



**CS319 Object-Oriented Software Engineering
Section 2
Group: 2.E**

Analysis Final Report

Maze Runner

Group members: Ali Sabbagh, Mehmet Furkan Dogan,
Umitcan Hasbioglu, Xheni Caka

Course Instructor: Ugur Dogrusoz

Date: 11 Mar 2017

Contents

1- Introduction	3
2- Overview	3
2.1 Play Mode	4
2.2 Bombs	4
2.3 Time	5
2.4 Patrols	5
2.5 Maze	5
3- Requirements	6
3.1 Functional Requirements	6
3.2 Non-functional Requirements.....	7
4- System Models	7
4.1 Use case models.....	7
4.2 Object and Class Model	10
4.3 Dynamic Models	11
4.3.1 Activity Diagram.....	11
4.3.2 Sequence Diagrams	14
4.4 User Interface	17
4.4.1 Main Menu	17
4.4.2 Single-Player Mode	17
4.4.3 Multi-Player Mode	18
4.4.4 Settings	18
4.4.5 Tutorial	19
4.4.6 High Scores	20
4.4.7 Credits	20
5- References	21

1. Introduction

We decided to implement an arcade game because arcade games give a better insight about objects and their interactions that can be viewed in the models. Thus, we decided to create a game which is a combination of “Pacman”[1] and “Bomberman”[2]. We decided to implement a stealth game "Maze Runner" that combines these two games and offers many new features. It will give you the present enjoyment of gaming plus the lovely nostalgia to the old days. The game can be played as a single player or multiplayer and the game purpose is to reach the end of a maze as soon as possible.

The goal of our game is to reach the finish point of a sophisticated maze as fast as possible with the least possible attraction of patrols. Both single and multiplayer modes can plant a bomb and try to keep the patrols out of your way or damage your enemy (in multiplayer version). When the player reaches the end, he/she will receive bonus points with respect to remaining time.

Overview section includes more details about the content of the project. The report contains overview, functional requirements, nonfunctional requirements and system models. The following chapter is overview that gives more information about the game.

2. Overview

The game consists of a Pac-man like moving object which goal is to finish the maze as soon as possible without being killed by the patrols which are continuously moving around and trying to stop the player from reaching the exit of the maze. The player has abilities that help him escape from the patrols when he is very close to them. The game has a time limit and also there are time related features of the bombs. The game can be paused and continued where it was left afterwards. A high score system will be accessible allowing the player to check his last five scores. In both single and multiplayer versions, the game will be played by using the keyboard. Player can access the help menu to get instructions on how to play the game.

2.1. Play Mode

- Single player mode: In this mode the player's goal is to reach to the exit as soon as possible. There will be the patrols trying to stop him and the challenge is finishing the maze before the time ends. The player can get rid of the patrols by throwing bombs to them but needs to escape in a short amount of time from the explosion range
- Multiplayer mode: While two players are using the same computer, the challenge is finishing before your opponent. The game is won from the player who made it first to the exit or, if the time limit is up and none of the players makes it to the finish line, then the winner is the one who has collected more points.

2.2. Bombs

- Area:
Bomb in own area: This kind of bomb is dropped in the current position of the player and has a small explosion area and the target is the patrols. It has an explosion time of 2 seconds which is enough for the player to escape from the explosion range.



Bomb in opponent's area: This kind of bomb differs from the other type because it is used to kill your opponent and resets the opponent's position to the beginning position unless the opponent makes it away untouched.

Image 1 - Bomb[3]

- Bomb usage ability: Planting a bomb on his maze costs specific amount of points (100) but planting on the other side costs double (200). Each player has 200 points when "MazeRunner" starts and the score of the player increases as the time runs out such as gets 100 points in every 15 seconds. He can spend these points according to his position in the maze. For instance, in the multi-player mode, if he is ahead of the other player, uses bombs to pass the patrols or if he is not ahead, uses bombs to bring back his enemy to initial point unless the enemy gets away of the explosion rate on time.

2.3. Time

Limit: There will be a time limit to finish the game and it will vary from level to level. In the first level the time limit is 1 minute 30 seconds. In the second level the time limit is 2 minutes 30 seconds and in the third level the time limit is 3 minutes 30 seconds.

Bonus Score: When the game is done the remaining time in seconds will be multiplied by 10 and it will be added to the score.

2.4. Patrols

- Blue: These patrols just move in one direction and they do not follow. They kill only if the player hits them.
- Yellow: These patrols move back and forth and they have the ability to follow the player when he is in a certain range. The player then has to escape or kill them in order to get rid of them.

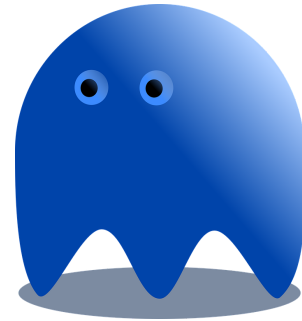


Image 2 - Patrol [5]

2.5. Maze

The player is supposed to solve the maze meaning that from the starting position he will have to move the Pacman to the exit of the maze. When in single-player mode there is going to be one maze which is going to change from level to level while in multiplayer mode the board will be split in two and the mazes are going to be the same but reflected to each-other.

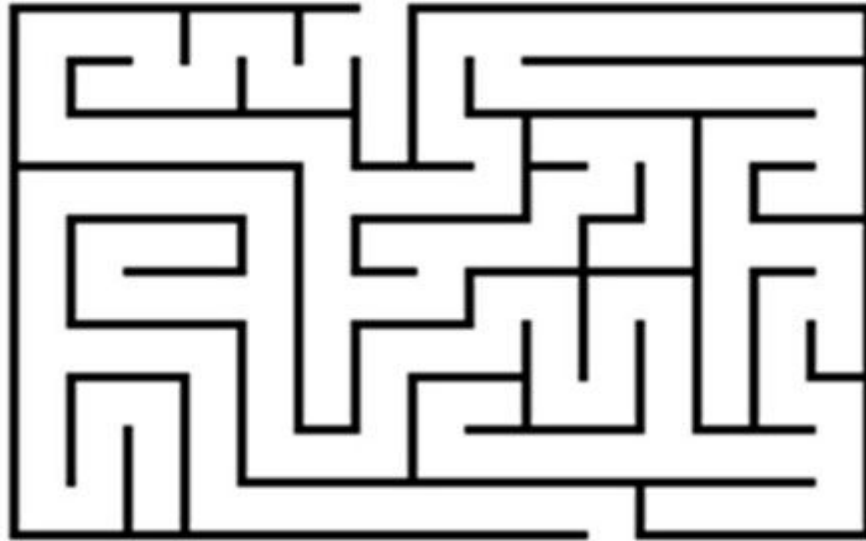


Image 4 - Maze [6]

3. Requirements

3.1. Functional Requirements

- Player(s) should be able to move freely except their direction is blocked by obstacles.
- Players(s) should be able to plant bombs in their area as well as in other player's area.
- Players(s) should die if a bomb explodes near them or any of computer controlled enemies touch them.
- Game should end if any of the players dies or reaches the end point.
- If game ends, player(s) should be able to restart the game.
- Game should be customizable such as number of bots, number of bricks that can be exploded by bombs or the speed of the bots or the resolution.
- Bombs that planted in other player's side should be placed in where the other player is.
- Bombs should explode after 2 seconds + randomly determined time up to 1 second.

- Player(s) movement should not be blocked by bombs.
- By default, player(s) speed should be 0.66 times of enemies.
- Player(s) should be able to plant bomb in their sides after 5 seconds they plant a bomb on their side and plant a bomb after 10 seconds they plant a bomb on other player's side.

