Software Requirements Specification

Hex Army

Prepared by Wizard Monkeys

University of North Texas - CSCE 5430

Table of Contents

Group members :

No.	Name	Student ID	Email
1	Sravanthi Tummala	11533397	Sravanthu Tummala@gmail.com
2	VinayKumar Bathula	11521515	vinaykumarbathula@my.unt.edu
3	Jeremy Glebe	11452290	JeremyGlebe@my.unt.edu
4	Haiyi Wang	11528159	haiyiwang@my.unt.edu
5	Prudhvi Krishna Jarabani	11509395	PrudhviKrishnaJarabani@my.unt.edu
6	Vishnu Sai Konka	11513706	VishnuSaiKonka@my.unt.edu
7	Jashwanth Korapati	11521394	korapatijaswanth@gmail.com
8	Anand Paul Kamadana	11535249	anandpaulkamadana@my.unt.edu

Version: 1.0

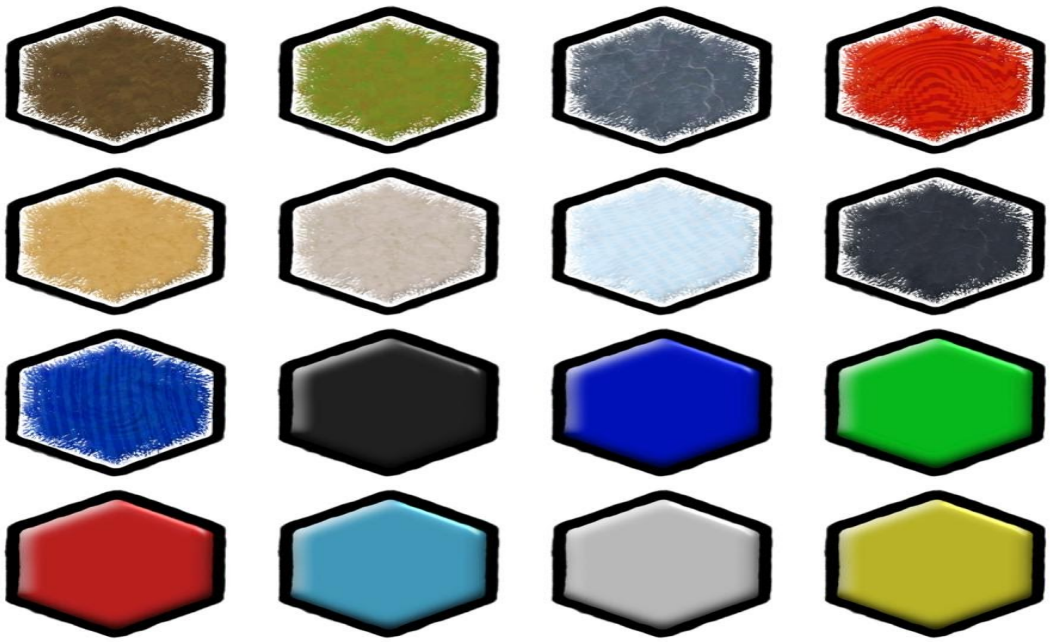
Document History

Version	Date	Author	Comments
1.0	15th Oct, 2021	Wizard Monkeys	Deliverable 3

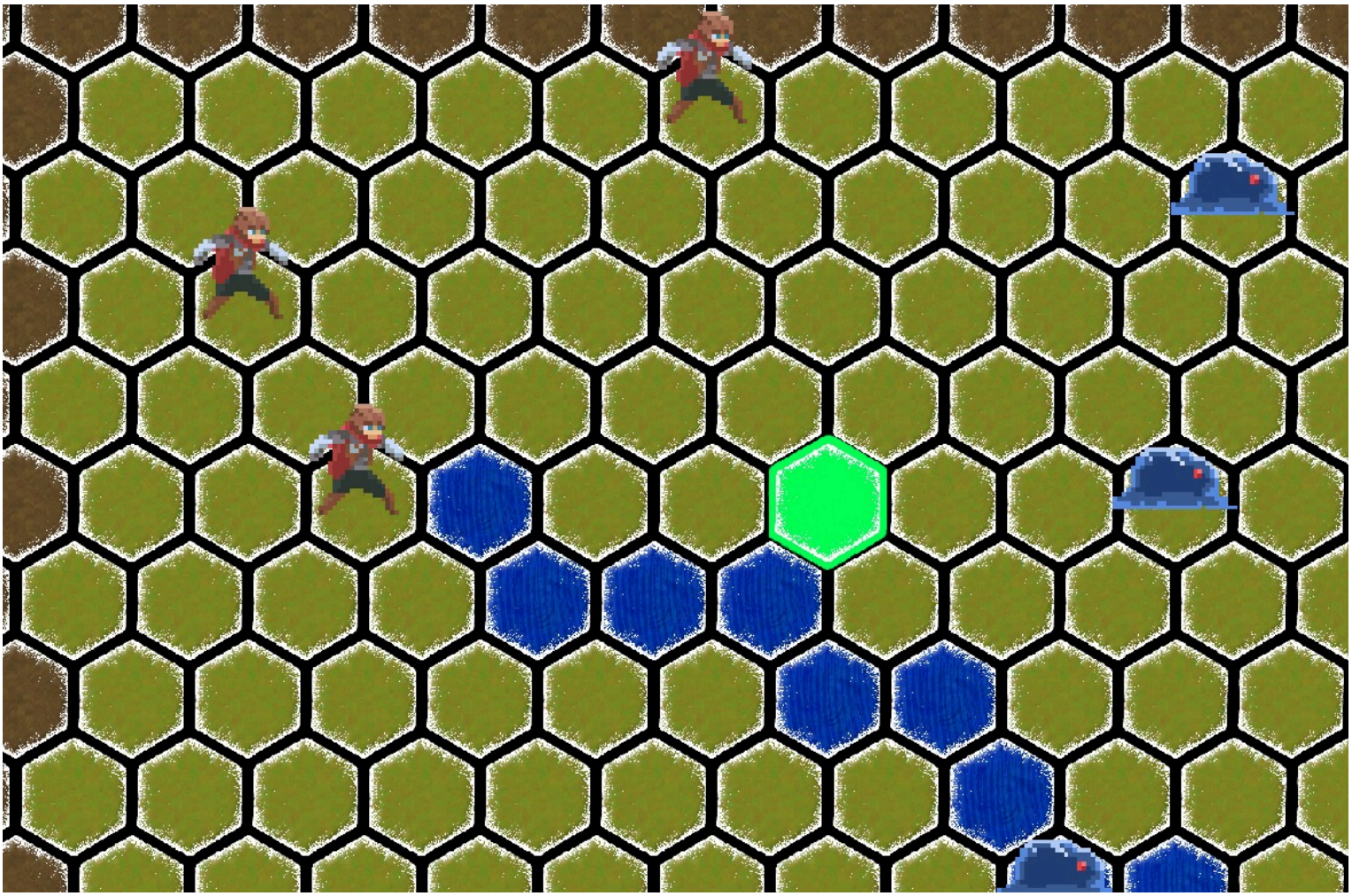
1.General Diagram:

The game is fitted with multiple hexagon grids with different colors. This leads to an effective layout for the game that not only looks good but also puts emphasis on ease of use. The player and enemies are separated by the different colors.

Hexagonal Grid:



Character Image:



User manual:

Description:

Controls:

There are two types of controls available for the users.

They are:

I) by using the mouse pointer, the player can move.

II) by using the arrows of the keyboard.

By pressing the arrow keys allows the player to move respective to the location of the arrow key towards the enemies.

1. By clicking the up arrow, the player moves upwards.
2. By clicking the down arrow, the player moves downwards.
3. By clicking the right arrow, the player moves towards the right side.
4. By clicking the left arrow, the player moves towards the left side.

Note:

The player can initially move from a standing point to any direction on a hexagonal grid.

