|  |
| --- |
| HCCOMP2, HCCOMPE2, BSHC2, BSHCE2, BSHTM2 |
| Requirements Specification (RS) |
|  |

|  |
| --- |
|  |

Domas Brazdeikis x22728815

Aaron Smyth x22433652

Adam Plesca

x22416436

**Table of Contents**

[1 Introduction 2](#_Toc1017999871)

[1.1 Purpose 2](#_Toc1443304883)

[1.2 Project Scope 2](#_Toc1723369104)

[1.3 Definitions, Acronyms, and Abbreviations 2](#_Toc1382336775)

[2 User Requirements Definition 2](#_Toc423794664)

[3 Requirements Specification 2](#_Toc1708360293)

[3.1 Functional requirements 2](#_Toc1691012263)

[3.1.1 Use Case Diagram 3](#_Toc1103636866)

[3.1.2 Requirement 1: Turn based gameplay 3](#_Toc1115050373)

[3.1.3 Requirement 2 Quest system 5](#_Toc142171152)

[3.1.4 Requirement 3 Teaching player to play 6](#_Toc841152165)

[3.1.5 Requirement 4 Starting the Game 7](#_Toc2017458735)

[3.1.6 Requirement 5 Skill Tree 7](#_Toc1785141379)

[3.1.7 Requirement 6 Saving Progress 8](#_Toc447071581)

[3.1.8 Requirement 7 Loading a Save File 9](#_Toc1850171023)

[3.2 Non-Functional Requirements 10](#_Toc604418075)

[3.2.1 Performance/Response time requirement 10](#_Toc2001646646)

[3.2.2 Availability requirement 10](#_Toc660463564)

[3.2.3 Recover requirement 10](#_Toc1914849898)

[3.2.4 Robustness requirement 10](#_Toc1040138602)

[3.2.5 Security requirement 10](#_Toc528274083)

[3.2.6 Reliability requirement 10](#_Toc2109647131)

[3.2.7 Maintainability requirement 10](#_Toc295296679)

[4 GUI 11](#_Toc1803694574)

[5 System Evolution 13](#_Toc1533996245)

# Introduction

## Purpose

The purpose of this document is to set out the requirements for the development of a turn-based web game.

The intended customers are gamers.

## Project Scope

The scope of the project is to develop a working foundation for a turn-based combat web game. The system will focus on player choice.

## Definitions, Acronyms, and Abbreviations

Skill Tree A set of abilities the character unlocks throughout gameplay

Boss Challenging enemy

Replayability Encourages players to play the game again

……..

# User Requirements Definition

A tactical turn-based experience that puts an emphasis on team building and strategy. The system we be web best for easy accessibility.

# Requirements Specification

The game mechanics should be easy enough for the player to understand but hard to master. Offering long term replayability to fully master the flow of combat. It should also allow players to use different strategies to achieve victory.

## Functional requirements

The game should have a working combat system.

The game should have quests for players to complete.

The game should teach players how to play.

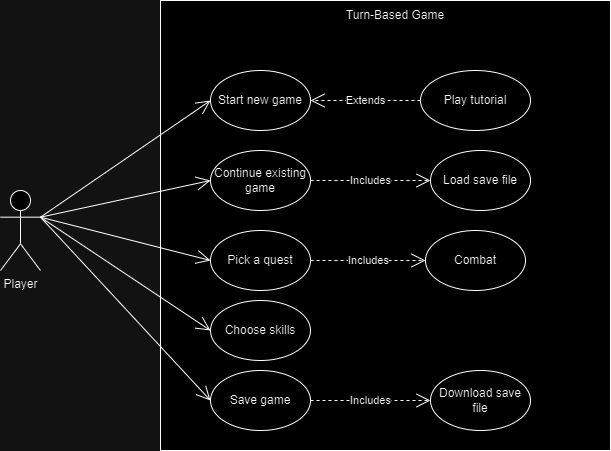
Players can start a new game.

Each character should have a working skill tree system that allows for different playstyles.

Players should be able to save their progress.

Players can continue an existing saved game.

