2018-2019 Fall
CS319 Object Oriented Software Engineering
Term Project

Section: 01
Group: 2E
Group Name: ProCODERs
Lords in Halls

Analysis Report

Group Members
1. Şamil Iraz
2. Ateş Bilgin
3. Enes Emre Erdem
4. Gülnihal Muslu
5. Can Ozan Kaş

Supervisor: Eray Tüzün
Teaching Assistant: Gülden Olgun

Table of Contents

1. Introduction

We, are named as ProCoders, choose Walls and Warriors as our course project. The reason behind of choosing from the options that are provided by instructor is that Walls and Warriors game is interesting, fun and improvable. Being interesting and fun are the main requirements of game to be preferred by gamers. On the other hand, what we mean by improvable is that, for the next version of this game, we can add some features to our game such as new game modes, new components depending on user's desires.

In our project, in addition to normal description of game, we add new features. Firstly, we add new piece to game named as white knight. Even though wide description of white knight will be given in later sections, it can be briefly said about white knight that it will be used to make game more different, funny, and interesting. Another new feature will be new game mods:

1) Arcade Mode
2) Time Challenge Mode
3) White Arcade Mode
4) White Time Challenge Mode

Furthermore of these features, we will add scores and leaderboard. With these features, we plan to make our game to be more preferred, more exciting, more interesting.

General definition of our game is that user will have finitely walls. Game will provide a map which has blue knights, red knights and white knights(depending on game mod). User will use walls to cover the knights depending on some rules. Rules will be explained clearly in the later parts. Positions of knights on map will be changed depending on the level. Furthermore depending on game mod, time will be used.

We will have three parts: Server, Database and User Interface. For the server part, we will use Java as implementing code language. For the database part, we will use MySql. For the part of user interface, we are planned to use Html and Javascript. The general reason to choose this languages and platforms is using them is easy and also all of the members almost have knowledge of them.

2. Overview

2.1 Main Page

Our application, after login game, will have one main page. This page provides understandable usage of our application to users. In this page, we will provide links to other pages of game: Game Mods, Scores, Options. As far as we observed, main page and layout are very important things for first view of user. As it is said "First impression is the last impression". As a result, we are planning to make this page as possible as ordered and also impressive. As we will provide links to other pages from main page, we will also provide a way to turn back to main page in other pages such as option page.

2.2 Maps

Maps consist of knights and walls basically. In depth, the type of the knights depends on the game mode. If the game mode is white knight game mode, there is additional knight type in games(rather than classic blue and red ones) and more points can be taken by surrounding them. We guarantee that each map has at least one solution. The thing that makes the difference is level. It means that more points can be achieved by reaching higher levels. Besides, levels will be harder as player passes the current map. The wall pieces will be placed next to the placed knights part. There will be small buttons near them to turn them. After turning the pieces, they can be located properly. In the time challenge map, there will be a stopwatch to measure the solving time. Like in the other mode, the logic of the walls are same. It checks whether the player exceeds the time or not and end the game if it is exceeded.

2.3 How to Play

Game consist of two main objects. They are wall and knights. The logic is that surrounding the knights with the wall pieces. However, there are some rules to do that. Red knights should be at out of the walls. Every map has at least one solution. The one that passes the level can have more points. In time challenge mode, time becomes another parameter for passing the levels. If player exceeds the time,

she/he fails that level and game ends. The same rules are valid for the white knight game mode. Additional to traditional ones, this mode adds the feature that if player surrounds the white knight too, she/he will get more points.

2.4 Game Modes

2.4.1 Arcade Mode

Arcade mode will be similar to the original Walls & Warriors game. Player need to place walls on the board in such a way that, walls will compose a closed area. The Blue Warriors must be inside that closed area, and red knights must be outside of that closed area. If player solves the puzzle, he/she can pass the next level. Player can rotate the walls with the button near them if needed. Additionally, players score gets higher as they pass more levels. The global high score ever done in the arcade mode will be displayed in the highscore menu.

2.4.2 Time Challenge Mode

In the time challenge mode, level rules will be the same as arcade mode. Player needs to place walls to cover all blue Warriors and exclude red warriors. However, player will have limited time to complete the level. As player passes the levels, harder levels will come up again and again. If player can not complete a level in the given time, the game will over and player will get a score according to the levels he/she pass and time she/he spent on each level.

2.4.3 Arcade Mode With White Knights

In the Arcade mode with White knights, again player needs to cover the blue warriors and exclude the red warriors with walls. In addition, if player also covers the White warriors, he/she will get higher points, but it is not necessary to pass the level.

2.4.4 Time challenge Mode with White Knights

In the Time challenge mode with White Knights, player needs to place the walls in such a way that they'll cover all the Blue Warriors and exclude the White Warriors in given time. Additionally, he/she will get higher points if covers the White Warriors also. If the time is up before player completes the level, game will over and player will get a score depending on the levels passed, time spent on each level, and White warriors s/he covered.

2.5 Scores / Leaderboard

In our game, users will be scored in game modes depending on rules and their performance. We will keep these scores in our database and we will provide users to see their scores in this page. We will also add leaderboard of game modes as an information in this page because we discussed and decided that if we give some global informations in addition to local information of user, it will lead users to contest with each other and it will help our game to be more preferred and interesting. These informations will be synchronized with the players and their local scores.

2.6 Options

The options menu provides some basic functionality for game experience. The player can adjust the music volume however she/he likes. Besides this page provides a password change opportunity too. It basically send a request to the database and database changes the password according to their usernames since usernames are unique.

3. Functional Requirements

3.1 General Features

· As it is mentioned, our project will be based on web browser. We will use many features of web but this will cause some problems. Therefore, our app will work on web browsers' some version. For example, it will be supported by Chrome version 60.0.3112 or more so as to Javascript and HTML5 work better(1). Other web browsers' such as Firefox(2), Opera(3), Explorer etc. versions also be selected and specified before implementing depending on their features.

· Our project's main language will be english so as to make it understandable for more users.

· For our project, admin is required for authorization. In this project, users will have to sign up and this authorization process will be held by administrators.

3.2 Pages Configuration

· Our game project will have an information page to make user informed about game before sign up process. This page will also be a transition page for the sign up and login pages.

· User will sign up to our web based project by using the sign up page. In that page, user will be questioned to enter valid username and valid password to enroll game. This sign up process is required for the security and keep progress of user's data.

· Our game project will also have login page. It will be used to enter the game using username and password by user.

· Our app will have main page which includes links to other pages of game such as scoreboard, settings etc. After login, user will be directed to this main page so as to make him/her to reach all of the possible pages of app.

·   There will be settings page which can be reached by game page. In this page, user can change some of the settings of app. For example, user can change the language of app or s/he will be able to change sound level.

3.3 Game

·   Our game requires many components such as knights, walls etc. As explained previously part of report, our game will depend on these components. For example to these components, knights which are separated by their colors which are blue, red or white have different points and roles. They will be used on scoring the user. Another example is the walls. Walls will be used by user to make areas in according to rules to earn points.

·   In our game, in order to make user more interested and have fun, we will create 4 different mods. They will be separated by defining different rules, game components, factors. Game components will be used in different mods of game. For example, arcade mod will not have white knight while some other mods will have. In addition, mods will differ from each other with respect to some factors such as time. In some game mods, time will be used to make game more competitive while other game mods don't have it so as to give user a relax game experience. User will be able to reach the game mods page from main page of game.

·   Our project will also have scoreboards to make game more interesting and competitive. As we get information and suggestions from possible users, having local scores is not attractive as much as global scoreboards. Users will be able to see their scores and also top scores of global players. This scoreboards page will be reachable by main page of app.

·   One of the main requirements of game is "How to play" page because there always be users who don't know the game and will want to learn game's rules, basics, components etc. As a result, this page will be added in our app to give users best enjoyment and experience. "How to play" page will be accessible from the game page of app.

3.4 Users

·   Our game will be based on users. Without users, it will be useless and waste of time and source. To keep users distinctly, information are required. For example, unique usernames will be used to make distinction. Furthermore, for users account safety, passwords will be kept as well.

·   Users will be authorized to make some changes in their accounts. For instance, they will be able to change their passwords, or their usernames dependently some rules such as being unique or acceptable.

3.5 System

·   In our project, we will create database by using MySql in order to keep some information such as username names and passwords. Instead of creating local databases, we will use some online providers for it such as DigitalOcean.

·   Our app will have user interface part and in this part, user will make some changes dynamically so we will implement these javascript codes to make our app successful in these dynamic actions. For the other parts of user interface, we have many options and we didn't decide yet but most probably, we will use Swing because it is coordinated with Java.

