

Team 7
Final Iteration

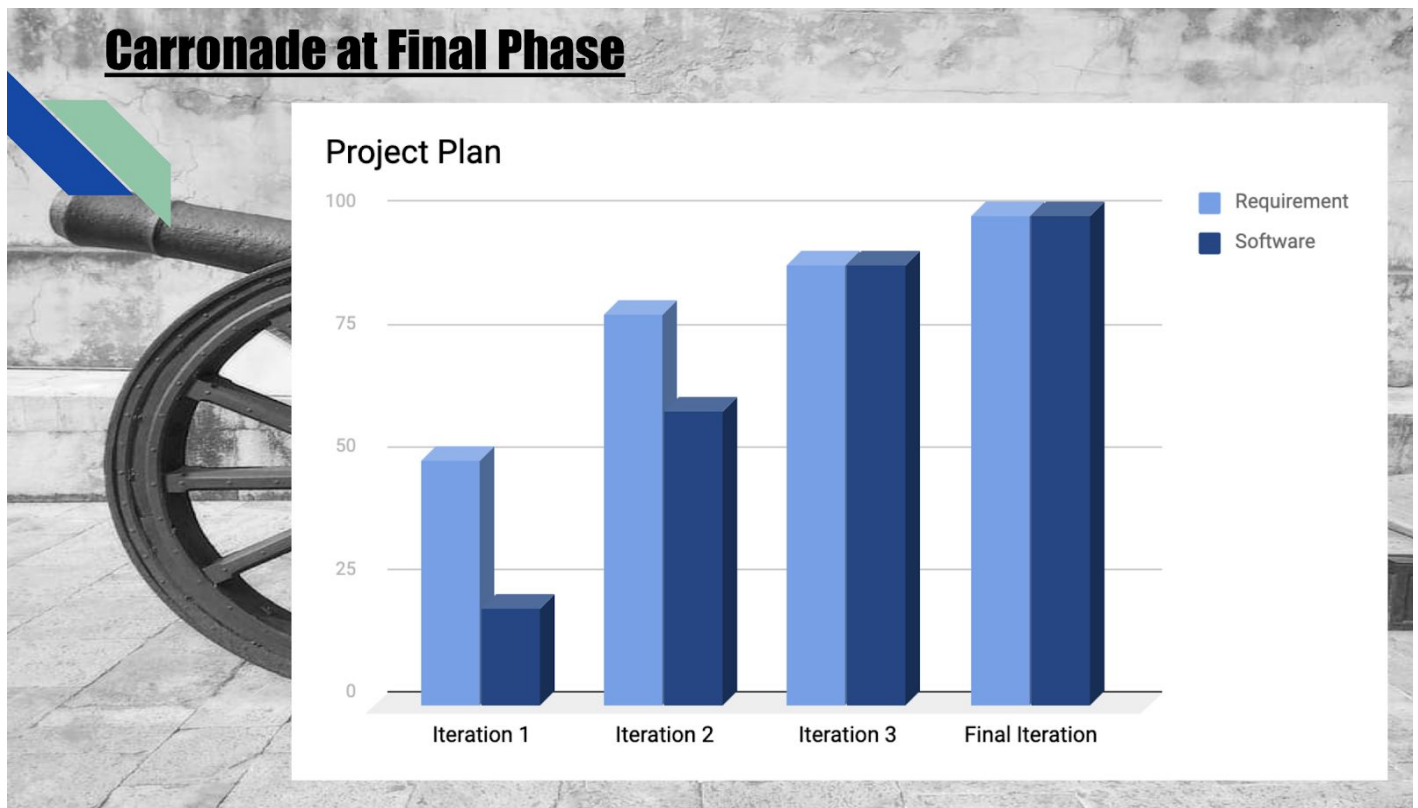
Members : Paul Patraca Pantoja
Utsav Dhungana
Bipul Karki

“Carronade”

Project Plan:

The game has final features introduced in this iteration. Game sound is implemented, additionally with settings so the player can mute and unmute the game as preferred. The user will be notified of a new high score and will be kept for throughout playthroughs. The settings menu has been implemented so that the settings for sound can be adjusted.

Drop-down of our Project Plan:



Iteration 1

- Content Management
- Test cases created for each feature implemented
- Basic Player Locomotion
- Basic Enemy Functionality

Iteration 2

- Player able to boost character's life with power-ups from cannon
- Projectiles with greater destruction ability
- Score tracking implemented
- Updated game based on previous feedback
- Test cases for all new features introduced
- Getting feedback from users(our colleagues) and fixing bugs.

