****

Bilkent Universtity

Department of Computer Engineering

CS 319 Term Project

Group 1D  
Wars & Warriors

Requirement Analysis Report

Project Group Members:

**Samet DEMIR  
Ahmet MALAL  
Mahammad SHIRINOV  
İbrahim MAMMADOV  
Huseyn ALLAHYAROV**

Supervisor: Eray TUZUN

Table of Contents

[1. Introduction 3](#_Toc527912421)

[2. Overview 4](#_Toc527912422)

[2.1. Gameplay 4](#_Toc527912423)

[2.2. Map 4](#_Toc527912424)

[2.3. Walls 4](#_Toc527912425)

[2.4. Warriors 5](#_Toc527912426)

[2.5. Menus 5](#_Toc527912427)

[2.5.1. Main menu 5](#_Toc527912428)

[2.5.2. Walls Menu 5](#_Toc527912429)

[2.6. Settings 5](#_Toc527912430)

[3. Functional Requirements 5](#_Toc527912431)

[3.1. Main menu 5](#_Toc527912432)

[3.2. Play 5](#_Toc527912433)

[3.3. How to play 6](#_Toc527912434)

[3.4. Settings 6](#_Toc527912435)

[3.5. Credits 6](#_Toc527912436)

[3.6. Quit 6](#_Toc527912437)

[4. Non-functional requirements 6](#_Toc527912438)

[4.1. Performance 6](#_Toc527912439)

[4.2. Interactivity 7](#_Toc527912440)

[4.3. Compatibility 7](#_Toc527912441)

[5. System Models 7](#_Toc527912442)

[5.1. Use Case Diagram 8](#_Toc527912443)

[5.1.1. Use Case: Play 9](#_Toc527912444)

[5.1.2. Use Case: How to Use 10](#_Toc527912445)

[5.1.3. Use Case: Credits 10](#_Toc527912446)

[5.1.4. Use Case: Pause 11](#_Toc527912447)

[5.1.5. Use Case: Select an Object 11](#_Toc527912448)

[5.2. Dynamic Models 12](#_Toc527912449)

[5.2.1. Sequence Diagrams 12](#_Toc527912450)

[5.2.2. Activity Diagrams 18](#_Toc527912451)

[5.2.3. Object Model 21](#_Toc527912452)

[5.3. User Interface: Navigational Path & Screen Mockups 23](#_Toc527912453)

[6. Improvements summary 24](#_Toc527912454)

[6.1. Map 24](#_Toc527912455)

[6.1.1. Different Sizes 24](#_Toc527912456)

[6.1.2. Immobile Objects 24](#_Toc527912457)

[6.2. Walls 24](#_Toc527912458)

[6.2.1. New Shapes 24](#_Toc527912459)

[6.2.2. Chains 24](#_Toc527912460)

[6.2.3. Strength 24](#_Toc527912461)

[6.3. Warriors 24](#_Toc527912462)

[6.3.1. Attackers 24](#_Toc527912463)

[6.3.2. Walkers 25](#_Toc527912464)

[7. References 25](#_Toc527912465)

# Introduction

The requirement analysis and development needs for the Walls and Warriors project will be explained in this document that is being implemented as a project for the Object Oriented Programming course by the group of 5.

Document is intended to describe the subsections of the development process. These subsections include overview, functional and non-functional requirements, system models, mockups and improvements list to clarify the status and needs of the project.

Overview and gameplay part will be depicting the final product in mind, its gameplay, attributes and features from the perspective of a user. More detailed, systematic and technical definitions will be given in the proceeding sections.

Functional requirements will include main features of the game that a user will be interacting during the gameplay. These features are consisting of menu options such as starting a new game, selecting levels, options and game instructions and guidelines.

Non-functional requirements will be describing developer needs and responsibilities for a fluid gameplay and responsiveness. This part includes performance and stability, interactivity and responsiveness and compatibility with all the platforms.

System models will be the section explaining the different scenarios and user decisions during the game with different diagrams developing in accordance with the details. Use case diagrams will be explaining user decisions and consequent scenarios. Dynamic models will go further in detail to explain gameplay from a technical perspective. These models include sequence and activity diagrams. The object model explaining the classes, instances and their associations is included in the document describing the game’s structure from a developer perspective.

Mockups are attached to visualize the final picture for better understanding and expectations for the project which will finalize the requirement analysis.

Improvements list will be summarizing the additional features that distinguish this project from the classical board game to increase user affinity and game complexity for a better experience.

# Overview

This section is intended to explain the general overview and gameplay of the game and key features that is associated with the gameplay. Additionally, main attributes and parts of the gameplay will be explained in subsequent subsections to describe functionalities of these attributes. Map is one of the features that will be explained with added features and main functionality which will be the most frequently interacted part of the game.The objects included in the map, such as the walls and warriors and their functions will be described in the proceeding subsections. Menus that are interacted to play the game will be explained with all the aspects. Finally, settings for the gameplay that are customized by the user will be described in the last subsection.

## Gameplay

Walls and warriors is a strategic board game that is being implemented in 2D for the desktop environment via this project. Game is consisting of a map that is populated with warriors of different kinds and walls for keeping the enemies outside the borders of the castle of a certain kind of warriors. Main responsibility of the user is to find a way to place the walls in such a manner that they do not overlap and the final castle includes only the warriors of one kind, mainly, blue warriors. The board game has a fixed size map and 2 kinds of warriors, blue and red, and 4 types of walls with different shapeswhich a user can pick one of each at most. The game always has one solution which means there is only one way to place the walls in such a manner that the user will win. This project is based on these key features with more additional ones for increased complexity. These additional features include different sizes of maps, immobile objects and obstacles on the map, different shapes of walls, new functionalities of certain warriors and more which will be explained in more detail in the improvements summary. With the additional features, game may have more than one solution and a certain algorithm will be used to determine if the game is over or not.

## Map

Map is the fundamental part of the game that all the action is happening on. Map is the base for the walls and warriors to be placed on. With the additional features, in the proceeding levels, map may have natural immobile objects such as lakes, forest or rocks to challenge the user to come up with a solution that circumvents these obstacles.

## Walls

Walls are the fundamental parts of the final castle which will determine if the user wins or not. With many shapes that satisfies different scenarios, user will be responsible for selecting a wall from a menu to place on the map. The walls will have some strength that may be decreased with the attacks of the enemy warriors which challenges user to think and play faster before they collapse. The strength feature will be included in harder and more complex levels.