3.2. Non-Functional Requirements

- Approximately, games FPS should be reasonable such as 100 FPS.
- The movements will be seen in a real time avoiding lagging in order for the user not to have a bad experience playing the game.
- Bright colours will not be used to avoid harming the user's eyes.
- Playing and understanding the game will be easy.

4. System Models

4.1. Use Case Models

Use Case # 1

Use case name:	PlaySinglePlayer
Contributors:	Player
Entry condition:	Game launched and main menu is entered and “Play Single Player” option is selected.
Exit condition:	Player is out of game and back to main menu.
Flow of events:	<ul style="list-style-type: none"> - Player starts the game. - System initializes the MainMenu with its options. - Player selects the “Play Single Player” option. - System passes selection to Console and a Board with one maze is drawn. - Player reaches finish point. - System displays top 5 scores - Player is able to save the score with name if he/she should be among the top 5. - The system records the score and name and updates the leaderboard - Player returns to the main menu.

Exceptions:

- Player wants to exit game directly.
- Player lost all lives.
- Time is up.

All lead to going back to main menu.

Use Case # 2

Use case name: PlayMultiplayer
Contributors: Player
Entry condition: Game launched and main menu is entered and “Play Multiplayer” option is selected.
Exit condition: Player is out of game and back to main menu.

Flow of events:

- Player starts the game.
- System initializes the MainMenu with its options.
- Player selects the “Play Multiplayer” option.
- System passes selection to Console and a Board with **two** mazes is drawn.
- One of the players reaches the finish point.
- Players are back to main menu.

Exceptions:

- Time is up.
- Both players lost all their lives.
- Players want to exit the game.

All lead to going back to main menu.

Use Case # 3

Use case name: SelectDifficulty
Contributors: Player
Entry condition: Game launched and main menu is entered, “Select Difficulty” option is selected
Exit condition: Player is back in main menu, with difficulty selected.

Flow of events:

- Player selects the “Select Difficulty” option.
- System shows a submenu with three options (Easy-Regular-Hard).
- Player selects one
- System saves new difficulty selected, returns to normal main menu.

Use Case # 4

Use case name:	CheckLeaderboard
Contributors:	Player
Entry condition:	Game launched and main menu is entered, “Check Leaderboard” option is selected
Flow of events:	<ul style="list-style-type: none">- System creates an instance of Leaderboard with top 5 scores loaded in a table.- Player checks scores.- Player presses “Back to main menu” button.- System goes back to main menu.
Exit condition:	Player is back to main menu.
Exceptions:	In case game is newly installed, table will be empty. “Take the chance and be on top of the leaderboard!” message will appear.

Use Case # 5

Use case name:	WatchTutorials
Contributors:	Player
Entry condition:	Game launched and main menu is entered “Tutorials” option is selected
Flow of events:	<ul style="list-style-type: none">- System creates an instance of Tutorials and goes to tutorials page.- First image is displayed with options “Next”, “Previous” and a caption of the image and rule stated under.- Player moves between images.- Player presses “Back to main menu” button.- System goes back to main menu.
Exit condition:	Player is back to main menu.

Use Case # 6

Use case name:	PauseGame
Contributors:	Player(s)
Entry condition:	Player(s) should be already playing.
Flow of events:	<ul style="list-style-type: none">- Player or one of the players in multiplayer mode presses the pause button on keyboard.- System saves locations of objects, time and scores.- System displays a popup with message “Game Paused”, and option “Continue Playing”.- Player presses “Continue Playing”.- System goes back to the game and enables timers and action handling.
Exit condition:	Game will resume.
Exceptions:	Pause is not available after a player reaches the end of the maze

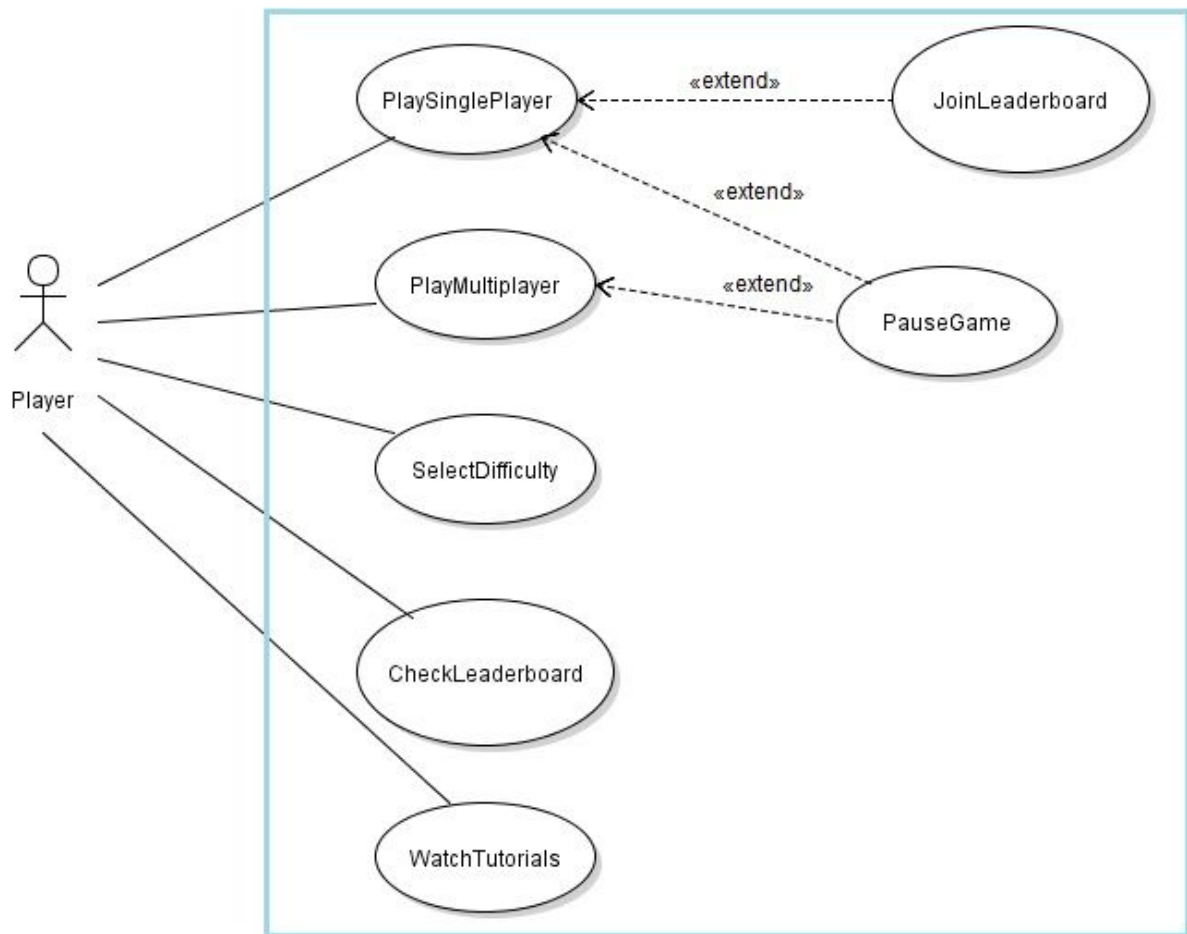


Figure 1 - Use Case Diagram

4.2. Object and Class Model

The MainMenu class will be constructed first. And according to player's input, the selectOption() method of the Console will be called to construct the following:

- Option 1: Play Single Player -----> a **Board** instance is constructed.
- Option 2: Play Multiplayer -----> a **Board** instance is constructed.
- Option 3: Select Difficulty -----> difficulty property is changed accordingly.
- Option 4: Check Leaderboard -----> a **Leaderboard** instance is constructed.
- Option 5: Tutorials -----> a **Tutorials** instance is constructed.

