SIMPERIA

Agüero, Julia

Besana, Luca

Simeunovic, Savo

Soares, Sophia

Susic, Marin

Our game takes you on a thrilling adventure through the vastness of space, where you find yourself in a precarious situation. As a space traveler, you are unexpectedly ambushed by ruthless space pirates, causing your ship to crash and leaving you stranded. These pirates have seized the card needed to open your ship, trapping you within the confines of an enigmatic asteroid.

The journey begins with a sense of mystery and urgency. You must navigate through the treacherous **asteroid** environment and uncover a way to escape. Along the way, you encounter a helpful **NPC** who provides a vital clue about the whereabouts of the enemies that hold the key to your freedom. To aid you on your quest, the NPC bestows upon you a valuable elixir to restore your health, which is securely stored in your **inventory**. This inventory serves as a repository for essential tools, including your trusty sword and powerful gun, both of which can be wielded for **melee** and **ranged** attacks. The inventory system allows you to access your collected items and their descriptions whenever needed.

As you explore the asteroid, formidable **enemies** cross your path, each presenting a unique challenge. These adversaries are relentless in their pursuit, skillfully maneuvering through **obstacles** to track your every move. Engaging in combat with these foes not only tests your skills but also rewards you with valuable **experience points** upon their defeat.

The NPC's earlier mention of a hidden **cave** piques your curiosity, compelling you to venture deeper into the asteroid's depths. Inside this cavernous domain, you encounter even more fearsome monsters and paths without return. Your journey reaches its climax as you confront the formidable **boss** lurking within the depths of the cave. Defeating this boss grants you the crucial card to access your ship that his minions had taken to him as an offering.

Returning to your ship after your adventure, you are now able to keep driving through the vastness of space. Armed with newfound abilities, you prepare to embark on your next thrilling adventure.

Our game features an intuitive and immersive GUI that greets you with our captivating **logo**. With a simple click of a button, the journey unfolds before you. At any point during gameplay, you have the flexibility to **pause** and **resume** the game, ensuring a seamless and uninterrupted experience. Additionally, you have the option to mute the **audio**, or revel in the multiple **sound effects** that enhance the game's ambiance, from the immersive GUI sounds to the intense shooting and the satisfying audio cues upon vanquishing enemies. Keeping you informed at all times, your **HUD** shows your health, experience points, and level, allowing you to monitor your progress and growth.

Our goal in creating this game was to provide an immersive RPG experience with a futuristic theme. We meticulously crafted a captivating storyline, complemented by synchronized audio and stunning visuals, to ensure an engaging and immersive gameplay experience. By incorporating mechanics of combat and achievement, we aimed to captivate players with exciting battles and rewarding progress as they navigate the thrilling world we have created.

Contributions of Marin Susic

During the course of development of Simperia I have worked the most on implementing features that would improve the user experience (sprites, sound effects). On the other hand, I have also worked on the animations of so called sprites, and solving bugs that occurred as well as performed the initial designing of the levels and later positioning of entities.

Level design

At the start of the project I was tasked with designing the 2 levels so that they make sense.

- The idea was to make the player have a quest that they will have to walk through to achieve the end of the game
- First level is designed so that the first thing the player encounters is an NPC that gives the quest
- After the quest is given, the player cannot reach the end part of the map without going around due to the crack (originally river), so they must follow the storyline
- The goal is to beat the octopuses in order to solve the quest
- After the first level the player travels to the second world to beat the "boss" or the leader of the octopuses, and this will also give the player the final piece to the quest
- The player has to navigate his way through the 2nd map, both ways are solvable, but out of the 4 that are offered, 2 are easier then the other 2.

Sprite and movement implementation

- Implemented the animation of the player moving
- Implemented the attack animations for melee attacks
- Helped implement the movement function with the keyHandler
- Implemented the ability of the player to have a sword
- Implementation of the bullet sprite animation

Positioning

- Due to the fact that I designed the levels (sketch), I was in charge of placing the NPC, opponents and warps
- Like stated before everything is positioned to allow the player a challenging but fun experience
- 2nd level is designed so the player has to make the right path choice in order to have an easier time passing the map and beating the boss

Sound implementation

- Implemented the sound effects on certain cues such as death of the player, shooting, moving (steps), etc.