Clear instructions on how to compile/run both your program and your test cases (the program must compile/ run):

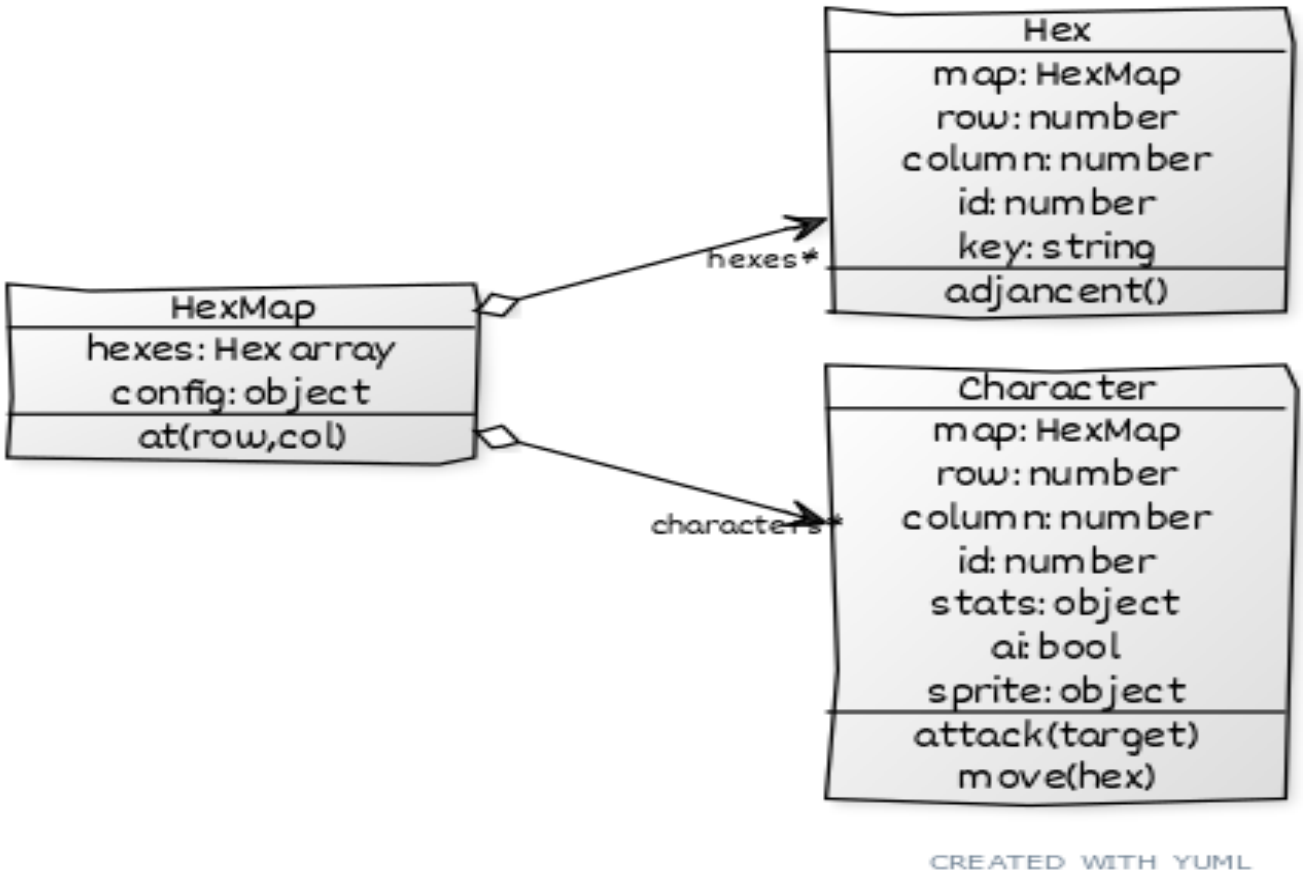
Install dependencies (must be done once initially and any time a new dependency is added to project)