Thus, the Console class will control the program until the game starts, then the Board class is responsible for the graphics (Patrol, Pacman, Bomb). All needed stored information will be taken from FileManager class that keeps images, mazes, scores, etc.

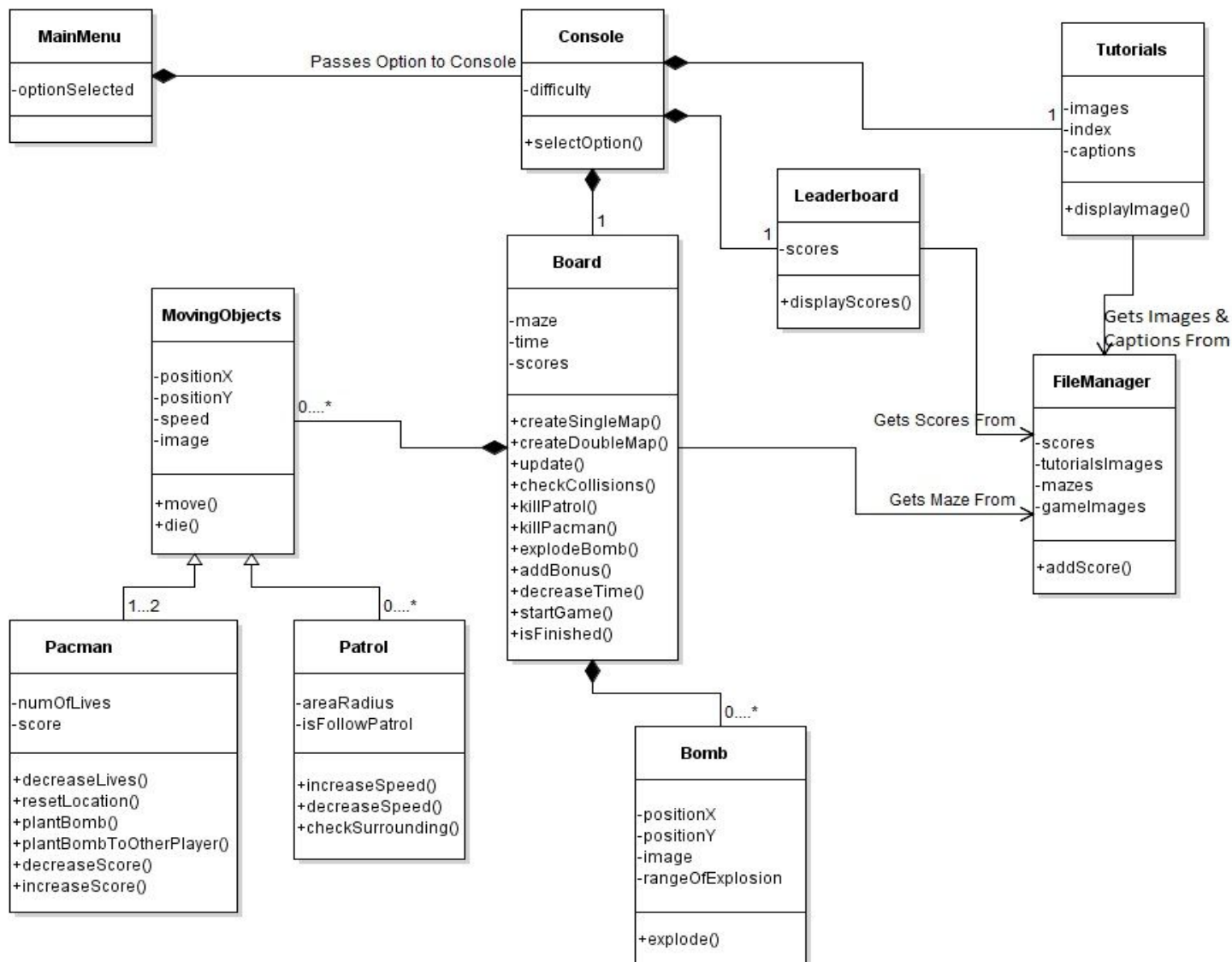


Figure 2 - Class Diagram Of The Game

4.3. Dynamic Models

4.3.1. Activity Diagram

The Activity Diagram shows how the system will handle playing the game. Each time a timer is triggered, the **Board** will move the **MovingObjects** and update scores and time. Collisions between **Pacman(s)** and **Patrols** or **Bombs**, and between **Patrols** and **bombs** will be checked and board updated accordingly. Meanwhile, the time remaining will be checked instantly to end the game when it is up or, when a player reaches the finish point, or when the player is out of lives. In case a player has a high score among the top 5, the player will be able to add it to **Leaderboard** otherwise, the game is ended directly.