Iteration 3

- Main Menu and Selection Menu
- Pause game feature implemented
- Able to select new different characters and arenas
- Fixing bugs and faults based on previous feedback

Final Iteration:

- Settings Menu
- High Score Tracking
- Game sound
- Fixing bugs and faults based on the feedbacks from users
- Thorough Inspection of Test Cases

Close Related App:

The most closely related games that are similar to marathon games:

Jetpack Joyride:

- No health bar that decreases overtime
- Game is over when player hits the obstacles instead to shot by enemy
- Not minimalistic
- Tokens to redeem for life makes game easier

Subway Surfers:

- Not much is handled by the AI of the game
- It is also not minimalistic
- Token redeem option makes game easy
- No health bar instead player dies when it hits obstacles

Top Risks experienced during the Final Iteration:

1. Changes in the requirements based on customer feedback.
 - Probability of the risk was 40%, and the effect of the risk is that we will have to spend 20 more hours. So, the Risk Exposure (RE) = $0.4 * 20 = 8$ hours.
 - Changes in the game functionality were made during each iteration based on customer feedback. This will also cause risk of exceeding the deadline.
 - To mitigate this risk we planned early for any possible changes in the game functionality.
2. Exceeding the deadline to complete the project i.e project not completed on time.
 - Probability of the risk is 30%, and the effect of the risk is that we will have to spend 20 more hours. So, the Risk Exposure (RE) = 6 hours.
 - Making a good timeline for each iteration has mitigated the risk, but at the time of writing was a concern
3. A member getting sick.
 - Probability of the risk is 15%, and the effect of the risk is that we will have to spend 10 more hours. So, the Risk Exposure (RE) = 1.5 hours.
 - Creating a healthy living environment and good hygiene would mitigate the risk. In the present context, good practice of social distancing would

mitigate the risk.

4. Communication and documentation difficulties due to network outage around team member's area.
 - Probability of the risk is 10%, and the effect of the risk is that we will have to spend 5 more hours. So, the Risk Exposure (RE) = 0.5 hours.
 - Communication between the members of the project using mediums like phone/text or using hotspot to access the internet. Files and messages can be transferred using mobile networks.
5. Poor programming practices leading to memory leaks and slowing down the user's machine; occurrence of bugs.
 - Probability of the risk is 5%, and the effect of the risk that we will have to spend 5 more hours. So, the Risk Exposure (RE) = 0.25 hours.
 - Inexperience in loading in assets to memory alongside time constraints make this the most possible scenario.
 - Careful and thoughtful use of documentation and available resources can mitigate the probability of the risk during each iteration.

Inputs and Outputs:

Input	Output
Keyboard Button (W, A, S, D)	On the play screen, Player moves up/left/right/down respectively
Keyboard Button (R)	On the play screen, Resets the game
Keyboard Button (Shift)	On the play screen, Player's given "movement skill" activates
Keyboard Button (P)	On the play screen, Pause the game
Keyboard Button (ESC)	On the main menu, Exit the game.
Play Button Clicked	Game switches screens to Selection Menu
Edit Button Clicked	Game switches screens to Settings Menu
Exit Button Clicked	On the Main Menu, exits the game. On the Pause Screen, returns to the main menu. On the Settings Menu, return to the previous menu
.xml Properly Structured	Game can load in assets safely and efficiently

Data Structures:

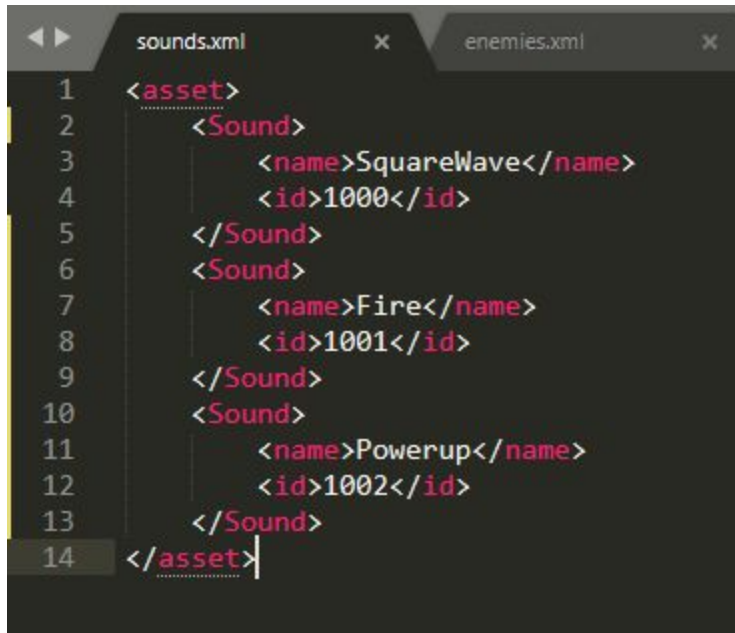
Asset linking, reads on-drive data (Images, Sounds, etc) and connects them to the game for use. This is the method for loading animations in particular (Most complex data structure)

```
public Animations BuildAnimations(System.Xml.XmlNode node) {
    Texture2D tex;
    int id, w, h;
    string defaultAnim;
    //TODO: Validate XML. Because there is absolutely 0 XML format validation going on here
    try {
        tex = contentManager.Load<Texture2D>(node.ChildNodes[0].InnerText);
        id = int.Parse(node.ChildNodes[1].InnerText);
        w = int.Parse(node.ChildNodes[2].InnerText);
        h = int.Parse(node.ChildNodes[3].InnerText);
        defaultAnim = node.ChildNodes[4].InnerText;
    } catch (Exception e) {
        Console.WriteLine(e.ToString());
        return null;
    }
    Animations anim = new Animations(tex, id, w, h, defaultAnim);
    Console.WriteLine(node.ChildNodes[5].InnerText);
    foreach (System.Xml.XmlNode animation in node.ChildNodes[5].ChildNodes) {
        Console.WriteLine(animation);
        string name = animation.ChildNodes[0].InnerText;
        int s = int.Parse(animation.ChildNodes[1].InnerText);
        int e = int.Parse(animation.ChildNodes[2].InnerText);
        anim.GenerateAnim(name, s, e);
    }
    return anim;
}
```