## Warriors

Warriors are the main characters of the game user interacts with. The placement of the warriors is generally fixed and determine the solution of the game. This project will be granting some warriors with new features to walk in certain directions for the blue warriors and attack the walls for the red warriors which adds a time constraint. These new features will increase the complexity in the harder levels.

## Menus

In the game there are 2 menus that a user interacts with. This subsection will be describing these 2 menus in the following paragraphs.

### Main menu

Main menu is the first screen that welcomes the player. The menu includes buttons to start playing, reading the guidelines and instructions, customize the game settings, display the credits and to quit the game. User needs to select the play option to start the game and select the level.

### Walls Menu

In the game, there will be a walls menu that displays the different kinds of walls a user may pick. This menu may present more options in accordance with the complexity of the current level.

## Settings

Settings menu will allow the user to customize the settings for the desired gameplay. The customizable settings will include adjustment options for the game sounds (SFX) and background music.

# Functional Requirements

This section includes the requirements and expectations for the key features of the game that a user will be interacting with. These features include main menu, play, how to play, settings, credits and quit options.

## Main menu

Main menu is the main screen that will be encountered by the user after the execution. This menu will let user to choose one of the options that are being displayed and need to respond accordingly. The main menu will be displaying play, how to play, settings, credits and quit options and will respond with the pre-defined functions for different scenarios.

## Play

To start the game, user needs to select the play option which will bring the available levels in response. Levels will be available consecutively after a successful gameplay in the previous level. After choosing a level from the available ones, user will be welcomed with the map that comes with the warriors placed beforehand. The size of map, placement and functionalities of the warriors, features and available shapes of the walls will be determined by the game in accordance with the level complexity. For that reason, features and availability of the walls, functions of the warriors and obstacles on the map will be calculated beforehand to guarantee at least one solution. User will be interacting with the walls via a mouse or other tracking device and drag and place the walls in a manner that they do not overlap and all the obstacles are ignored. Game will detect whether the selected wall can be placed on a certain location that user desires with considering conditions and limitations explained above. Game will be dynamically checking if the user has won yet or not and will respond immediately if the game is over either by a win or because of collapse of the walls. Finally, proceeding level will be made available to play if user won.

## How to play

How to play section will describe the gameplay, rules and instructions as its name suggests. Basic rules of the board game and features of the walls and warriors with the additional ones will be explained thoroughly in this section.

## Settings

This settings section will allow the user to customize the game setting for a desired experience. These settings will allow the adjustment of the game sounds (SFX) and background music volume. Additional options may be included such as brightness or graphics.

## Credits

Credits will display the information about the developers and the contributors of the project. Their names will be displayed on the screen. Some additional information about the developers may also be included in addition to the names such as linkedin accounts or github repository of the project.

## Quit

As the name suggests, quit option will allow the user to quit and terminate the game from the main menu. This option will appear only in main menu but a terminate button will also work any time to close the window and terminate the process immediately.

# Non-functional requirements

Main intention of this section is to explain the non-functional requirements which serve the developer responsibilities. These responsibilities include performance, interactivity and compatibility issues of the game.

## Performance

The game should not bottleneck the CPU and use many resources of the system. Lags and performance decrease or FPS drops should be solved for the final product. Main reason to choose the Java language for the implementation is to use tools and features of the JVM to tackle these issues with the help of garbage collector, for example. Screen tearing should be avoided. In other words, game should maintain a FPS rate accordance with the display in use to not to respond with a higher refresh rate that is used by the displaying device. An ideal 30-60 FPS rate should be maintained to fix any probable bugs related to this issue as it is the case with most of the monitors in use and considering the idleness of the game. CPU usage should also be reasonable for a smooth experience and sound production. Finally, game should not use many data resources and be as compact as possible for a smaller and more desirable disk and memory usage.

## Interactivity

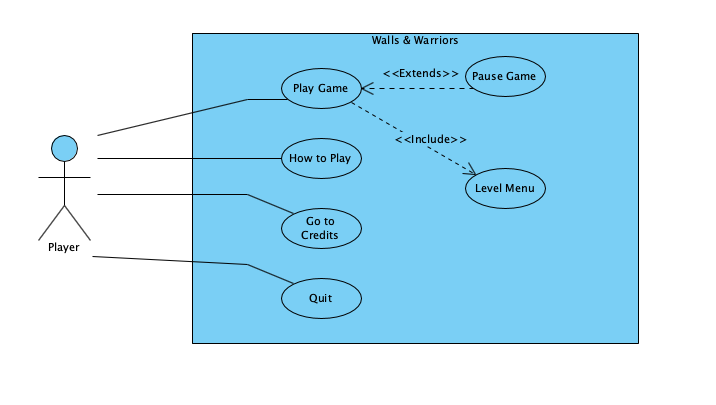
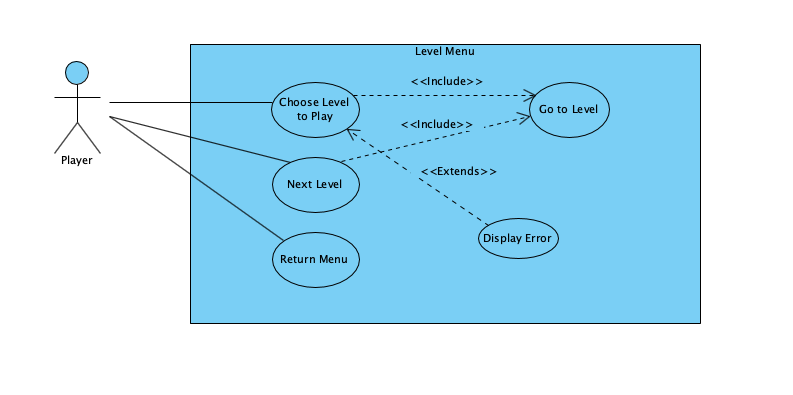
Game’s responsiveness is another challenge to be tackled for a smooth user experience. User interface also should be as simple as possible to increase satisfaction of the customer. Response time of the game for the monitor and mouse should be reasonable for these purposes.