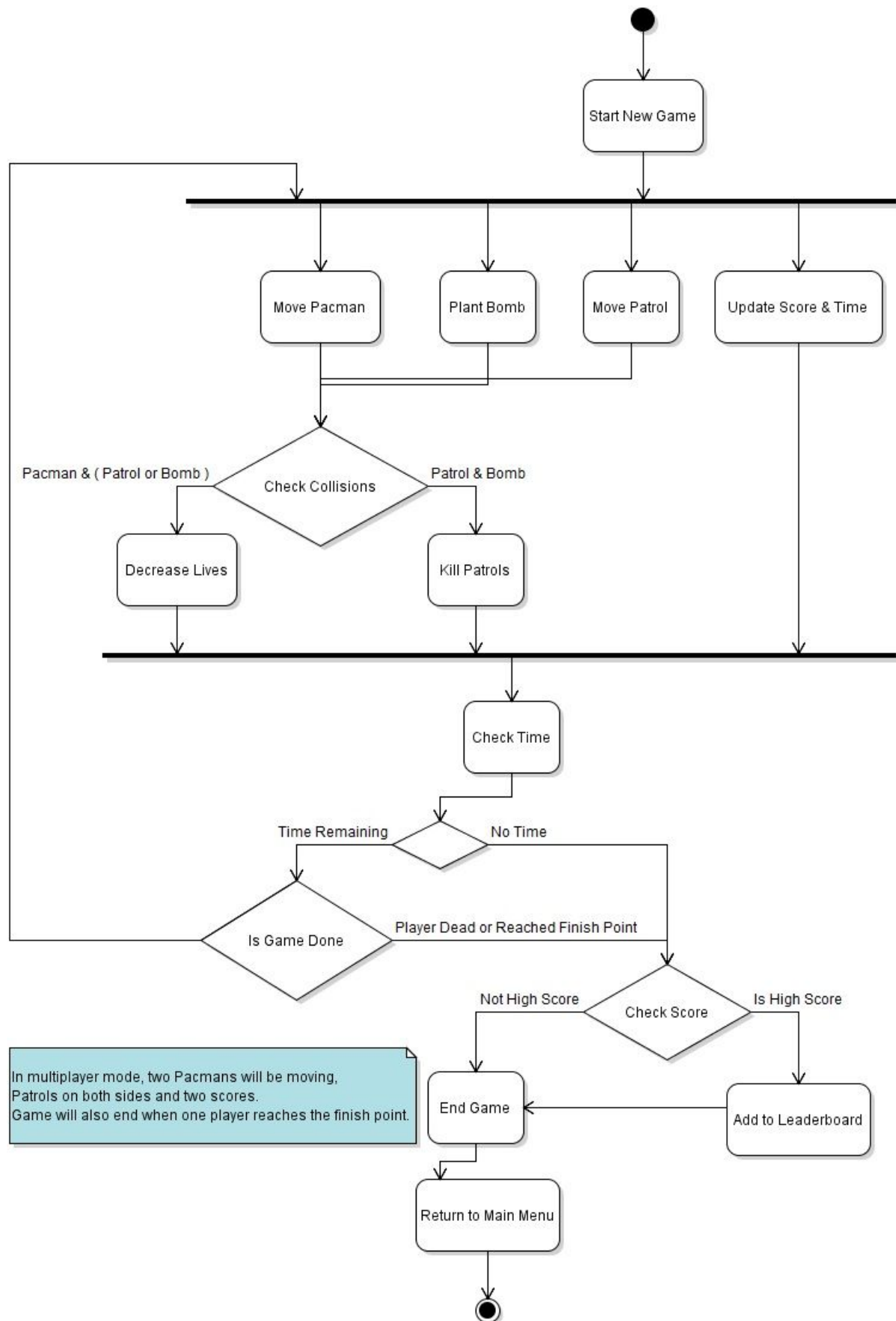


Figure 3 - Activity Diagram of Playing the Game

4.3.2. Sequence Diagrams

Scenario name: Create Game

Scenario description: Eysan wants to play the game. She executes it and the main menu shows up. She is on her own so she selects the first option which is single-player. Then the game starts.

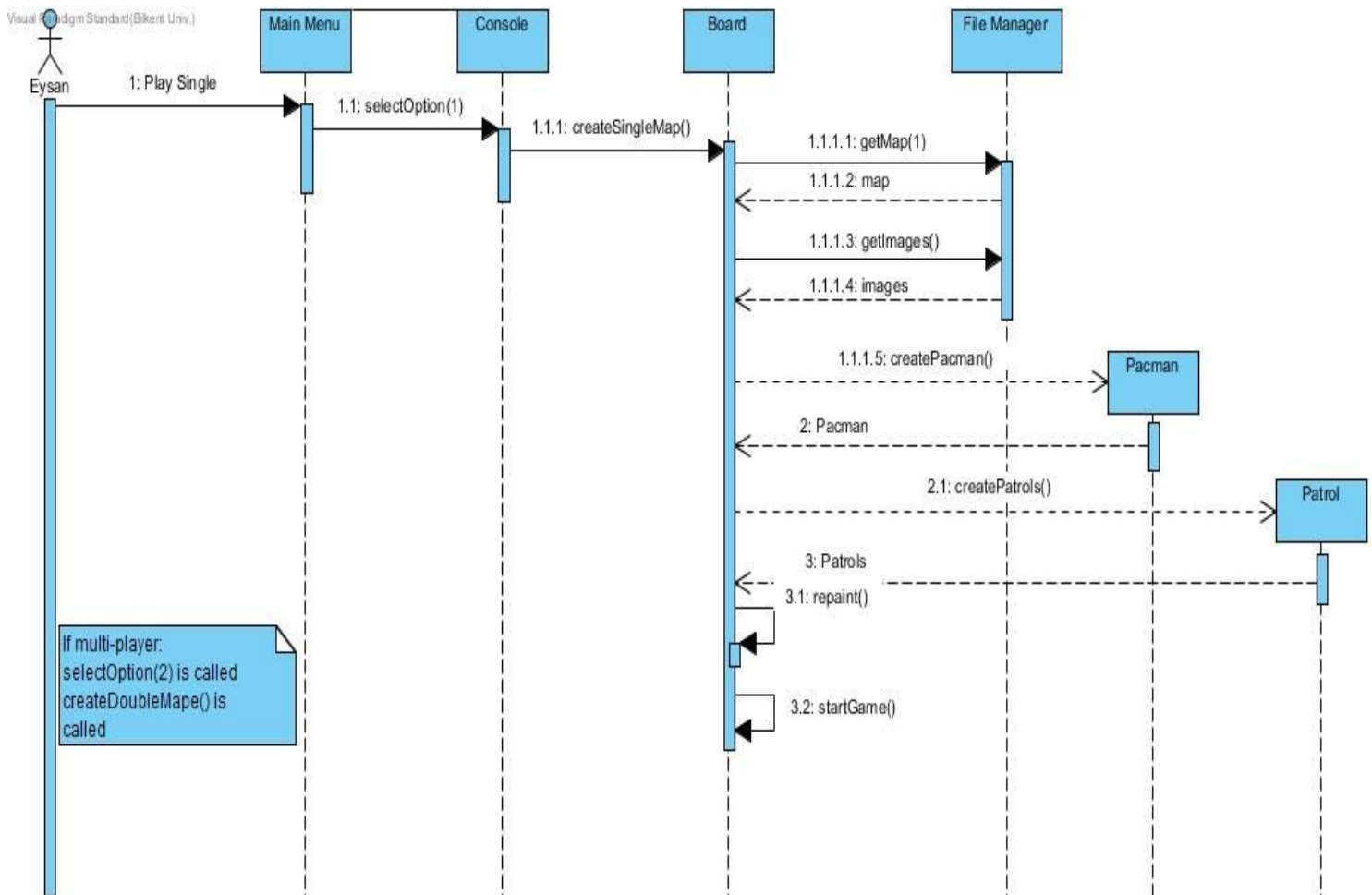


Figure 4 - Sequence Diagram of Create Game

Description: After the player chooses the first option which is single-player the game is created. The Board asks for the map, the images, the Pacman and the Patrols to be loaded. The map and the images are loaded from the File Manager, the Pacman is an instance of the Pacman Class and the Patrols are instances of the Patrol Class.

Scenario Name: Play Game

Scenario Description: Eysan starts to play the game and presses direction button which is down. Then, she wants to plant a bomb on her side to kill the patrols which are on her way.