·   We will also create server to make connections between database and user interface parts. We will implement server in Java decided by votes of group members. We will also use Maven for the project structure.

·   When implementing, we will use Intellij as an idea. Intellij is very useful for Java implementations and also it is synchronized with Maven and Github. Intellij makes us to use these technologies easily and safely.

4. Non-Functional Requirements

4.1 Performance

While implementing our project, we will care the performance in the first hand. The reason behind it is that performance issues will cause some problems for users and it will lead them to disuse our game. To increase our game's performance, we will consider its features which may cause some performance issues such as database queries and we will implement these features efficiently as much as possible by using new technologies or decreasing unnecessary heavy features of coding languages.

4.2 Maintainability

Maintainability is very important requirement for our game application. As we experienced from the possible users of our application, users prefer applications which keep maintaining according to bugs. In addition to fixing bugs, we will add new features to our application. We are planning to add new game mods if possible and also we will add some new features to users such as giving authority users to add new maps. Furthermore, we will use new technologies to make our application better when they are released. While adding new features to our application, we will care mostly the users' opinions, experiences, suggestions etc in order to make application more usable, being preferred.

4.3 Security

Security is also another important point of our project since we will keep some information and data about users and these data will be used for some features of game such as scores. There is also another possibility that in further updates, we might need more information from user such as payment methods so security is required in our application. To provide the security, we will use unique secret names and passwords, also we will use online reliable providers for our database such as DigitalOcean.

4.4 User Interface and Usability

Good looking, understandable and usable user interface is one of most important parts of our project. We care about it because these are main factors that attract attention of possible users. Users mostly care about two things and one of them is user interface and its properties. On the contrary, their opposite situations can also cause users to not use our application so we pay attention to both user interface and usability in our application.
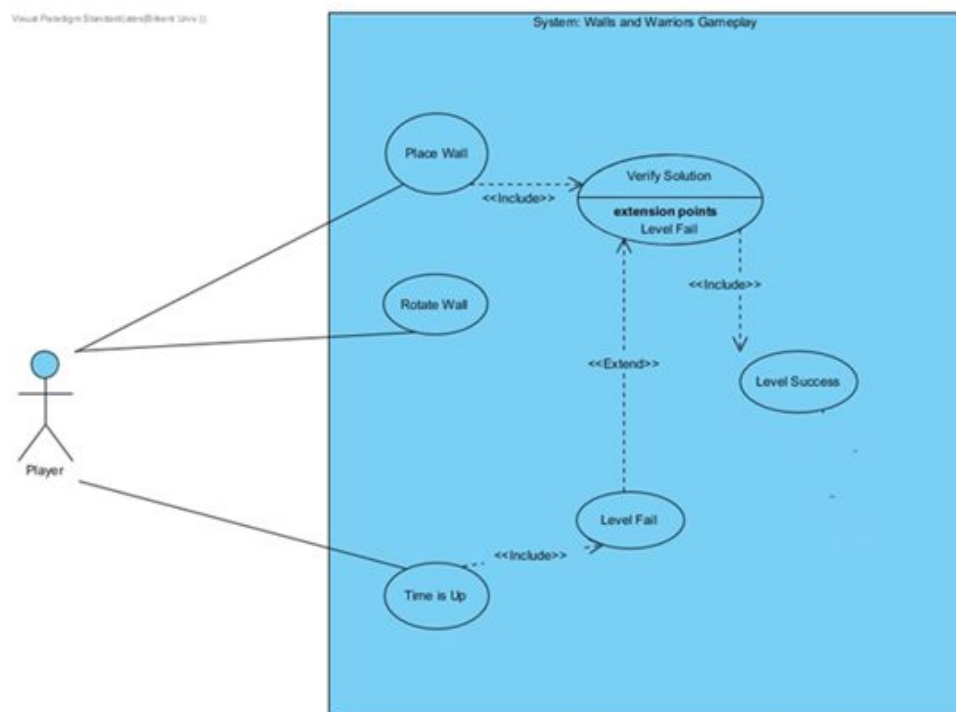
5. System Models

5.1 Use Case Models

In the Use Case Model, there are 2 use case models, one for the Menu's in the game, other for the play part of the game.

5.1.1 Gameplay Model

Gameplay Use Case Diagram:

Use Case 1:

Name: Place Wall.

Participating Actors: Player.

Stakeholders Interest: Player wants to place particular Wall in a particular position.

Entry Condition:

- Player starts particular game after entering the game system with id and password.

Exit Condition:

- Player drops the Wall in a particular position with Mouse.

Main Flow of Events:

- Player clicks on a Wall.

- Player drags the Wall on a particular position.

- Player drops the Wall.

Alternative Flow of Events:

- The position player tries to drop the Wall is full.

- Wall returns back to initial position(The beginning position, seperate from main game board).

Use Case 2:

Name: Rotate Wall.

Participating Actors: Player.

Stakeholders Interest: Player wants to change the rotation of the particular Wall.

Entry Condition:

- Player starts particular game after entering the game system with id and password.

- There must be a wall to rotate in the usable walls part.

Exit Condition:*

- Event ends when user clicks rotate button.

Main Flow of Events:

- User clicks the rotate button near the wall he/she wants to change the rotation.

- Wall's rotation changes (to 90 degree counterclockwise).

Alternative Flow of Events:

- None.

Use Case 3:

Name: Verify Solution.

Participating Actors: Player.

Stakeholders Interest: The system checks whether the walls are in correct position or not, to let player to pass next level.

Entry Condition:

- Player must be playing the game.

- Player must have placed needed walls (All walls may not be needed to placed).

Exit Condition:

- If all needed walls placed in correct places, case finishes and player passes the level.

- If all needed walls did not placed in correct places, system do not let player to pass the level.

Main Flow of Events:

- System controls whether placed walls forms a solution to the level, or not.

- Game finishes and player passes next level.

Alternative Flow of Events:

- Placed walls does not form a solution.

- Current game continues.

Use Case 4:

Name: Level Succeed.

Participating Actors: Player.

Stakeholders Interest: When player places the walls to the next level, the level succeeds and player can play the next level, or see the score.

Entry Condition:

- Player must be playing the game.

- Player must place all needed wall into correct positions.

Exit Condition:

- Player must choose to continue the next level or quit game.

Main Flow of Events:

- Player must be playing the game.

- Player places needed wall to the correct positions.

- The level is succeeded.

- Screen appears to let player to choose to continue the next level or quit.

Alternative Flow of Events:

- None.

Use Case 5:

Name: Time is Up

Participating Actors: Player.

Stakeholders Interest: Player should not let to pass the level, if given time is finished.

Entry Condition:

- Player must be playing game.

- The time for the given level must have finished and player must not placed

  walls in correct positions.

Exit Condition:

- After player is shown time is up text, level fail case starts.

Main Flow of Events:

- Player didn't places walls in the correct places.

- The time finishes and the text "Time is Up!" appears.

Alternative Flow of Events:

- None.

Use Case 6:

Name: Level Failed

Participating Actors: Player.

Stakeholders Interest: Player must shown a menu after failing a particular level,

which he/she can return back to main menu, or play again.

Entry Condition:

- Player must be playing game.

-      Player does not complete the level in the given time.

