Bilkent University

Department of Computer Engineering

CS 319 Course Project

Group 1C-SS

Final Report

- Alperen Öziş
- Arda Göktoğan
- Cemal Gündüz
- Eren Şenoğlu
- Duygu Nur Yaldız
- Yavuz Faruk Bakman

Supervisor: Eray Tüzün

Contents

1. Introduction

We implemented all type of rooms. That includes fight rooms, unknown rooms, merchant rooms, event rooms, treasure rooms. We created the Map composed of these rooms. The user can play the game by simply using mouse and interacting with user interface elements. After the game, the user can save the game and can start the game where it is saved. We used the factory design pattern mostly. All implemented elements are visually very similar to the actual game.

2. Design Changes

We did not lots of change after the first iteration in design. However, there are some issues obviously. In first iteration, we were holding cards in database but we changed this implementation and we started to hold cards in classes. Because all cards behave differently and it does not make sense to hold them in database.We faced the same issue with the relics. We employed the same method with relics. Also, we employ the factory design pattern for almost every objects of the game such as cards, relics, rooms, potions, buffs. Lastly, we had to add Enemy and Pet Controller differently. Because their interfaces are different. As pets don’t have hp they are not fightable.

3. Lessons Learnt

We have learnt how to use factory design pattern for different subsystems. We understand the importance of sharing of works properly. As we can distribute the work among all group members in a proper way, we prepared the project on the time. Also, we realized that subsystem decomposition is a very crucial part for the design. As we achieve this step well, we only need minor class changes in these subsystems and debug the code easily. Analysis part is also very important. As we started the design step without having all information about the cards, buffs, relics etc., we could not add all features of these elements. We could not arrange the power of enemies and fight rooms because of the analysis part. Lastly, we have decided which objects are hard code or stored in database late. Therefore, we need to change implementation in some places.

4. User’s Guide

In this chapter, we explain how to play game.

4.1. Main Menu Explanation

In the main menu, user can start a new game, load game change settings, quit the game and view statistics, patch notes and compendium. In setting it is possible to adjust the volume. If he chooses new game, he selects the character he want to play then it starts. If he wants to load game, he must choose the save file that he wants to load.