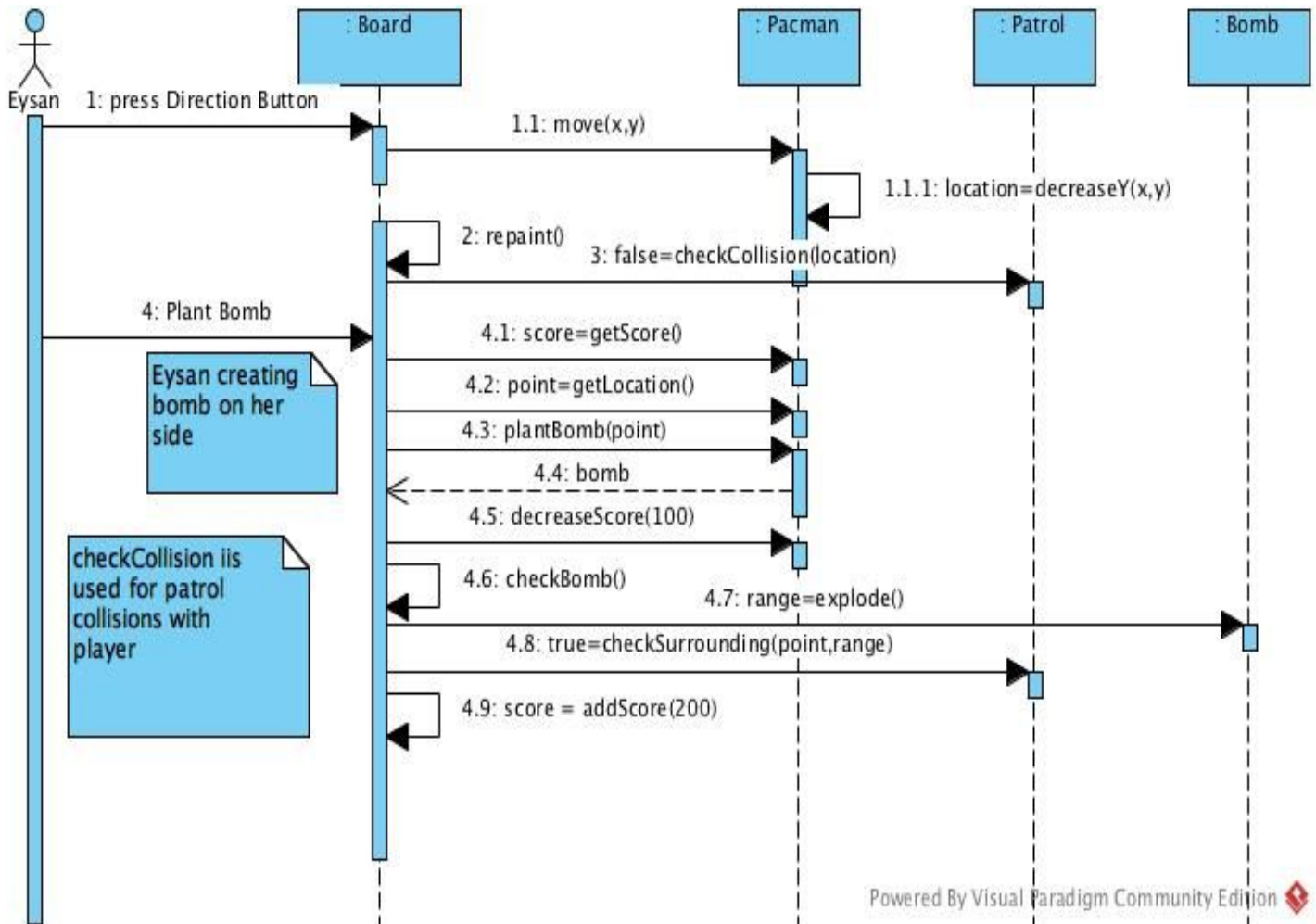


Figure 5 - Sequence Diagram of Playing Game

Description: When player wants to move down, she press the down button which invokes to move method from the board. Then the location of pacman is repainted. Than player chooses to plant a bomb on her side. Before the bomb is planted, the Pacman score is checked by Board. If she has enough points to plant a bomb, location of the Pacman is returned to Board. After that, Bomb class is invoked to plant bomb. The Board counts the time for a while before the bomb explodes. Than, the bomb has exploded. The patrol is in bomb range and it is dead after explosion. Thus, Board gives bonus points to player for the patrol kill.

Scenario Name: Check Tutorial

Scenario Description: Ali doesn't know how to play the game so before starting to play he thinks that it's better if he checks the tutorial and he goes for it.

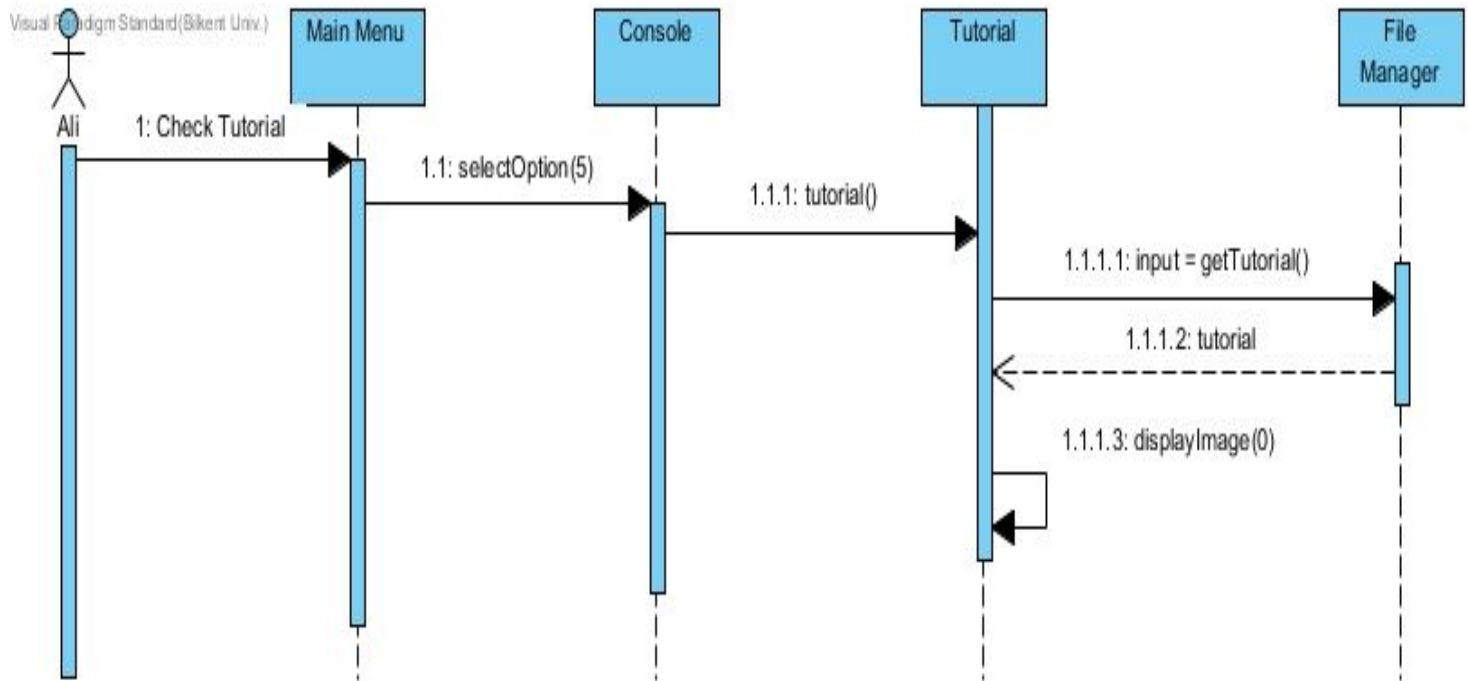


Figure 6 - Sequence Diagram for Check Tutorials

Description: In the Main Menu the player chooses option 5 which corresponds to Check Tutorial. After that the Console asks from the Tutorial Class to show the tutorial. The class then interacts with the File Manager which sends the tutorial image to the Tutorial class and the image showing the instructions shows.

Scenario Name: Check Leaderboard

Scenario Description: Ahmet had allowed his friends to play the game using his computer and he wants to know whether they broke his record. So he goes ahead and checks the leader board.

