to strategize and learn about different enemy abilities and weaknesses.
* Engagement: The game should be engaging and fun, motivating players to continuously improve their skills and explore new gameplay mechanics.

5.2 Game Mechanics:

* Game Over Condition: Define clear conditions for ending the game, such as player health reaching zero due to enemy attacks.
* Scoring System: Design a scoring mechanism that rewards players for defeating enemies and enhancing highest store, providing a sense of achievement and progression.
* User Feedback: Provide immediate feedback to players after each action, indicating successful enemy hits, health changes, and point accumulation.

5.3 User Experience:

* User-Friendly Interface: Design a visually appealing and intuitive user interface, including menus, HUD elements, and in-game notifications.
* Sound and Graphics: No sound needed and sprites are mainly used in the game.

5.4 Platform Compatibility:

* Operating Systems: Ensure compatibility with common operating systems such as Windows, macOS, and Linux, allowing a wide range of players to enjoy the game.
* Graphics Libraries: Specify the use of appropriate graphics libraries for smooth rendering and compatibility across different hardware configurations.

5.5 Licensing and Distribution:

* Open Source: Declare the project's open-source nature, including the licensing terms and where the source code can be accessed.
* Distribution: Determine the distribution method, whether through online platforms, direct download, or physical media.

5.6 Documentation:

* User Manual: Provide a user manual or in-game instructions explaining how to play the game.
* Code Documentation: Document the code comprehensively to assist future developers or contributors.

5.7 Testing:

* Quality Assurance: Define a testing plan that includes unit testing, integration testing, and user testing to ensure the game is error-free.
* Bug Reporting: Specify a mechanism for users to report any bugs or issues encountered during gameplay.

Appendix A: Glossary

* SRS: Software Requirement Specification
* GUI: Graphical User Interface
* C#: A programming language used for the development of the project.
* Assets: Graphics, sound effects, and other media used in the game.

Appendix B: Analysis Model

The analysis model for this project involves the creation and maintenance Unity based structure to store and create data. This model guides the game's functionality and logic.

Appendix C: Issues List

This is a dynamic list of open requirements issues. It may include items related to coding, testing, user feedback, or future enhancements. The issues list is maintained and updated throughout the project's development cycle to track and address any pending tasks, challenges, or improvements.