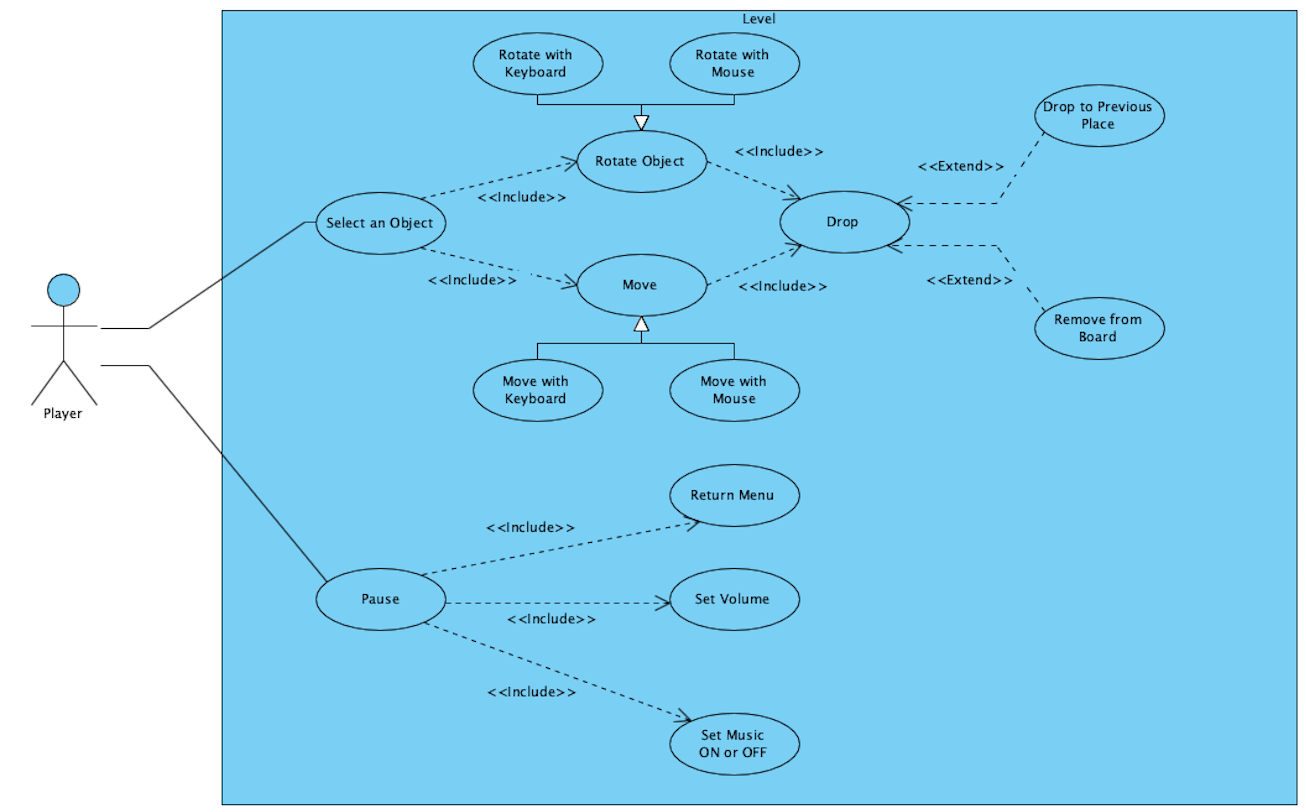
## Compatibility

The game should be compatible with all the desktop platforms and operations systems in use. Java’s JVM tackles this issue as it comes with its own VM to run the Java programs.

# System Models

## Use Case Diagram





### Use Case: Play

Primary Actor: Player

Interests: 1. Player wants to play the game.

2.System launches the game.

Pre-Conditions: 1. Player has to start the game.

2.Player has to be in main menu.

Entry-Conditions: 1. Player clicks “Play” button in the main menu.

Exit-Conditions: 1. Player clicks “Pause” button in the game menu.

2. Player clicks “Back” button in the level menu.

Success scenario event flow:

1. Player clicks “Play” in the main menu.
2. Player selects an unlocked level from level menu.
3. Player chooses a wall and changes its positions by using direction keys on the keyboard.
4. Player drops the selected wall in suitable place.
5. Player chooses a wall and changes its positions by using mouse.
6. Repeat 4.
7. Player chooses a chain and changes its positions by using direction keys on the keyboard.
8. Player drops the selected chain in suitable place.
9. Player chooses a chain and changes its positions by using mouse.
10. Repeat 8.
11. Repeat 3-10.
12. Level is completed.

Alternative Event flow:

1. Player presses pause key while in game.
2. Player selects quit game option.
3. Walls & Warriors closes the game.

### Use Case: How to Use

Primary Actor: Player

Interests: 1. Player wants to know about the game.

Pre-Conditions: 1. Player has to be in the main menu.

Entry-Conditions: 1. Player clicks “How to Play” button in the main menu.

Exit-Conditions: 1. Player clicks “Back” button in the how to play screen

Success scenario event flow:

1. Player presses How to Play button.
2. Player views tutorials to learn how to play Walls & Warriors
3. Player presses Back button
4. Player goes to main menu.

### Use Case: Credits

Primary Actor: Player

Interests: 1. Player wants to know credits.

Pre-Conditions: 1. Player has to be in the main menu.

Entry-Conditions: 1. Player clicks “Credits” button in the main menu.

Exit-Conditions: 1. Player clicks “Back” button in the how to play screen

Success scenario event flow:

1. Player presses Credits button.
2. Player views the developers.
3. Player presses Back button.
4. Player goes to main menu.

### Use Case: Pause

Primary Actor: Player

Interests: 1. Player wants to change the settings.

Pre-Conditions: 1. Player has to be in a level.

Entry-Conditions: 1. Player clicks “Pause” button in the game menu.

Exit-Conditions: 1. Player clicks “Back” button in the pause screen.

Success scenario event flow:

1. Player presses Pause button.
2. Walls & Warriors pauses the game screen
3. Walls & Warriors displays the pause menu.
4. Player selects to continue level.
5. Walls & Warriors continues the game.

Alternative Event flow:

1. Player presses Pause button.
2. Walls & Warriors pauses the game screen
3. Walls & Warriors displays the pause menu.
4. Player selects to continue level.
5. Walls & Warriors continues the game.
6. Player presses sound up button.
7. Player presses sound down button.
8. Player presses music up button.
9. Player presses music down button.
10. Player presses return the main menu.
11. Walls & Warriors display the main menu.

### Use Case: Select an Object

Primary Actor: Player

Interests: 1. Player wants to pick an object like a wall or chain.  
2. Player wants to change the position of an object it is already in the map.