npm install

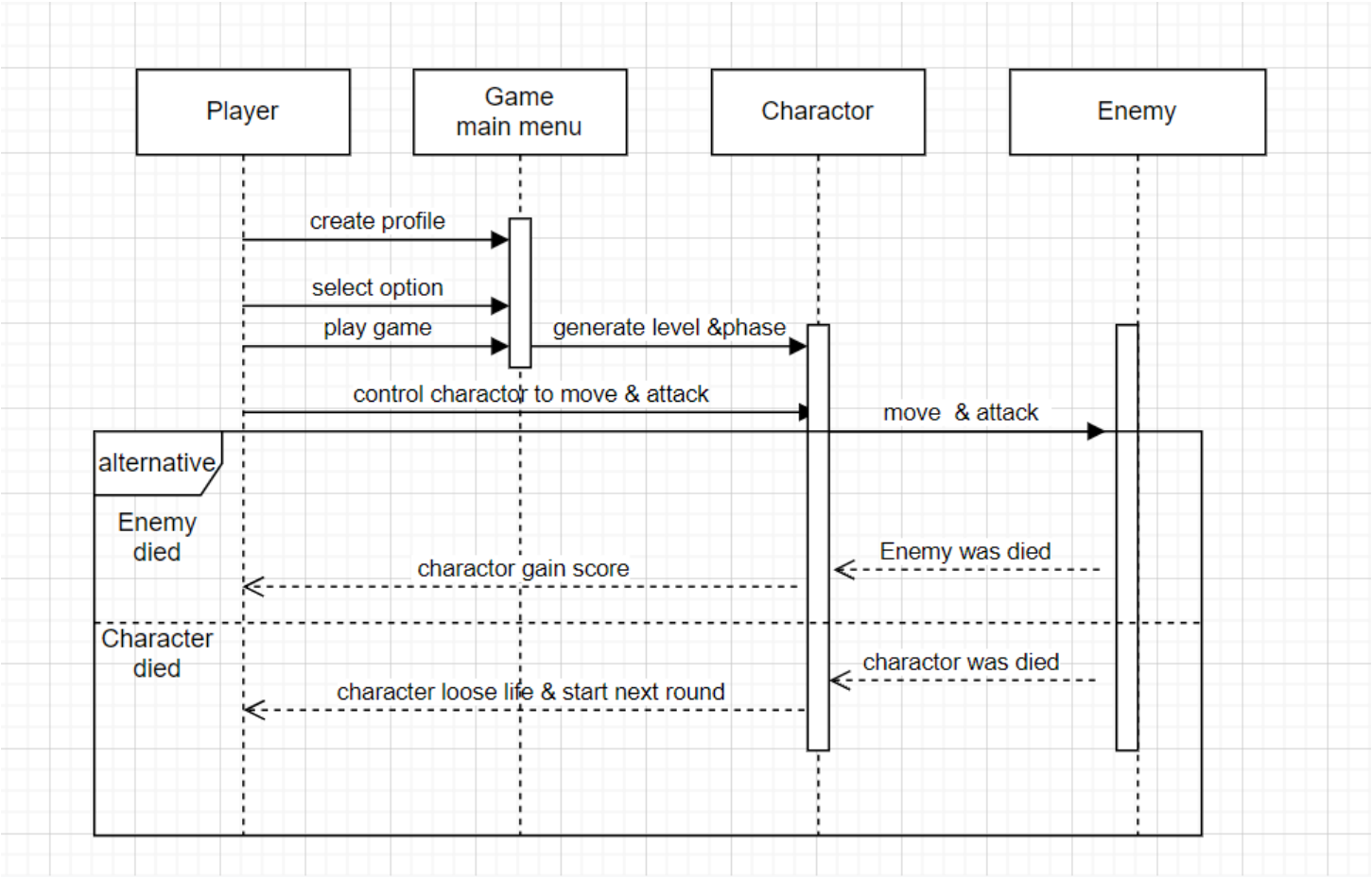
Run the project in a server (connect with your browser to test)
npm run serve # Use this command
npm run test # OR this, both commands are the same
Run the project in a server open to LAN devices (can be useful for mobile testing)
npm run serve:lan
Just build the project to www/ folder (production mode)
npm run build

#clone the repo
git clone https://github.com/jeremyglebe/5430-Software-Engineering.git
#use visual studio code to compile and run the code
code .
run start.js file