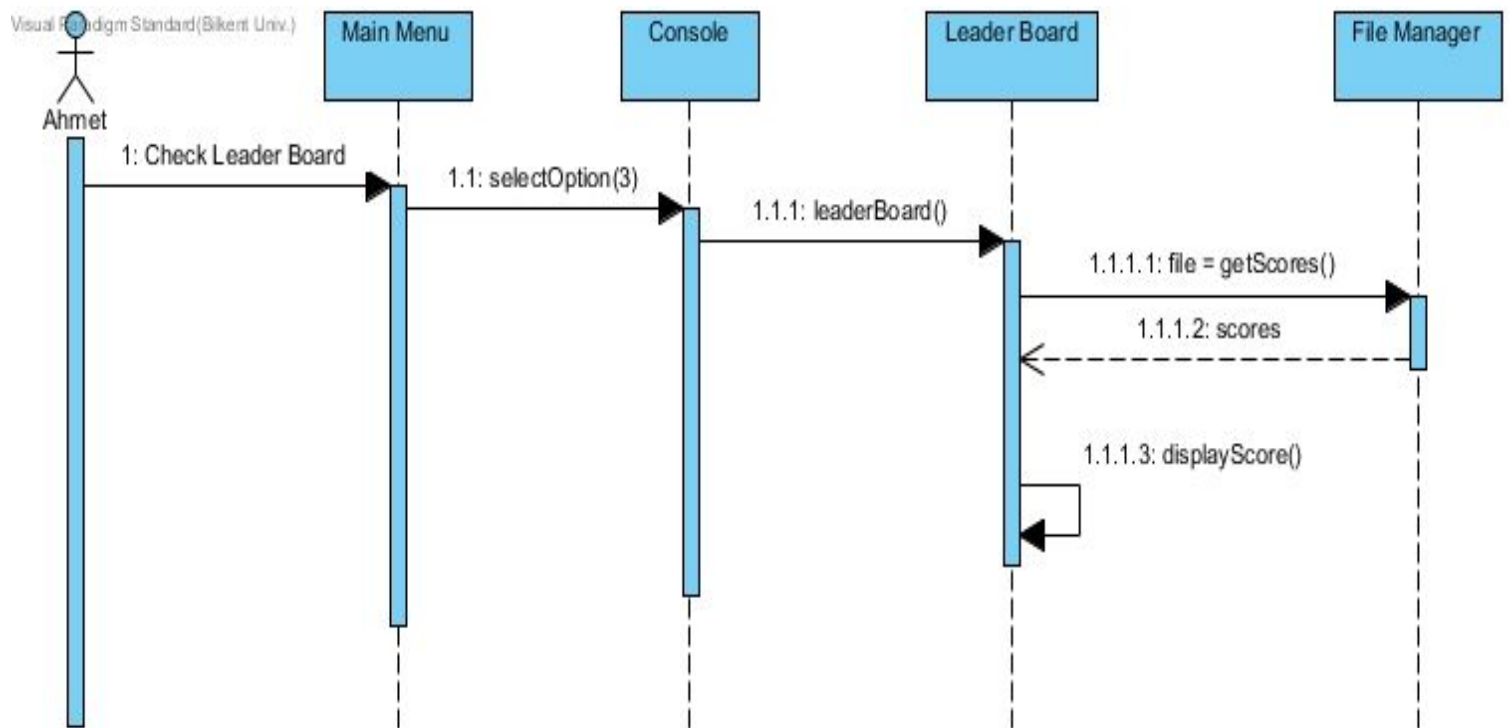


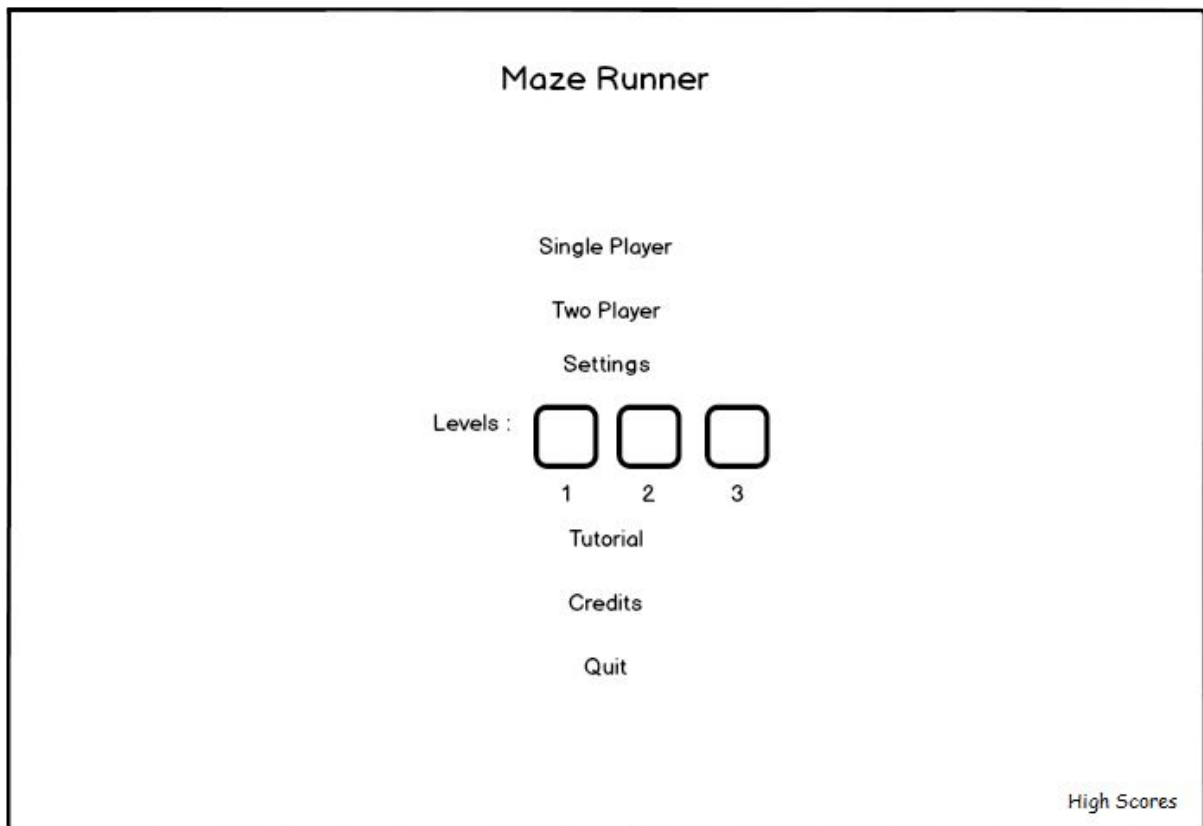
Figure 7 - Sequence Diagram for Check Leaderboard

Description: After the player chooses option 3 which corresponds to Check Leaderboard, the Console asks the Leaderboard class to show the highest scores. The class gets the file from the File Manager and then displays it in the screen.

4.4. User Interface

4.4.1. Main Menu

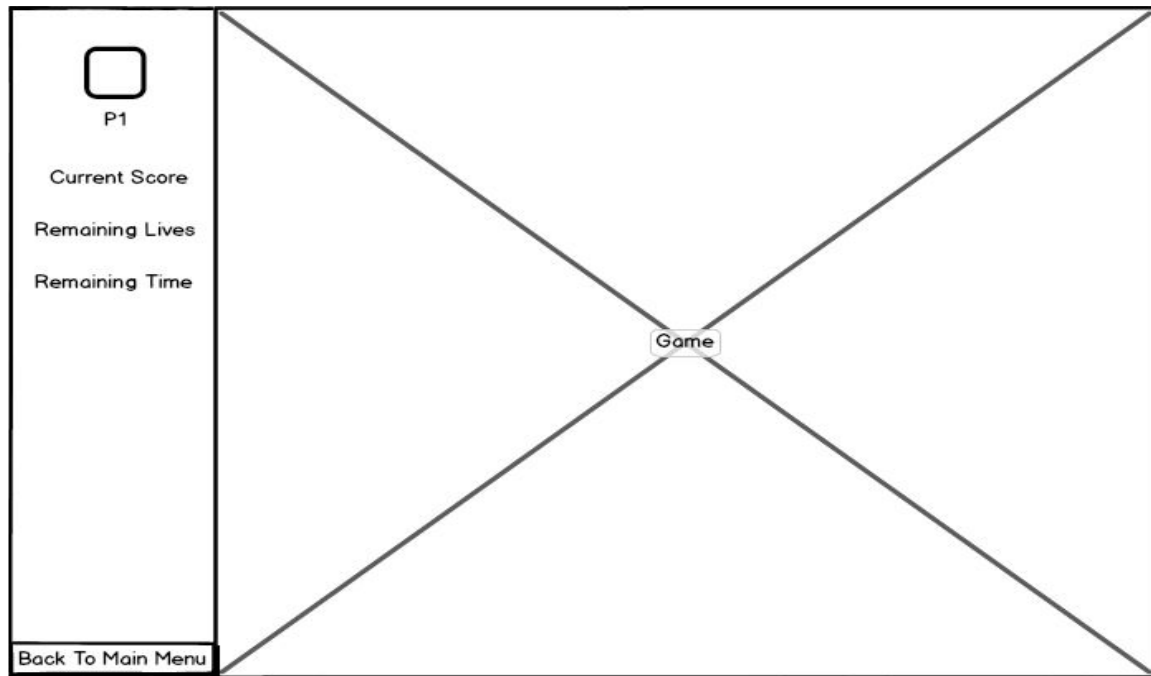
When application begins to run, Player will see main menu screen. Main menu screen shows seven options to Player which are Single Player, Multi Player, Settings, Levels (Difficulty), Tutorials, Credits and Quit. If player does not choose any level, it will give default level 1.