Pre-Conditions: 1. Player has to be in the level.

Entry-Conditions: 1. Player has to use mouse.  
2. Player has to use specific keys on the keyboard.

Exit-Conditions: 1. Player releases the mouse.  
2. Player releases pressed key.

Success scenario event flow:

1. Player clicks an object in the wall/chain panel.
2. Player clicks the rotate button on the specific object.
3. Walls & Warriors rotates the selected object
4. Player presses the selected object and drops into the map.
5. Walls & Warriors changes the position of the selected object.

Alternative Event flow:

1. Player clicks an object in the wall/chain panel.
2. Player clicks the rotate button on the specific object.
3. Walls & Warriors rotates the selected object
4. Player presses the selected object and drops into the map.
5. Walls & Warriors does not change the position of the selected object.

## Dynamic Models

### Sequence Diagrams

#### Scenario 1 - Discovering Main Menu

Entry Condition:

Player is on main menu

Exit Condition:

Player wants to exit the game

Main Flow of Events:

1- Player clicks on Play button

2- View Controller displays the level menu

3- Player clicks on the first level

4- Game Controller checks if level 1 is currently open

5- Game Controller tells the model to create level 1

6- Game Controller tells the view to display level 1

7- Player wants to go to level menu

8- Player wants to go to main menu

9- Player wants to go to How to Play screen

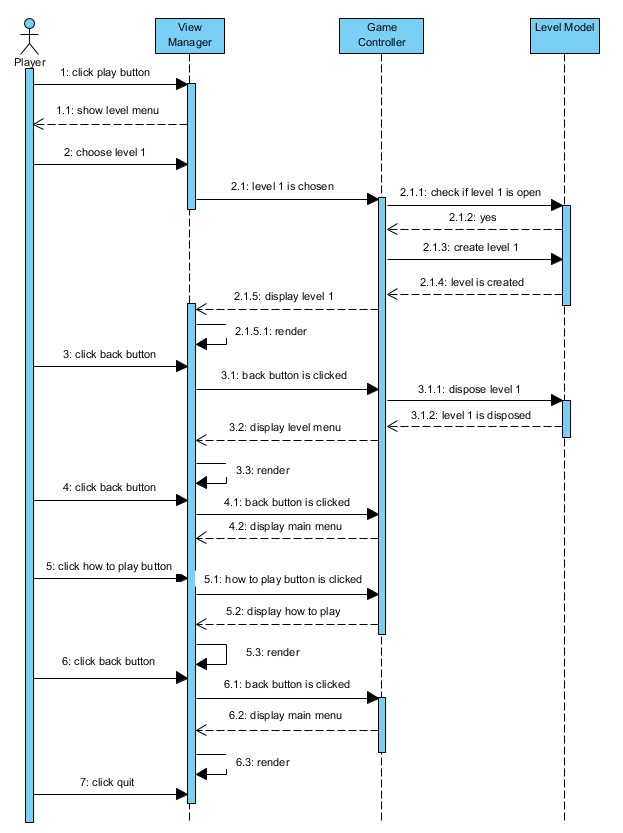
10- How to Play is displayed

11- Player presses the back button to go to main menu

12- Player presses the credits button

13- Credits screen is displayed

14- Player exits the game



#### Scenario 2 - Settings

Entry Condition:

Player is on a level

Exit Condition:

Player wants to go back to level menu

Main Flow of Events:

1- Player clicks on the Settings button

2- Game Controller will pause the game and show a pop-up for Settings

3- Player will turn off the music

4- Game Controller will update the Sound Controller

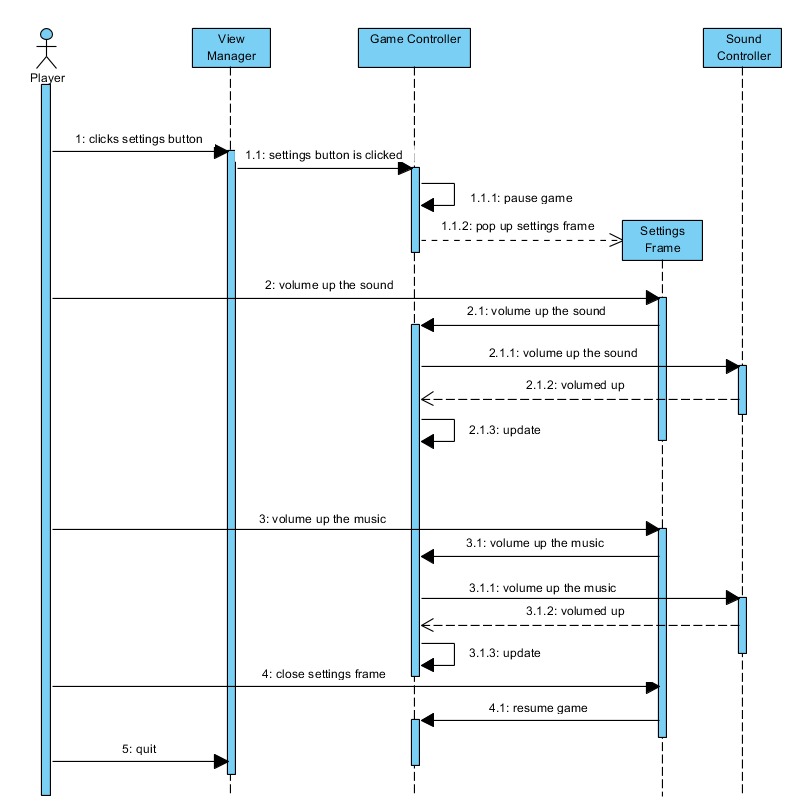
5- Player will increase the volume of the sound of the game

6- Game Controller will update the Sound Controller

7- Player will close the pop-up

8- Game Controller will resume the game

9- Player wants to go back to level menu



#### Scenario 3 - Placing the Wall

Entry Condition:

Player is on a level

Exit Condition:

Player wants to go back to level menu

Main Flow of Events:

1- Player drags a wall and places it on an obstacle such as lake, tree and another wall