Sound Asset Linking

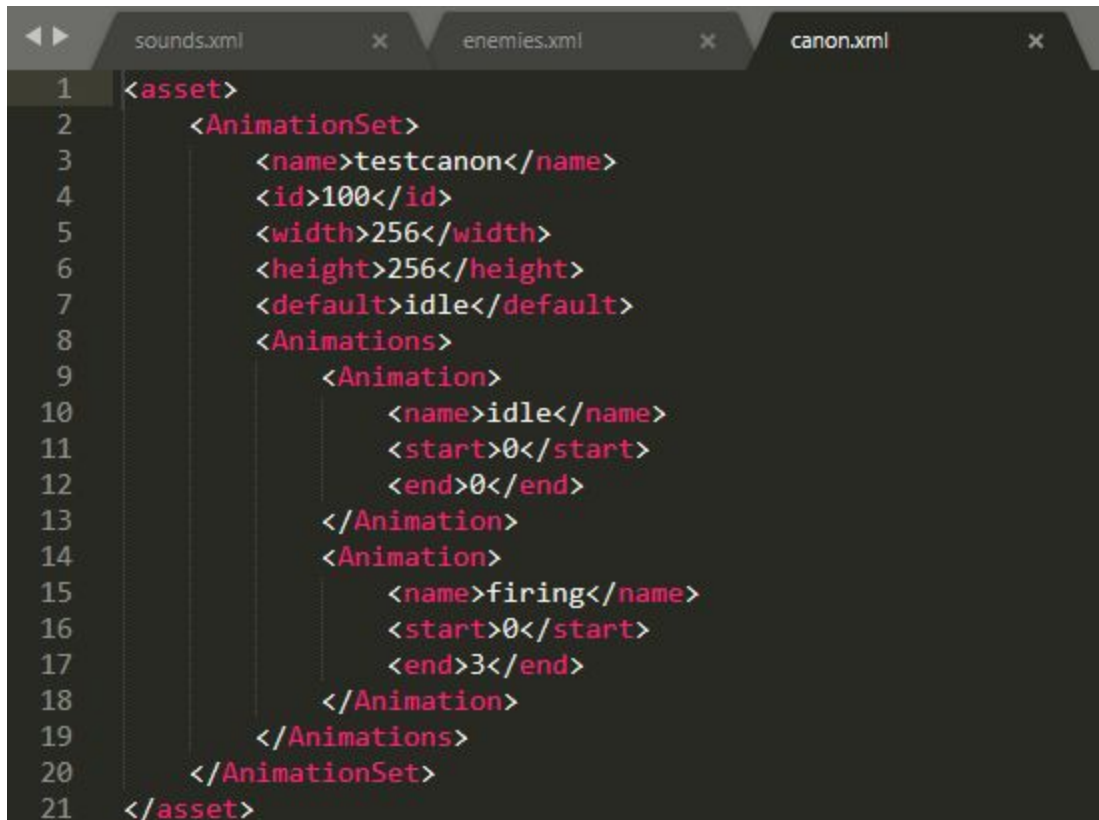
```
public Sound BuildSound(System.Xml.XmlNode node) {
    SoundEffect aud;
    int id;
    try {
        aud = contentManager.Load<SoundEffect>(node.ChildNodes[0].InnerText);
        id = int.Parse(node.ChildNodes[1].InnerText);
    } catch (Exception e) {
        Console.WriteLine(e.ToString());
        return null;
    }
    Sound sound = new Sound(aud, id);
    return sound;
}
```