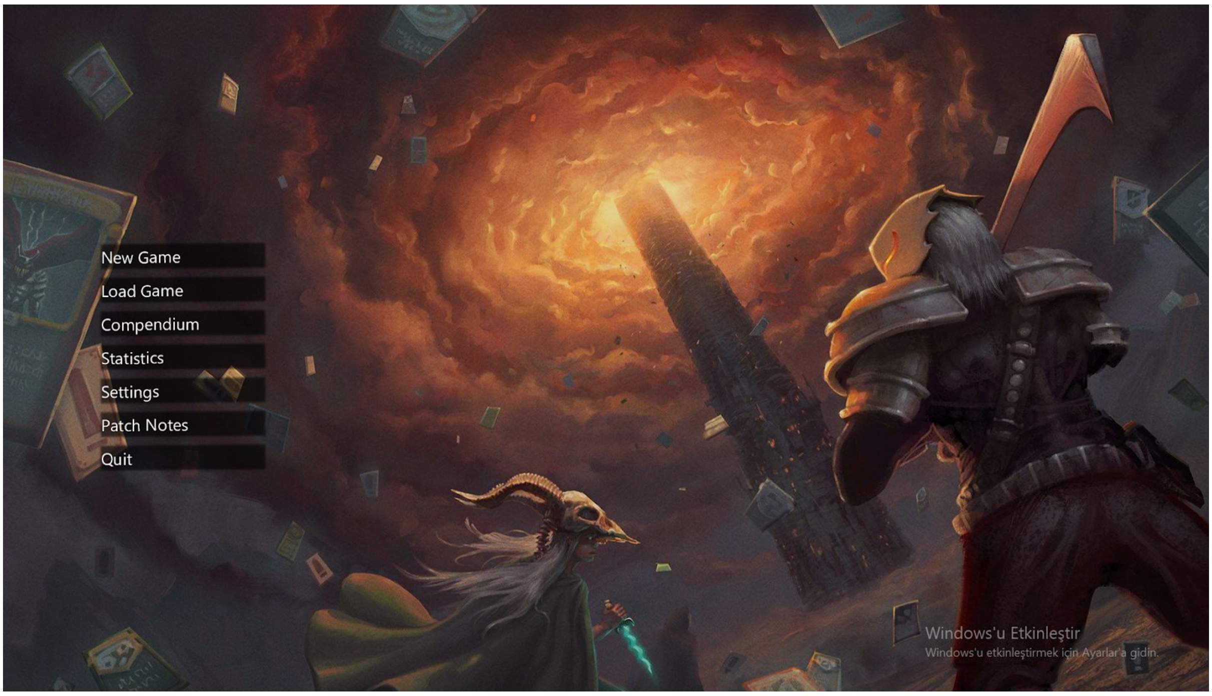


Figure 1: Main Menu

4.2. Map Explanation

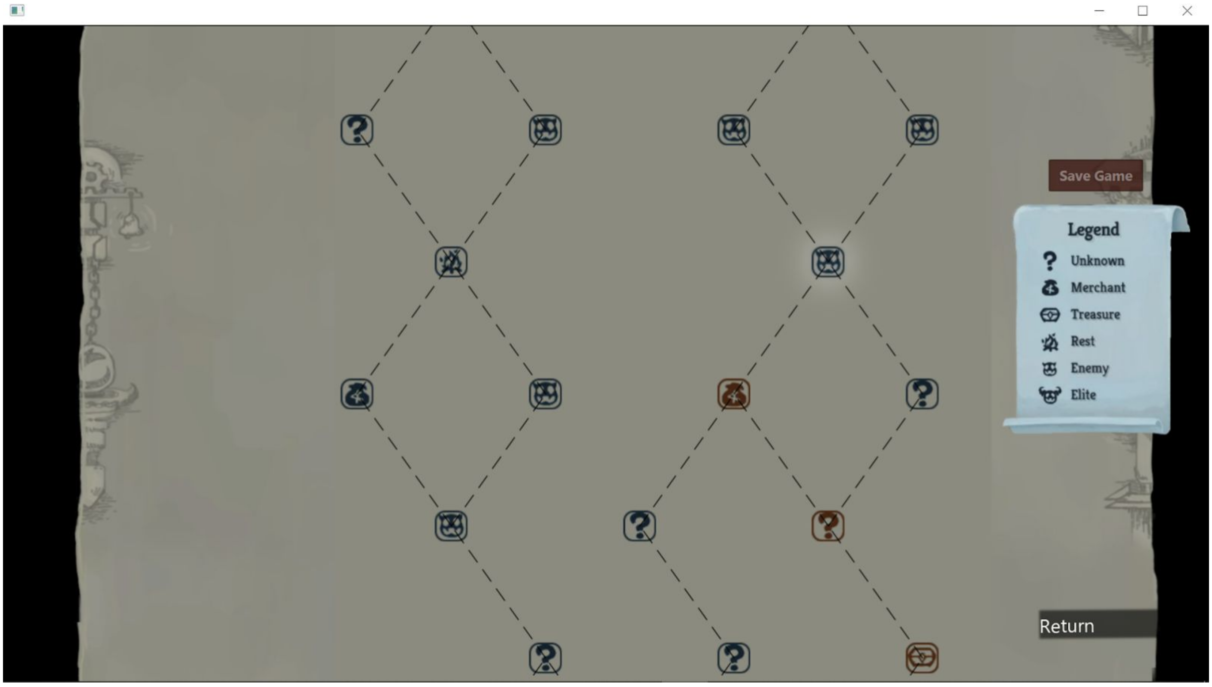


Figure 2: Map

In the Map, the icons show the type of the rooms. The user clicks the icon and enters the room. All rooms are not available obviously at the same time, it follows the path. The legend shows the meaning of the icons.

4.3. Fight Room Explanation

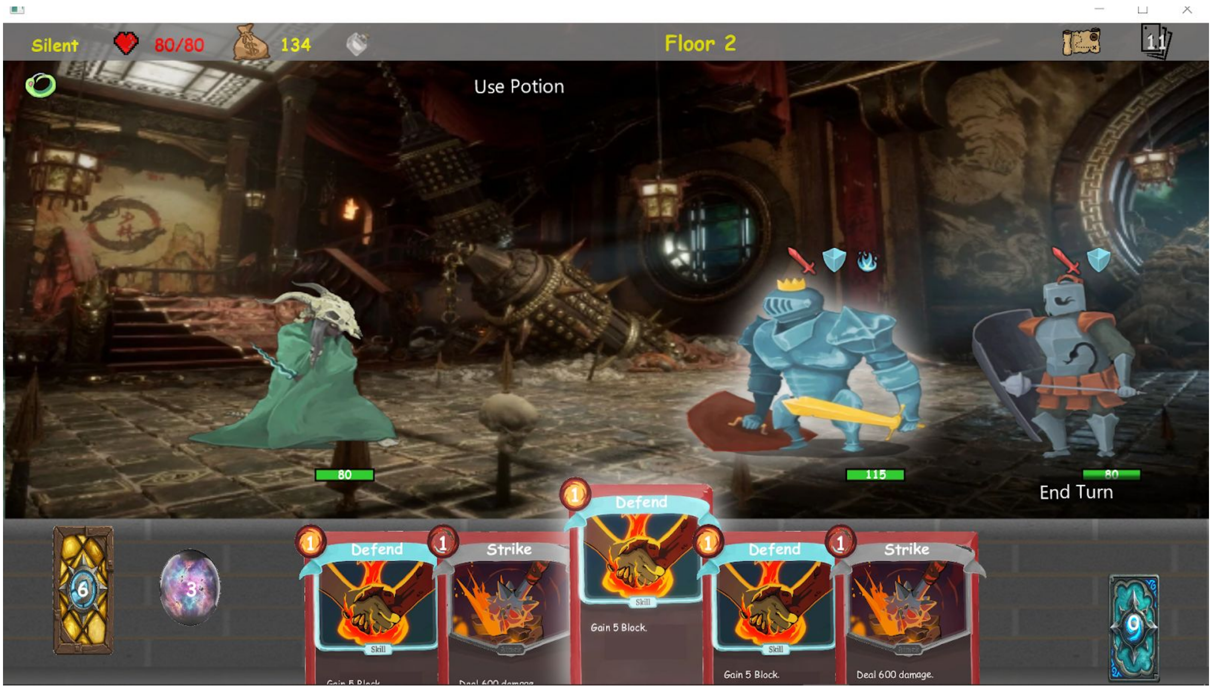


Figure 3: Fight Scene

In the fight room, the user can click the cards below and after choosing card by the mouse, the user can choose one of the enemies or the character. If energy is enough for the card, the effects of the card are realized. The player can see the stats of the character from the upper bar. When the mouse come over the cards, the card become bigger and more visible. After each play of the user, the user clicks the end turn button and enemies play its turn.