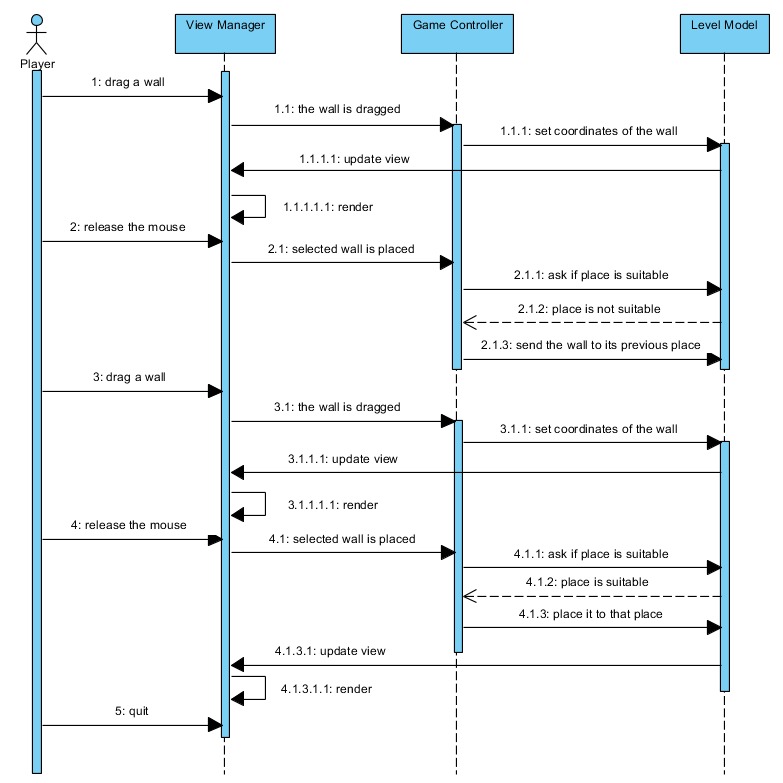
2- Game Controller sends the wall back to its previous place

3- Player drags a wall and places it outside of the board

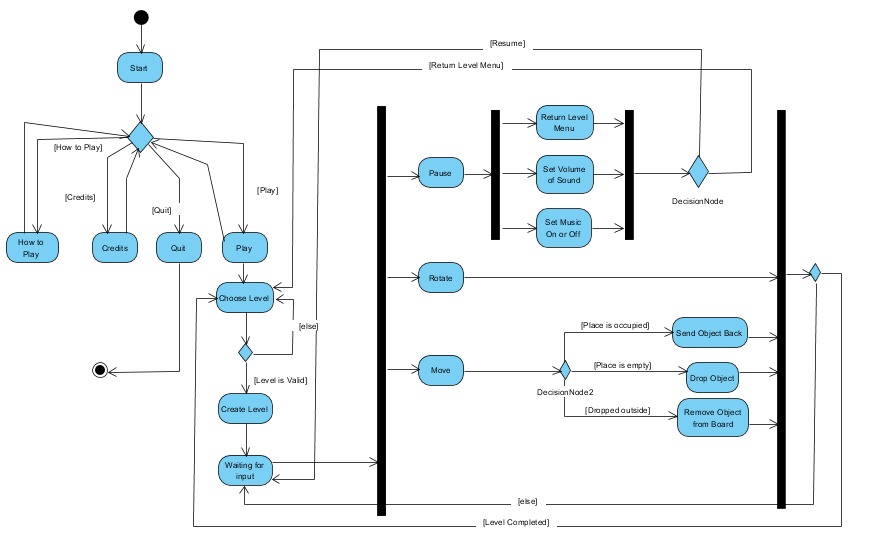
4- Game Controller removes it from the board

5- Player drags the wall and places it to a suitable place

6- Player wants to leave the level and goes to level menu



### Activity Diagrams

  
  
- When the player starts playing the game, the game will show the main menu, which includes Play, how to Play, Credits and Quit options.

- The system will wait for the selection of the player. Depending on the player's selection, what the next page is going to be will be decided.

- If the user clicks on How to Play button, a panel having a text which describes what the game is and how the game is played will appear on the screen.

- If the user clicks on Credits button, a panel having a text which have the names of the members of our project team will be displayed

- If the user clicks on the Quit button, the current situation of the game will be saved in a file and the game will be disposed.

- When the user selects Play option, level menu will appear and the system will wait for the user to choose a level.

- If the prerequisites of the selected level are not satisfied, the player will not be able to play the level and the system will wait for the player to choose another level.

- If the prerequisites of the selected level are satisfied, the player will start playing the level.

- When the user clicks the back button from any one of the pages that comes right after the main page such as Credits page and How to Play page, main menu will reappear.

- In the starting of the level the system will wait for an input such as dragging a wall/chain and clicking the rotation button of a wall.

- When the user clicks on the rotation button of a wall/chain, the wall/chain will be rotated 90 degrees clockwise.

- The player will be able to drag the wall, when s/he presses the wall/chain and drag the mouse.

- When the player drops the wall, there are three possible events

1- If there is another object at the dropped place of the wall/chain, the object will be sent to its previous place.

2- If the wall/chain is dropped outside of the board (it is enough for one piece of the wall/chain to be outside) the wall/chain will be removed from the board and sent to the wall/chain panel.

3- If the place is available, wall/chain will be placed on that place.

- When an input is executed, we will check if the level is completed or not.

1- If it is not completed, we will wait for the player to enter another input.

2- If it is completed, it will return to the level menu.

