CS 319
Object-Oriented Software Engineering Project
Analysis Report

Section 1 / Group 1H

Ece Altınyollar

Arif Can Terzioğlu

Raza Faraz

Emin Tosun

Instructor: Bora Güngören

TABLE OF CONTENTS

1) Introduction

Battle For Middle Earth is an arcade game similar to well-known game ‘Space Invader’. We’ll basically stick the basics of gameplay of the original game. Our game’s aim will be similar to the original game, which is killing all the enemies but in a Middle Earth Theme. Instead of bombing, we’ll use weapons suitable to Middle Earth environment. There will be three types of races which are humans, elves and wizards. Respectively weapons are swords, arrow and staff. The main object of the game is to reach the final checkpoint on the map. Map representation will have a similar structure to candy crush level map, however in a Middle Earth manner.When player unlock a level he/she can replay this level. Within each level the user will begin the game at the bottom of the screen and the enemies will occupy the first couple of rows from the top of the screen. The user can attack the enemy, likewise the enemy can attack the user. The user’s health will be indicated by a health bar, while his armor will be represented by an armor bar.

To complete the requirements of the course and make game a better, we will make the following additions to the original game.

- Power ups
- Extended Weapons
- Enhanced and various characters
- Music and sound options
- Ammo boxes
- Middle Earth Environment

The target platform of the game is JAVA SE. Keyboard and mouse will be used to control the game.

2) OverView

2.1) Game Play

In “Battle for Middle Earth” you are the last hope to defend Middle Earth against the orks invaders and bring back peace to the land. “Battle for Middle Earth” is an arcade game inspired by the then popular game known as ‘Space Invaders’. In this game you, the user , would be required to shoot at the waves of orks armies until all of them are eliminated. But in order to carry their invasion forward the orks too can fire back at the player. The player would be allowed to step in the shoes of different characters (Humans, Wizards and Elves) and play as them.Each of these characters will possess a unique abilities ,in order to aid for the task. The player would start off his journey by being positioned at the bottom of the screen and would be allowed to move in all directions. Whilst the enemies will appear at the top couple of rows of the screen, whose movement will only be restricted to the left and right side.This game consists of 5 Levels. For the first four levels the user will face off against normal soldier orks and finally in the fifth level the user will face against the orks King, whose abilities will be far greater than a normal soldier orks.

2.2) Levels / Map

The game includes a game map with a middle earth theme.This map will be similar to the level map representation of candy crush where the players can see the levels of the game. At the beginning of the game all levels except the first one is locked. (See mockup page:) To unlock these other levels players should pass the levels one by one. And when a level is unlocked player can replay this level. The Battle for Middle Earth will include 5 different levels. The arrangement of these levels are in the order of difficulty meaning While the levels goes up the enemies gets more powerful and it gets comparatively harder to destroy them. This level system allows the user to experience the game in a more entertaining environment and makes the game experience more challenging.

2.3) Characters

Three different characters are offered for players.This diversity of the choosing characters offer players more detailed game play. These characters are from the middle earth theme. Players can choose whether they want to play as a wizard, elf or a human. Each of these characters will posses unique and special features. In addition in order to distinguish each character , the image representation of the each character is will be different.

2.4) Weapons & Ammo

Weapons in this game have some resembles to both original space invaders and Lord Of the rings film series. While users choose their character, they will also select the weapon automatically because weapons are specific to Middle Earth Races. Basically, as we mentioned we have three races which are humans, elves and wizards. Each possessing its respective weapons such as sword for humans , arrow for elves and staff for wizards. As user progress in game, new weapons (specific to character) may come up during the battle moments. In other words, user can gain new item synchronously with the game. Also, experience coming from achievements will enable users to upgrade their weapons (This will be mentioned more detailed in Power-ups sections).

During the game if user can hit the target 5 times without taking any damage, this will also change the structure of weapons in a more damage manner. Weapons can be changed also with the character changed. In our game, ammo will not be infinite,and must be refilled by catching drops (as will be mentioned later).

On the enemy side, weapons will also be specific to characters and they will fire their weapons in a random order.

If we have a time, we may also make the weapons sensitive to type of enemies. For example that: wands may cause more damage to orcs with swords more than other weapons.

2.5) Power Ups

There will be some power up packs. Power up packs will appear on random places, between the character and its enemies, and they will appear for only some certain time period. When the power up run out of time, it will disappear and can not be reachable. There are three types of power up packs. They will affect player's weapon, health or armor. The power ups can be taken by just moving your character on that power up. When the player gets the power up, it will affect immediately, which means power ups can't be controlled by player.

2.5.1) Weapon Upgrade Pack

When the player gets the weapon upgrade pack, character’s weapon will be changed. As its changes, the damage of new weapon can be higher or lower than the previous one.

2.5.2) Health Pack

If player obtains the health pack, character’s health points will be regenerated. The amount which health pack has will be determined randomly.

2.5.3) Armor Pack

The armor pack will appear the same as the other power ups. This pack increases character’s armor points. The amount of armor that player will receive by getting armor pack will be determined randomly.

2.6) Health

The character's initial health will depend on each character, thus each character will posses a different amounts of health points. Health of a character will be decreased by taking hits from enemies.In order to help the player to observe his health, on the top of the game page there will be health bar.

2.7) Armor

The strength of armor depends on character. Each character has different amounts of armor points. Simply, armor decreases the damage that is taken by enemies. Under the health bar, there will be a grey bar which represents character’s armor. This bar helps user to see his armor level.

2.8) Atmosphere

Our game will provide the user with Middle Earth Environment. Alongside the Middle Earth, because game dynamics will be similar to legendary game ‘Space Invaders’, feeling of nostalgia will take place. Battle screen background and map screen will be supported with images related to Middle Earth. Weapons structure will also be similar to ones in the movie. During the interactions between user and enemies, damage can be seen with some effects such a fire, incision... Characters and music will also be chosen among film.

2.9)Scoring System

Like most games this game too includes a scoring system. The scoring system will be based by keeping track of the number of enemies killed by the user. Initially each enemy killed would be worth a single point, but later on as the player avoids getting hit and at the same time manages to score 5 kills ,his score would increase proportionally to his ‘perfect streak’.In a scenario for example, initially each kill is worth 1 point but after 5 kills,while maintaining the perfect streak, each kill would be worth 2 points then 3 points and so forth as long as the player avoids getting hit.

3) Requirement Specification

3.1) Functional Requirements

3.1.1)Play Game

Before starting the game, player have to choose his character. After choosing a character the game will start. Player’s character will be located on the bottom of the screen and the enemies located on top of screen. The enemies will distributed and positioned randomly at the top of the screen. The amount of enemies and their shoot frequency is changed level by level. The enemies only objective is killing the character by shooting to it. While on the other hand , the player has to kill all of the enemies in order to finish the level.

All levels have different difficulties, whose difficulty will increase after each level. The character has some health points and armor points depending on his type. When the player loses all his health points, the level will end and player will lose. However, he has the chance to play same level for infinite amounts of times. In order to kill an enemy, player needs to shoot it more than one time depending on enemies' health points. Some power up packs will appear at random times and on random locations. Player would be required to hover his character over that power-up drop get that power up. These power-up can be one of three types ,armor pack, health pack and weapon pack. Player will get score points based on killing his enemies and the amount of 'perfect streak' as mentioned before. The main objective of the game is to stop the orks invasion by killing all enemies and finishing all levels.

3.1.2) Player Control

The player's movement will not be restricted, thus he/she would be allowed to move anywhere they please necessary. There is one exception to the above mentioned player movement rule, the player will not be allowed to move over the rows occupied by the enemies. This movement of the player would be done though the keyboard keys, more specifically through the up, down, left , right button. Shooting the weapon will be accomplished by pressing the 'Spacebar'.

3.1.3) Settings

The user will be allowed to change the game settings to his/her convenience while playing the game. Inside the setting options the user will have the authority to turn the game sound on/off.

3.1.4) Help

Users who are new to game need to check the help section. Users will be able to get game information from this help menu. This menu helps users to learn which keys they need to use. Also this menu shows the aim of the game and how they can complete the levels.In addition they will be able to learn what each power-up will represent

3.1.5) High Score

Users can see high score table by clicking High Scores button. The ten high scores will be stored in database. The account names and their scores will be given in the table format.

3.1.6) Choosing Characters

The screen of the choose character section opens after the play game screen. Before the players start the game, they have to choose which character they are going to play with. There are three different characters for players. In the choose character screen, players see three parts that explain all of the three specific character’s features. In order to choose one of the characters they click the button which is located down below of them.

3.1.7) Save Game

When the players login the game with their accounts, they can start the game at the level that they reached. While the players play the game they will not be allowed to save their games manually, but the game is saved when a user,for that specific account passes that level in order words, the passed level and the unlocked level is saved for this account. In order to save the levels that the players reach, database is used to keep the information of the account.

3.1.8) Create & Login Account

Create Account & Login screen is the one of the main functional requirement for the game. Once account created, users will not encounter with this screen again unless logout. User information will be stored in database. By creating account, user will able to see their high score, latest level they played, preferences and other information. Below are the information, the user has to enter when creating or logging onto his/her account.

- Create Account
    - Enter User-name

    - Enter Password
    - Enter E-mail
- Login
    - Username
    - Password

3.2) Non-Functional Requirements

3.2.1) Usability

The game is really easy to understand how to play and finish it. The reason why it will be high in its usability,because there are a few keys to play the game, thus making the game more usable. Also users will be allowed to create a new account without any modifications to the previous account's data.

3.2.2) Supportability

Since the goal of this project is to reach many people, supportability is an essential element for this software. The Battle for Middle Earth is designed with using Java language. Since the Java is suitable for the many platforms, people can run The Battle for The Middle Earth easily.

3.2.3) Reliability

In terms of Robustness, our game will be focused on handling unexpected termination and unexpected actions. Users will be informed with proper error messages. Especially with the principle of robustness, during the battle screen, user will be able to see the fluent gaming experience. Also, as we see from our PL course, other programming languages ,such as python etc, are not flexible and give errors most of time at compile time. Therefore, with java our game will be safe and more error-free. However, if out program does encounters any error , programmers should be informed as quick as possible after error occurred.

3.2.4) Performance

Transition between screens should be as quick as possible. System must support smooth characters changes if user wants. Run time performance of the game must be proper for user such as fast loading.

3.3) Pseudo Functional Requirements(Constraints)

The game will not be made using any game engines but will be made the primarily by using Java. As Java will allow easier Memory management i.e allow us to avoid memory leaks. Another reason why Java has been selected to make this game is because the UI will be made from scratch, which java will allow us to write in an efficient manner. For the Database part of the system the game will be implemented in SQL.