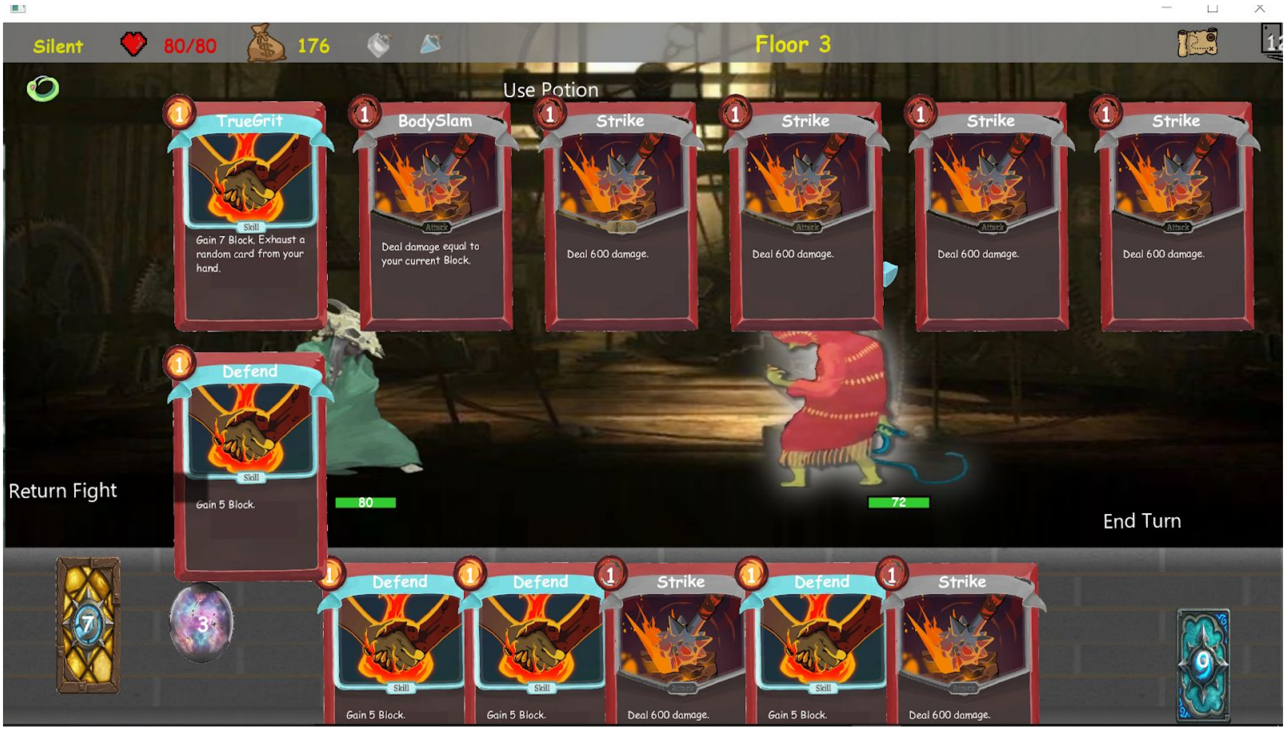


Figure 4: Fight Scene - See Pile

When the user clicks the piles, the list of the cards can be seen. Figure 4, shows this event.

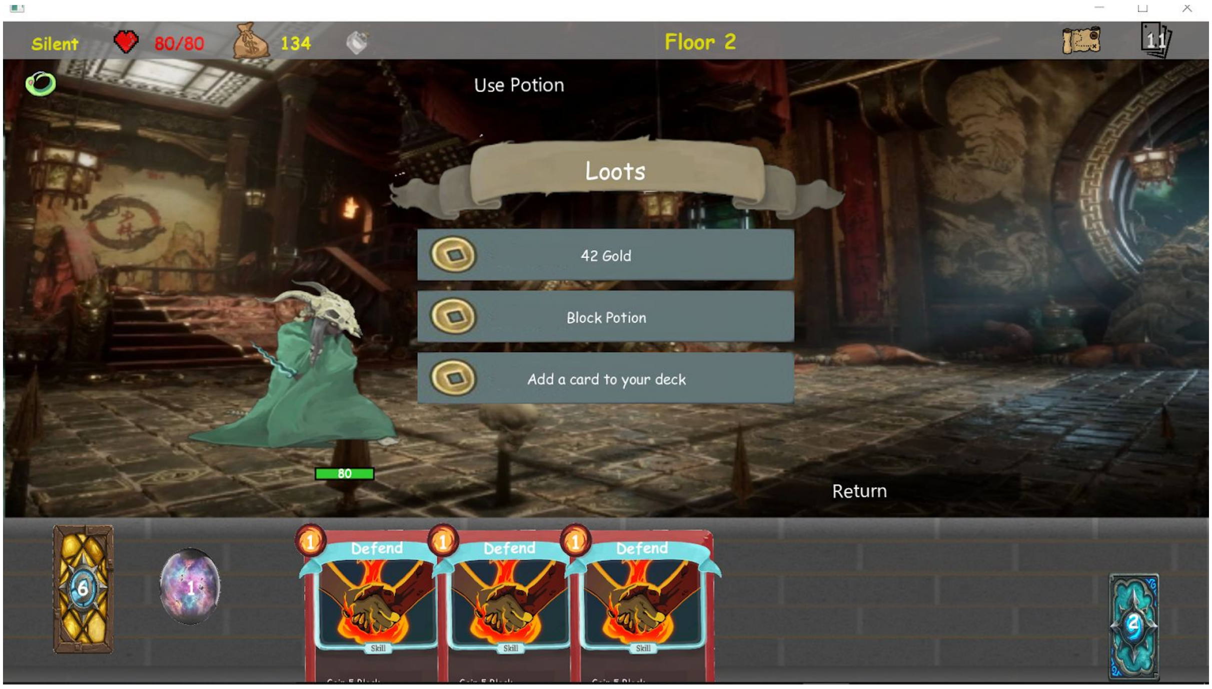
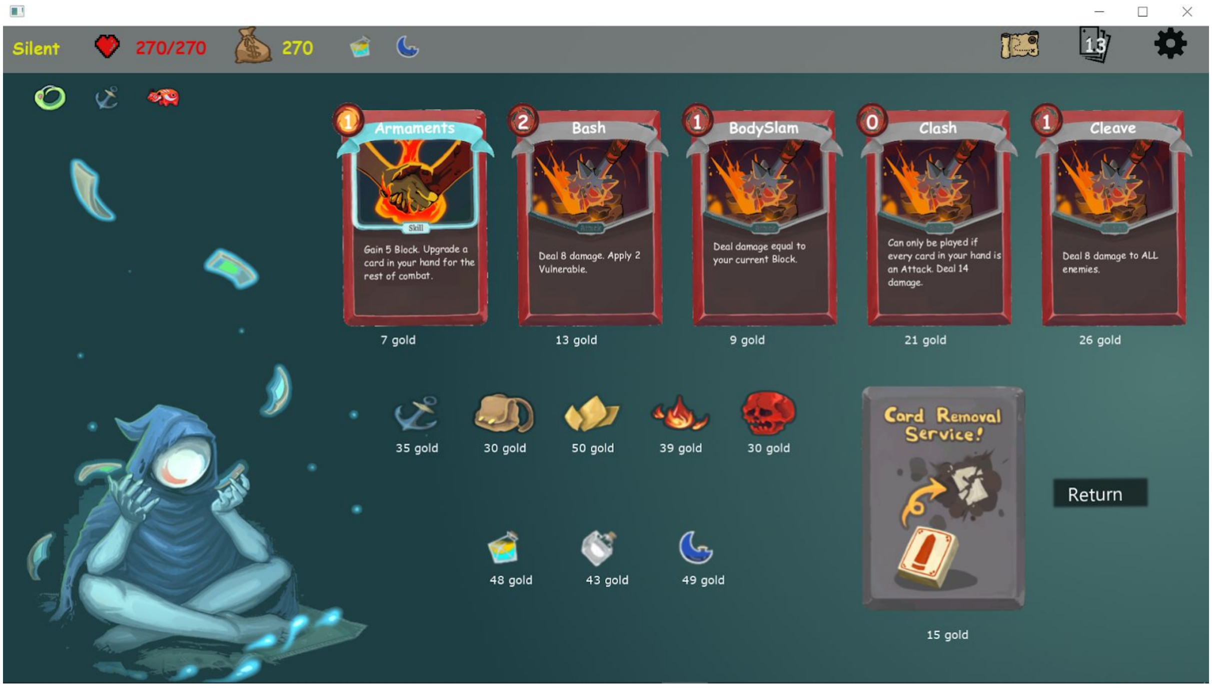