Created with Balsamiq - www.balsamiq.com

Figure 8 - MainMenu

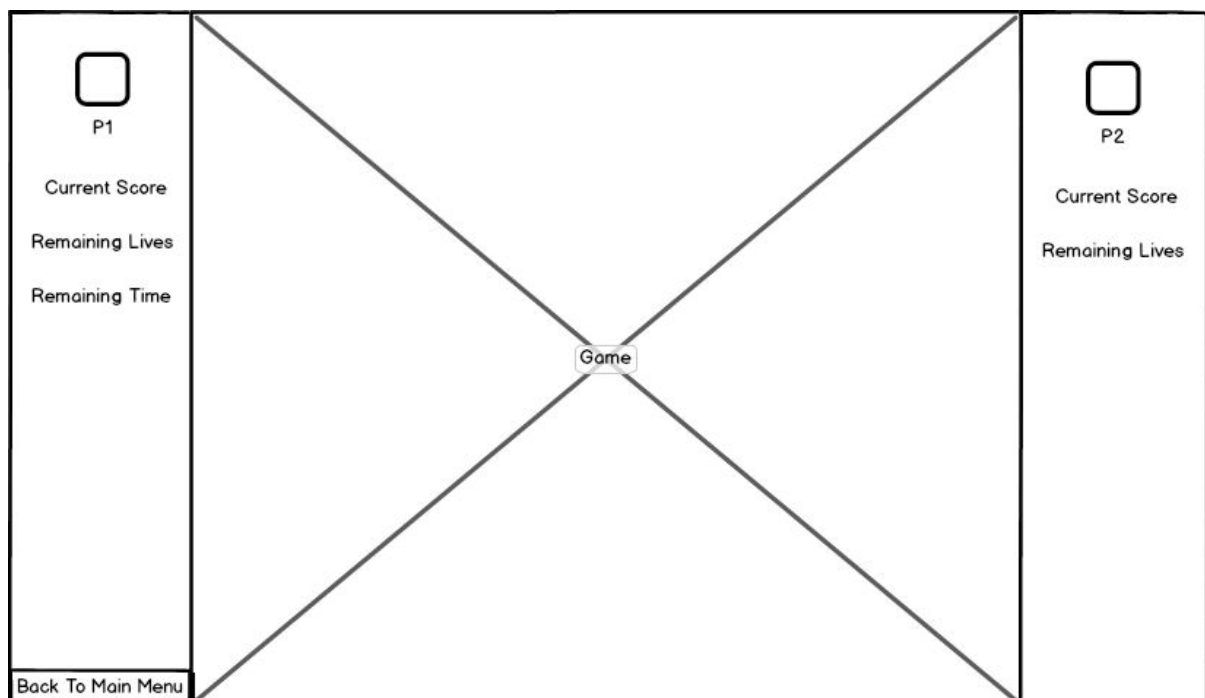
4.4.2. Single Player Game Mode



Created with Balsamiq - www.balsamiq.com

Figure 9 - Single Player Board

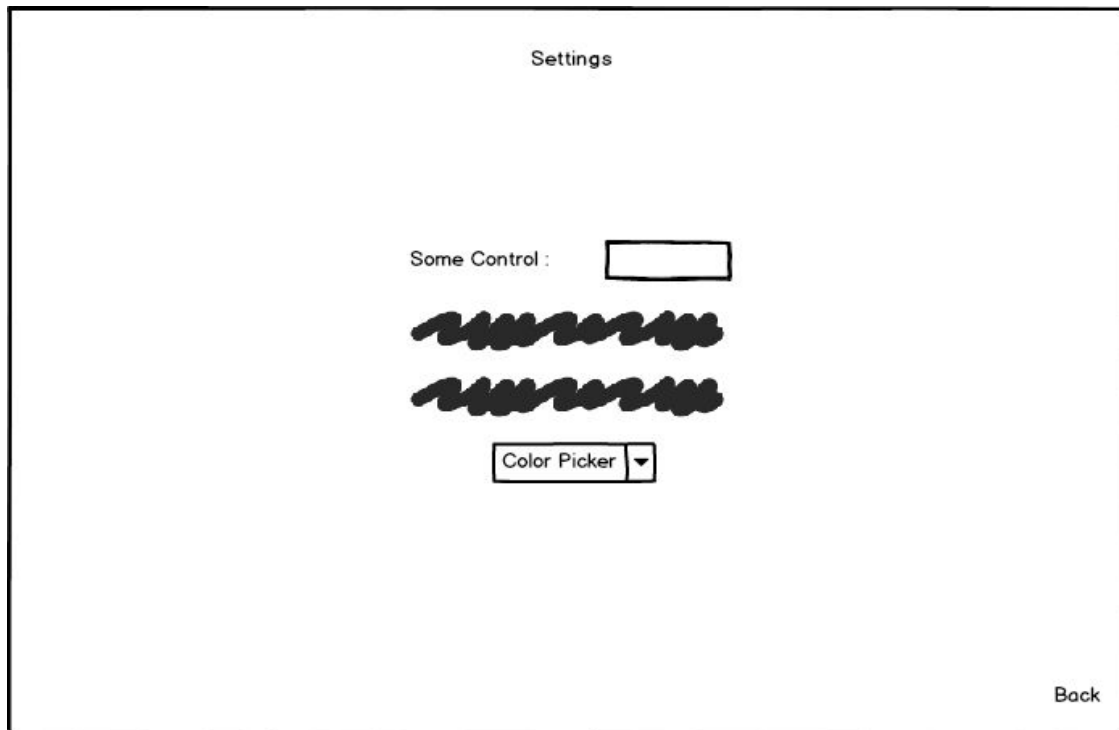
4.4.3. Multi-Player Game Mode



Created with Balsamiq - www.balsamiq.com

Figure 10 - Multiplayer Board

4.4.4. Settings

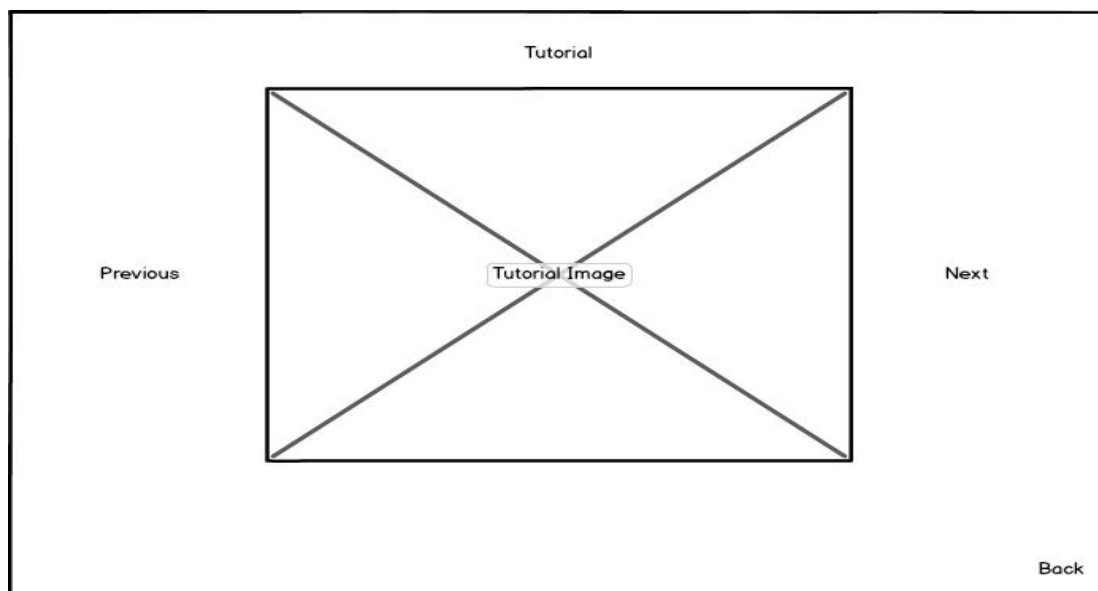


Created with Balsamiq - www.balsamiq.com

Figure 11 - Settings

When change settings button is pressed a list of options are shown on screen. These settings can be changed by Player manually. If Player does not make any change in settings, default settings are set and used by system.

4.4.5. Tutorial

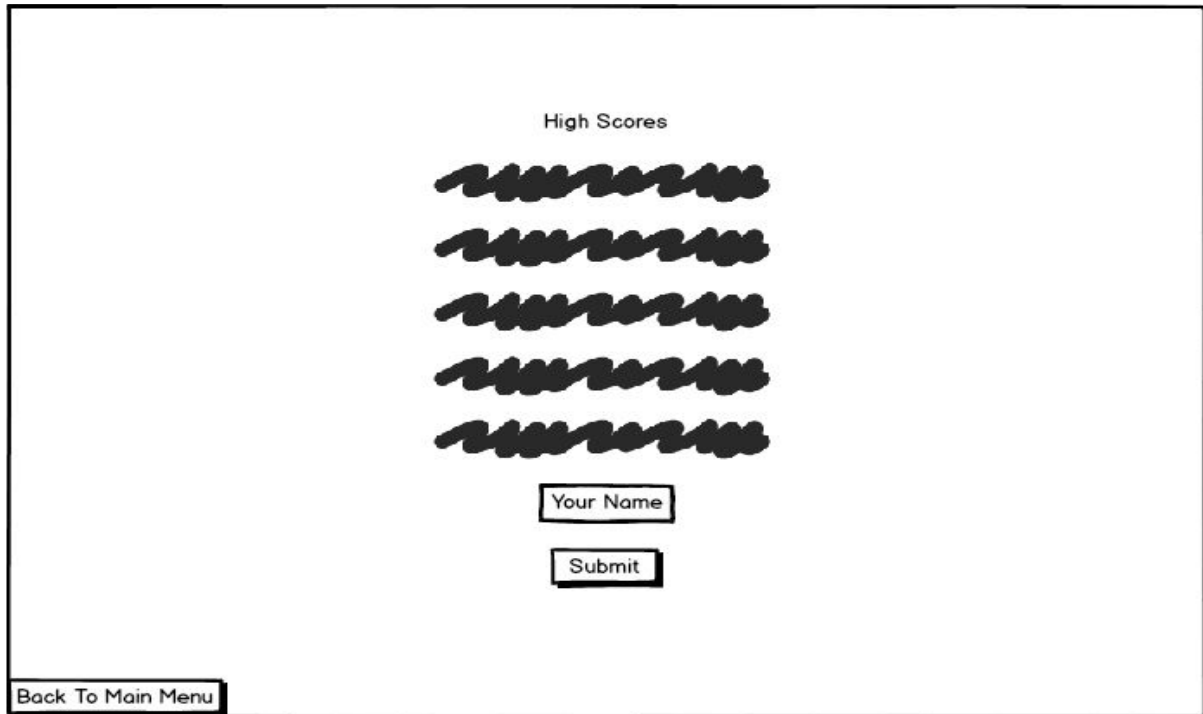


Created with Balsamiq - www.balsamiq.com

Figure 12 - Tutorial

Player can look at tutorials for game assistance.

4.4.6. High Scores



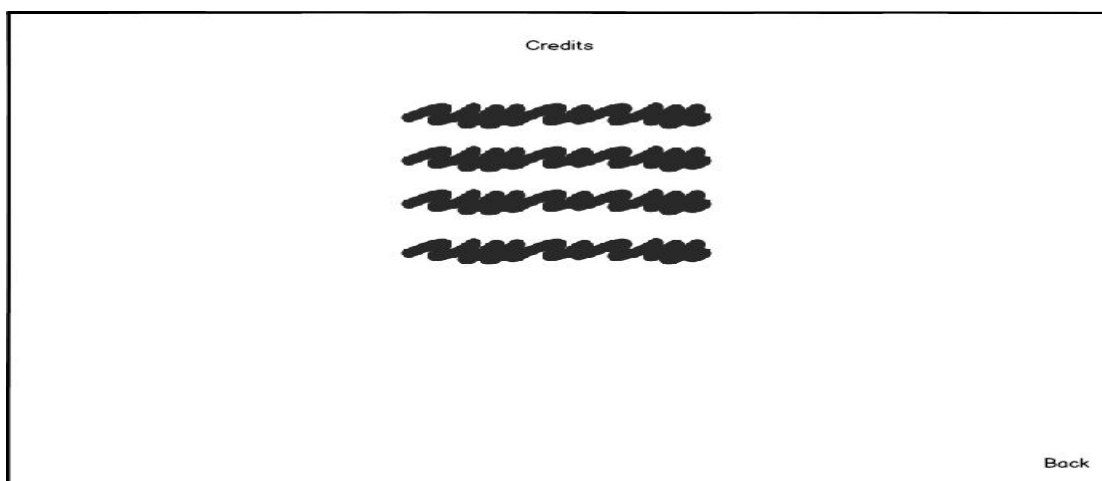
A wireframe mockup of a 'High Scores' screen. At the top center is the title 'High Scores'. Below it are five rows of placeholder text, each represented by a series of black, wavy, scribbled lines. Under the placeholder text is a text input field labeled 'Your Name' and a 'Submit' button below it. In the bottom left corner, there is a button labeled 'Back To Main Menu'.

Created with Balsamiq - www.balsamiq.com

Figure 13 - High Scores

If the player passes someone on high-score list at the end of the game, the player will be able to place his/her name within the top 5. The list rearranges automatically.

4.4.7. Credits



A wireframe mockup of a 'Credits' screen. At the top center is the title 'Credits'. Below it are four rows of placeholder text, each represented by a series of black, wavy, scribbled lines. In the bottom right corner, there is a button labeled 'Back'.

Created with Balsamiq - www.balsamiq.com

Figure 14 - Credits

5. References

- [1] <https://en.wikipedia.org/wiki/Pac-Man>
- [2] [https://en.wikipedia.org/wiki/Bomberman_\(1983_video_game\)](https://en.wikipedia.org/wiki/Bomberman_(1983_video_game))
- [3] <https://clipartfest.com/categories/view/a6216f025cb6e55dc6a622b58e73bfa1639db42f/free-clipart-animated-atomic-bomb.html>
- [4] <https://pixabay.com/en/blue-ghost-pacman-horror-fantasy-2045189/>
- [5] <https://pixabay.com/en/yellow-ghost-pacman-horror-fantasy-2045191/>
- [6] <https://scratch.mit.edu/projects/95364380/>