### Use Case Diagram



### Requirement 1: Turn based gameplay

#### Description & Priority

This focuses on implementing the core functionality of the game. It is of high priority as it forms the foundation of the whole gaming experience for the user. The system must allow for the player to take turns using a card system, make strategic decisions, and interact with the game environment within the confines of the game mechanics

#### Use Case

Combat

**Scope**

The scope of this use case is to illustrate the flow of actions during a players turn in the turn-based system

**Description**

This use case describes the sequence of events when a player takes their turn during the game involving strategic decision making

**Flow Description**

**Precondition**

The player has chosen a quest

**Activation**

The player initiates this use case by selecting a quest.

**Main flow**

1. The system identifies the active player and presents the relevant game elements to duel
2. The player makes strategic decisions, such as selecting which card to use which will set who goes first, if the player has a higher card he will go first, if the ai has a higher card, it will go first (See A1)
3. if the player has the higher card he will continue, selecting which enemy to attack and which ability to use
4. The system processes the user's inputs and updates the game state accordingly
5. The player’s go is done and the ai will now proceed with its attack if it was not defeated

**Alternate flow**

A1 : AI goes first

1. The ai choose a higher card so its turn and attack take place first
2. The system processes the ai’s inputs and updates the game state accordingly
3. The use case continues at position 3 of the main flow.

**Exceptional flow**

E1 : player encounters connectivity issue

1. The system detects the connectivity issue
2. The user is informed of the issue and the turn is temporarily paused
3. The use case continues at position 4 of the main flow

**Termination**

The system presents the player at the start of their next turn

**Post condition**

The system goes into a wait state

### Requirement 2 Quest system

#### Description & Priority

This feature allows players to choose which quests they would like to do, think of it like game levels. It is essential for initiating combat and is therefore in high priority.

#### Use Case

Pick a quest

**Scope**

Should offer quests for the player to choose from.

**Description**

It describes how players can play the game. Players need to choose a quest to then start a battle.

**Flow Description**

**Precondition**

The player must have started the game from the menu, either by starting a new game or continuing from where they left off.

**Activation**

This use case starts when the player is redirected from the main menu

**Main flow**

1. The system displays available quests
2. The player chooses a quest
3. The system redirects the user to combat**Alternate flow**

### Requirement 3 Teaching player to play

**3.1.4.1 Description & Priority**

Players must be taught the basics of the game so that they can understand how to play it. This is on high priority as the game systems are complex.

**3.1.4.2 Use Case**

Tutorial

**Scope**

Should teach the basics of combat, learning skills and doing quests.

**Description**

A popup will appear when the player creates a new game. It asks if the player would like to play the tutorial. The player may choose to to decline the tutorial if they are already familiar with the game.

**Preconditions**

Player starts new game.

**Activation**

Player accepts tutorial popup.

**Main Flow**

1. System asks player if they would like to play the tutorial

2. Player accepts

3. System plays tutorial

**Alternative Flow**

1. System asks player if they would like to play the tutorial

2. Player declines

3. System starts game without playing tutorial

### Requirement 4 Starting the Game

**3.1.5.1 Description & Priority**

The player must start the game before they can play. This is also on high priority to give the game its structure.

**3.1.5.2 Use Case**

Start new game

**Scope**

Player should be able to start a new game.

**Description**

Players can start a fresh game without a previous save file.

**Activation**

Player selects “New Game” from main menu.

**Main Flow**

1. Player selects “Play” from main menu
2. System offers the user “New Game” and “Continue” options
3. Player chooses “New Game”
4. System initialises default parameter and starts game.

### Requirement 5 Skill Tree

**3.1.6.1 Description & Priority**

Player should be able to choose skills for their characters that encourage different styles of play. This should also make combat more exciting. This is a high priority as it adds much more flavour to the combat system.

**3.1.6.2 Use Case**

Choose skills

**Scope**

Presents the player with a variety of skills to choose from for each character.

**Description**

As players progress through the game, they will earn skill points for their characters. These skills points can be spent in the skill tree to enhance character skills an learn new ones and can change the flow of combat.

**Preconditions**

Player must have skill points.