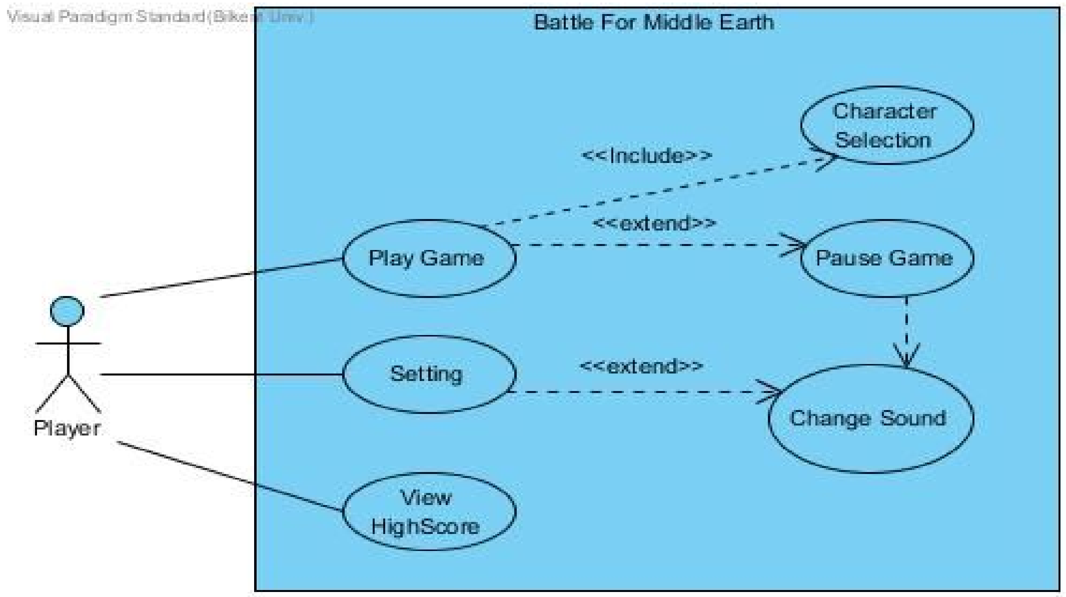
4) System Model

4.1) Use Case Model

The Use Case Diagram is a graphical representation of the interactions among the elements of a system. It shows the relationship between the user and the program.
The Use Case Diagrams of the Battle For Middle Earth is below. There are two different Use Case Diagram for the Battle For Middle Earth. First one is for describing the game play and the other one is for the other steps of the game in detail.

4.1.1) Play Game

The first Use Case Diagram for the Battle For Middle Earth is demonstrates the steps of playing the game.



Use Case#1

Use case name: Play Game

Participating actor: Player

Stakeholders and Interests:

- The objective of the player is to eliminate every enemy in each level and reach to the end of the game.
- Tracking of the player's score will be kept

Pre-Condition: The player should open the game, they should create or login their account. After that they can click the new game button from the main menu.

Post-Condition: The player should select a character from the choosing character screen. If they prefer they can click pause game option while they playing the game. The player can also change sound of the game through the pause game sections.

Entry Condition: The player interaction with “Play Game” button which is in the main menu.

Exit Condition:

1. The player interaction with the “Go To Main Menu” button which is in the pause
game section.
2. When the player dies during the game play, after which the player is directly sent to the ‘Main Menu’ screen.

Main Flow of Events:

1. Player logs-into his account
2. Players click the button to starts the game
3. The system loads the game
4. The player manages to eliminate all the enemies
5. Player advances to the next level until he reaches the end
6. The system returns the user to the main menu

Alternative Flow of Events

- The player closes the game,without ending the level

__Use Case#2__

Use case name: Setting

Participating actor: Player

Stakeholders and Interests:

- Player wants to change the sound volume or music volume.

Pre-Condition:

- Player needs to be in main menu.

Post-Condition:

- Volume levels are updated.

Entry Condition:

- Player is in interaction with volume change buttons and mute button in ‘Setting’ menu.

- Player is in interaction with volume change buttons and mute button in ‘Pause Game’ menu.

Exit Condition:

- Player clicks Go Back button in Settings Panel.

Main Flow of Events:

- Player clicks settings button in Main Menu.
- System opens settings panel.
- Player changes volume by using volume slider.
- System saves volume changes.

<u>Use Case#3</u>

Use case name: View HighScore

Participating actor: Player

Stakeholders and Interests:

- The objective of the player is to view the high scores.
- System brings last ten highest point from the database.

Pre-Condition: The player should open the game, they can click view high score button from the main menu.

Post-Condition: High scores are shown.

Entry Condition:

- Player is in interaction with high score button in the main menu.

Exit Condition: Player clicks ‘Go Back’ button in the high score screen.

Main Flow of Events:

1. User interacts with the high score button in the main menu.
2. System brings last ten highest point from the database.

<u>Use Case#4</u>

Use case name: Pause Game

Participating actor: Player

Stakeholders and Interests:

- Player wants to pause the game.

- System displays Pause Game menu.

Pre-Condition:

- Player needs to be playing the game.

Entry Condition:

- Player needs to press ESC button while playing the game.

Exit Condition:

- Player needs to press ESC button in Pause Game menu.

Main Flow of Events:

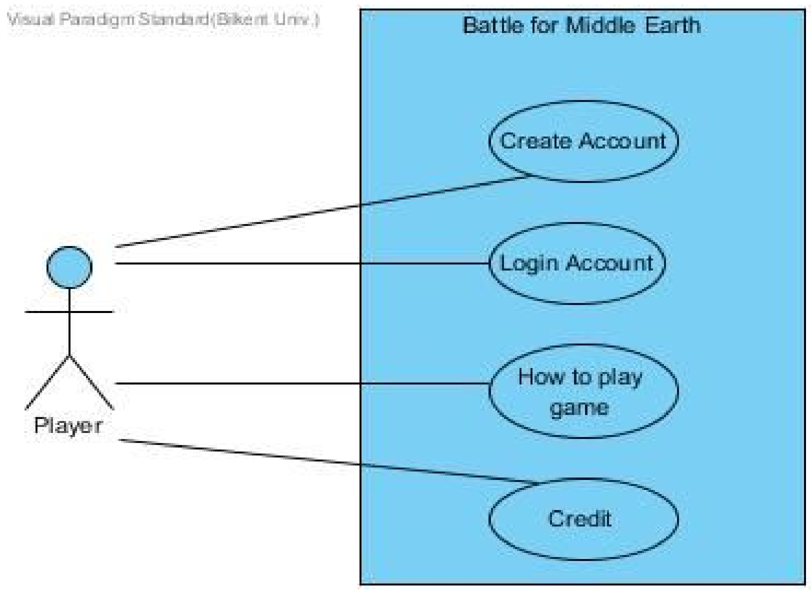
- Player clicks the ESC button.
- System displays Pause Game menu.
- Player interacts with volume slider or mute button.
- System saves volume updates.

Alternative Flow of Events

- Player clicks ESC button.
- System displays Pause Game menu.
- Player clicks Exit to Main Menu button.
- System returns to Main Menu.

4.1.2)Non-GamePlay Interactions

The second Use Case Diagram for the Battle For Middle Earth is demonstrates the steps of create account, login account, how to play and credit sections.



Use Case#1

Use case name: Create Account

Participating actor: Player

Stakeholders and Interests:

- · The player want to create an account in order to play the game
- · System inserts the account detail into the database

Pre-Condition: None

Post-Condition:

- · Account is created

Entry Condition:

- · Player runs the application

Exit Condition:

- · Player clicks Go Back button in Settings Panel.

Main Flow of Events:

1. The Player is directed to enter a non-existent user name and a password
2. The system confirms his creation of the account

Alternative Flow of Events:

1. User enters a username that already exists in the DataBase
2. A prompt is popped informing the user about the taken username
3. User is directed to enter the information again

Use Case#2

Use case name: Login Account

Participating actor: Player

Stakeholders and Interests:

- The objective of the player is to enter existing account on the database.
- Game opens the game relative to the existing account, which the user has enter.

Pre-Condition: Played must create account before login.

Post-Condition: Game opens with existing account which user enter.

Entry Condition:

- Player is in interaction with entering existing username and password.

Exit Condition: Player should click the logout button.

Main Flow of Events:

1. User enter the login button
2. Write the username and password section, then click enter button.
3. System opens the game with existing account.

Alternative Flow of Events:

1. User enter the login button
2. Write the username and password section, then click the enter button.
3. System check the information with database and there will be no matched account.
4. Proper error message will appear.
5. Steps goes back.

Use Case#3

Use case name: How To Play Game

Participating actor: Player

Stakeholders and Interests:

- Player wants to get information about game
- System displays the information about the game and a button to go back the main menu.

Pre-Condition:

- The player should click the "How To Play" button which is on the main menu.

Post-Condition: -None

Entry Condition:

- The player interaction with "How To Play" button which is in the main menu.

Exit Condition:

- The player interaction with the "Go To Main Menu" button.

Main Flow of Events:

1. Player should click the "How To Play" button.
2. After the player learn about the game he/she should click "Go To Main Menu" button.

<u>Use Case#4</u>

Use case name: Credit

Participating actor: Player

Stakeholders and Interests:

- Player wants to get information about the developers.
- System displays the information about the developers and a button to go back the main menu.

Pre-Condition:

- The player should click the "Credits" button which is on the main menu.

Post-Condition: -

- Player must be in main menu

Entry Condition:

- The player interaction with "Credits" button which is in the main menu.

Exit Condition:

- The player interaction with the "Go To Main Menu" button.

Main Flow of Events:

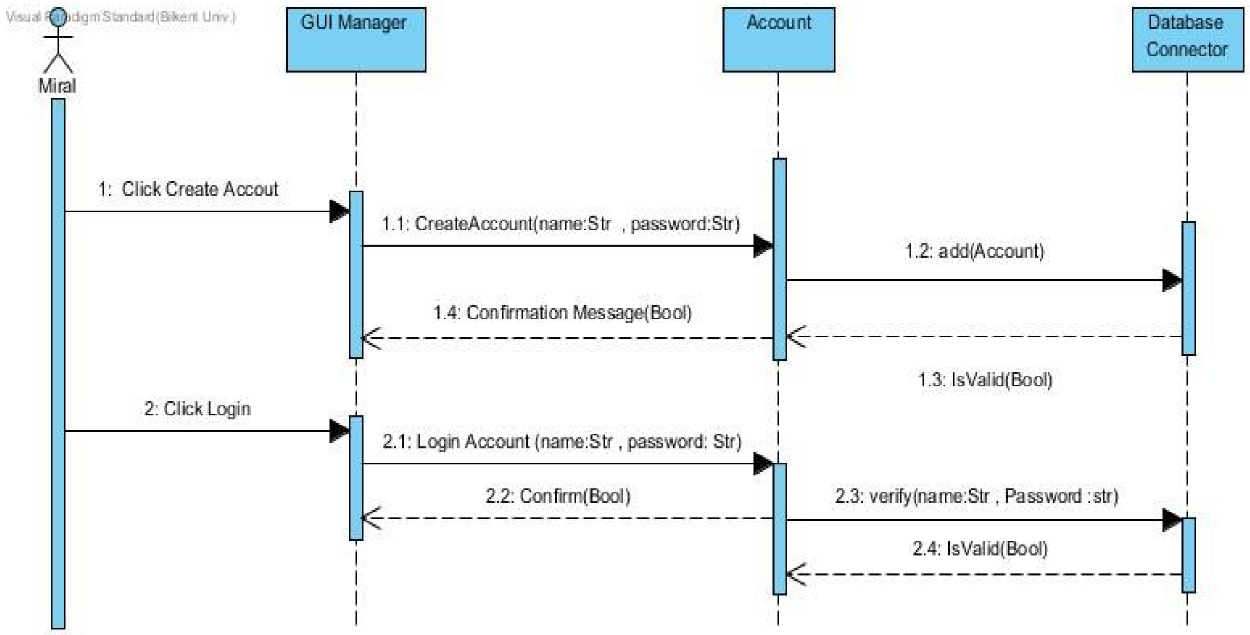
1. Player should click the "Credits" button.