Figure 5: Fight Scene - Rewards

If the character's hp become zero before the enemies' hp, the game is over and returns the main menu. If the player wins the fight, the loots scene become visible. The player can click the loots and take it. After taking them, by the return button, the game moves the map scene.

4.4. Merchant Room Explanation



Gain 5 Block. Upgrade a card in your hand for the rest of combat.

Deal 8 damage. Apply 2 Vulnerable.

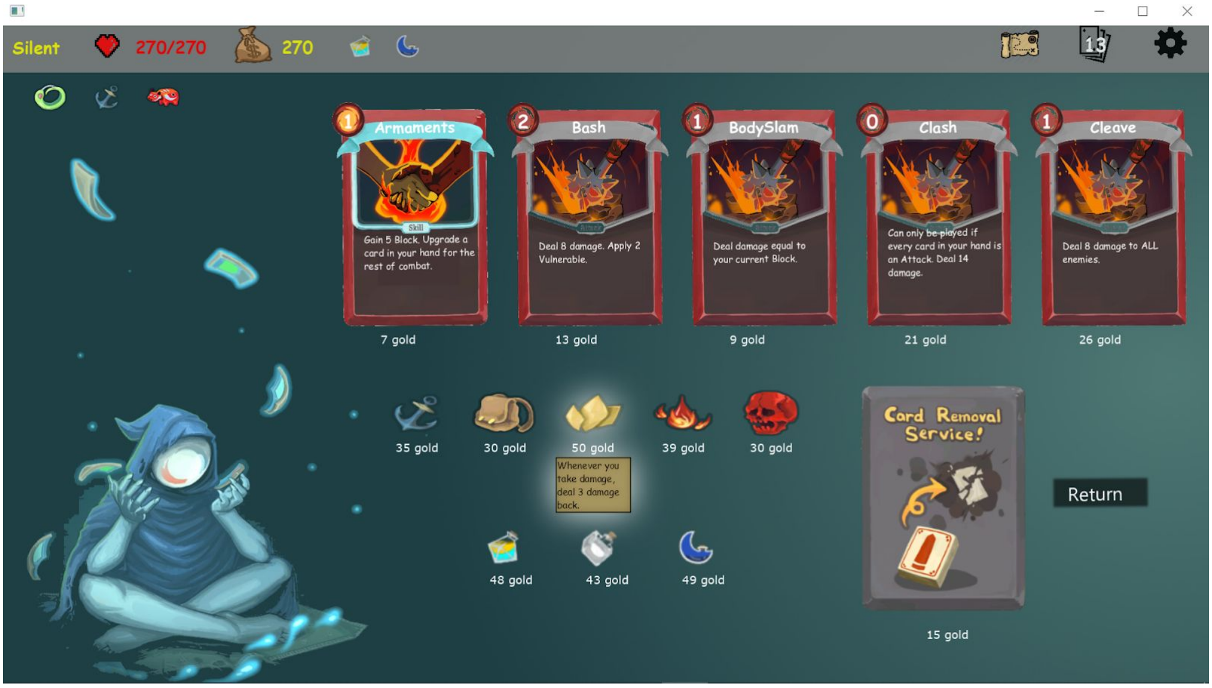
Deal damage equal to your current Block.

Can only be played if every card in your hand is an Attack. Deal 14 damage.

Deal 8 damage to ALL enemies.

Figure 6: Merchant Room

In the merchant room, the player can buy cards,potions,relics. To do that, the player clicks the element the player wants to buy.



Gain 5 Block. Upgrade a card in your hand for the rest of combat.

Deal 8 damage. Apply 2 Vulnerable.

Deal damage equal to your current Block.

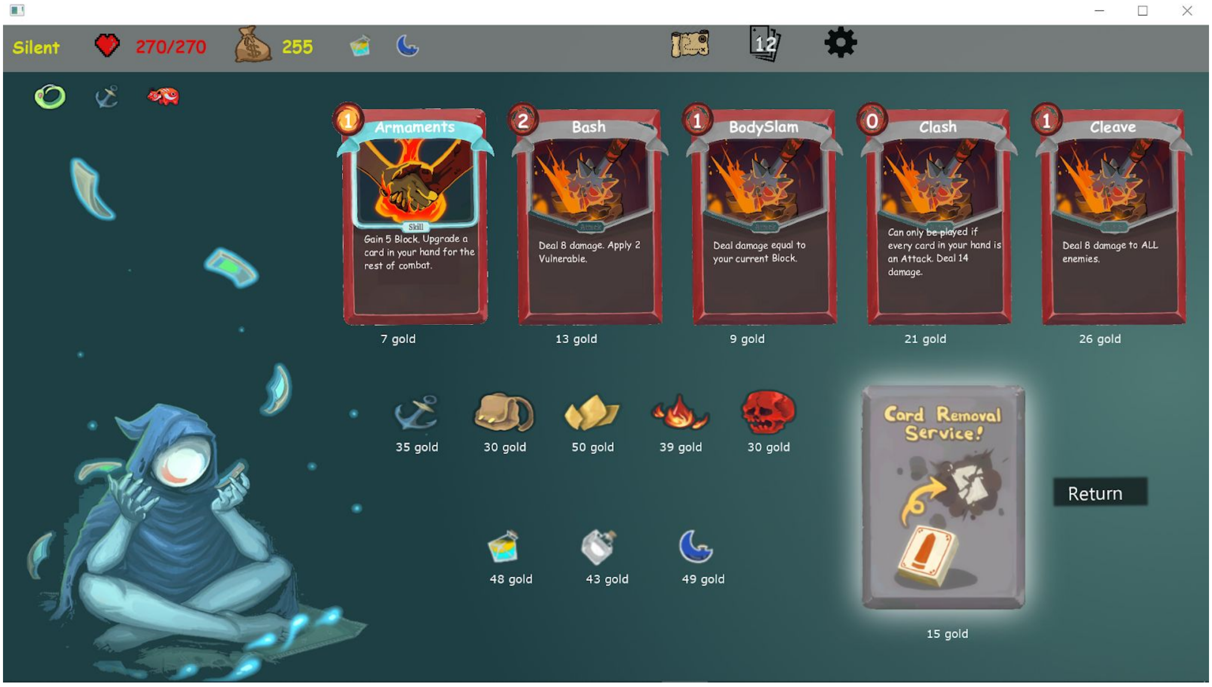
Can only be played if every card in your hand is an Attack. Deal 14 damage.