-      Player wants to stop and quit

Exit Condition:

-      Player wants to quit the game and click quit button.
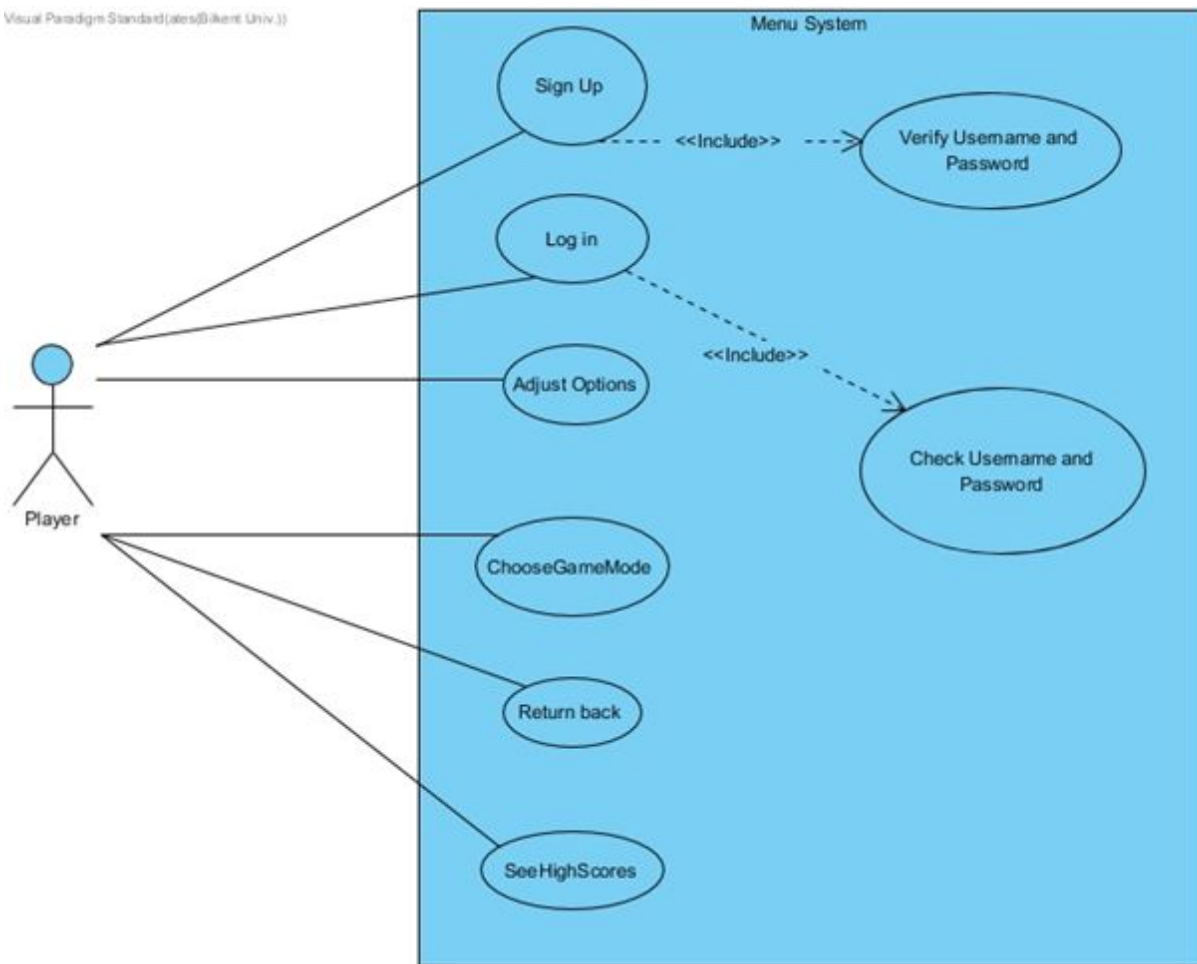
Main Flow of Events:

-      Player is trying to pass the level.

-      Given time ends, and player fails.

Alternative Flow of Events:

-      Player is trying to pass the level.

-      Player wants to quit.

5.1.2 Menu Model

Menu Use Case Diagram:

Use Case 1:

Name: Sign Up

Participating Actors: Player.

Stakeholders Interest: Players need to sign up the game, it is needed for recording their high scores and comparing the highscores with other players.

Entry Condition:

-       Player need to click the sign up button in the main menu.

Exit Condition:

-       Player takes the username and password.

Main Flow of Events:

-       Player clicks the sign up button in main menu.

-       Player writes the username and password.

-       Username and password is sent to the check.

- User signs up the game with specified name and password.

Alternative Flow of Events:

- Player clicks the sign up button in main menu.

- Player writes the username and password.

- Username and password is sent to the check.

- Username is already taken.

- Player is warned with message says that specified username is already taken.

Use Case 2:

Name: Log in

Participating Actors: Player.

Stakeholders Interest: Player need to log in the game. User accounts makes it possible to save the highscores of the players, and compare them.

Entry Condition:

- User clicks the log in button in the main menu.

Exit Condition:

- User enters the correct name and password.

Main Flow of Events:

- Player clicks the log in button in main menu.

- Player enters the username and password.

- Player clicks the enter button.

- Play game screen comes.

Alternative Flow of Events:

- Player enters the wrong username and password.

- Wrong username or password text is displayed in the screen.

Use Case 3:

Name: Check Username and Password

Participating Actors: Player.

Stakeholders Interest: Systems checks whether if player entered the correct username and password.

Entry Condition:

- User enters the username and password

Exit Condition:

- Username and password is checked

Main Flow of Events:

- System checks if the username and password are correct.

- System sends feedback to let the player to log in.

Alternative Flow of Events:

- System checks if the username and password are correct.

- System does not let the player to log in if password or username are incorrect.

Use Case 4:

Name: Choose Game Mode

Participating Actors: Player.

Stakeholders Interest: Player can choose which game mode he/she wants to play. There are 4 game modes. Arcade Mode, Time Mode, White Knight Arcade Mode, White Knight Time Mode.

Entry Condition:

- Player need to be logged in the game

Exit Condition:

- Player choses a game mode to play.

Main Flow of Events:

- Player chooses one of the game modes he/she wants to play.

- Chosen game starts.

Alternative Flow of Events:

- None.

Use Case 5:

Name: Return Back

Participating Actors: Player.

Stakeholders Interest: Player wants to return one screen back.

Entry Condition:

-       Player need to be logged in the game

-       Player clicks the return button.

Exit Condition:

-       Screen return one step back, old screen appears.

Main Flow of Events:

-       Player clicks the return button.

-       The screen from one step before appears.

Alternative Flow of Events:

-       None.

Use Case 6:

Name: See High Scores

Participating Actors: Player.

Stakeholders Interest: Player wants to see his/her high score, also wants to see who has the highest score in the whole game.

Entry Condition:

-       Player clicks the High Score button

Exit Condition:

-       Player clicks the return back button.

Main Flow of Events:

-       Player clicks the high scores button.

-       Highscore table appears. Player can see his/her high score, also can see the highest score achieved in a particular game mode.

 Alternative Flow of Events:

-       None.

Use Case 7:

Name: Adjust Options

Participating Actors: Player.

Stakeholders Interest: Player wants to change game options.

Entry Condition:

-       Player need to be logged in the game.

- Player clicks the options button.

Exit Condition:

- Player clicks the return back button.

Main Flow of Events:

- Player clicks the options button.
- In options menü, there will be two buttons, sign up button and voice adjustment button.
- Player increases or decreases the volume.

Alternative Flow of Events:

- Player clicks the options button.
- In options menu, there will be two buttons, sign up button and voice adjustment button.
- User clicks the change password button, and enters its new password.

Use Case 8:

Name: Verify Password

Participating Actors: Player.

Stakeholders Interest: Player's username and password is needed to be verified.

Entry Condition:

- Player need to click the write the desired username and password and click okay button.

Exit Condition:

- Username and password are verified or not.

Main Flow of Events:

- Player writes the username and password.
- Username and password is checked and appropriate.
- Two passwords player wrote match.

Alternative Flow of Events:

- Player writes the username and password.
- Two passwords do not match, or username is already taken.

5.2 Dynamic Models

5.2.1 Sequence Diagrams

5.2.1.1 Launching Application

Scenario: User opens the application and starts using it.

User starts using the application by going to the First Page named as Lords in Halls Page. In this page, there are two options. One goes to login page and the other one goes to sign up page. If user already has an account, s/he can directly go to login page and log in by entering username and password. After passwords controls are done, user is automatically redirected to the Main Menu Page. If user does not have an account yet, s/he can create an account by following the sign up page. In sign up, username and passwords are required to create the user. Once the account is

created, users can log in to their account and start using the application. In the Main Menu Page, user can move on to the Play, Options, Highscore and Credits pages across the application.