2. After the player learn about the game he/she should click "Go To Main Menu" button.

4.2) Dynamic Model

4.2.1) Sequence Diagrams

4.2.1.1) Create Account / LogIn



Visual Paradigm Standard(Bilkent Univ.)
Miral
GUI Manager
Account
Database Connector
1: Click Create Accout
1.1: CreateAccount(name:Str , password:Str)
1.2: add(Account)
1.4: Confirmation Message(Bool)
1.3: IsValid(Bool)
2: Click Login
2.1: Login Account (name:Str , password: Str)
2.2: Confirm(Bool)
2.3: verify(name:Str , Password :str)
2.4: IsValid(Bool)

Primary Actor : Miral

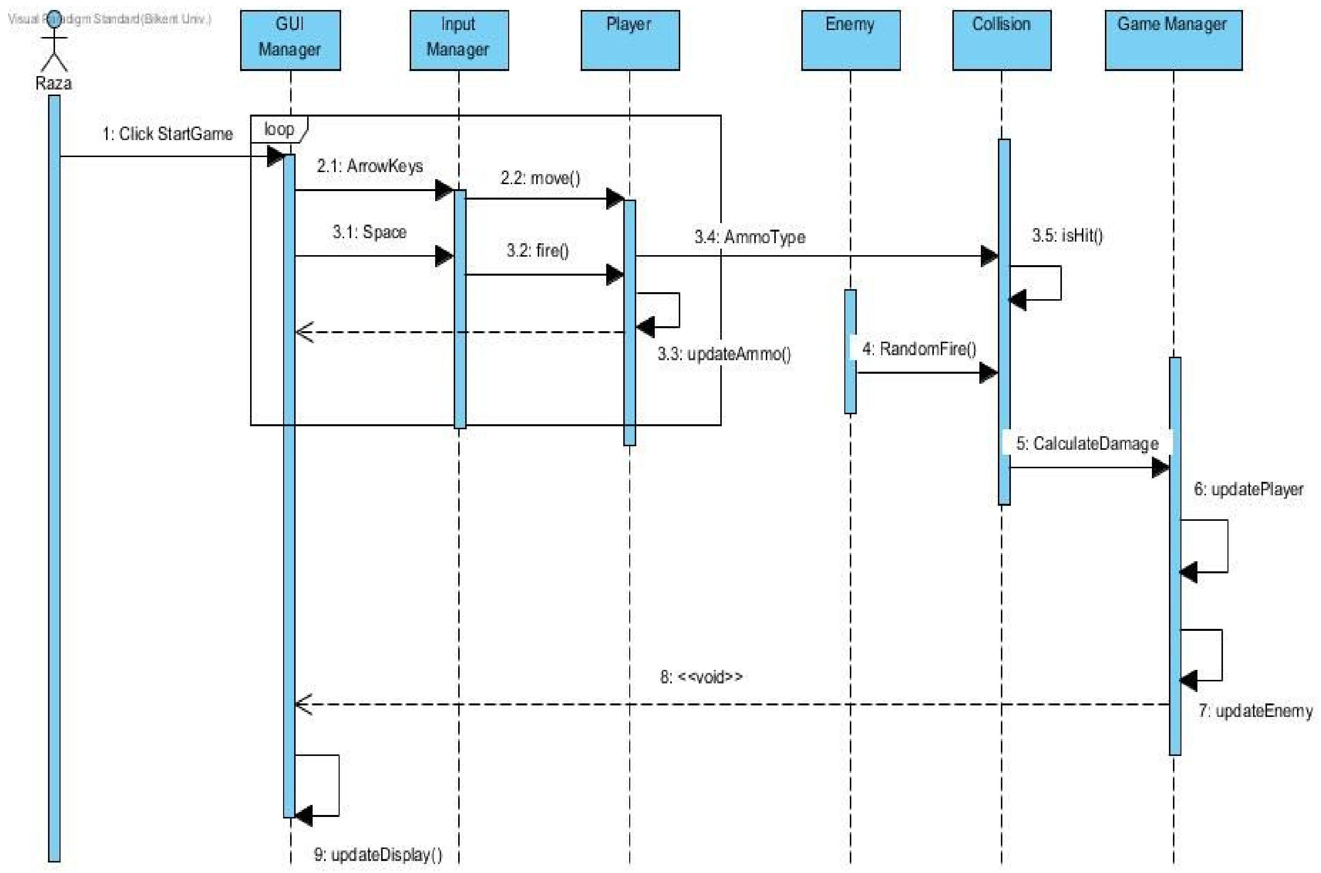
Scenario :

Miral is a new user and she creates a new account by inputting her name and password. If the name already exists she is informed and is required to enter the name again.After confirmation of her account creation, she logs in to her account by putting her appropriate information.

Description:

When Miral Clicks the Create Account Link she is directed to another Page where she is required to input her information in order to create a new account. When she has entered her information, her data is made into an Account object by the Account Class , via the CreateAccount(name:str, Pass:Str) method. This new account object is checked if there already exists a similar matching account based solely on the other objects username. If there already exists another similar account the IsValid() method from the DataBaseConnector class returns false and prompts Miral to try another username. Upon inclusion of an acceptable data, the account is added to the Database. The same checking procedure is applied when Miral wants to login into the game. This includes her information checked in the DataBaseConnector class , thought the 'verify(name:Str , Password :str)' method and a prompt is displayed according the input entered.

4.2.1.2) Gameplay



Primary Actor : Raza

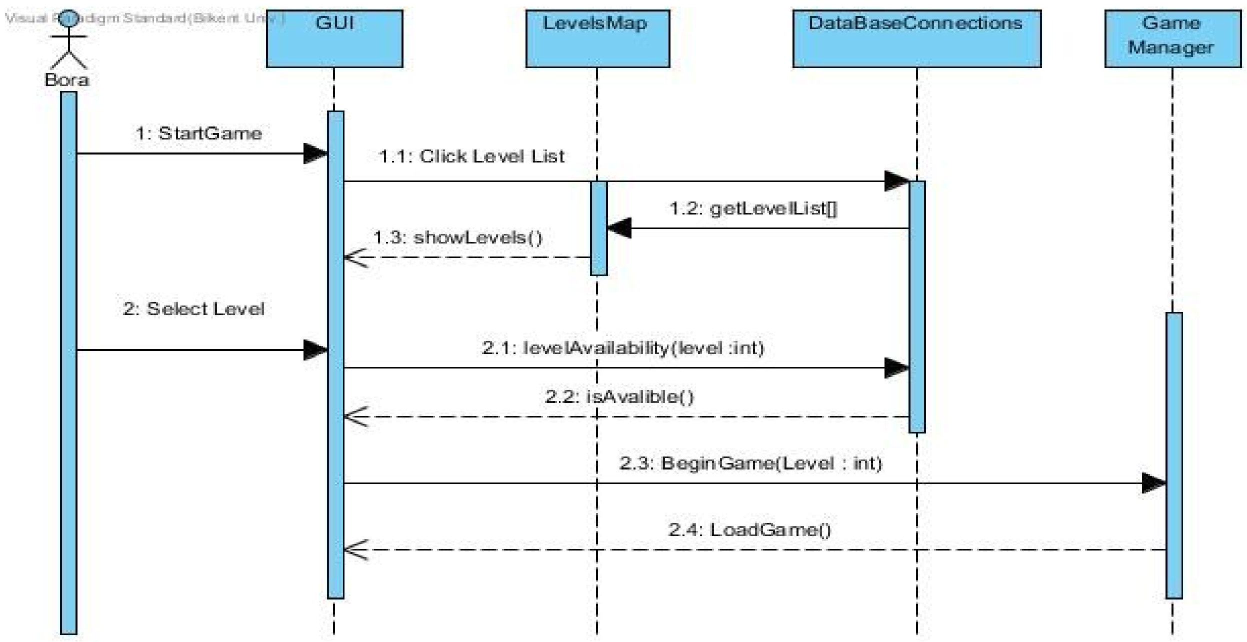
Scenario:

After selecting the appropriate level Raza clicks on the 'StartGame' Button , which immediately starts the game. Raza starts the game with full health and full armor. Raza can move his character by using the arrow keys on his keyboard and he fire his bullets by tapping the space button. When these bullets hit the enemy they are destroyed.Similarly Raza has to dodge the line of fire from the enemy.If Raza is shot, first his armor will decrease after that which his will health.

Description:

After the game has been loaded. Raza is able move the player using the arrow keys.By pressing the arrow keys the Input Manger class invokes the move() method, which updates the player's position coordinates , these player coordinates are then send to the GUI manager to represent the updated player position during the game.Similarly when Raza presses the space key, the fire() method is invoked which fires an ammo and in order to keep track of the ammunition remaining, the updateAmmo() is called every time is the fire() method is invoked.As each character have a different ammunition type, thus its Ammotype is categorized under AmmoType .In addition on random time intervals the Enemy Class will invoke a randomFire() method , this will allow the enemy to fire at Raza during random time intervals .In Both cases where the either Raza fired or Enemy fired, the collision class will calculate whether contact has indeed taken place between the enemy and player , and the bullet. If there is indeed contact then the isHit() method returns true which in turn invokes CalculateDamage() method in the Game Manager class in order to update the player and enemy. Finally the Game Manager constantly gives updates to the GUI Manager ,which excercies the updateDisplay() method to update the GUI.

4.2.1.3) Load Level



Primary Actor : Bora

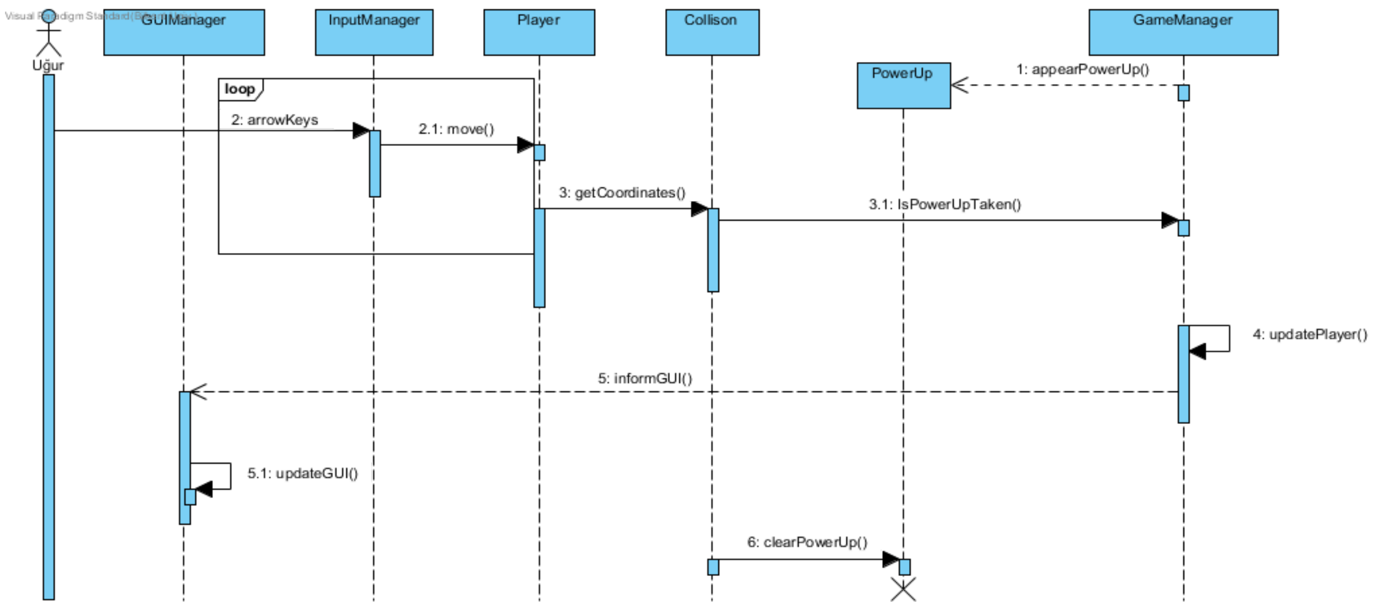
Scenario :

After Bora has selected his character. He clicks the ‘Level List’ button from which he is directed to another screen where the game level shows up.Considering its his first time playing the game only the first level is playable. Had he played these levels before, those played levels would be unlockable.When Bora click a locked level nothing happens but after selecting the appropriate level Bora is able to clicks on the BeginGame button which loads the game and allows Bora to play.