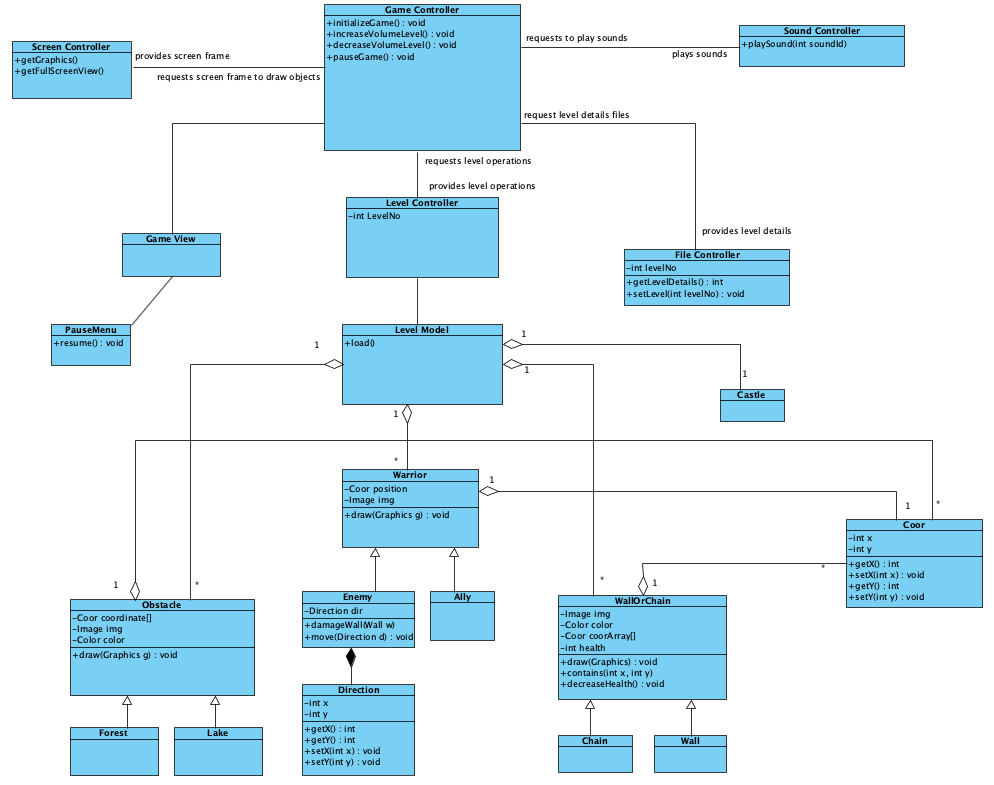
- If the player pauses the game, pause frame will appear and the player will be able to set settings.

- The user will be able to resume the game and to go back to level menu.

1- If the player resumes, the system will wait for an input from the player

2- If the player wants to go back to level menu, the level will be disposed and the level menu, which allows the player to choose a level, will be displayed.

### Object Model



#### Boundary Classes

**GameView**: It is the user interface class. It simply takes the inputs from the user and sends the required information to Game Controller

**PauseMenu**: It is menu that is shown when the game is paused. It allows the user to set the setting and to go back to the level menu

#### Controller Classes

**GameController**: GameController is the fundamental controller class of the game. When the GameController is initialized, it will create instances of other controller class objects. It will be also a bridge between GameView and GameModel. When an input is entered, GameView class will inform GameController. GameController will manipulate GameModel.

**LevelController**: LevelController will keep the information of the levels such as where are the objects in the level and whether a level is locked or not. It will also load the levels by using FileController class.

**ScreenController**: It deals with screen operations

**FileController**: It deals with the file that keeps the information of levels

**SoundController**: It deals with the sounds and music.

**LevelModel**: It is the model class of the game. It is manipulated by GameController class and updates the screen through GameView

#### Entity Classes

**Warrior**: It is the parent class of Enemy and Ally classes.

**Enemy**: It is the warrior of enemy of the player. Some of them will be able to break the wall when they are near any wall or able to move through a specified path.

**Ally**: It is the warrior of the player. As the aim of the game implies, Allies should be inside the walls.

**Coor**: It simply keeps the x and y coordinates

**Obstacle**: It is the parent class of Forest and Lake classes. This class will not allow the wall/chain to be placed on that place.

**Forest**: It is a type of obstacle

**Lake**: It is a lake in which there can be enemy ships or ally ships. The player is supposed to separate them by using chains.

**Direction**: It is simply vector class which has x and y directions

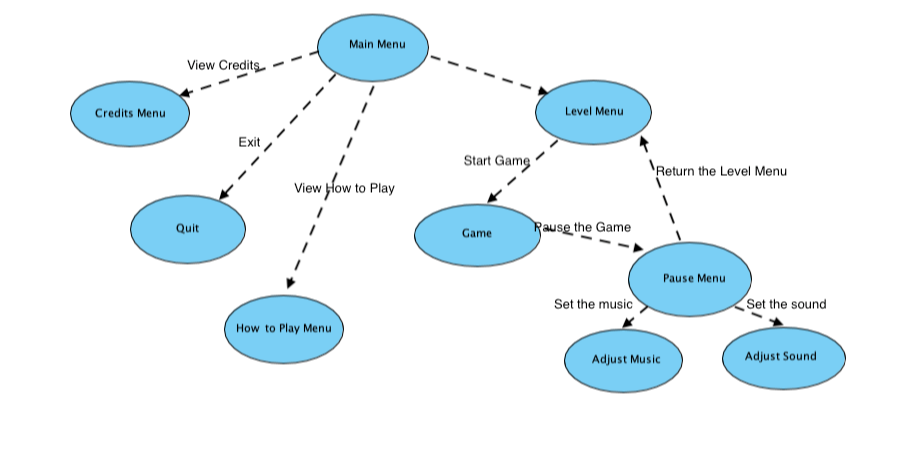
**Castle**: It is the castle of player around which player has to wrap by walls

**WallOrChain**: It is the parent class of Wall and Chain

**Chain**: It separates ally ships from enemy ships

**Wall**: It separates ally warriors from enemy warriors

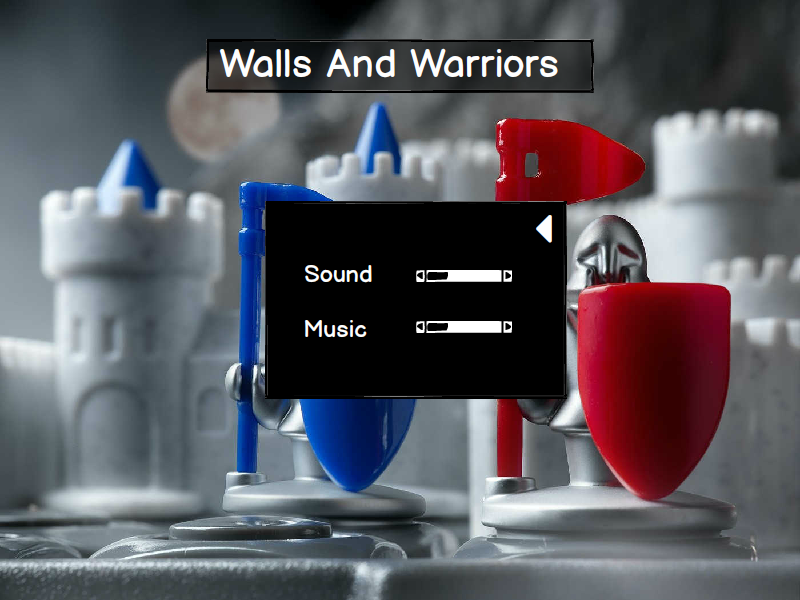
## User Interface: Navigational Path & Screen Mockups