Sound Data



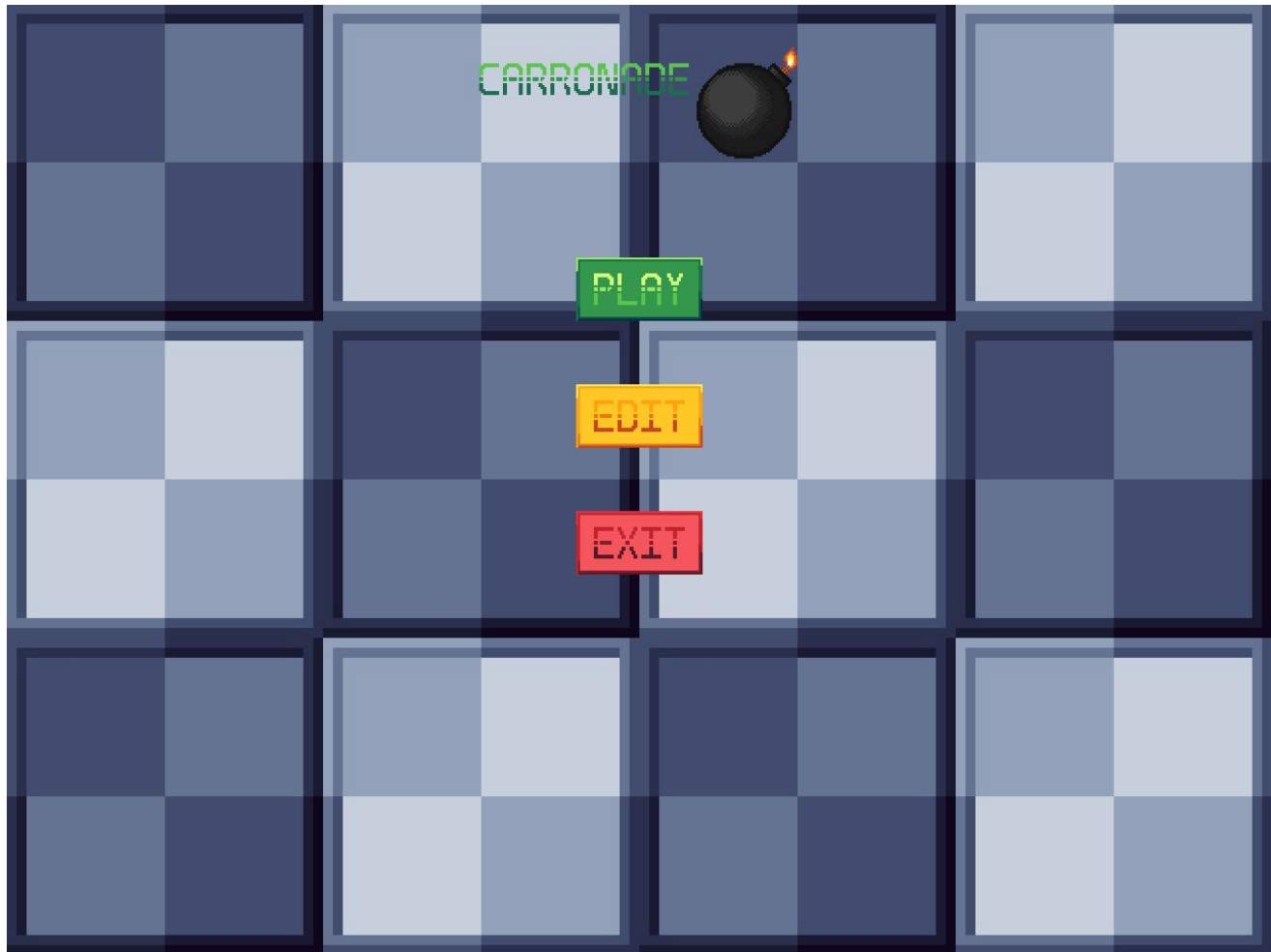
```
1 <asset>
2   <Sound>
3     <name>SquareWave</name>
4     <id>1000</id>
5   </Sound>
6   <Sound>
7     <name>Fire</name>
8     <id>1001</id>
9   </Sound>
10  <Sound>
11    <name>Powerup</name>
12    <id>1002</id>
13  </Sound>
14 </asset>
```

