# SOFTWARE REQUIREMENTS SPECIFICATION (SRS) FOR

Project Name: Project Horseman

Github:

https://github.com/Jean-H77/2DGame

Team Name: RoguePixel Games

Class: COMP490

Instructor: Edmund Dantes

## Revision History

| Revision Letter | Primary Author | Description of Changes | Date completed |
|---|---|---|---|
| 0/- | All Team Members | Original Document | 10/08/23 |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

Table of Contents

**Table of Figures**

**List of Tables**

## 1. INTRODUCTION

### 1.1 Scope

This Software Requirement Specification (SRS) outlines the specific requirements and design considerations for Project Horseman, a 2D Halloween-themed gaming experience. Beyond merely providing a blueprint for gameplay mechanics and narrative elements, this document also encompasses design specifications, and potential user interface (UI) and user experience (UX) pathways. Developed by the RoguePixel Games team of the California State University Northridge located in Northridge, CA. This game aims to merge the rich cultural backdrop of spooky folklore with contemporary gaming techniques, ensuring an engaging and atmospheric playthrough.

### 1.2 Product Value

Project Horseman offers players a unique blend of narratives and challenging gameplay. Set against a backdrop of horror landscapes, it taps into the nostalgia of classic 2D games while introducing innovative mechanics.

### 1.3 Intended Audience

Our primary audience encompasses gamers who have a liking for adventure and mystery. This includes individuals who appreciate a balanced blend of story-driven quests and skill-based challenges.

### 1.4 Intended Use

The game serves as an entertaining medium for players to immerse themselves in a virtual ghostly world. Beyond mere entertainment, Project Horseman also provides an opportunity for players to solve in-game puzzles, communicate with other characters, and escort someone, fostering individual skill enhancement.

## 2. FUNCTIONAL REQUIREMENTS

**FUNC_SRS_001:** The game shall provide the ability to move the character up/down/left/right

**FUNC_SRS_002:** The game shall have a quest system.

**FUNC_SRS_003:** The game shall support single-player mode.

**FUNC_SRS_004:** Project Horseman will incorporate a save and load system, allowing players to continue from their last checkpoint.

**FUNC_SRS_005:** The game's design shall include interactive NPCs (Non-Playable Characters) that provide quests, and trade items.

**FUNC_SRS_006:** Project Horseman will have an inventory system that players can access to view collected items, equip gear, and use consumables.

**FUNC_SRS_007:** The game shall be optimized for desktop operating systems including Windows 7-10, MacOS 10.12-11, and Linux distributions.

**FUNC_SRS_008:** The game will have adaptive background music and sound effects that respond to in-game events and player actions.

**FUNC_SRS_009:** Project Horseman shall feature a distinct map area. It will consist of challenges, enemies, and narratives.

**FUNC_SRS_010:** The system will be using libGDX and written in Java.

## 3. EXTERNAL INTERFACE REQUIREMENTS

3.1 User Interface Requirements

**Design Guides:**

- Menu elements must be centrally located or presented in a manner that facilitates rapid access to desired functions, such as starting a game or loading content.
- During gameplay, in-game UI elements should be strategically positioned along the screen's periphery, ensuring that the central action space remains unobstructed to maintain smooth gameplay flow.
- Player Status UI elements should be designed for clarity, making it easy for players to understand and utilize the provided information to enhance their gameplay and decision-making abilities.

**Style Guide:**

The game will feature a pixelated art style inspired by classic Nintendo games, such as Pokémon and Legend of Zelda. It will utilize a 32-bit pixel art design, aiming to evoke nostalgia and pay homage to retro gaming aesthetics.

**EXTINTF_SRS_001:** The game shall have a Main Menu with the following elements:

1. Play button
2. Settings button
3. Load Game button
4. Quit button
5. About button

**EXTINTF_SRS_002:** The game shall have a Settings Menu with the following elements:

1. Audio: Sound Effects Volume
2. Audio: Music Volume
3. Brightness: Increase
4. Brightness: Decrease
5. Screen Resolution: Adjust the screen size

**EXTINTF_SRS_003:** The game shall have a Player Status UI with the Health Bar

**EXTINTF_SRS_004:** The game shall have a Player Inventory UI with the following elements:

1. Player's inventory
2. Method to navigate said inventory
3. Selected Item Name
4. Selected Item Description

**EXTINTF_SRS_005:** The game shall have a dialogue system.

1. Dialogue Box
2. Space bar to go to next dialogue

**EXTINTF_SRS_006:** The game shall have a quest overlay.

1. Transparent overlay
2. Displays the current stage of the quest and description.

## 3.2 Hardware Interface Requirements

**EXTINT-HI-001:** The game shall run on personal computers
Intended Minimum Hardware Requirements: AMD Ryzen 5 5500U, 8GB RAM, 256GB Storage, AMD Radeon 7 Graphics, DirectX 10 compatibility.

## 3.3 Software Interface Requirements

**ExtInt-SI-001:** The game shall be developed for Windows, Mac, and Linux OS`

## 3.4 Communication Interface Requirements

**ExtInt-CI-001:** TBD

## 3.5 CSCI Internal Interface Requirements

**ExtInt-IR-001:** TBD

## 3.6 CSCI Internal Data Requirements

**ExtInt-ID-001:** TBD

## 4. NON-FUNCTIONAL REQUIREMENTS

4.1 Security
**NONFUNC_SRS_001:** The game can only be downloaded from secure applications, such as Steam and Itch.io

4.2 Capacity
**NONFUNC_SRS_002:** TBD

4.3 Compatibility
**NONFUNC_SRS_003:** The software will specify minimum hardware requirements for different platforms (Windows, MacOS, Linux) to ensure that it runs smoothly. These requirements are detailed in the "3.2 Hardware Interface Requirements" section of the project plan.