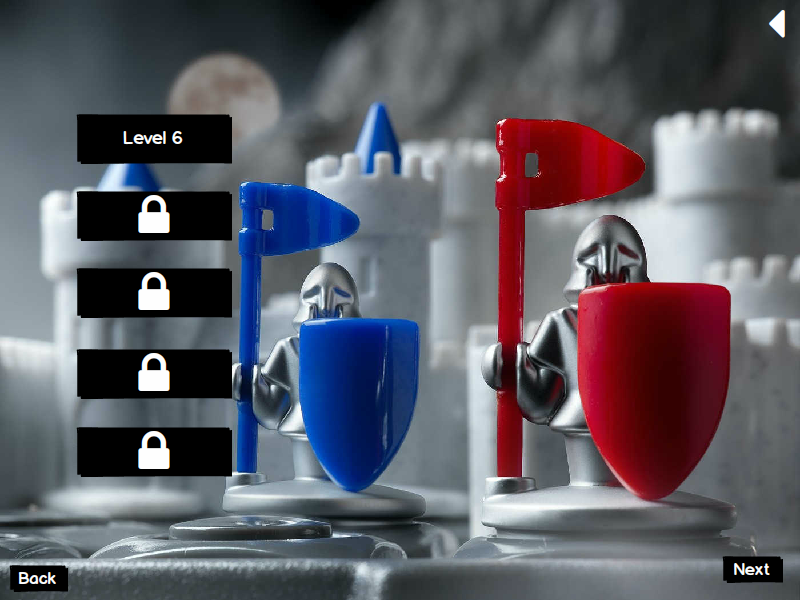
The main menu which is displayed at startup contains buttons to start and quit the same and learn about its creators or about the game. The “Settings” button will open the Settings page (below).

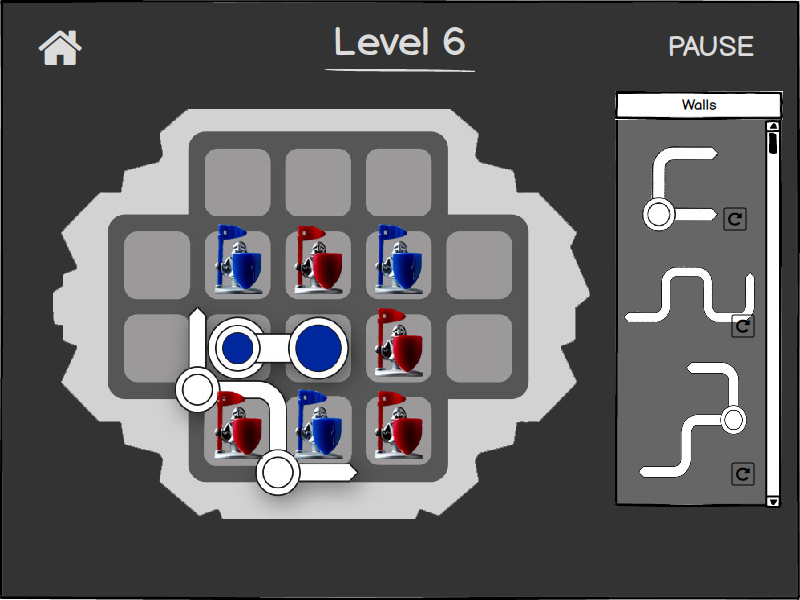


In Settings menu, you can change the volume of background music and sound effects.

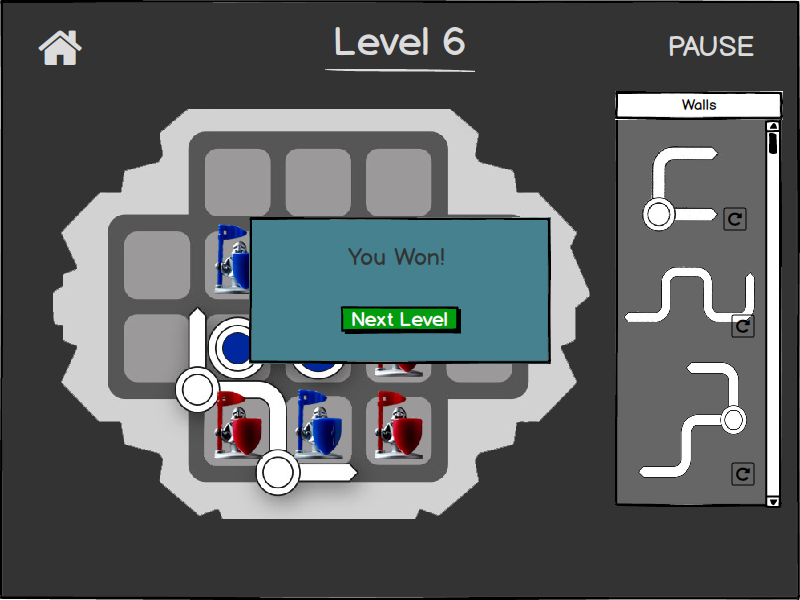


When “Play” button is pressed in main menu, the user is directed to the “Selects Levels” menu, to choose which level they want to play.

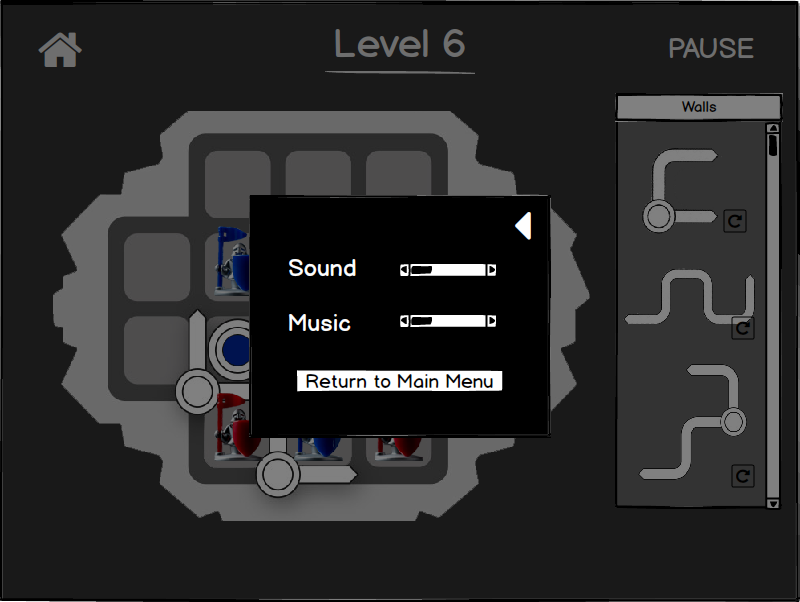


Choosing a level brings one to the game screen. The game screen displays the game board and the walls the player can select from. The player can rotate the walls by 90 degrees using the ‘rotate’ button next to each wall item. It also has buttons to go back to the main menu or pause the game (see below).