5.2.1.2 Playing Arcade and Time Challenge Modes with White Knight

Scenario: Player starts the white knight mode. If player chooses the white knight mode with time challenge, timer starts to count, if time is up, game restarts. Player places the walls and movement manager checks the position of walls. If red knights are inside the walls, the game restarts. If all the blue knights are inside the walls and red knights are outside of the walls, level is succeeded. If white knights are also inside the walls, level is succeeded and player score increases according to the number of white knights inside the walls.
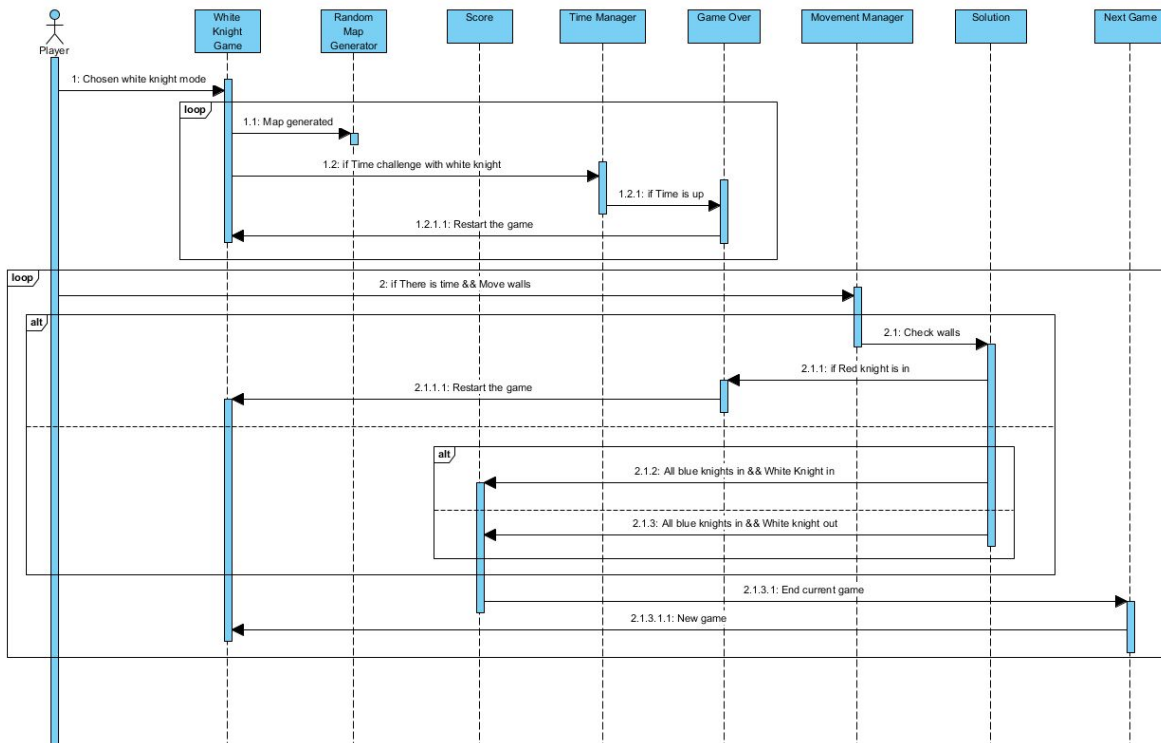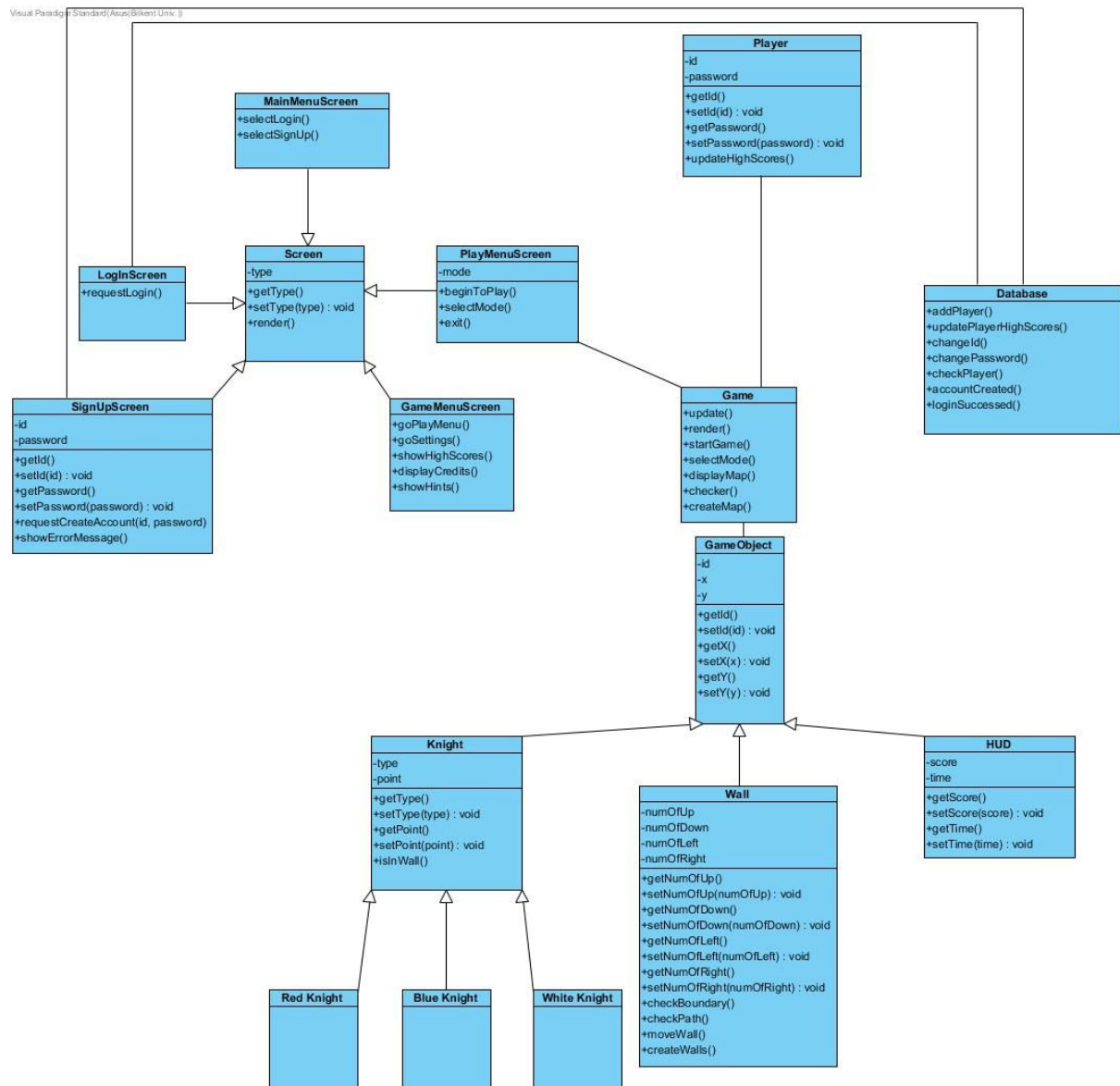
5.2.1.3 Playing Arcade and Time Challenge Modes without White Knight

Scenario: This sequential diagram shows how our game controls the game mechanics. In our application, while user is playing the game, movement will be controlled in every step according to rules such as controlling the knights positions according to their colors. Another control mechanism is about the time for some game mods. If player exceeds the time limits, then game will be ended by the control mechanism. In the if user both did not exceed time and did not cover the red knights, game will go over the next map.
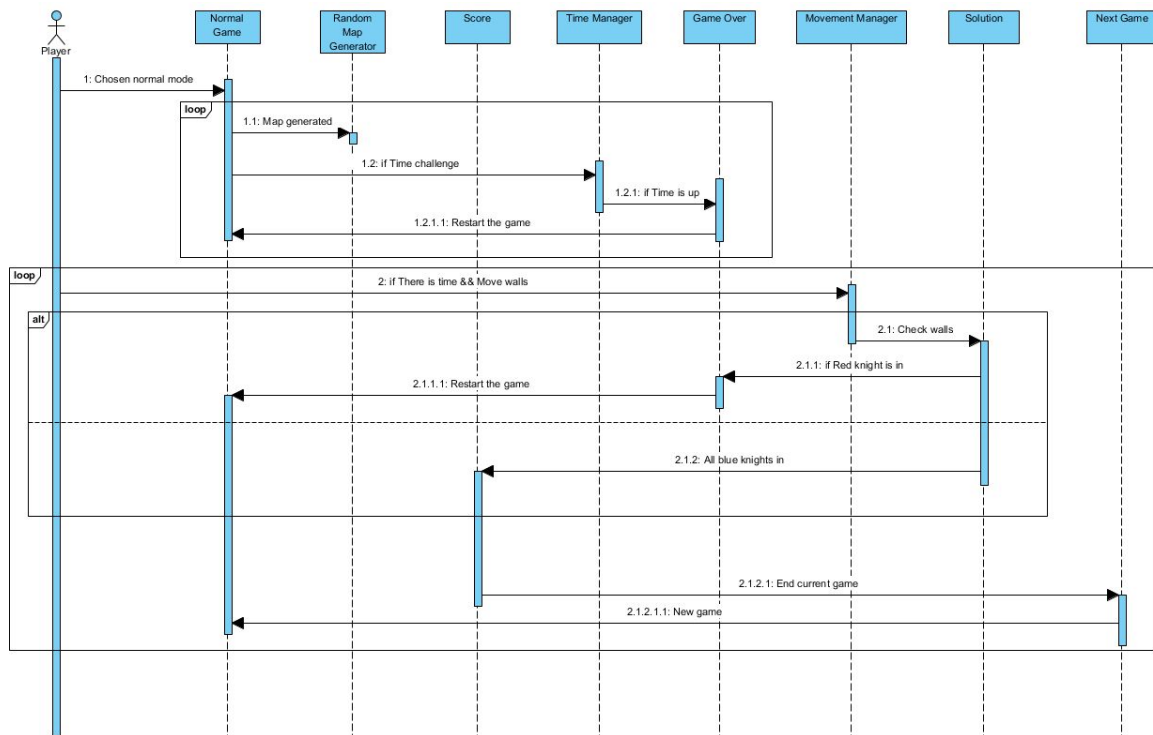
5.3 Object and Class Model