Deal 8 damage to ALL enemies.

Whenever you take damage, deal 3 damage back.

Figure 7: Merchant Room - Relic Explanation

If the player moves the mouse on the potions or relics, the description appears.



Gain 5 Block. Upgrade a card in your hand for the rest of combat.

Deal 8 damage. Apply 2 Vulnerable.

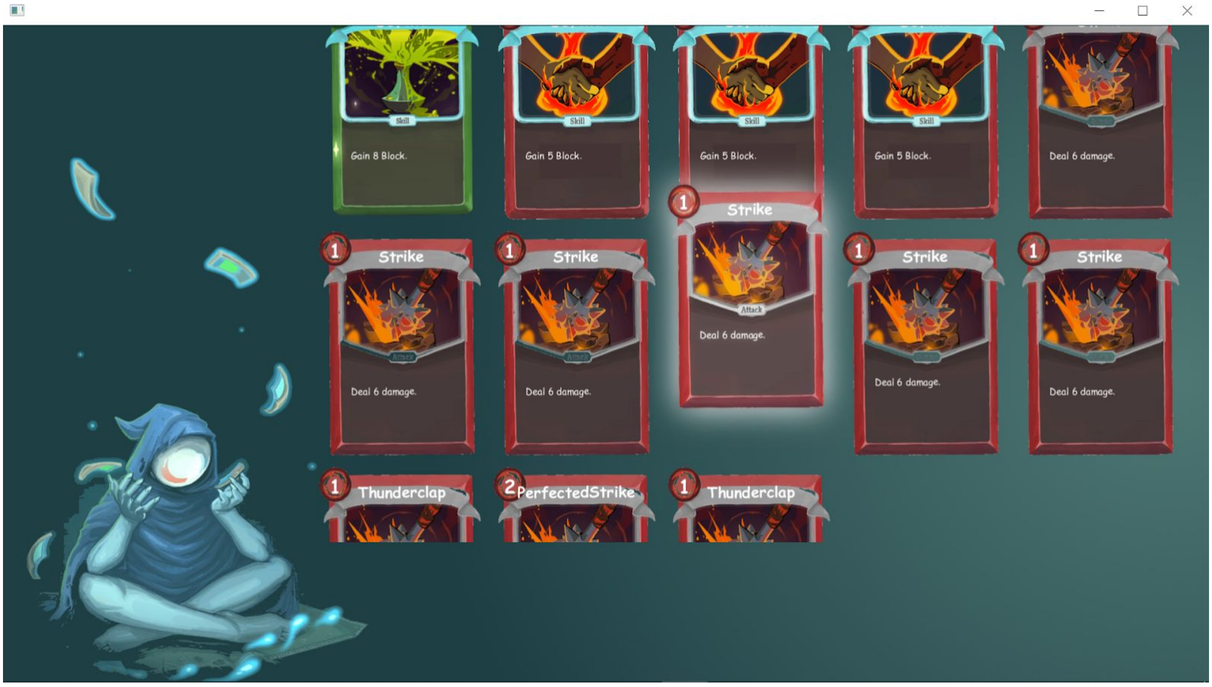
Deal damage equal to your current Block.

Can only be played if every card in your hand is an Attack. Deal 14 damage.

Deal 8 damage to ALL enemies.

Figure 8: Merchant Room - Remove Card

There is also a card removal service. The player enters this service by clicking the Card Removal Service icon.



Gain 8 Block.

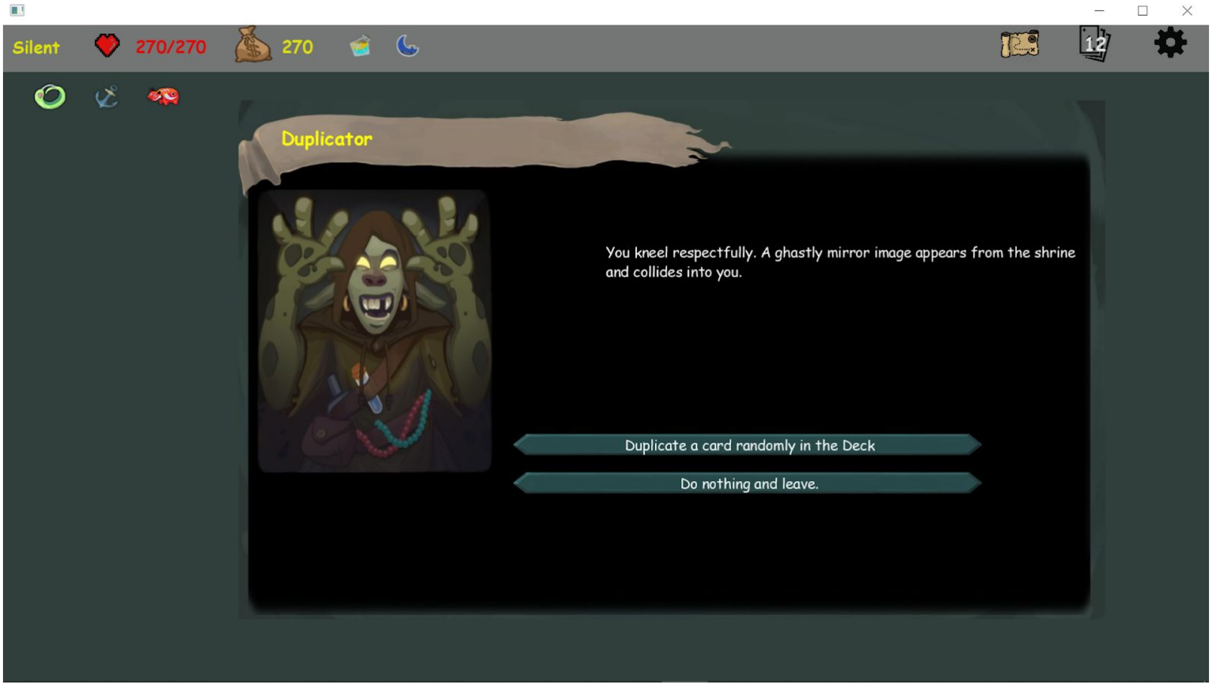
Gain 5 Block.