Description:

When Bora clicks on the ‘Level List’ button, immediately the DataBase Connection class sends the levels already played by the user to the LevelMap class. These already played levels are placed inside a LevelList integer array. The GUI manager calls this array by using the showLevels() method. Those integers already present in the list are highlighted by the GUI, thus showing their accessibility to Bora. When Bora selects a non-highlighted level it is send as a parameter to the DataBaseConnection via the ‘levelAvailability(level :int)’ method to check whether the integer value passed exists in the playedLevel array in the database Connection. As it isn’t isAvalible returns false and the game does not advance until Bora presses the correct level. When Bora click the correct level isAvalible returns true and the ‘BeginGame’ Button becomes selectable. When clicked the BeginGame(int) method is invoked by the Game Manager and the LoadGame() method is send to the GUI to begin the level.

4.2.1.4) Power-up



Primary Actor: Uğur

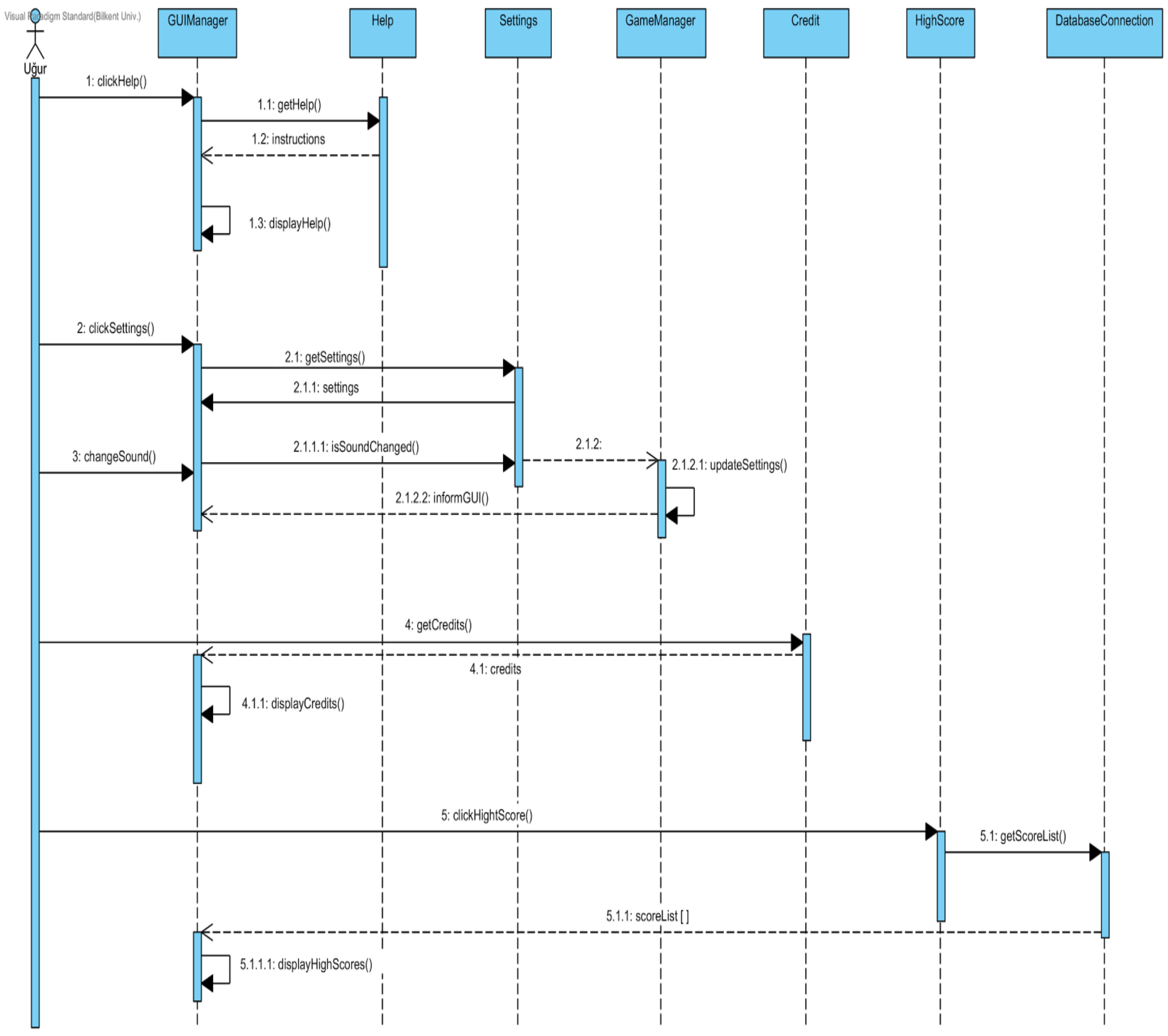
Scenario:

While playing the game, Uğur encounters some sudden power-ups on any place of game screen. He tries to take them by moving on the screen and go over the power-ups. Based on what he take, his health may increase, his damage can increase and etc.

Description:

When power-ups are appeared by the game manager, it creates a power- up object which type is determined randomly inside the power-up class. GUIManager keeps power up on the screen for a while. By using arrow keys, player has to go over the power ups in order to take it. In game screen player can move all four ways (up, down, right, down). Within this time collision class check whether power ups and player's characters coordinates intersects or not. Then return the result to the GameManager. Based on result and power-up type, GameManager updates the player with necessary calculations and to reflect the changes on screen it sends modifications to GUIManager. At the end created PowerUp object is deleted. The detection requires separating the object when there is an intersection with a static collision object. The collision involves calculating the exact pixel that is hit and is used for small objects. Based on type of power-up, either it updates the attribute of weapons or updates the attribute of the players such as health.

4.2.1.5) Game Menu



Primary Actor: Uğur

Scenario:

When Uğur enters the game, he will see four different option that are help, settings,credits and highscores.With these options, he can learn how to play the game. If he’s bored of sound, he can close it. He can see the who behind the this awesome game and see how he is awesome by looking at his scores.

Description:

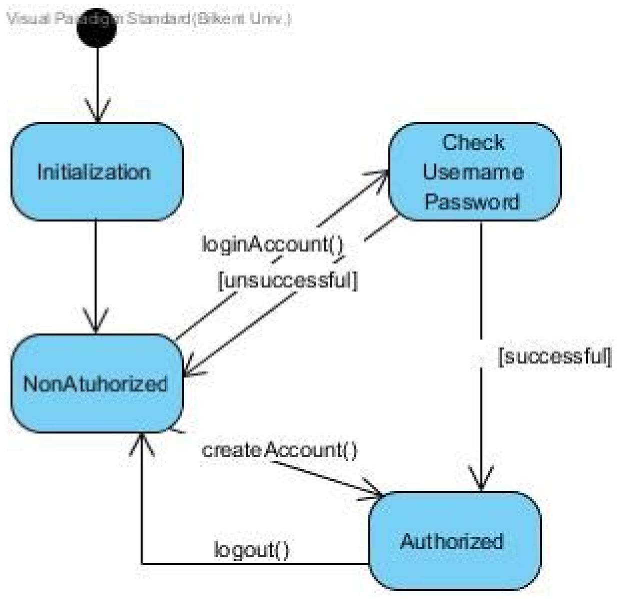
Basically all menu components are initiated with the button on the menu. For help part, GUIManager call the getHelp() function to get the instructions. Help object return the instructions and GUIManager displays them on the help screen. Display credits works in the same way with credits text instead of instruction.

In settings part, GUIManager calls the options which is setSound(). Based on user input that is whether sound is opened or closed, last situation goes to GameManager. It updates the system based on conditions and informs the GUI about changes.

High score works connected to databased different from others. When player click the high scores, high score object get the list of scores from the database and return it to GUIManager. On the high score screen, date and scores are displayed.

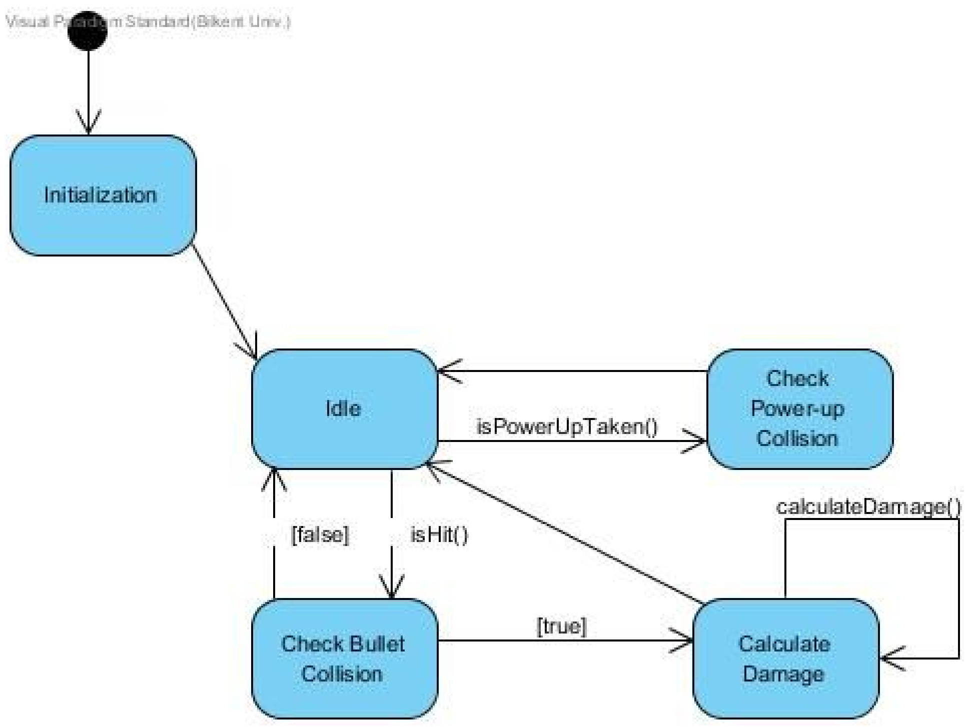
4.1.2) State Diagrams

4.1.2.1) Account's State Diagram



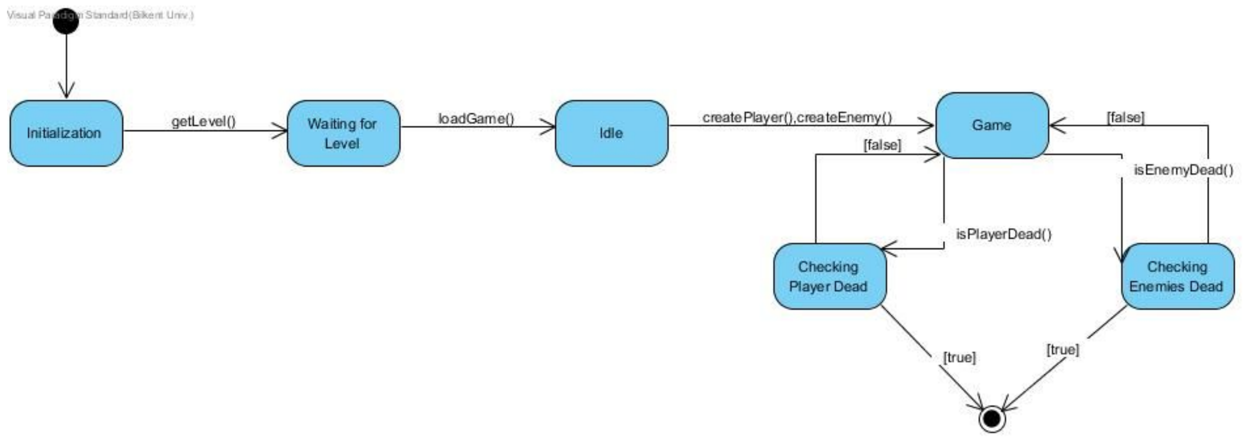
In the beginning user starts in non-authorized state. User can get authorized by logging in with correct username and password or creating new account. User can log out whenever he wants so that he became in non-authorized state.