Player
-id
-password
+getId()
+setId(id) : void
+getPassword()
+setPassword(password) : void
+updateHighScores()

MainMenuScreen
+selectLogin()
+selectSignUp()

LogInScreen
+requestLogin()

Screen
-type
+getType()
+setType(type) : void
+render()

PlayMenuScreen
-mode
+beginToPlay()
+selectMode()
+exit()

Database
+addPlayer()
+updatePlayerHighScores()
+changeId()
+changePassword()
+checkPlayer()
+accountCreated()
+loginSuccessed()

SignUpScreen
-id
-password
+getId()
+setId(id) : void
+getPassword()
+setPassword(password) : void
+requestCreateAccount(id, password)
+showErrorMessage()

GameMenuScreen
+goPlayMenu()
+goSettings()
+showHighScores()
+displayCredits()
+showHints()

Game
+update()
+render()
+startGame()
+selectMode()
+displayMap()
+checker()
+createMap()

GameObject
-id
-x
-y
+getId()
+setId(id) : void
+getX()
+setX(x) : void
+getY()
+setY(y) : void

Knight
-type
-point
+getType()
+setType(type) : void
+getPoint()
+setPoint(point) : void
+isInWall()

Wall
-numOfUp
-numOfDown
-numOfLeft
-numOfRight
+getNumOfUp()
+setNumOfUp(numOfUp) : void
+getNumOfDown()
+setNumOfDown(numOfDown) : void
+getNumOfLeft()
+setNumOfLeft(numOfLeft) : void
+getNumOfRight()
+setNumOfRight(numOfRight) : void
+checkBoundary()
+checkPath()
+moveWall()
+createWalls()

HUD
-score
-time
+getScore()
+setScore(score) : void
+getTime()
+setTime(time) : void

Red Knight

Blue Knight

White Knight

Our game has 16 classes for now. This game starts with the Game class' startGame method. It basically creates the first screen in our game which is Main Menu Screen. It includes the options for logging in and signing up. If the user selects the login button, an id and a password section shows up to allow the user write down his information. If user does not have any account, an account can be made in the sign up screen.

5.3.1 Login and Sign up Screen Classes

The logic behind for login class is that this class is communicating with the database. It sends a login request to database and database firstly searches for the

account itself. After searching for the account, if it finds the account, it allows user to enter the play menu screen. If the attempt fails, it displays an error message to inform the user that the attempt failed and adds a recommendation for signing up. This loop continues until a successful attempt occurs. The same situation is valid for sign up screen. The user enters a password and an id to sign up and the database checks whether the account is creatable or not. It also displays an error message.

5.3.2 Game Menu Screen Class

After passing the entrance procedure user finally becomes be able to play the game. The GameMenuScreen class contains different options for features like settings and score table. It also contains a hint part for the game and settings choices. Credits shows the creator of this game. The most important that should be mentioned is that the game can be started by going to the play menu screen which can be reached from goPlayMenu method.

5.3.3 Play Menu Screen Class

This class is the beginning of the actual game. It contains four modes which are normal mode, normal-time challenge mode, White Knight mode and White Knight-time challenge mode. It basically sets the necessary compounds for game which are knights and walls. The logic is to make every wall and knight draw itself.

5.3.4 Game Class

This is the core logic class of our game. The adjustment of the maps, objects are made here and the modes are actually formed in there too. The selectMode method is the place where the necessary updates and render calls made. It is also responsible to check whether the given map is solved successful or not based on the rules that we decide.

5.3.5 Game Object

This class is a parent class for all game objects. It keeps the information of location and an id to distinguish the objects. It is the parent class of three subclass which are Knight class, walls and HUD(Heads Up Display).

5.3.6 Knight Class

This is the class that almost defines the rules entirely. The characteristics of the knights are very important. It is the parent class of knight's common properties basically. It has an additional knight called White Knight.

5.3.7 Red Knight- White Knight - Blue Knight Classes

The logic of the game is to place the walls such a way that no red knights will be inside the walls. However, all blue knights must be surrounded by the walls. Normally, in the original games there is just one solution for each map but we wanted to extend this idea. We added a White knight which causes the player make more points and the efficiency of the maps become important too. For example, two person with each solved identical maps, the one that solves the puzzles more efficiently, he/she gets more points than the one that has not done.

5.3.8 Wall Class

This class is to create the walls. The map's logic is created by the createMap method. It generates a closed path by game rules. This closed path is created according to positioning of knights and it assures the solvability of the map. It checks the boundary of the walls to ignore the issues. It also checks the closeness of the map and with those permissions createMap works properly. After all checks are done, the program generates the walls.

5.3.9 HUD class

This class is for the display the player's score and time for time-challenge mode.

5.3.10 Database Class

This class is to handle the database side of the project. It checks the login and sign up attempts. It prevents the identical name problem and collects some data from the users to display the top scores. It adjust the login and sign up permissions too. Apart from that, it provides a change password and change id option too.

5.4 User Interface and Screen Mockups

5.4.1 Lords in Halls Page

This page is the very first of page of our application. When we first run the app, we see this page. Since our application requires authorization with valid user, users need to log in to our account in order to play. If the user does not have an account yet, sign up is required. Thus, in this page, our application has two links. One of them is sign up and the other one is log in. They redirect user to the related pages.

5.4.2 Sign Up Page

Back

SIGN UP

User Name :   enter username here

Password :   enter password here

Password :   enter password here again

Sign Up

In sign up page, users create their account with a username and password. The application asks users to enter their password twice so that they do not enter wrong passwords unconsciously.   Same requirement is unnecessary for username since they see, while entering the username. However, the password is hidden while entering. If a valid username and password is given, and blue sign up button is pressed, new account will be created and user will be redirected to login page.   In this page, there is also a button "Back" on top left corner. It redirects to the first page.

On the other hand, if user enters a username that is already taken by someone else, obviously user cannot sign up with that username again. Therefore, the page refreshes itself and displays the error: "This username is already in use!". Usernames are unique and must be different for every user.

Similar to the error above, while creating the account if user enters two different passwords for each password area, again the page refreshes itself and shows the error message: "Passwords do not match!" Once the passwords are the same and a unique username is entered, users are redirected to the login page, to login with their account.

5.4.3 Login Page

The login page is very simple. Basically, it requires users' username and password to log into their account. If the username exists in the database and the password corresponding to that username is entered correctly as well, users are redirected to Main Menu page. In this page, there is also a button "Back" on top left corner. It redirects to the first page.

In this page when the login process failed, there could be two reasons of it. First is, if users enter an invalid username, which does not exist in our database, the application refreshes the page and displays the error message: "No such user exists!" The reason of failure here is not about the password but still password might be wrong as well. The application only states that the given username is not proper.

The second reason of failure is password. Once users enter a username that exists in the database, application checks the database and pulls the password data. If the password in the database for the given username and the given password do not match, log in page refreshes itself and displays the error message: "Wrong Password!" In this case, we definitely know that there exists an account in the database with the same username entered by the user. Only the passwords are different.