Debugging

- Debugged the sprite problems (resolution)
- Debugging of the player resetting to looking down when not moving
- Debugging the fact that the player would always shoot down when in STILL state, fixing the getAimAngle() function
- Debugging the player being able to run out of the map
- Helped debugging of the maps being drawn weird when the player would reach the end of the screen, the issue was with the camera continuing even though the player is stable

Contributions of Sophia Soares

In the game project, I took responsibility for a portion of the front-end development. My focus was on creating various panels that were crucial to the game's interface.

Home panel: it serves as the central hub for the player. The design is simple and objective, and it consists of the game name and a button that allows the player to start a game.

Pause panel: enables players to take breaks and resume gameplay seamlessly. It features intuitive controls for pausing, and resuming, and a mute/unmute button for the game's soundtrack and effects, aiming for a more personalized gaming experience.

Inventory panel: plays a vital role in managing and organising in-game items. The panel has 12 slots for displaying different items and their quantities and I incorporated visual cues such as distinct colours for selected slots, making it easier for players to identify the items they have chosen. The item name and description are also visible when selecting it. When using or removing an item, the inventory panel reflects the changes accordingly. This means that players can easily use items with the click of a button or delete unwanted items, resulting in an up-to-date inventory display.

Dialog panel: essential for conveying storylines and interactions with NPCs. The primary objective was to create a simple yet clean interface that effectively displays quests for the player to undertake within the game, maintaining a minimalistic design while ensuring that the displayed content is easily readable and accessible.

Information panel: designed to appear at the beginning of the game, providing players with a brief overview of the story and its context.

Victory/fail panel: provides players with feedback based on their game performance. When the player successfully completes the game, the panel appears, accompanied by a congratulatory message. On the other hand, if the player fails to achieve the game's objectives, the panel is triggered, displaying a message reflecting the outcome. From this panel, the player can also start a new game.

In addition to the panels, I also developed the **HUD (Heads-Up Display)**, which constantly displays critical player information. To ensure an intuitive and visually appealing HUD, I implemented dynamic elements for health and experience points (XP) display. Instead of just numeric values, I used progress bars that visually represent the current levels of health and XP. These bars change in length depending on the respective amounts, allowing players to measure their status at a glance. A profile picture of the player is also visible from the HUD, adding a personal touch to the gameplay.

Overall, my contributions to the front-end development of the game project included designing and developing the home, pause, inventory, dialog, information and victory/fail panels, and HUD. These elements collectively enhanced the user experience, making the game more enjoyable and immersive for players.

Contributions of Savo Simeunovic

Mob implementation

There are 3 types of mobs in the game: Octopus (range), Maurice (mele) and NPC. I was dealing with all implementation of these mobs and their bug fixing whenever something popped up. Drawing the sprites of these mobs (depending on the direction of movement) was done by me as well. I created a utility function to flip the sprites horizontally and vertically so we don't have to store additional sprites, improving our game's performance.

Detailed:

- Movement of mobs (including the random roaming around the map which we decided not to keep) toward a player when player is in range
- Mobs and Player taking damage and state changes if health goes below 0. Enemies get removed.
- Changing the sprites based on if Octopus is trying to attack the player or not. At closer range, it shoots but at a bit further range it is just following the player.
- In each fixed timeframe, the Octopus shoots in 8 directions. Creating these bullets is done by me as well which are following certain directions until they hit a player (in Enemy Bullet case)
- NPC interactions with the Player.
- Quest implementation, returning right strings to be displayed (without spamming, just once). If the quest is done, the player gets a reward in its inventory.
- Removing the bullets after some amount of pixels so we don't waste the memory or when the button is collied with an entity

Bigger Bugs

- Fixed bug: hitboxes were shifted 24 pixels for the player to be in the center
- Fixed bug with a Boss Octopus hitboxes and the range not correct
- Panels (muting, inventory) were flickering
- Home panel wasn't starting the game
- When loading warps, the bullets were still showing
- When a bullet killed the player, an exception is called when removing that player bullet

Contributions of Luca Besana

In this project my role was to manage the environment and to take on some side tasks and overall help the other team members if they were having trouble.