**Activation**

Player selects a skill to learn

**Main Flow**

1. System presents user with list of skills
2. User selects a skill to learn
3. System checks with player has sufficient skill points
4. If player has sufficient skill points, system activated selected skill

### Requirement 6 Saving Progress

**3.1.7.1 Description & Priority**

Players should be able to save their progress via downloading a save file to their machine. This is medium priority for now as our main focus is to make the game functional before including this feature.

**3.1.7.2 Use Case**

Save Game

**Scope**

The player will download a save file to their machine.

**Description**

This use case describes how the player will save their game progress.

**Preconditions**

Player must have sufficient storage capacity on their machine.

**Activation**

Players must interact with the “save game” button within the game.

**Main Flow**

1. Player interacts with “save game” button
2. System generates a JSON file
3. System prompts player to download the file
4. Player downloads the file to their machine

**Exceptional Flow**

1. Player interacts with “save game” button
2. System generates a JSON file
3. System prompts player to download the file
4. Player does not have sufficient storage space
5. Browser aborts download process

### Requirement 7 Loading a Save File

**3.1.8.1 Description & Priority**

The player should be able to load their save files to continue the progress they have made within the game. This is a medium priority requirement as we will first be focusing on making the game functional before we include save features.

**3.1.8.2 Use Case**

Continue Existing Game

**Scope**

Player will be able to submit their save file to continue their previous progress.

**Description**

This use case describes how the user will load their saved progress.

**Preconditions**

Player must have a save file on their machine.

**Activation**

Player interacts with “Load Game” button in main menu.

**Main Flow**

1. Player interacts with “Load Game” button
2. System prompts player to submit their save file
3. Player submits their save file
4. System generates the game with certain parameters set

**Exceptional Flow**

1. Player interacts with “Load Game” button
2. System prompts player to submit their save file
3. Player submits invalid file
4. System notifies the player that the file is invalid
5. System aborts the process

## Non-Functional Requirements

### Performance/Response time requirement

Our games performance/ response time will be maintained to a standard which will provide the user with an enjoyable and seamless gaming experience. The response time will be quick to maintain engagement with the user

### Availability requirement

Since the game will be a single player game as of launch the runtime of the game span most of the year. If there ever was a maintenance patch it would be done at a time of least convince for the player base.

### Recover requirement

As stated previously with file saves this ensures the games data integrity is kept. If a bug occurs, there's a decent probability it would not make the game unplayable to the player.

### Robustness requirement

Ensuring our game is bug free, if a player encounters a bug, the game can be restarted at the last autosave point and or point where the player decides to save the game. Ensure a restart with minimal data loss for the player's in-game file.

### Security requirement

Our product will not store or transmit sensitive data as it will be a web-based game, there will be no login or input of data so there are minimal security risks

### Reliability requirement

The game will be reliable ensuring consistent and predictable behaviour. Consistent performance will be necessary for an enjoyable gaming experience for the user

### Maintainability requirement

The games source code will be written to a high standard, as well as clearly documented to facilitate easy maintenance, making for ease of updates or patches in the future if needs be.