5.4.4 Main Menu Page
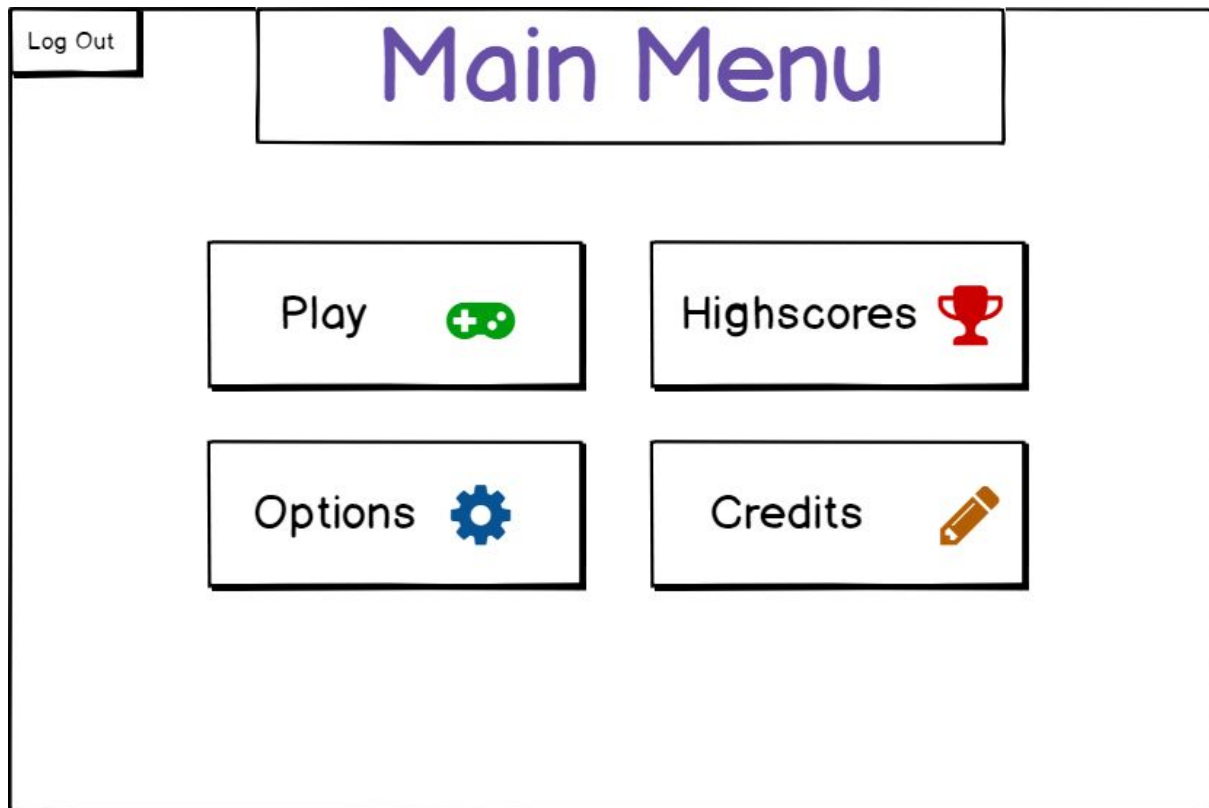
After a successful log in process, the application recognizes the user as an authorized user and main menu page opens up. In this page, users can go to four different pages.

1.   Play Menu Page
2.   Highscores Page
3.   Options Page
4.   Credits Page

The main function of this page is to offer the different options that our application provides to authorized users. They can start playing the game, they can check High Scores for games, they can go to options and they can check credits. Moreover, there is another button "Log Out" on the top left corner. It logs out the current user and redirects to First Page.

5.4.5 Play Menu Page
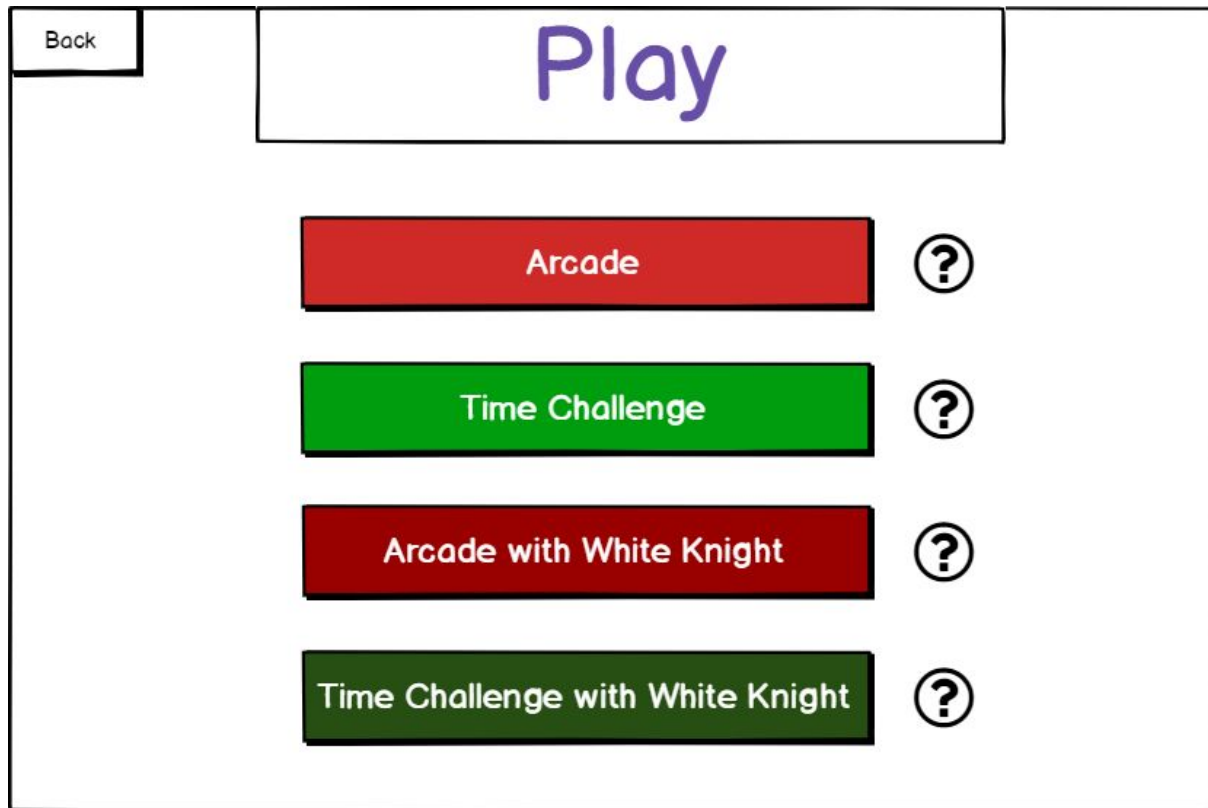
After users click on the "Play" button in the Main Menu Page, they are redirected to Play Menu Page. In this page, our application provides four different game modes of our game Lords in Halls. Every one of the game modes are well explained before. By clicking the game mode buttons, users can move on to the game pages and start playing instantly. There also question mark buttons next to each game mode. These buttons are there to explain that specific game mode in detail to the users so that, they learn how to play. For that, these buttons redirects to How To Play Page. There is also another button "Back" on the top left corner. It redirects to the Main Menu Page.

5.4.6 How To Play Page
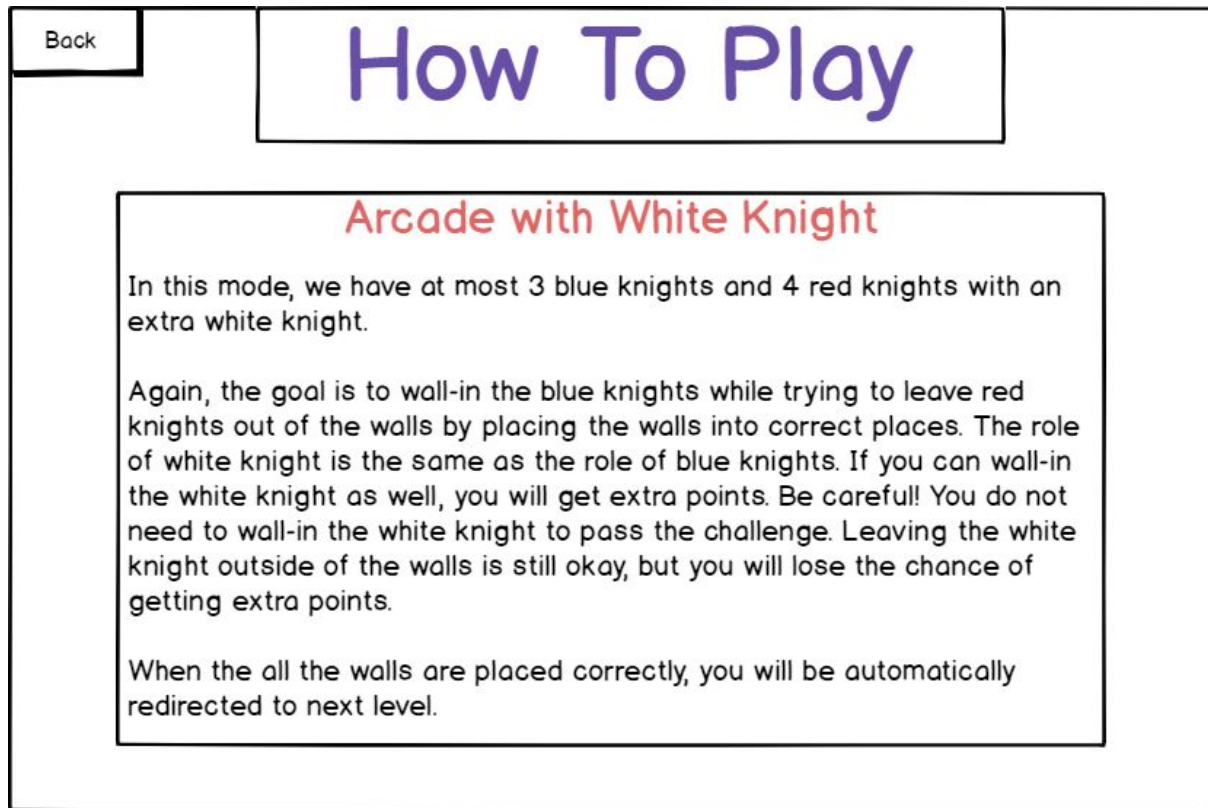
How to play page is one of the simplest pages in our application. It does nothing but displaying a text message. The text message provides detailed information about the game next to the question mark button, which users clicked. There is also a button "Back" on the top left corner, which redirects to Play Menu Page.