In detail, my contribution has been to:

- Project setup: I have created the first shell of the project that we used, including the graphic setup and frames. Since we started from scratch, I used a structure similar to that provided by professor Hasbarge's examples that I had already tested previously with a smaller project.

- User inputs: I created the classes and methods to handle user inputs, they're first received and sorted from a class and then called in the main game loop, where they are reacted to according to our needs.

- Camera: I implemented the camera, which holds the player in the middle while he's far from the map's borders and when he approaches them, the camera gets locked and he moves freely towards the border. The camera works on a big drawing made by Julia Agüero, so it's essentially cutting that bigger picture into the frame surrounding the player.

- Map class: Since this is an adventure game, we needed to be able to load different levels one after the other, so I made loadable maps that contain mobs, items, warps, a picture, a grid for collision detecting. These maps are loaded in memory only when accessed and the previously loaded one gets removed.

- Map collision: Since our maps were designed first by a drawing, to manage collision detection, I created a grid-based special partitioning. This way the collision detection doesn't require too much processing power and it's easy and reliable.

- Warps: To access different maps, I created warps that can also be used to teleport the player around the same map. These warps are created with a map's ID on them and when collided, they are the starting point of the map loading and unloading process.

- Sound class: I created a simple class to load and handle sounds, with a few methods to start a sound, loop it, pause it and other minor things. This gave Marin Sušić the tools to work on the audio aspects of our project.

Contributions of Julia Agüero

Throughout the development of the game, I have made significant contributions in various aspects, showcasing a diverse range of skills. From managing the project's organization to implementing essential classes, gameplay mechanics, and providing debugging support to creating all the artwork. This document highlights the number of versatile tasks I have accomplished:

1. Game Organization: I organized the game development process by breaking it down into smaller manageable tasks and creating a clear structure. I used the project management software Asana to track and assign these tasks to team members, ensuring efficient project management and progress tracking.
2. Sprite Drawing: I implemented the drawing of all the sprites for different entities in the game, including the Asteroid map, Cave map, NPC, enemies such as Octopus and Monster, bullets, player animations (with sword and gun), and inventory items. I created these sprites using Procreate on the iPad, ensuring they have multiple frames for smooth animations and different directions.
3. Project Diagrams and Java Class Shell: In collaboration with Luca Besana, we created initial project diagrams to define the relationships and inheritances between different classes and how they would interact. I then implemented the basic shell of Java classes based on these diagrams, providing the necessary attributes for further development by the team.
4. Inventory Logic: I implemented the logic for the game's inventory system. This includes handling floating inventory items in the game world and placing them in the inventory. Items are automatically stacked up to a maximum quantity, and they can be used by the player or removed from the inventory as needed. Each inventory object implements its own logic when is used.
5. Player Experience Management: I added functionality to the Player class for experience management, including tracking experience points, leveling up, and other related features.
6. Rocks dropping: I implemented the logic for the rocks appearing on the cave map when the player enters a certain corridor, locking him inside.
7. Ship and ship card: I also developed the ship and card system, with the victory of the game happening when this item is used when the player is colliding with the ship.
8. Lore Creation: As part of my contributions, I also took on the task of crafting captivating lore for the game. This involved writing the text for NPCs, as well as developing the initial narrative text that introduces the game and the final text that concludes the player's journey.
9. Mixed tasks: Throughout the development process, I participated in various small tasks such as creating a Coordinate class to have consistent access to size and position, creation of BossOctopus adapting the code from the normal Octopus, changing the floor pattern color in the cave when the boss fight starts and finishes, and searching and adding some sprites into the project. I also provided assistance and support to teammates, helping them with their tasks if they needed it.
10. And last but not least, I wrote and designed this booklet with the full information about our game, Simperia.