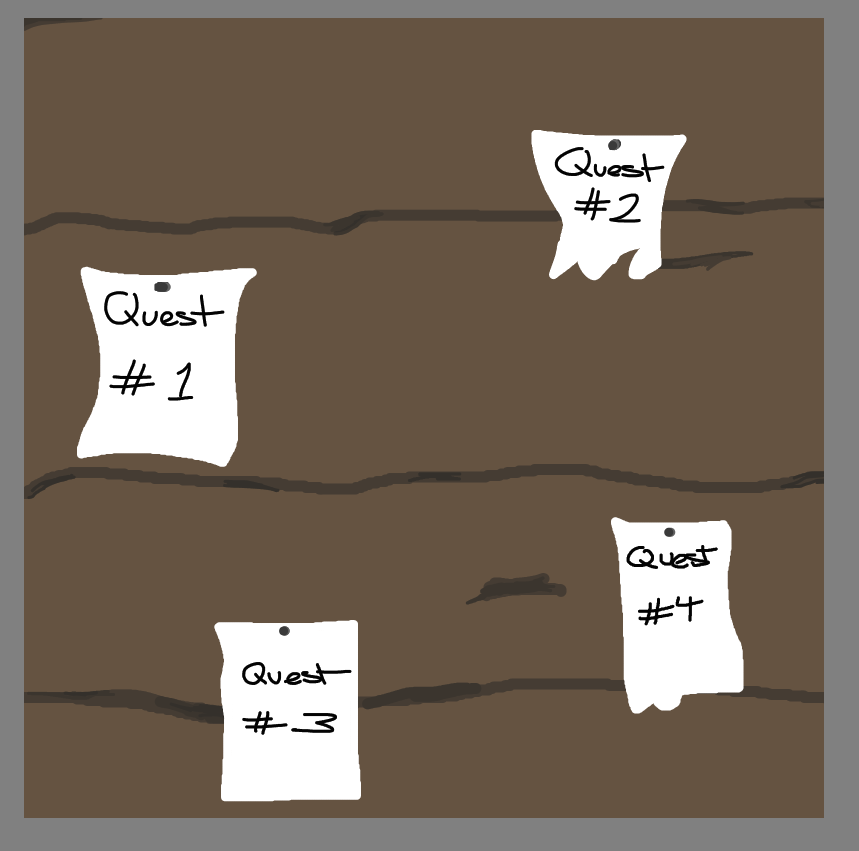
# GUI

Here is the main menu concept for the game. We decided to keep it as simple as possible for many reasons.

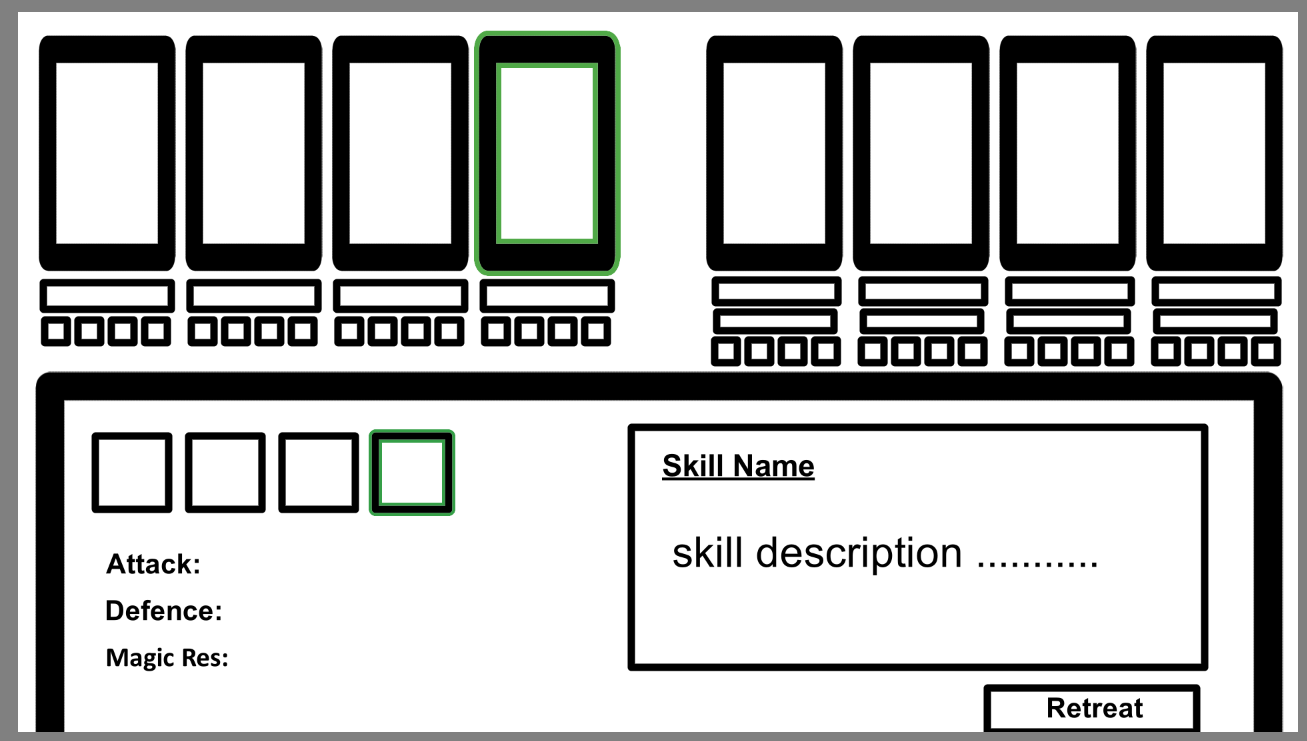
* Game will run smoother
* Easy to implement with code
* Easy to understand, no matter the age of the player