5.4.7 Highscores Page
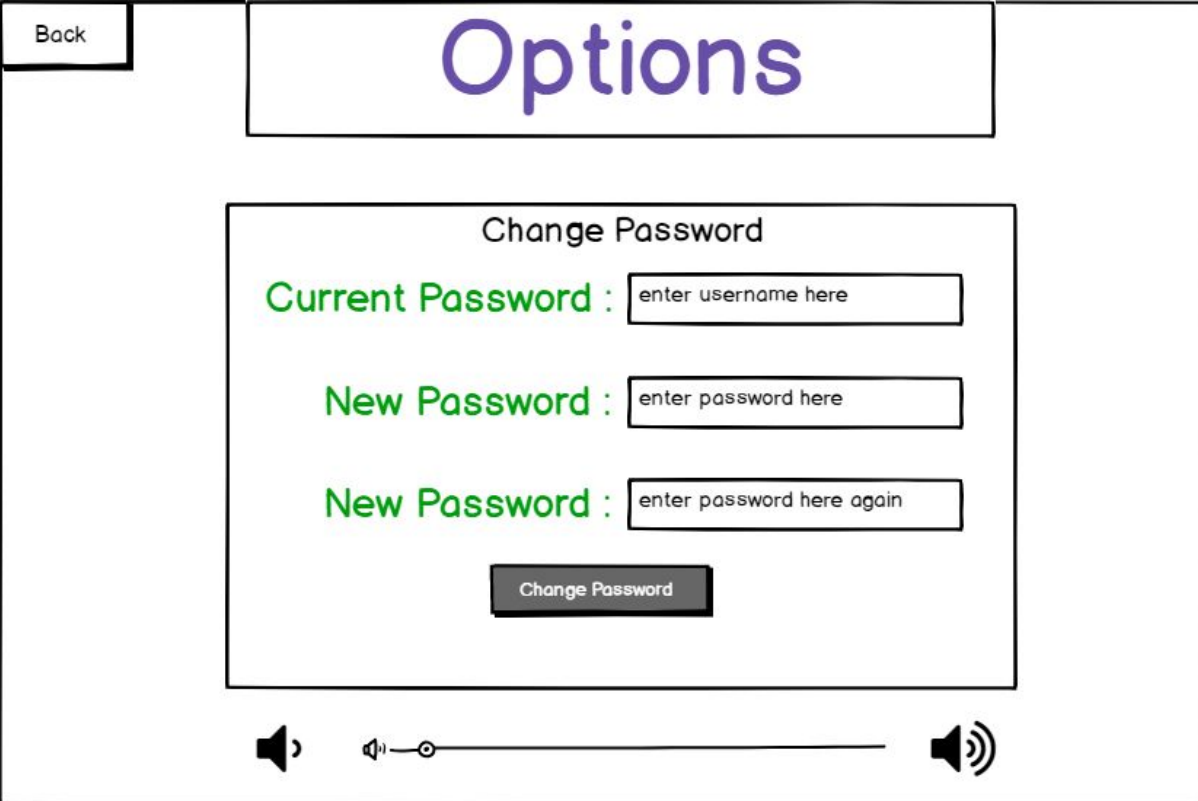
In Highscores page, we have four different Highscores tables for each game mode we have. For each game mode, we have a different highscore table in the database. The tables are always updated and displayed here. The reason we have different highscore tables for different games is that the score calculation of each game is different. Thus, it would be complex to merge them into one table. That is why each of them are displayed separately. There is also a button "Back" on the top left corner. It redirects to the Main Menu Page.

5.4.8 Options Page

Options page, as the name implies, provides the options of application. For now, it only has the option of volume up and down but it is open to development for future. Besides the options of applications there is another area that enables users to change their passwords if they wish to. To do that, the current password must be given with the new passwords for security. If current password is correct and new passwords are matching, password changing request will be sent to server and the password will be changed.

Password changing process might fail again because of two reasons. One of them is user enters the current password wrong. If the password user enters, differ from the password that is stored in the database for the current user, application rejects the password change request, refreshes the options page and displays the error message: "Wrong Password!" In this case, the new passwords are not important since password changing is not executed.

The second reason of password changing failure is the mismatch of the new passwords. If user enters the new passwords different, password changing request will be rejected. Again, application will refresh the page and display the error message: "Passwords do not match!" One important thing to remember, if user sees this error, he/she knows that the current password is entered correctly because the previous error has priority over this error. First, the current password is checked, once it is proven correct, the new passwords are checked. When everything is in order, the password changing request will be sent to server.
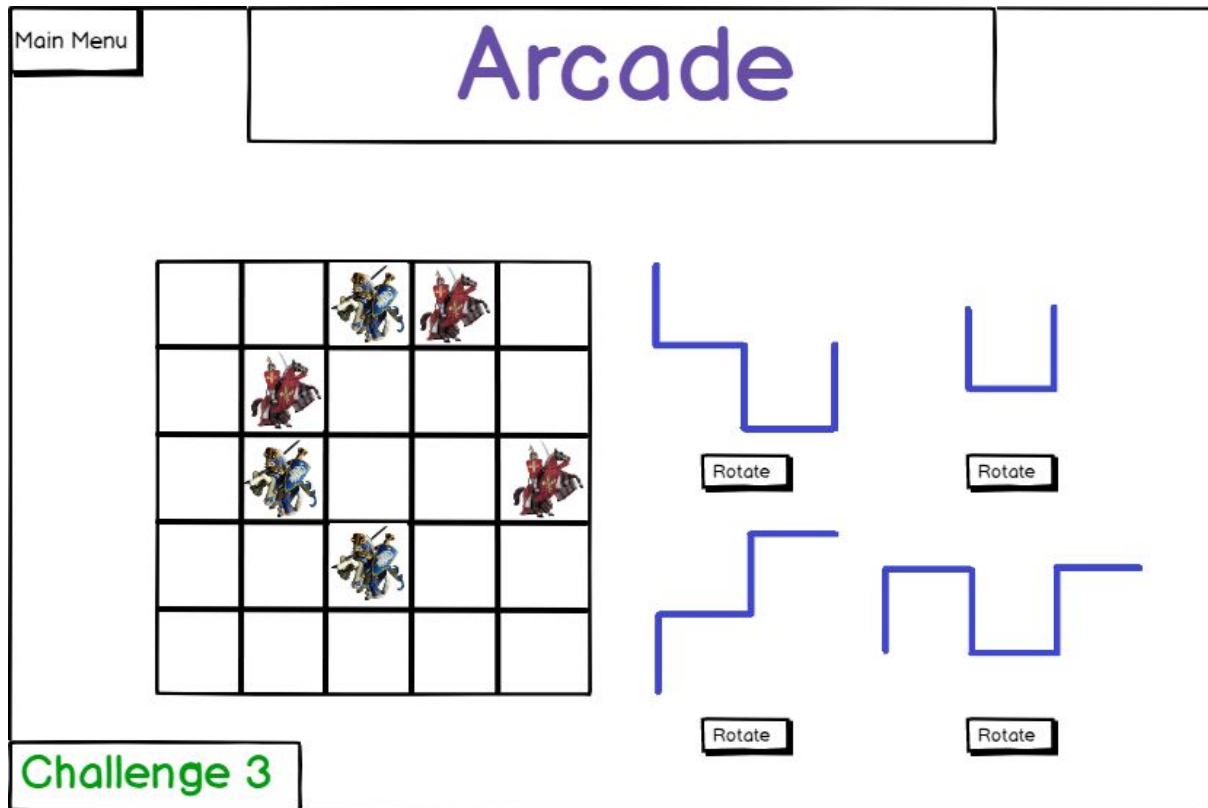
5.4.9 Credits Page