Animation/Image Data

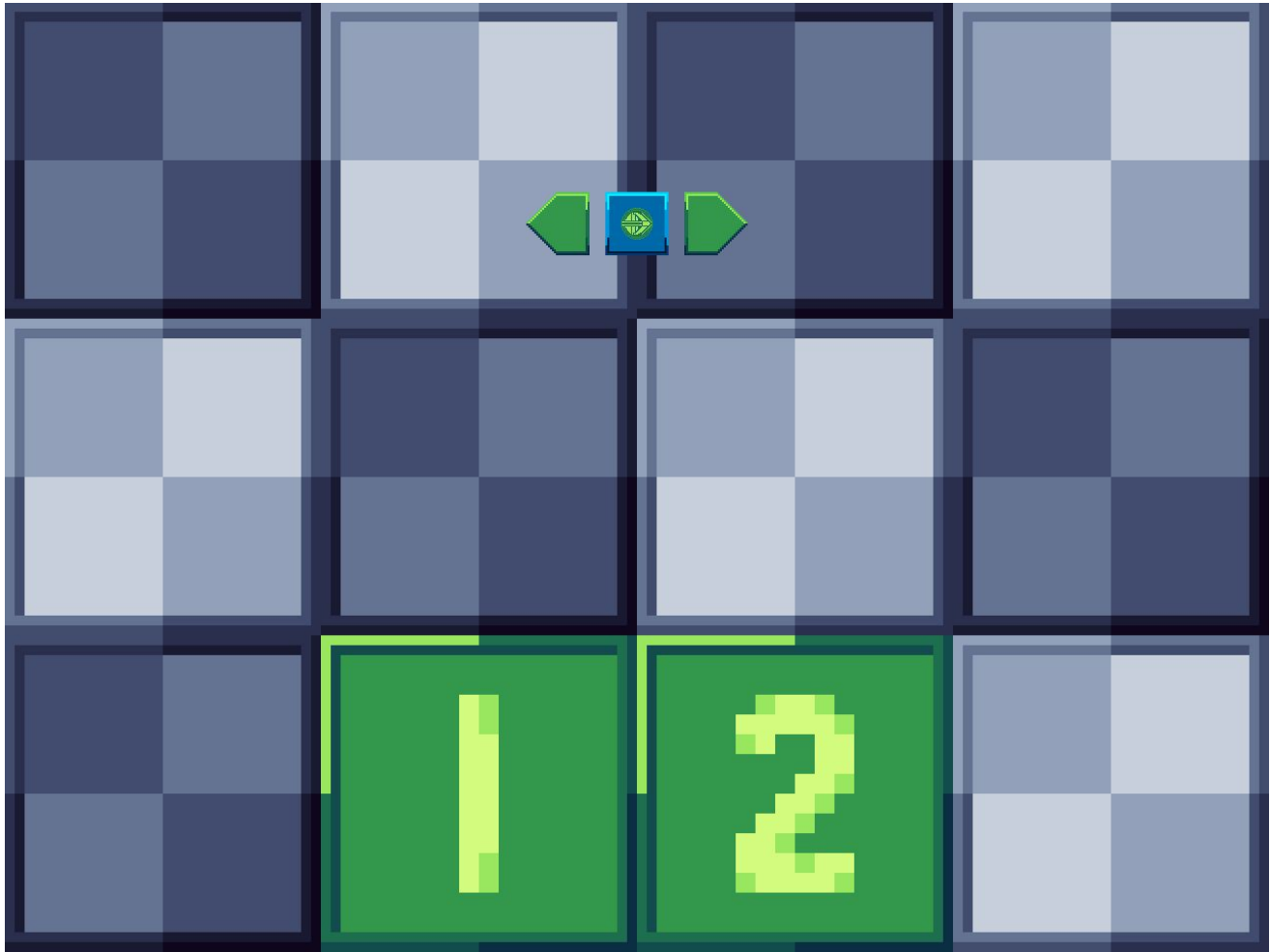


```
1 <asset>
2   <AnimationSet>
3     <name>testcanon</name>
4     <id>100</id>
5     <width>256</width>
6     <height>256</height>
7     <default>idle</default>
8     <Animations>
9       <Animation>
10        <name>idle</name>
11        <start>0</start>
12        <end>0</end>
13      </Animation>
14      <Animation>
15        <name>firing</name>
16        <start>0</start>
17        <end>3</end>
18      </Animation>
19    </Animations>
20  </AnimationSet>
21 </asset>
```