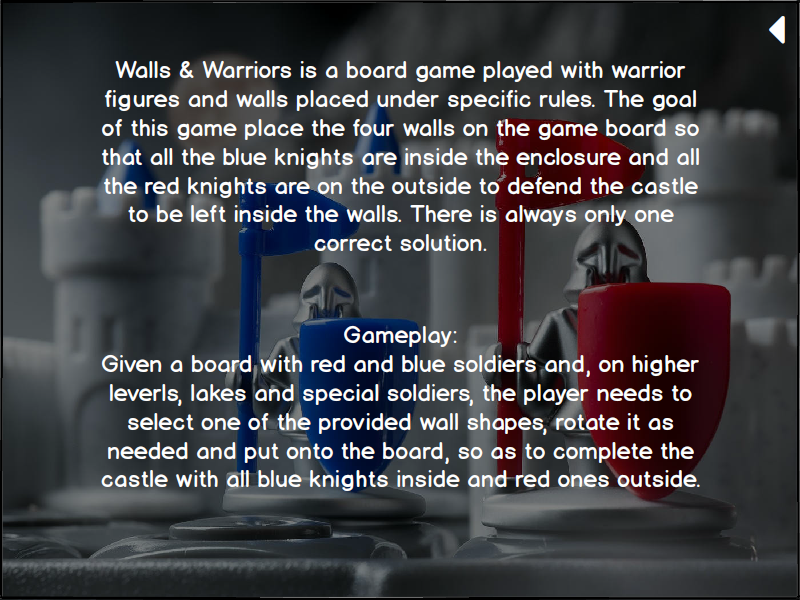
When you complete the challenge, a pop-up appears and you can proceed to the next level.



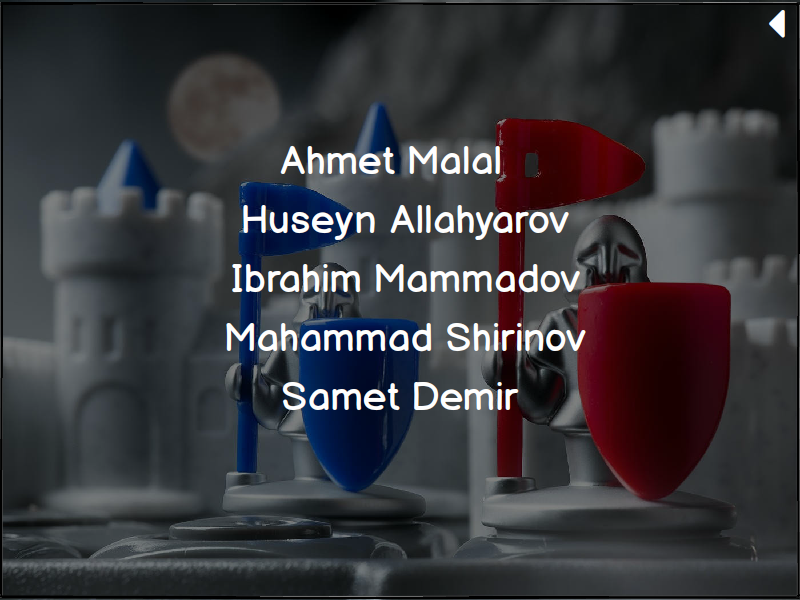
When the game is paused, the player can change some basic settings or return to the main menu.



The “How To Play” screen gives the user information about the game and instructions about how to play.



Finally, the “Credits” screen gives a list of the creators of the game. Some additional information can be given here.



# Improvements summary

To challenge the user and increase the complexity of the game, many additional features have been added to the classical board game. These features will be explained in the proceeding subsections which will describe the additions to the map, walls and the warriors.

## Map

### Different Sizes

Unlike the board game, this game comes with different sizes of the maps to increase the complexity. Levels will start easy and advance accordingly. One of the advancements include the increase in the map’s size which will allow it to house more warriors and therefore, more walls and obstacles.

### Immobile Objects

As the map expands, immobile objects may be generated accordingly to increase complexity and challenge the player. These objects include lakes, forest and rocks that cannot be moved and need to be ignored to build the castle. Lakes may house ships of different warriors. Blue ships should be in the castle’s area while the red ships should remain outside. In that scenario, walls may contain the blue ship by the chains that are attached to the walls.

## Walls

### New Shapes

In the board game there are 4 wall shapes and they all can be placed on a single fixed-size map. Because of the varying sizes of maps in this game, new wall shapes are also introduced accordingly.

### Chains

Like the walls on surface, there may be chains available to replace the walls’ functionality in the water. In certain scenarios where there is a lake on the map and that houses 2 different kinds of ships, chains may be used to house the blue ship in the castle’s field. These chains will appear in these scenarios on the walls menu and should be attached to the walls by the both ends.

### Strength

Walls in the harder levels will have some strength levels that can be decreased by attackers on the red ships. This same rule applies to the chains as well which may be damaged and collapsed by the attackers. This rule adds a virtual time constraint and challenges the user think and decide faster to win the game. If the strength of the walls decreases to 0, game is over and player loses.

## Warriors

### Attackers

As mentioned in the previous subsections, in the maps that house a lake with red ships, there will be red warriors, in other words enemies, who will attack and try to collapse the walls or the chains. These warriors will be seen in the harder levels.

### Walkers

Rather than being in the same location and fixed place, some blue soldiers will be walking in certain direction. Player needs to place the wall accordingly to contain these walking warriors. Like the attackers, these warriors will appear in the more complex levels.

# References

Balsamiq – Mockup Software

<https://balsamiq.com>

Visual Paradigm – Diagram Design Software

<https://visual-aparadigm.com>