4.1.2.2) Collision's State Diagram



The main state of collision object is Idle. After object is created state is remain unchanged in Idle state. When isPowerUpTaken() method calls the state changed to Check Powerup Collision state. And then it goes back to the Idle state. If isHit() method is called, the state is changed to Check Bullet Collision state. If isHit() returns true it goes to Calculate Damage state and damage is calculated. And then it goes back to Idle state. However, if isHit() returns false, it goes back to Idle state.

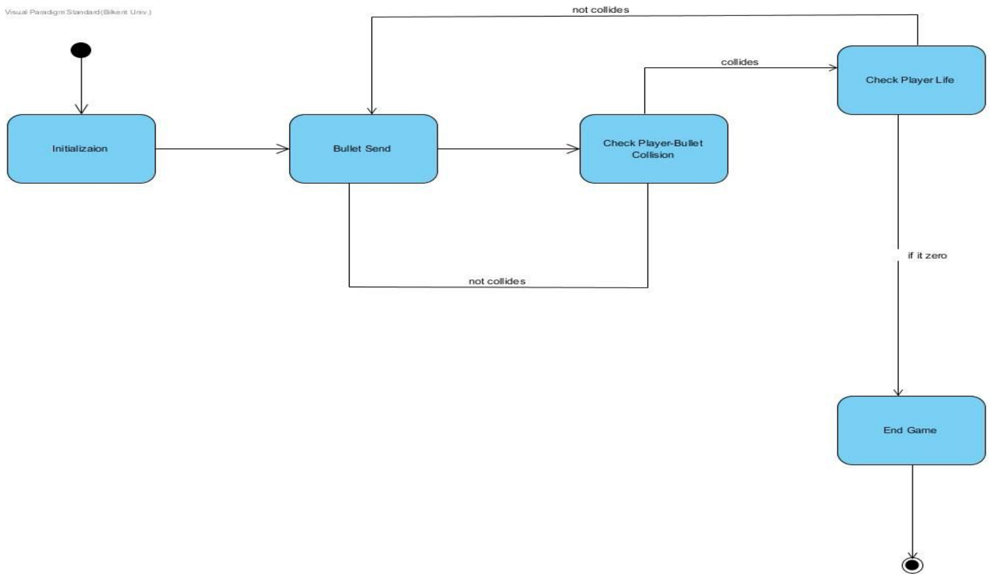
4.1.2.3) Game Manager State Diagram



In the beginning, gameManager gets level in order to start the game. Then it loads that level. After that it creates game objects. While game is being played, it checks whether the player is dead or not. If the player is dead the object is destroyed and game ends. However, if the player is not dead, it goes back to game state. The same sequence is valid for enemies. If all enemies are dead, the game ends. However, they are not dead, it goes back to game state.

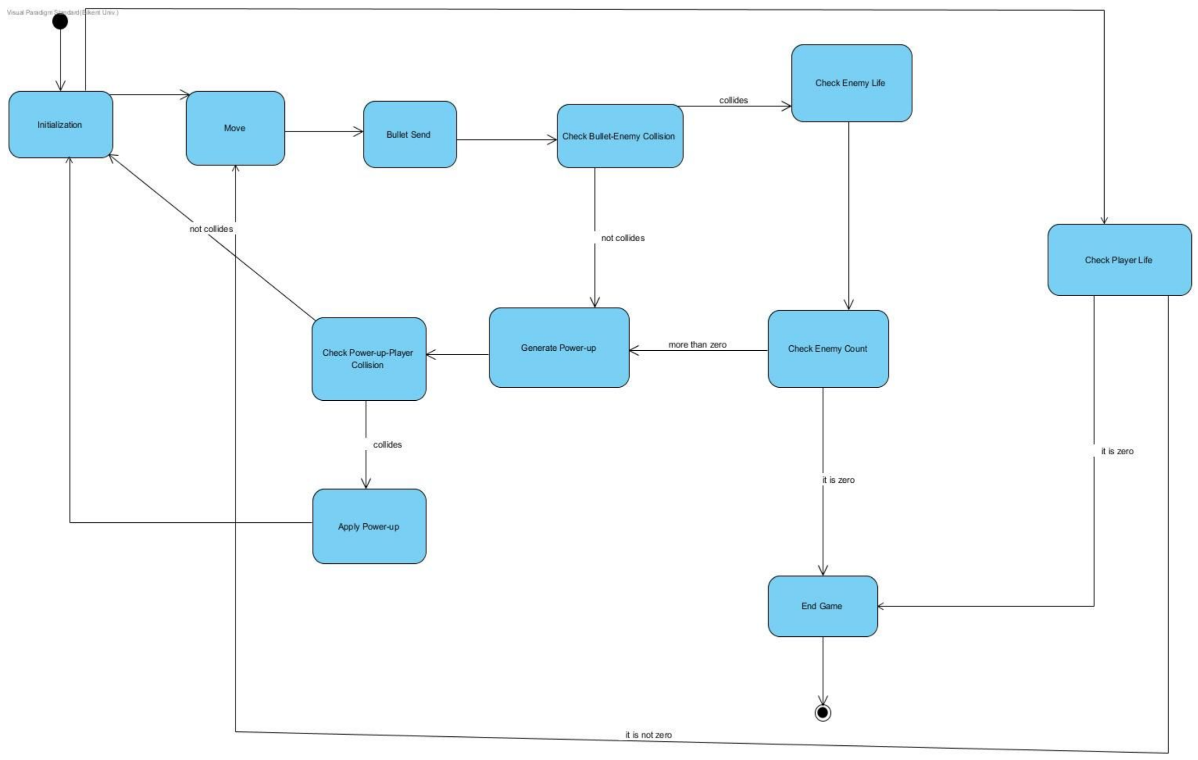
4.1.3) Activity Diagrams

4.1.3.1) Enemy's Life Activity Diagram



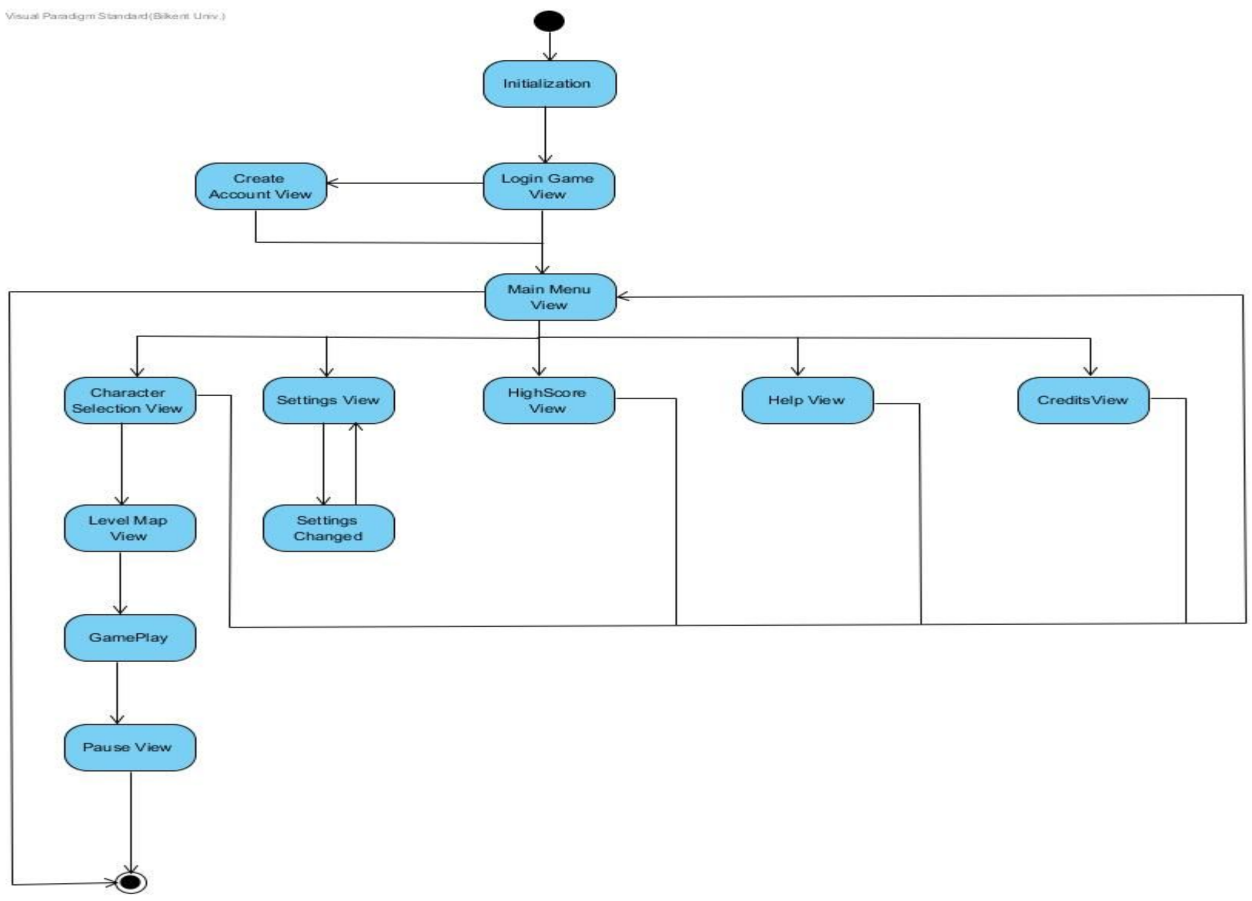
Enemy's Activity Diagram shows one enemies life cycle in the game. It sends bullets to the player. Then checks if it hit the player or not. If the bullet collides with the player, it checks player's life. If player's life is zero, the game ends. However, if player's life is more than zero, enemy goes back to sending bullets.

4.1.3.2) Player's Life Activity Diagram



Player's Life Activity Diagram shows the lifecycle of player in the game. After starting the game, player can move and send bullets. After sending bullets, the system checks the bullet-enemy collision. If there is collision, it checks all enemies are dead or not. If all enemies are dead the game ends. If they are not dead, system can generate power-ups. After generating power-ups, system needs to check collision between character and power-up. If they collides each other, the power-up will be applied. Then it goes back to moving. If collision did not come true, it continues its life. While these are happening, player's life is checked. If his life points are decreased to zero, the game ends. If they are more than zero player continues its life cycle.