4.4 Reliability
**NONFUNC_SRS_004:** The game would crash if the player does not have the assets.

4.5 Scalability
**NONFUNC_SRS_005:** The game will have the option to disable weather effects and change the width and height of the game screen.

4.6 Usability
**NONFUNC_SRS_006**: The software should be user-friendly, with an intuitive user interface and clear instructions. It should be easy for players of various experience levels to navigate, control the game character, interact with NPCs, and manage their inventory. Usability testing will be conducted to ensure the software meets these criteria. Steam availability is TBD.

4.7 Other
Not Applicable

## 5. QUALIFICATION PROVISIONS

Qualification in this specification is interpreted as requirement verification. The following are the base definitions for the verification methods.
A – Analysis: Use of analytical data or simulations under defined conditions to show theoretical compliance. Used where testing to realistic conditions cannot be achieved or is not cost-effective. Analysis (including simulation) may be used when such means establish that the appropriate requirement, specification, or derived requirement is met by the proposed solution. Examples include the reduction, interpretation or extrapolation of test data.
D – Demonstration: A qualitative exhibition of functional performance, usually accomplished with no or minimal instrumentation. Demonstration (a set of test activities with stimuli selected by the developer) may be used to show that the CSCI, or a part of the CSCI, response to stimuli is suitable (e.g. observation of fin deployment, etc.). Demonstration may be appropriate when requirements or specifications are given in statistical terms (e.g. mean time to repair, etc.).
I – Inspection: The examination of the CSCI code against applicable documentation to confirm compliance with requirements. Inspection is used to verify properties best determined by examination and observation.
T – Test: An action by which the operability, supportability, or performance capability of the CSCI, or a part of the CSCI, is verified when subjected to controlled conditions that are real or simulated. These verifications often use special test equipment or instrumentation to obtain very accurate quantitative data for analysis.
An example verification table is provided below but may be replaced by tables auto-generated from the requirements management tool if used

**Table I. Requirements Verification**

| **SRS Req. ID** | **Paragraph Title** | **Verification Method** |
|---|---|---|
| FUNC_SRS_001 | The game shall provide the ability to move the character up/down/left/right | Demonstration |

| | | |
|---|---|---|
| FUNC_SRS_002 | The game shall have a quest system. | Test |
| FUNC_SRS_003 | The game shall support single-player mode. | Test |
| FUNC_SRS_004 | Project Horseman will incorporate a save and load system, allowing players to continue from their last checkpoint. | Test |
| FUNC_SRS_005 | The game's design shall include interactive NPCs (Non-Playable Characters) that provide quests, and trade items. | Test |
| FUNC_SRS_006 | Project Horseman will have an inventory system that players can access to view collected items, equip gear, and use consumables. | Test |
| FUNC_SRS_007 | The game shall be optimized for desktop operating systems including Windows 7-10, MacOS 10.12-11, and Linux distributions. | Test |
| FUNC_SRS_008 | The game will have adaptive background music and sound effects that respond to in-game events and player actions. | Test |
| FUNC_SRS_009 | Project Horseman shall feature a distinct map area. It will consist of challenges, enemies, and narratives. | Analysis |
| FUNC_SRS_010 | The system will be using libGDX and written in Java. | Inspection |
| EXTINTF_SRS_001 | Main Menu | Demonstration |
| EXTINTF_SRS_002 | Settings Menu | Demonstration |
| EXTINTF_SRS_003 | Player Health Bar | Demonstration |
| EXTINTF_SRS_004 | Player Inventory System | Demonstration |
| EXTINTF_SRS_005 | Dialogue System | Demonstration |
| EXTINTF_SRS_006 | Quest System | Demonstration |
| EXTINT-HI-001 | The game shall run on personal computers | Test |
| ExtInt-SI-001 | The game shall be developed for windows, mac, and Linux OS` | Test |

| | | |
|---|---|---|
| NONFUNC_SRS_001 | Actual requirement (SHALLS) | Test |
| NONFUNC_SRS_002 | TBD | TBD |
| NONFUNC_SRS_003 | The software will specify minimum hardware requirements for different platforms (Windows, MacOS, Linux) to ensure that it runs smoothly. These requirements are detailed in the "3.2 Hardware Interface Requirements" section of the project plan. | Demonstration |
| NONFUNC_SRS_004 | The game would crash if the player does not have the assets. | Test |
| NONFUNC_SRS_005 | The game will have the option to disable weather effects and change the width and height of the game screen. | Test |
| NONFUNC_SRS_006 | The software should be user-friendly, with an intuitive user interface and clear instructions. It should be easy for players of various experience levels to navigate, control the game character, interact with NPCs, and manage their inventory. Usability testing will be conducted to ensure the software meets these criteria. Steam availability is TBD. | Demonstration |

## 6. NOTES

6.1 Acronyms and Abbreviations

**Table II. Acronyms and Abbreviations**

| Abbreviation | Full name |
|---|---|
| **2D** | **Two Dimensional** |
| **NPC** | **Non-player Character** |
| **OS** | **Operating System** |
| **UI** | **User Interface** |
| **UX** | **User Experience** |