Deal 6 damage.

Figure 9: Merchant Room - Choose a Card to Remove

After clicking the icon, the cards that player have displays. And the player clicks the card to remove it.

4.5. Event Room Explanation



You kneel respectfully. A ghastly mirror image appears from the shrine and collides into you.

Figure 10: Event Scene

In the merchant room, the player can see a dialogue and clicks one of the options. After choosing the option, the effects of the option are realized.

4.6. Rest Room Explanation

In rest site player can choose either smithing or resting.

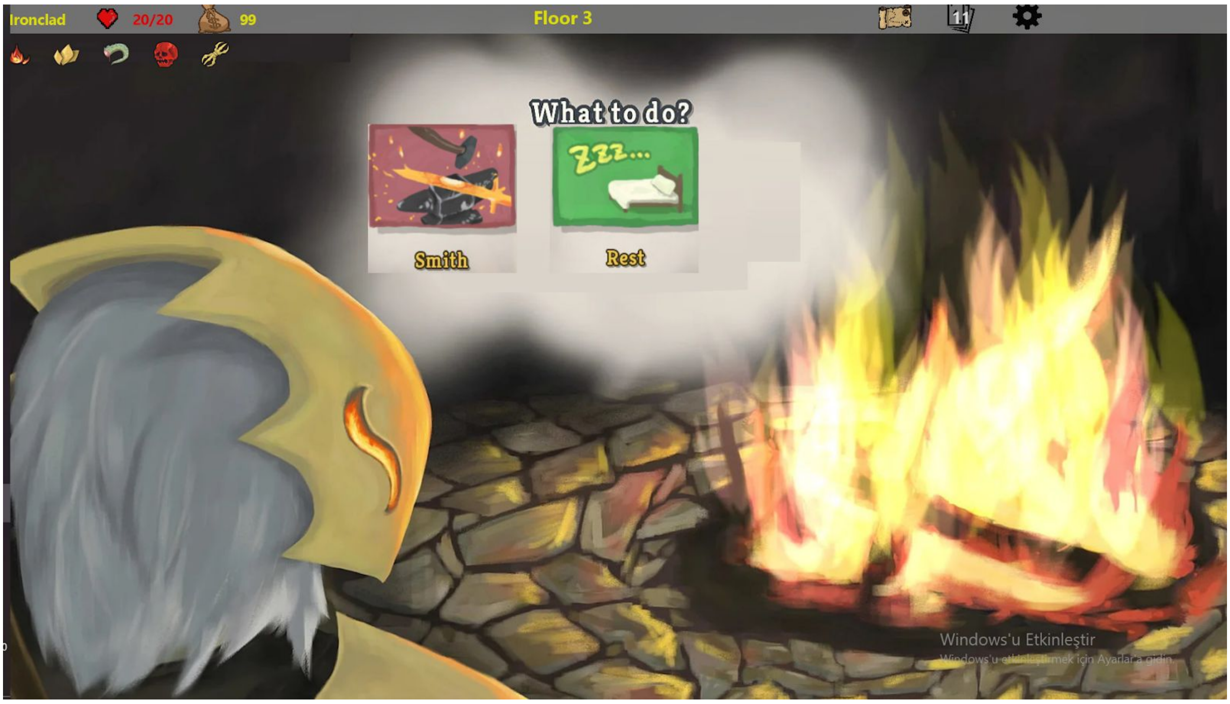


Figure 11: Rest Room

If he chooses to smith, he should select which card he wants to upgrade in his deck.

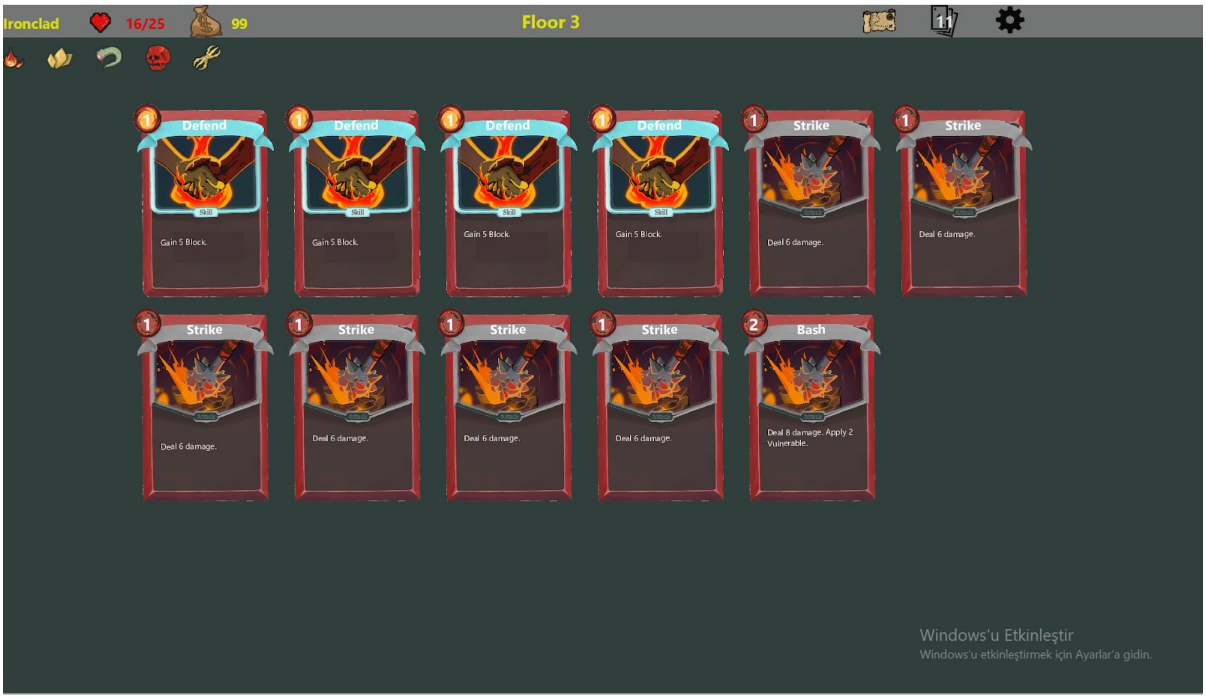


Figure 12: Rest Room - Choose a Card to Upgrade

If he clicks a card, he can see the upgraded version of that card. By clicking the “Upgrade” button, the user upgrades that card, and return to the menu. If the user choses “Return”, the game returns the previous scene to choose a card to upgrade.