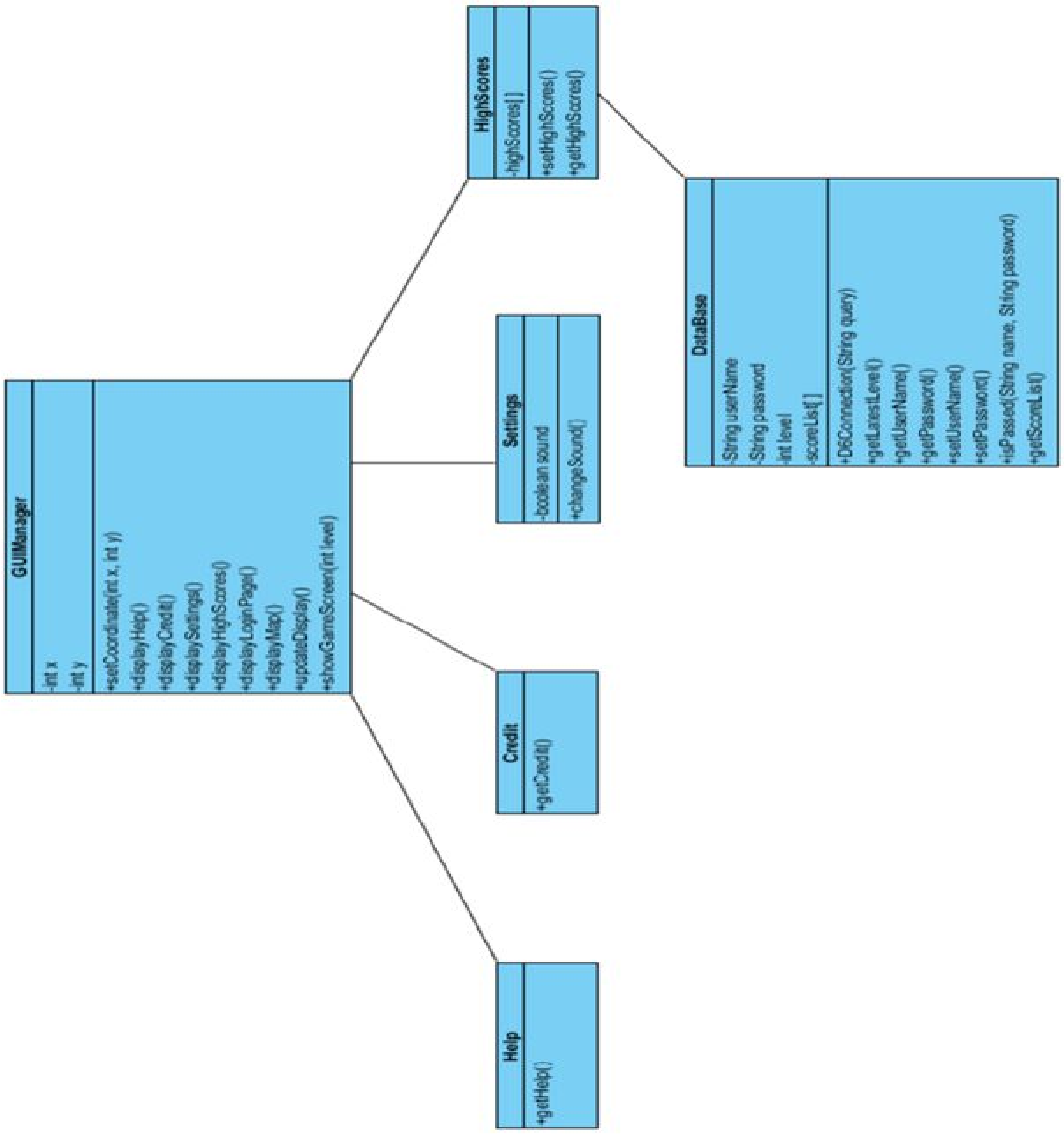
4.1.3.3) Overall Game Activity Diagram



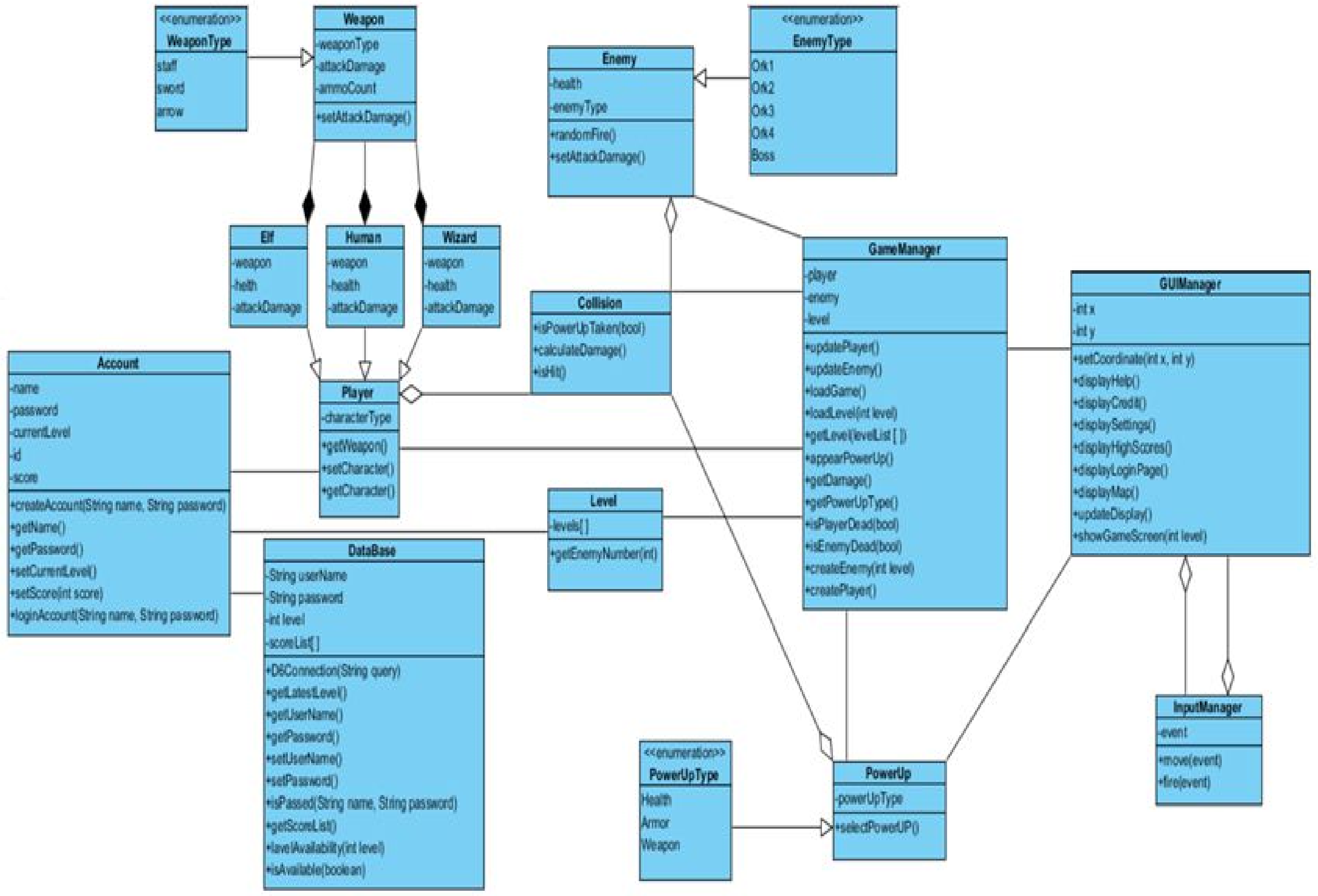
The Overall Game Activity Diagram shows the software's overall views. The first view player is met is Login View. After logging in with correct username and password or after creating new account user face's with the Main Menu View. User can see how to play the game, high scores, credits, can change settings or play game. Even if user chooses to play game, the Character Selection View appears on the screen. After character selection, in the Level Map View player chooses the level he wants to play. After level selection, user interacts with Gameplay. While he is playing the game, he can pause the game and Pause Game View will appear. He can exit the game in Pause Game View. When player closes the game or chooses exit selection, the life cycle will ended.

