SOFTWARE REQUIREMENTS SPECIFICATION

for

Endless Pinball

Version

Prepared by : Tristan Zippert
Donato Apon
David DiFrumolo
Gedeon Rugema
Connor Lariviere

Organization : CircuitCity

March 30, 2022

Contents

Revision History

Name	Date	Reason for Changes	Version
Tristan, Donato, David, Gedeon, Connor	2/18/2022	Added Functional Requirements	0.01
Tristan, Donato, David, Gedeon, Connor	3/2/2022		0.02
Tristan, David	3/30/2022	Added Overall Description	0.03

1 Introduction

1.1 Purpose

The purpose of this document is to outline the requirement specification for the Endless Pinball game and its connected website. The scope of this document is to outline the interactions the user would have with the game, as well as to highlight what needs to be done during the development phase of this project.

1.2 Document Conventions

1.3 Intended Audience and Reading Suggestions

1.4 Product Scope

1.5 References

2 Overall Description

2.1 Product Perspective

This SRS defines a new self-contained product that will exist on its own on mobile devices. The context of this SRS explains both the application and the accompanying website.

2.2 Product Functions

The application is designed to be a continuous point scoring video game until failure.The player will have tools to move across their infinite board while collecting points which will be counted until the player loses its life variable.

- Touch Input to move pinball
- Avoid Obstacles to maintain lives
- Rewards/Objectives on the board to increase score

2.3 User Classes and Characteristics

We anticipate that iOS and Android users will use this product, and that they will have a simple understanding of how menus work. We also anticipate that the players would already be familiar with being able to read online leaderboards.

2.4 Operating Environment

The software will operate on modern smartphone devices, such as the most recent update of IOS and Android (as of 3/22/2022).

2.5 Design and Implementation Constraints

This application will be created with the Bevy game engine, which is a relatively new game engine compared to Unity or Godot. Bevy doesnt currently have much support for smartphone builds, however pull requests are being added everyday that add functionality. With the Bevy game engine, the game will be architectured in the ECS (EntityComponentSystem) paradigm. This software will also be written in the Rust programming language due to the fact that it's a low level language, but also easy to set up and doesnt have as many safety vulnerabilities compared to C and C++. The application will use POST and GET requests to communicate with the associated website.

The associated website will use the Rust Sauron and Diesel.rs to display and manage the score database. Because of the access to a persistent database, we will be taking extra consideration in using an ORM (Diesel.rs) instead of directly communicating with the Postgres database.

2.6 User Documentation

Documentation will be provided in the form of Youtube videos explaining aspects of the application, such as uploading your score or how to play the game.

2.7 Assumptions and Dependencies

We assume that the Rust programming language is able to communicate with and make HTTP requests to the website through a mobile application, and we also assume that it will be able to access the filesystem of the mobile device without any restrictions as to save data about the game.

3 External Interface Requirements

3.1 User Interfaces

The software outlined in this document will have two sets of user interfaces, one for the website and one for the Endless Pinball game itself.

Endless Pinball: The Endless Pinball game will be designed using the **MVC** architecture design pattern for its UI as we want the controller to handle any interactable scenarios for the user. The controller would handle the events that the user would use to control the pinball, and to select in the pause menu. The view, in our case, would display data such as the score the user currently has and what power-up they have. The Model part would manage all the incoming data and the View would display it.

Endless Pinball website: Our website would use the Client-Server architecture, as it is intended just to display the high scores from the game. The website would receive POST requests from the game, and would have no controller for user interaction. This will allow the user to visit the site, and the site will update the view and display it to the user when a new high score is received.

3.2 Hardware Interfaces

The game will interact with the phones vibration motors to make the game more interactive, and could also interact with the phones gyro sensor to control the pinball. The game will also interact with the phones internal GPU in order to render the game.

3.3 Software Interfaces

3.4 Communications Interfaces

The game will use GET requests in the form of JSON to the game application, and the game application will use POST to post user uploaded pictures and the high score from the user.

4 System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

4.1 Pinball Game System Feature

4.1.1 Description and Priority

4.1.2 Stimulus/Response Sequences

4.1.3 Functional Requirements

Functional Requirements are as follows for both the Endless Pinball app and the Endless Pinball website.

REQ-1: The Endless Pinball system shall display a continuous path for the user until failure
REQ-2: The Endless Pinball system shall save the users high score data.
REQ-3: The Endless Pinball system shall register inputs from the user, to manipulate the pinball.
REQ-4: The Endless Pinball system shall select a specific color for their pinball.
REQ-5: The Endless Pinball system shall allow the user to adjust the volume.
REQ-6: The Endless Pinball system shall allow the user to replay the game after failure
REQ-7: The Endless Pinball system shall allow the user to select the music that plays during the game.
REQ-8: The Endless Pinball system shall allow the user to select a theme for their pinball board.
REQ-9: The Endless Pinball system shall play specific sound effects when the pinball collides with objects.
REQ-10: The Endless Pinball system shall simulate realistic physics bounces for the pinball.
REQ-11: The Endless Pinball system shall save the users highest score.
REQ-12: The Endless Pinball system shall vibrate the users device when the pinball collides into objects.
REQ-13: The Endless Pinball system shall subtract a life point from the user when they fail.
REQ-14: The Endless Pinball system shall subtract from the users high score when a life point is removed from the user
REQ-15: The Endless Pinball system shall reset the round when all of the life points are subtracted from the user.

4.2 System Feature 2

5 Other Nonfunctional Requirements

REQ-1: The Endless Pinball system shall load the players pinball in 3 seconds 95% of the time.
REQ-2: The Endless Pinball system website shall be able to load within 1 second 98% of the time.
REQ-3: The Endless Pinball system shall display a replay button in 3 seconds after failure 95% of the time
REQ-4: The Endless Pinball system shall indicate a life point loss within 1 second 95% of the time

REQ-5: The Endless Pinball system shall provide feedback within one second of the users request 99% of the time

REQ-6: The Endless Pinball system website shall be able to receive web requests from the endless Pinball system within 1 second of data being sent 98% of the time.

REQ-7: The Endless Pinball system shall be able to send data successfully to the website 96% percent of the time.

REQ-8: The Endless Pinball game system shall be available to the users 99%z of the time.

REQ-9: The website system shall be able to log data without error to the database 95% of the time.

REQ-10: The website system shall be available to users twenty-four-seven 90% of the time.

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

5.5 Business Rules

6 Other Requirements

Appendix A: Glossary

MVC (Model View Controller): Software design pattern commonly associated with user interfaces that divide into connected elements such as: the *Model*, *View*, and the *Controller*

ECS (Entity Component System): System architecture paradigm that follows composition of entities with associated types ECS architecture description

Appendix B: Analysis Models