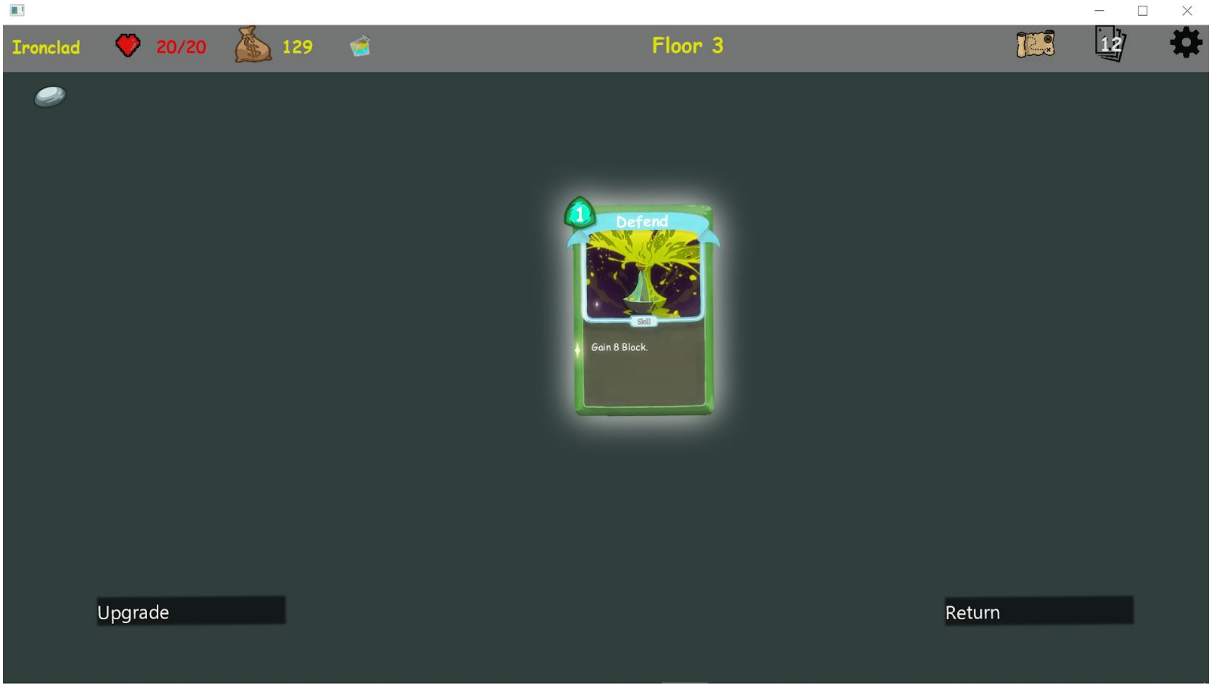


Figure 13: Rest Room - Upgrade a Card

4.7. Treasure Room Explanation

In treasure room player can take the treasures in the chest.

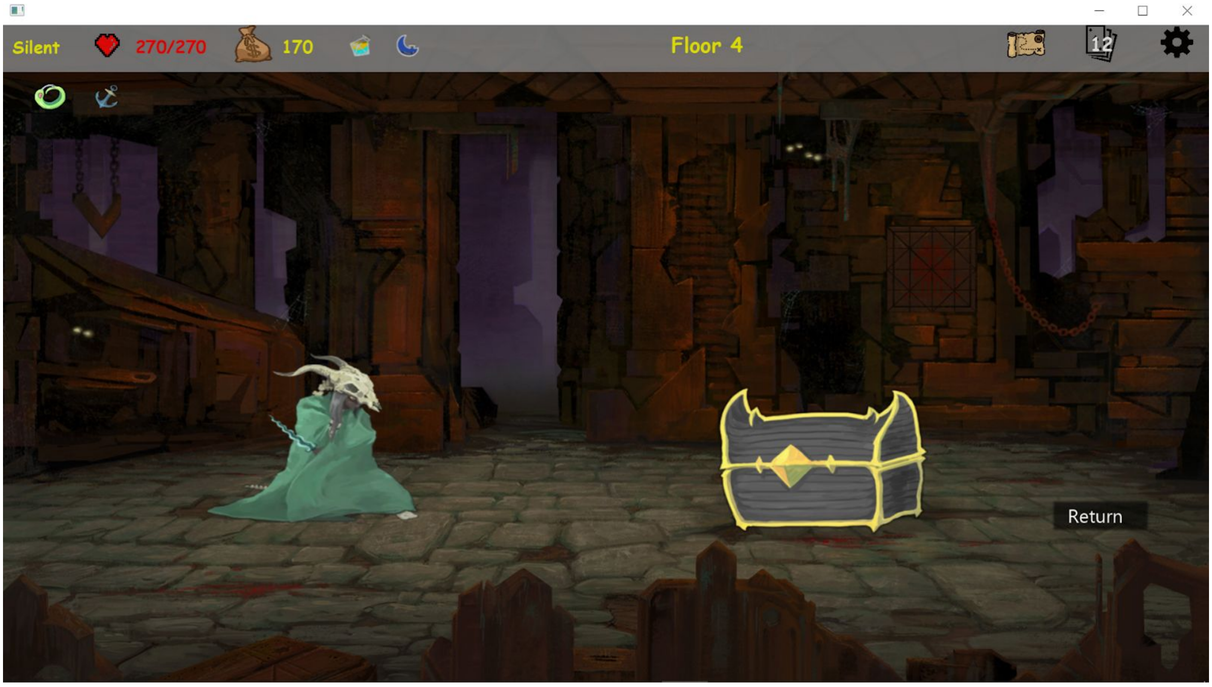


Figure 14: Treasure Room

First, the player clicks the chest image and chest become open.