4.3) Class and Object Model

4.3.1)Main and Player Creation Class Diagram

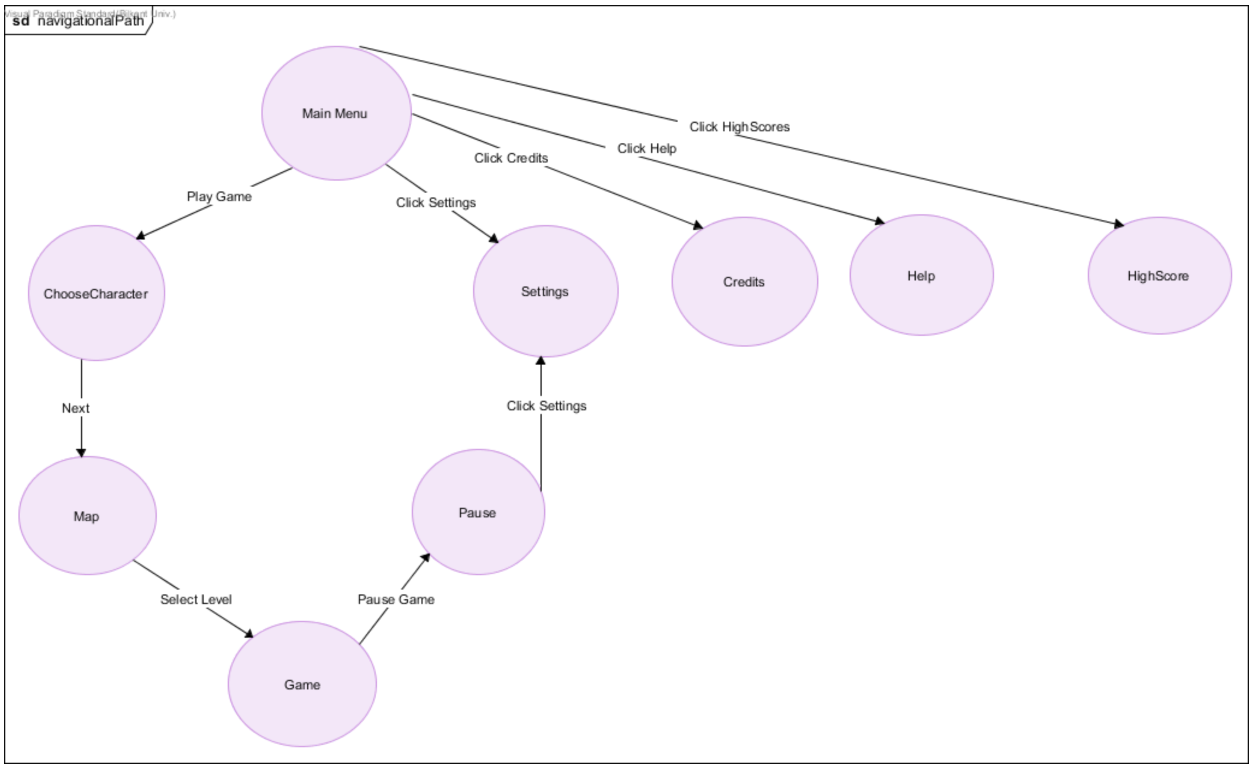


4.3.2) Game Class Diagram



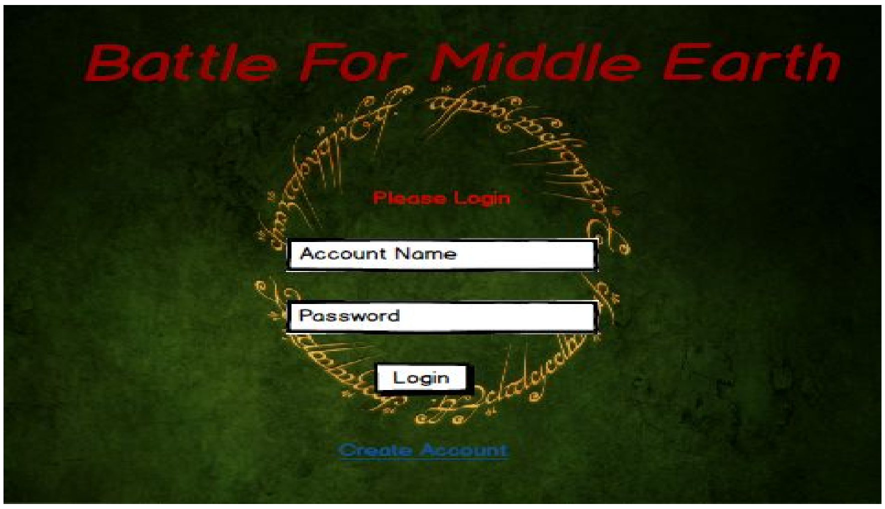
5) User Interface

5.1) Navigational Path



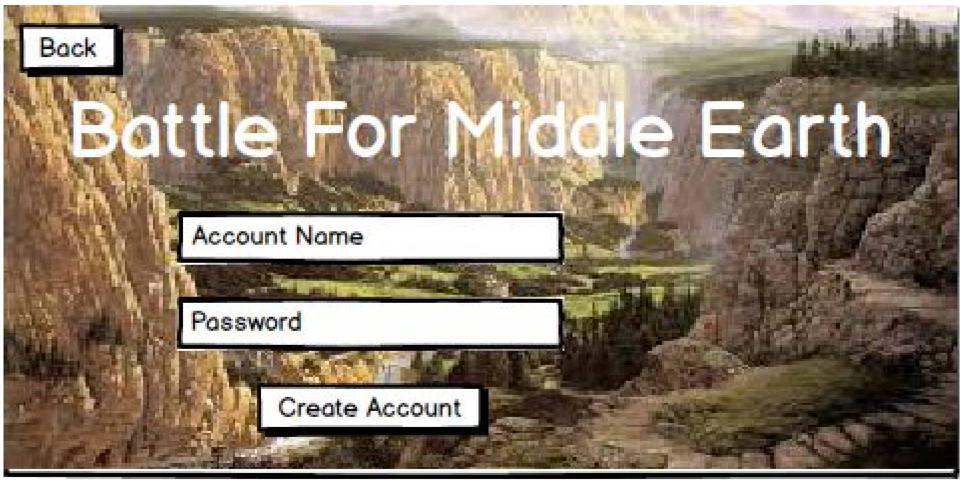
5.2) Screen Mock-ups

5.2.1) Login Account



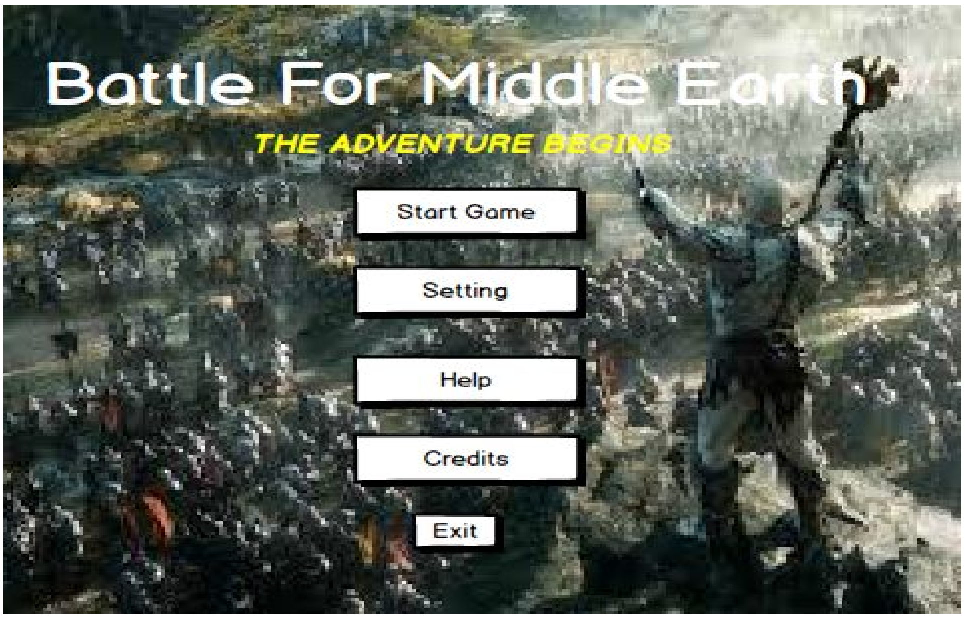
In the following screen the player would be required to enter his account name and password in their respective fields. Upon insertion they need to click the login button in order to login into their game's account. If it is a new player then he/she is required to clicks the 'Create Account' link to be directed to another screen

5.2.2) Create Account



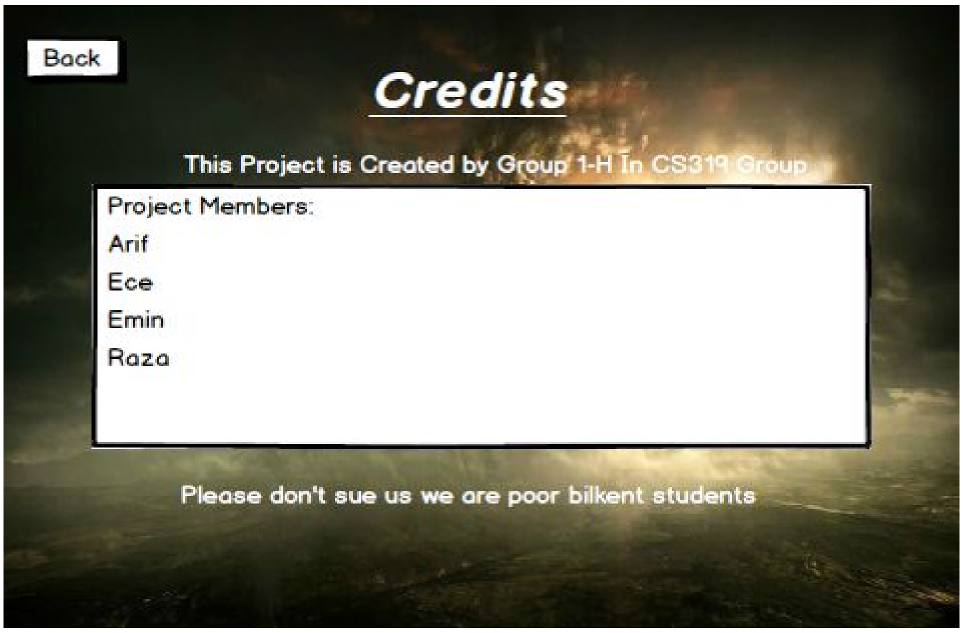
In this the user is required to enter his username and password in the respective field. Upon clicking the 'Create Account' button the user account will be created.

5.2.2) Menu



In this the user is presented with different options. ‘Start Game ’ will start the game. ‘Setting’ will allow the user to change the game sound and music. ‘Help’ and ‘Credits’ button will send the user to their respective screen

5.2.3) Credits



Credits

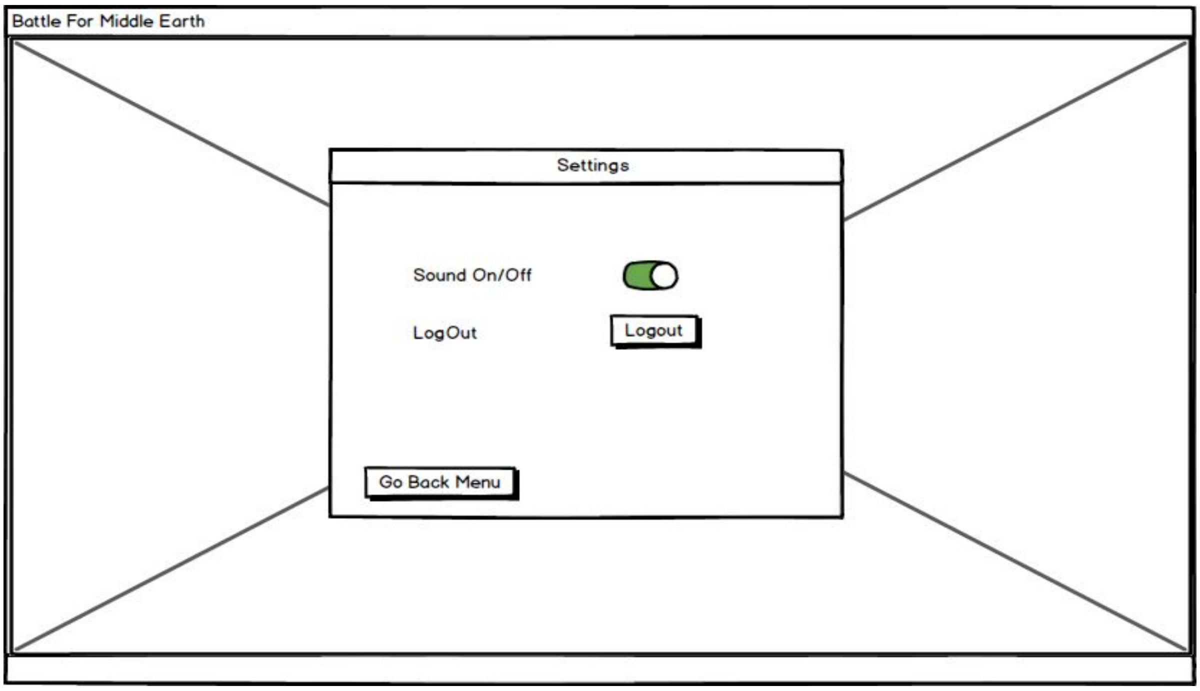
This Project is Created by Group 1-H In CS319 Group

Project Members:
Arif
Ece
Emin
Raza

Please don't sue us we are poor bilkent students

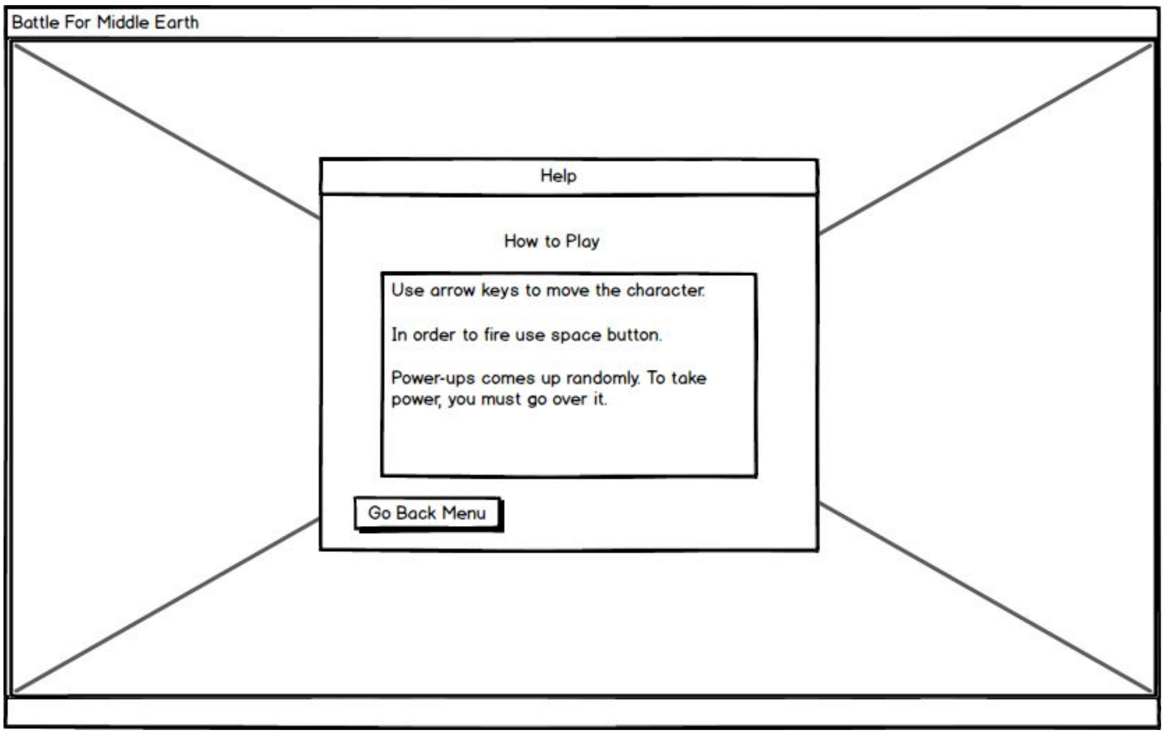
This screen displays the genius and hardworking engineers, who put their time and effort in this game.