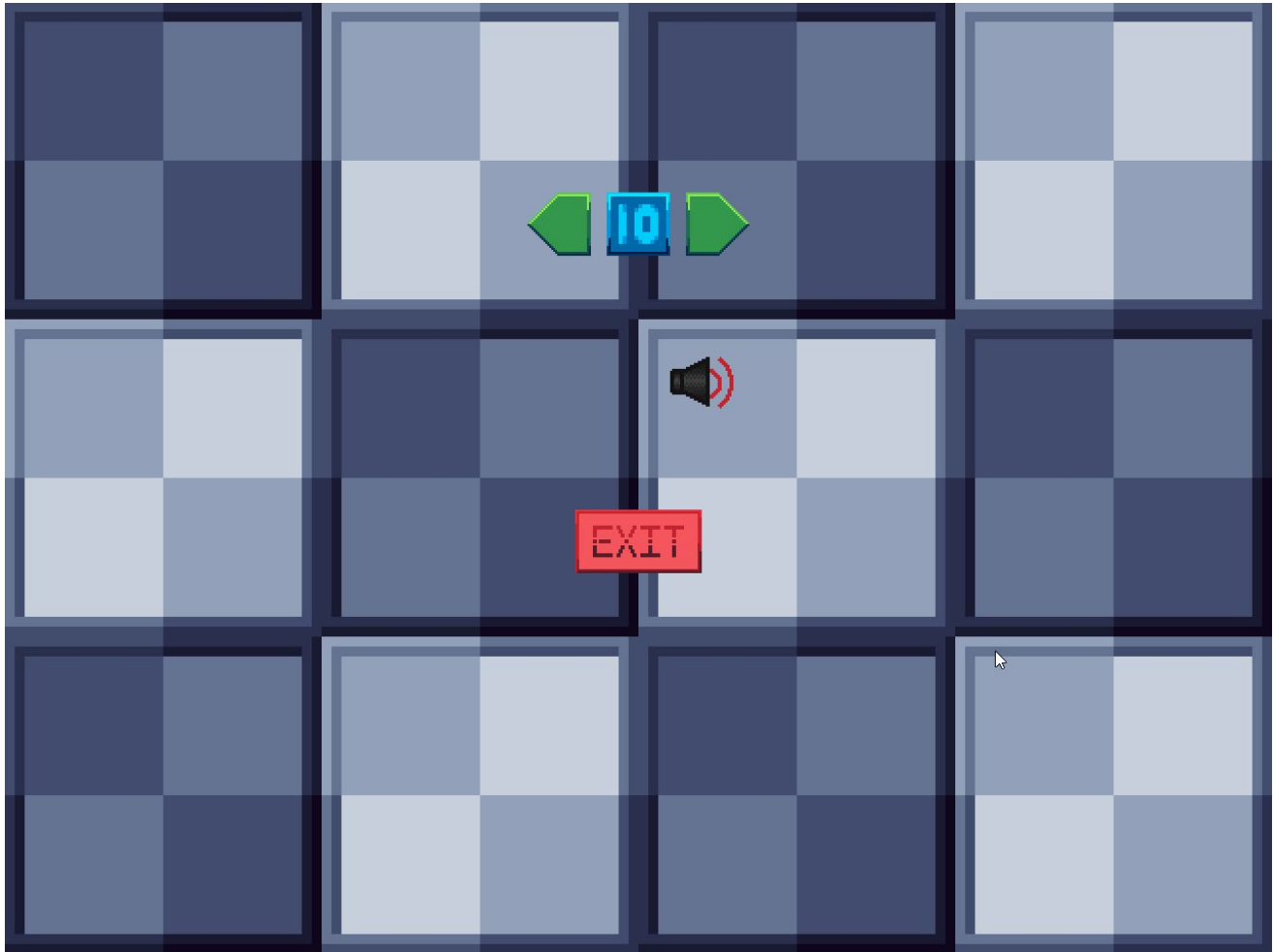
Finished Screens



While an odd name for Settings, Edit was chosen to fit the theme of the rest of the menu. Play will take the user to the Selection Screen, and Exit quits the game.



Selections screen has a selector to tab through characters and then the two large buttons allow for selection between the two arenas and gameplay.



Settings menu, while bare, serves its functionality. The arrows adjust the volume and the speaker allows for muting and unmuting, while Exit returns to the previously active screen.



Pause screen, which shows up whenever the Pause Key is pressed, giving options of what to do and where to go next.



Final Play Screen. The high score will appear whenever a new high score is achieved in the play session.

Requirements

Req. ID	Requirements	Functional/ Non-Functional	Implemented in Iteration	Updated/ Deleted/ Added Req. to Document	Priority
R1	The enemy shall throw objects towards the player when game starts	Functional	1		Does
R2	The System shall display health for the convenience of the Player	Functional	2		Does
R3	The Player shall be able to mute the sound of the game.	Functional	Final		Does
R4	The System shall not be error prone in a manner that will inconvenience the user	Functional	1		Does
R5	The Enemy director shall create at least two boosts for the Player	Functional	2		Does
R6	The Player shall be able to choose from at least two different characters.	Functional	2		Does
R7	The Player shall be able to choose from at least two different arenas.	Functional	3		Does
R8	There shall be tracking of scores so the player can make a note of their own progress.	Functional	3		Does

R9	The Enemy director shall create at least three different types of enemies.	Functional	2		Does
R10	The Player shall be able to change the graphical settings of the game.	Functional	Final	Deleted	Doesn't
R11	The player's avatar shall move responsively to the inputs of the player.	Functional	1		Does
R12	The player shall be able to restart the game.	Functional	2		Does
R13	The system shall inform if a high score is reached.	Functional	Final		Does
R14	The player shall be able to pause the game.	Functional	3		Does
R15	The system shall have a main menu and selection menu.	Functional	3		Does
R16	The system shall have a settings menu.	Functional	Final		Does

Use-Cases:

Use Case 1 - Load up the game

- TUCBW the user launches up the executable
- TUCEW the program loads in all assets

Use Case 2 - Generate Main Menu

- TUCBW the user leaves play or first launches the game
- TUCEW the user opens the settings menu, quits the game, or loads up the selection screen

Use Case 3 - Open Settings Menu

- TUCBW the user opens the settings menu from the main menu or pauses.
- TUCEW the user closes the menu, returning to the menu or play screen

Use Case 3.1 - Load Saved Settings

- TUCBW the settings menu is loaded, putting everything to its previous value
- TUCEW everything is properly loaded

Use Case 3.2 - Apply Changed Settings

- TUCBW a setting is changed
- TUCEW a setting is then applied to begin working immediately.