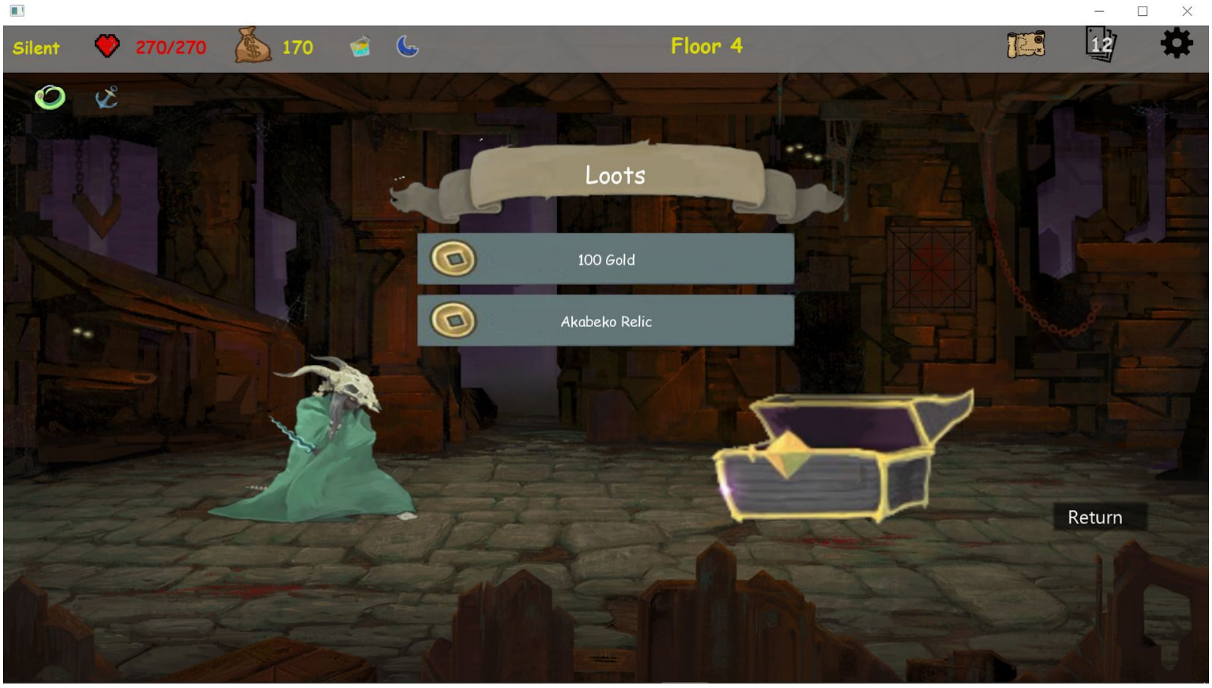


Figure 15: Treasure Room - Get Rewards

After clicking the chest image, the list of loots appears and the player clicks the loot to take them. After that, by clicking the return button, the map scene appears.

5. Build Instructions

To install the game, the user needs two JREs installed on the computer. Here are the libraries and links to download:

- JSON simple library:

https://code.google.com/archive/p/json-simple/downloads

- Javafx library version 11.0.2:

https://gluonhq.com/products/javafx/

The .jar files of the libraries should be added to the library paths of the project. (In the intelliJ, it could be added as follows : File>Project Structure>Libraries)

User should add the PATH_TO_FX to PATH variable as follows :
path/to/javafx/lib

Then use should set the vm options in the IDE as follows (for intelliJ it is in run>Edit Configurations>vm options):

```
--module-path ${PATH_TO_FX} --add-modules=javafx.controls,javafx.fxml
--add-modules=javafx.controls,javafx.fxml --add-modules javafx.controls,javafx.media
```

For configuration settings, MainMenu.java file (inside GUI package) should be selected as the main class.

For different operating systems, folder paths might be different. Adapting this little difference is very easy in our code. The user should only open the GameSaver.java file which is in the DBConnection package. And inside this class, the user should only follow the instructions at the top of the class, which is uncommenting the related static properties and commenting the other os related properties. Clear instructions are present in the code.

6. Work Sharing

Alperen Öziş

- Model class implementation of Buffs and Relics
- Testing and debugging of Buffs and Relics.
- Help the gui of merchant room.
- Design of gui class diagram.
- Design of fightroom use case.

Arda Göktoğan

- Application Domain class diagram
- System decomposition with Duygu
- Choosing architectural pattern
- Design of Fight Control Subsystem
- Design algorithm of Random map generation and implementing it

Cemal Gündüz

- Design and implementation of user interfaces of Main Menu, Fight Scene, Map Scene, some minor classes for GUI.
- Collecting and editing visual components of GUI.
- System decomposition with Duygu
- Assisted in the implementation of Control and Model classes.
- Use case diagrams of Map, Menu and Fights.
- State diagram of game and activity diagram of a turn in the fights.

Duygu Nur Yaldız

- Application Domain class diagram
- System decomposition with Arda
- Design and implementation of Menu, Map and Room Controller subsystems except Fight Controller.
- Design and implementation of game saving and loading operations, and GUI implementation of it.
- GUI implementation of Merchant room and Treasure Room with the help of Alperen and Eren.
- Collecting and editing some of the visual components of GUI.
- Helped implementations of model classes.
- Relic, Card, and Character Factory Classes implemented.

Eren Şenoğlu

- Design and implementation of user interfaces of Main Menu, Fight Scene, Map Scene, Event Room, Rest Site and some minor classes for GUI.
- Assisted implementation of databases of enemies.
- Collecting and editing visual components of GUI.
- Implemented Buffs with Alperen.
- Assisted in the implementation of Control and Model classes.
- Use case diagrams of Map, Menu and Fights.
- State diagram of game and activity diagram of a turn in the fights.

Yavuz Faruk Bakman

- Sequence diagrams of Fight Room

- Subsystem decomposition of GUI
- Design and implementation of Rooms.
- Database services of Rooms, enemies, pets.
- Assisted in debugging process of control classes.
- Helped implementations of model classes.

.