5.2.4) Settings



User can change the sound options. By sliding the button, player will be able to set the sound. Also, if player wants to login with another account, logout option is provided here.

5.2.5) Help



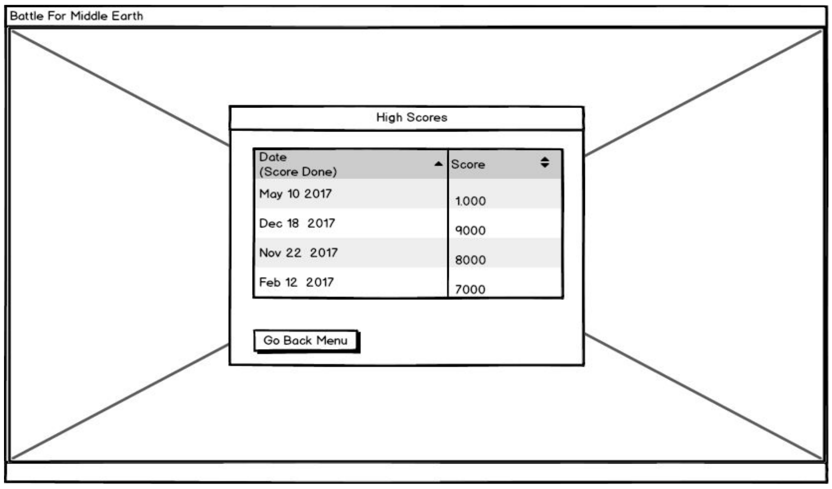
Use arrow keys to move the character.

In order to fire use space button.

Power-ups comes up randomly. To take power, you must go over it.

Player will be able to display how to play via help screen in a documentation style.

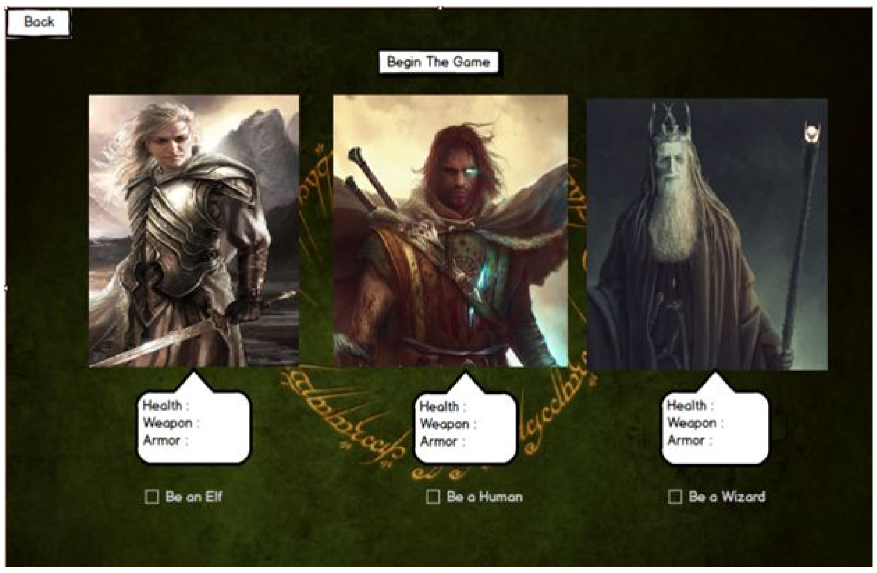
5.2.6) High Score



Date (Score Done)	Score
May 10 2017	1.000
Dec 18 2017	9000
Nov 22 2017	8000
Feb 12 2017	7000

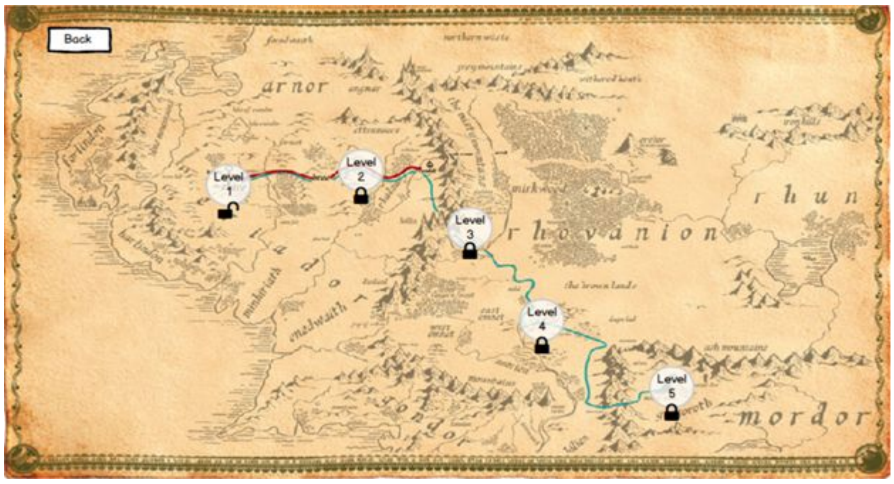
In this screen, the players' top scores in the game are listed.

5.2.7) Choose Character



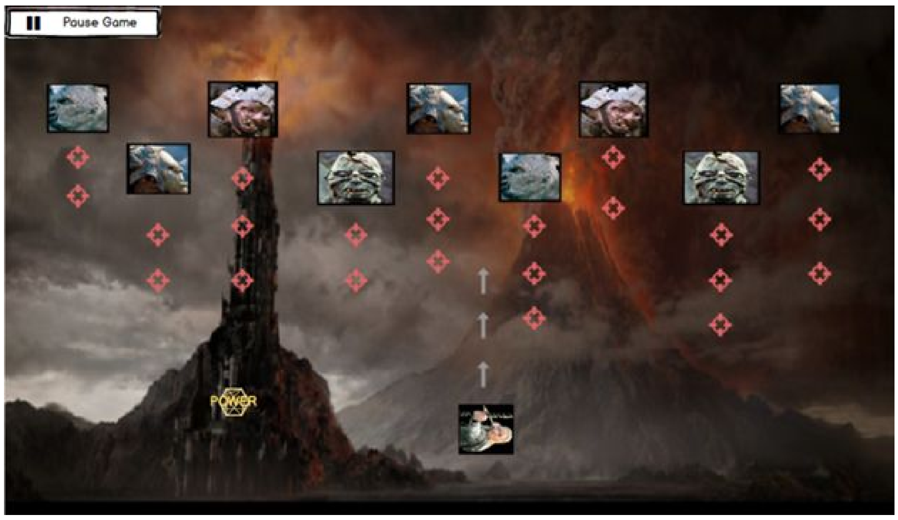
Three different characters are offered for players. These characters are from the middle earth theme. Players can choose whether they want to play as a wizard, elf or a human. Each of these characters will posses unique and special features which are shown below each of these characters. To choose a character user should click check box which is below of the specific character. To go to the into map page(meaning the next screen) he/she should click 'begin the game'. To go back to the main menu, player clicks the back button.

5.2.8) Map



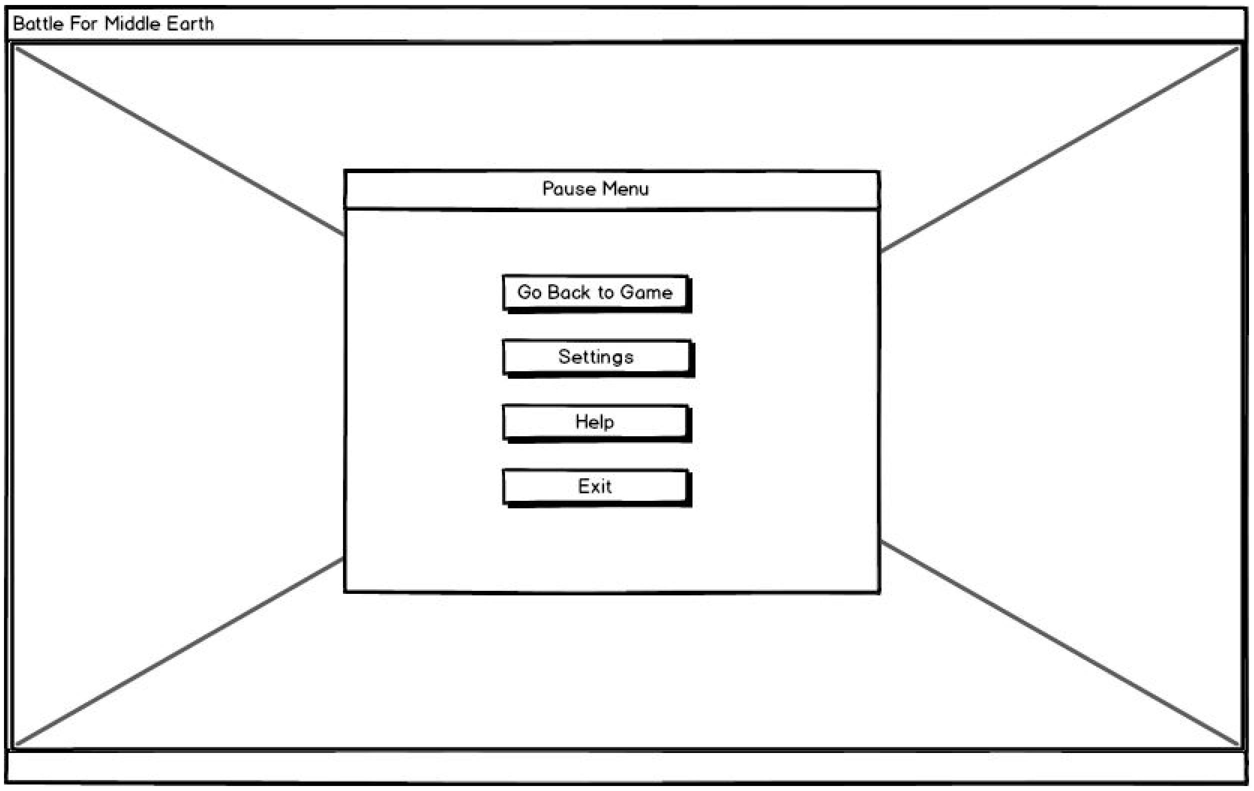
As seen above each level is presented by a checkpoint. The player begin from the first level and if he/ she can be abe to pass this level the second level's will be unlocked and playable to the user. It is go on like this as the user passes each level. To go back player clicks the back button, which sends the player to the character selection menu.

5.2.9) Game Screen



The character is positioned at the bottom of the screen and would be allowed to move in all direction, whicle at the same time allowing the user to shoot at the enemies. The enemies will appear at the top couple of rows of the screen ,who they begin to fire towards the player in a random manner. To pause the game player clicks on the 'Backspace' button.

5.2.10) Pause Game



In this screen, while playing the game, player can pause the game and reach the settings, help and exit options.

6) Conclusion

In the analysis report, we have investigated requirements, system models and some mockups for the game "Battle For Middle Earth". We have 5 parts to specify these parts. First part is the introduction part. In this part we briefly explained what is the game is about. In the overview part we explained the essential knowledge prior to the game which includes detailed description and explained of the mechanism of the game. The requirement specification part has three subsections which are functional requirements, non-functional requirements and the pseudo functional requirements (Constraints).
We specified our requirements in these parts clearly. The system modeling section was the most important and demanding part for us. Our project's use case models, dynamic models and class-object models are also included in this section , which is an integral part because the infrastructure of the project is composed of these parts. The user interface section include

some mockups that illustrate our game screens. We made our mockups easy to understand and also we wrote some descriptions in order clarify confusions ,if any. To conclude analysis report is so useful that our job will be much easier while we design and implement our project.