Use Case 4 - Go To Selection Menu

- TUCBW the user opts to play the game.
- TUCEW the user confirms their selection

Use Case 4.1 - Register Selections

- TUCBW the user confirms that the current play conditions are satisfactory.
- TUCEW the play screen is reconfigured alongside the selections and is then loaded in

Use Case 4.2 - Load in additional assets

- TUCBW the play screen needs additional assets
- TUCEW the additional assets have been loaded in

Use Case 5 - Play Screen

- TUCBW user returns from the settings menu or enters from the selection screen. In either case, the AI will begin functionality
- TUCEW users lose the game or pauses, taking them to the main menu or selection screen respectively.

Use Case 6 - Keyboard Input

- TUCBW users press an interactive key for the game.
- TUCEW the game processes the input and takes the corresponding action

Use Case 7 - Enemy Locomotion

- TUCBW the enemy is created ingame
- TUCEW the enemy collides with the player or with the game border

A use case implemented in Final iteration
Name : Use Case 3 - Open Settings Menu
Actors: Players/Users
Precondition: The User should be on the Main Menu or Game Screen (Paused).
Main success scenario: <ol style="list-style-type: none"> 1. The User clicks the Settings menu. 2. The settings menu is loaded. 3. The system loads up all the settings.
Extensions: <ol style="list-style-type: none"> 1.a) The system fails to load the main menu. <ul style="list-style-type: none"> - The system tells the actor that an error has occurred and suggests restarting the game itself. 3.a) The system fails to load the new settings <ul style="list-style-type: none"> - The system loads default settings, overriding the new settings.
Post-conditions: The settings are shown and properly applied as set by the user.

Test Cases

#	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
1	The Player Avatar responds to input	WASD keys used to test cardinal directions.	W goes up, A left, S down, D right.	W goes up, A left, S down, D right. Combinations nullify each other or add diagonality.	Pass
2	The XML loader reads in XML and produces usable in game assets.	See: canon.xml	Files should be able to refer to ID:100, all animations present	ID:100 exists, works as intended	Pass
3	The Asset Manager should not allow for multiple IDs to collide	.xml asset image made with ID:0 with pre-existing ID:0	Console spits an error and disallows this from happening	Console prints an error, ID:0 stays as is, newly loaded asset is disregarded	Pass
4	The Enemy should track the Player and fire at them	Player Avatar made to run around the canon	Canon's muzzle should follow the player, fire at them	Canon's muzzle follows the player, fires	Pass
5	The R key should completely reset the game and swap the player	R key is pressed	Canon and player both reset to their reset state, player changed for different type	Game resets, player is swapped to a different type	Pass
6	The Blue Enemy should pursue the player	Player Avatar made to run around the map with	Blue Enemies should slowly turn towards the player and follow.	Blue Enemies Chase the Player	Pass

		Blue Enem. onscreen			
7	The Invincibility Power-up should render the player impervious to damage	Player Avatar picks up the invincibility powerup	Enemies that collide with the player should have no effect	Player takes no damage from the exchange	Pass
8	The Player shall be able to choose from at least two different arenas.	Player clicks between options 1 and 2	Entering the game with a different selection gives a different arena	Button 1 gives the classic 1 canon setting while Button 2 creates 2 canons	Pass
9	The Score Tracker shall make note of changes in score	An enemy flies off screen	The Score Tracker should be updated to reflect changes	The Tracker updated to reflect the gained score	Pass
10	The Player shall be able to pause the game by pressing P	P key is pressed	The Game should stop locomotion	The Game pauses	Pass
11	The Game shall have a main menu and selection screen	Player clicks on Play	The Game should transition from Main Menu -> Selection Menu	The Game switches from the Main Menu to the Selection Menu	Pass
12	The Exit button shall function differently depending on the screen	Player clicks on Edit on the Play/Main screen	The game should exit if on the Main Menu, returns to Main Menu on Play Screen	The Game exits and returns correctly.	Pass