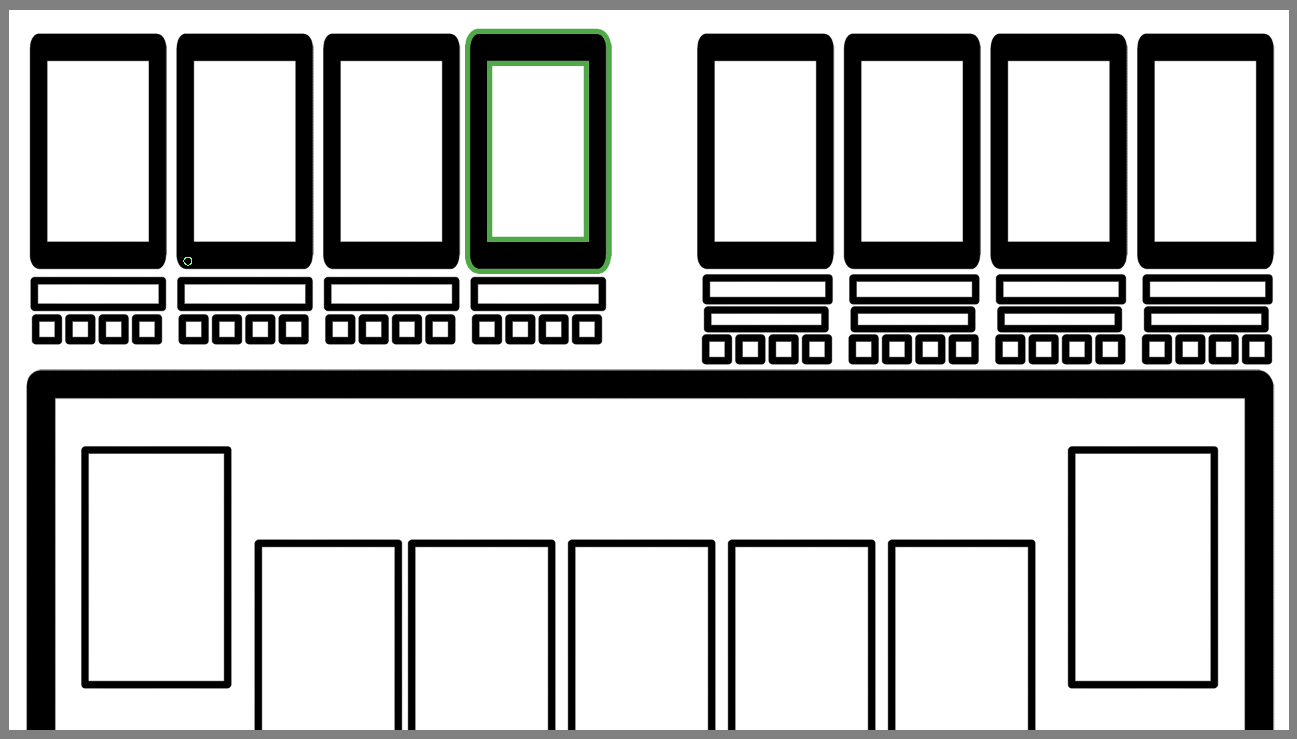


The quest drawing is a simple notice board where quest items will span as a piece of paper nailed to the board. Where the player can choose from a variety of quests based on what seems the most enticing.



The combat GUI will be split into two halves. The top half of the screen will present the player characters and enemies, as well as their health and status effects. The bottom half will show the selected character’s skills or the players cards if the combat is in its dueling phase.





# System Evolution

Moving forward if everything is going well. Performance wise and if our intended audience likes the game we make. We can add more playable characters, weapons, more enemies to fight. A bigger map, more unlockable and unique skills and maybe multiple story line quests with side characters. And even a multiplayer system where players fight each other for unique items and prizes.