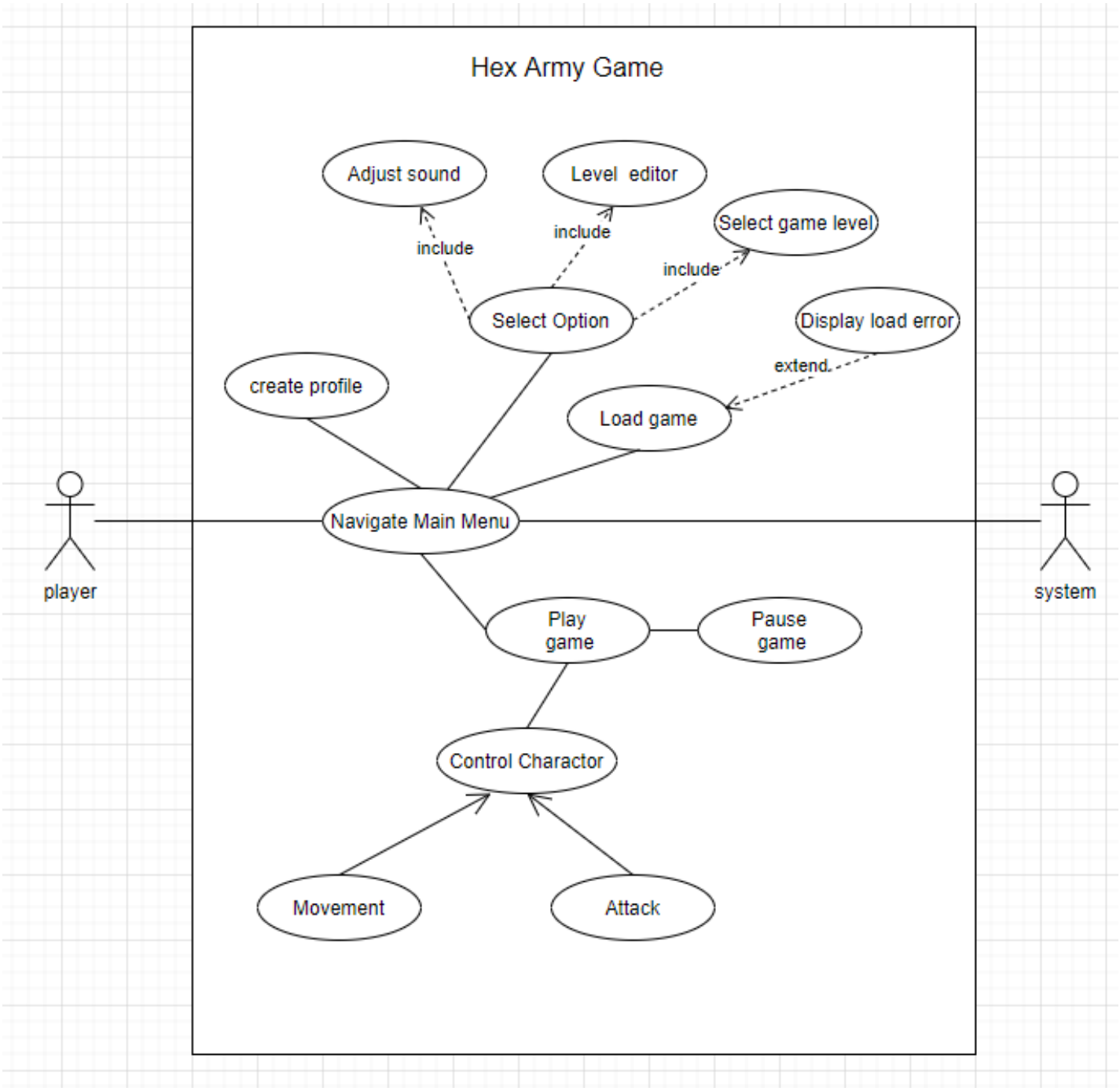
2. Class Diagram:



3.Sequence Diagram:


Player
Game
main menu
Charactor
Enemy
create profile
select option
play game
generate level &phase
control charactor to move & attack
move & attack
alternative
Enemy
died
charactor gain score
Enemy was died
Character
died
charactor was died
character loose life & start next round

4.Use case Diagram:



3. Software Requirement Specifications:

1. Functional Requirements:

1. The game board(User Interface):

- Created hexagon-based digital board
- Created 20*20 tile grid
- Created 2-dimensional table of the board with odd rows visually stagger to display as hexagons

2.Avatars/Characters:

- Created character of the player and enemy
- Created character will display animations when moving and attacking
- Created health bar of the player and enemy
- Created the weapon of the player and will display animations when attacking

3.Inputs

- Accomplished the moving that player can move to any of their six neighbors and could not exceed the board

4.User main page

- Created the main page, the player could choose to play and level editor.
- Create the setup of the board, the player can choose the color, appearance, size of the board.

5.Sounds

- Added the sound of some backgrounds
- Adder the sound of some buttons
- Added the sound of moving with player and enemy
- Added the sound of the player attack.

2. Non-Functional Requirements:

- Performance Requirements
    a. The application should respond to user quickly to user input.
    b. Have error handling mechanism that deals with slow internet or no internet issues.
- Security Requirements
    a. Protecting the application data.
    b. Make sure that the application data should not be stealed.

- Usability Requirements

  Ensure each navigation works like how they should

  Ensure that user interface is user friendly and intuitive.

- Availability Requirements

  a. The system should be available round the clock that help for user.

3. Interface Requirements:

3.3.1 User Interface:

A frontend will be created with html, CSS and Java Script

Initially, the game can be started by giving a player name, after inputting the player's name it leads to the new page where the player can select the controls and the sounds of the game and start playing the game. It also consists of an exit button if the player wants to leave.

The hex game is the game that takes place within the hexagonal grids. The user is considered as a player who continuously moves towards the enemies and kills them respectively. This is a one on one play where the player moves towards the enemy by selecting the path and attacks the enemy after the player kills the enemy the next one starts attacking. If the player defeats all the enemies in level one and heads to the next level. For each kill, the player can gain a certain amount of health which helps the player to survive in the game if required.

The goal of the game is to kill the enemy who was placed in random places on the hexagonal grid. The game gets tougher while getting to the next levels.

- **3.3.2 Hardware Interface**:

Our application can be accessed and can be opened in devices like laptops, computers as well as mobile phones.

- **3.3.3 Software Interface:**

a. Backend

JAVA SCRIPT

b. Development Languages:

HTML, CSS, JAVA

SCRIPT

4. **Test Cases**:

Test Cases (unit tests) for **phase 1**.

List a set of test cases used for testing the working program including descriptions of tests (e.g., what functionality they test, and inputs/outputs for them).

Test case:

1. Test all hexagonal grids, walls and characters

- Make sure the characters can move to each and every hexagonal
- No character is moving outside the grid
- When two characters came across make sure they don't pass through each other

2. Test the main page

- Check all the buttons and their functionality
- Make sure all the buttons serve their purpose

3. Test for health bar and weapons

- Check if the health is being reduced for every attack
- Make sure both players and enemy are attacking each other
- No two enemies should fight with each other

4. Test the controls

- Test every possible control button
- Make sure character moves according to the controls
- Test the whole GUI (colors, buttons, appearance, etc,.)

5. Member Contribution Table:

Member	Contribution description	Overall Contribution (%)
Sravanthi Tummala	Created a menu for a game using HTML	12.5
Vinay Kumar Bathula	Short path algorithm for Hero character	12.5
Jeremy Glebe	Created a Hero and enemy character and moving character animation using Javascript.	12.5
Haiyi Wang	Added sound for Background, buttons, moving with player and enemy, player attack.	12.5
Prudhvi Krishna Jarabani	Created Menu, health bar and weapons for character(Hero and enemy character)	12.5
Vishnu Sai Konka	Menu, health bar and weapons(Hero and enemy character)	12.5
Jaswanth Korapati	Added sound for Background, buttons, moving with player and enemy, player attack. Update the meeting and discussion about the project	12.5
Anand Paul Kamadana	Created a menu for a game using HTML	12.5

6.**Screenshots for Interfaces:**

Welcome Page:

Home Page:


HEX GAME
Player Name
OK

References

https://photonstorm.github.io/phaser3-docs/