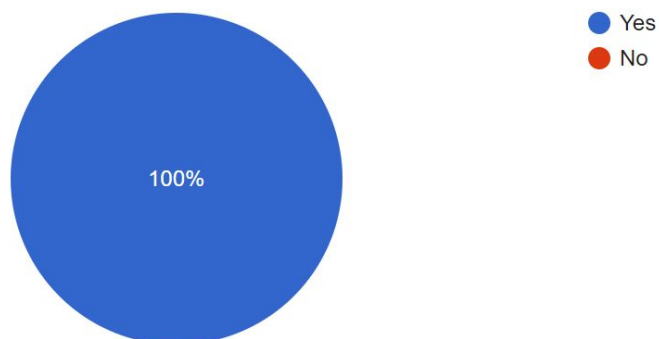
Customers and User:

All the users who will play this game are our customers. For testing and getting feedback we provided our game in the game feedback community with the game and asked for constant feedback from them about the features we implemented in each of the iterations. To gather feedback we have provided an online form which asked several free response questions to the user concerning the gameplay, flow of the program, functionality check, errors, and bugs. The users who provided feedback have some experience with different types of games. In the case of faults and bugs in the system we set up a meeting with users virtually to get further information.

Screenshot of user feedback:

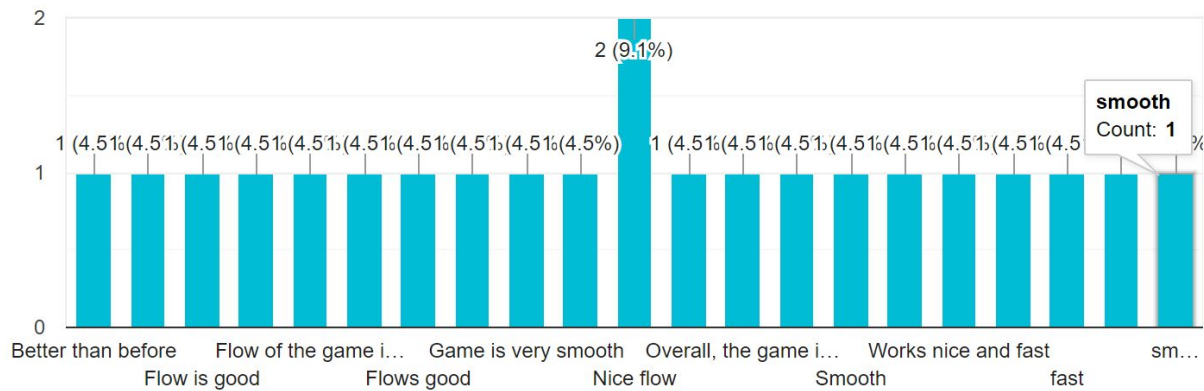
Are you able to open setting option in the main menu?

29 responses



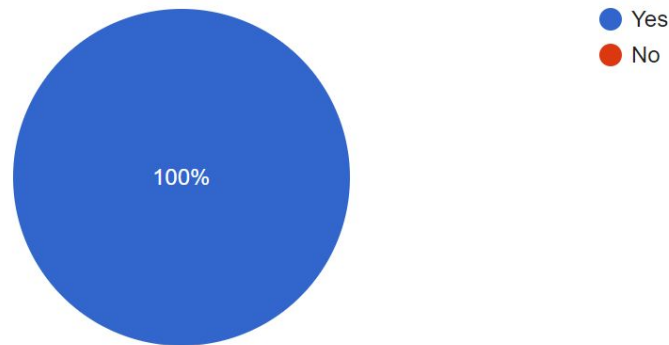
How do you like the flow of the game?

22 responses



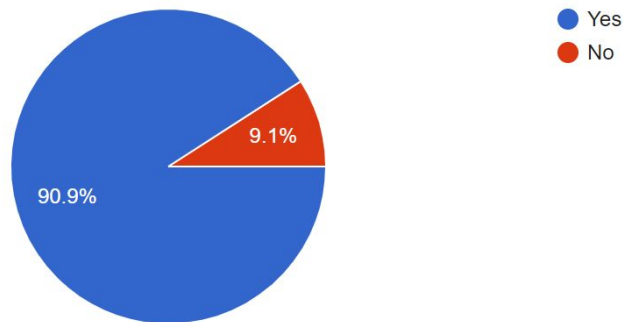
Is Main Menu Showing up?

27 responses



Was all inputs to the game working for you?

22 responses



How to compile the game?

The game is written in C# using framework software Monogame that runs on the .NET 4.6.1 framework, and is compiled with Microsoft Visual Studio. You need to have Microsoft Visual Studio installed for this purpose. First, clone the game directory from github (link provided below) to your local directory. Open Microsoft Visual Studio, in your first screen choose open a project or solution, now look for Carronade(.sln) in the Carronade subdirectory inside the main directory you cloned from github. Open that file and click on the start button located below the menu bar. Game will compile and start running.

<https://forms.gle/uAfbRY5uW3UBUzep6>

Github Link:

<https://github.com/udhungana/team7cse3311>

Carronade Version 0.4