Credits page is simple as it is. It displays our project name, developers' names, our group name and two cute knight figures.
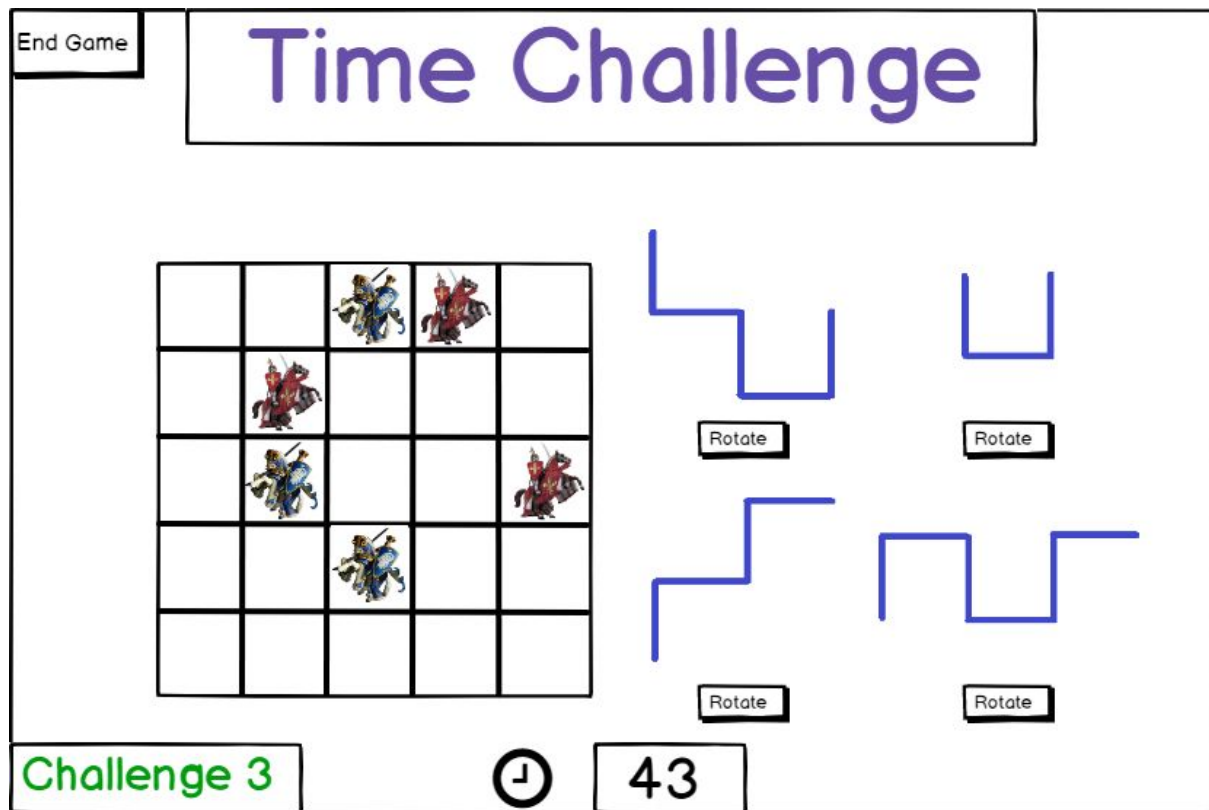
5.4.10 Play Arcade Page

This is the gameplay page of arcade mode. When users click on the "Arcade" button on the Play Menu Page, they are redirected to here and start playing. The game board is positioned on the left side of the screen. Game board consists the blue knights and red knights. The walls are on the right side. Walls can be placed on the board by drag and drop. There is a "Rotate" button under every wall piece. Before placing the walls, players can rotate them in order to get the walls into correct positions. On the bottom left corner, players can see the challenge they are on.

There is also another button "Main Menu" on the top left corner of the page. It leaves the game and redirects to Main Menu Page.
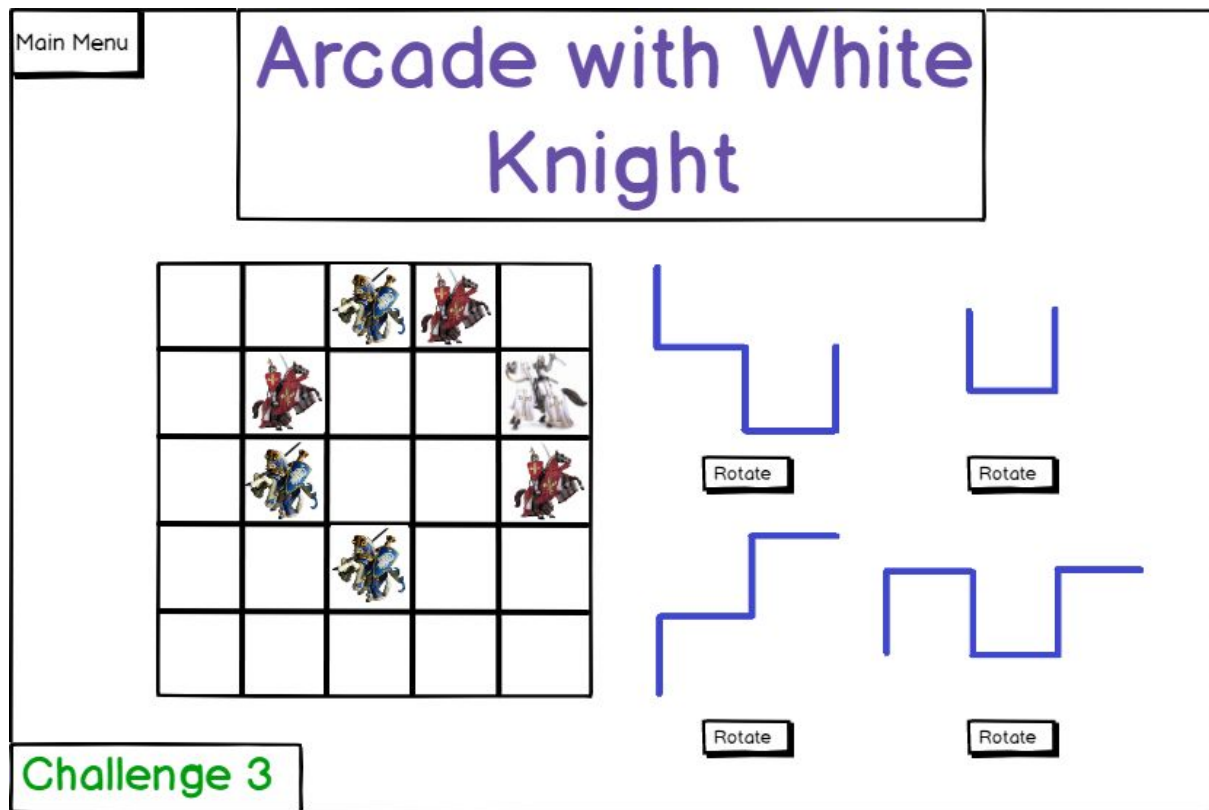
5.4.11 Play Time Challenge Page

This is the gameplay page of time challenge mode. When users click on the "Time Challenge" button on the Play Menu Page, they are redirected to here and start playing. The game board is positioned on the left side of the screen. Game board consists the blue knights and red knights. The walls are on the right side. Walls can be placed on the board by drag and drop. There is a "Rotate" button under every wall piece. Before placing the walls, players can rotate them in order to get the walls into correct positions. On the bottom left corner, players can see the challenge they are on. Next to it, there is a timer. Since this is a time challenge, players should solve the challenge before the timer reaches 0. When the timer reaches 0 before the challenge is solved, the game will be over.

There is also another button "Main Menu" on the top left corner of the page. It leaves the game and redirects to Main Menu Page.
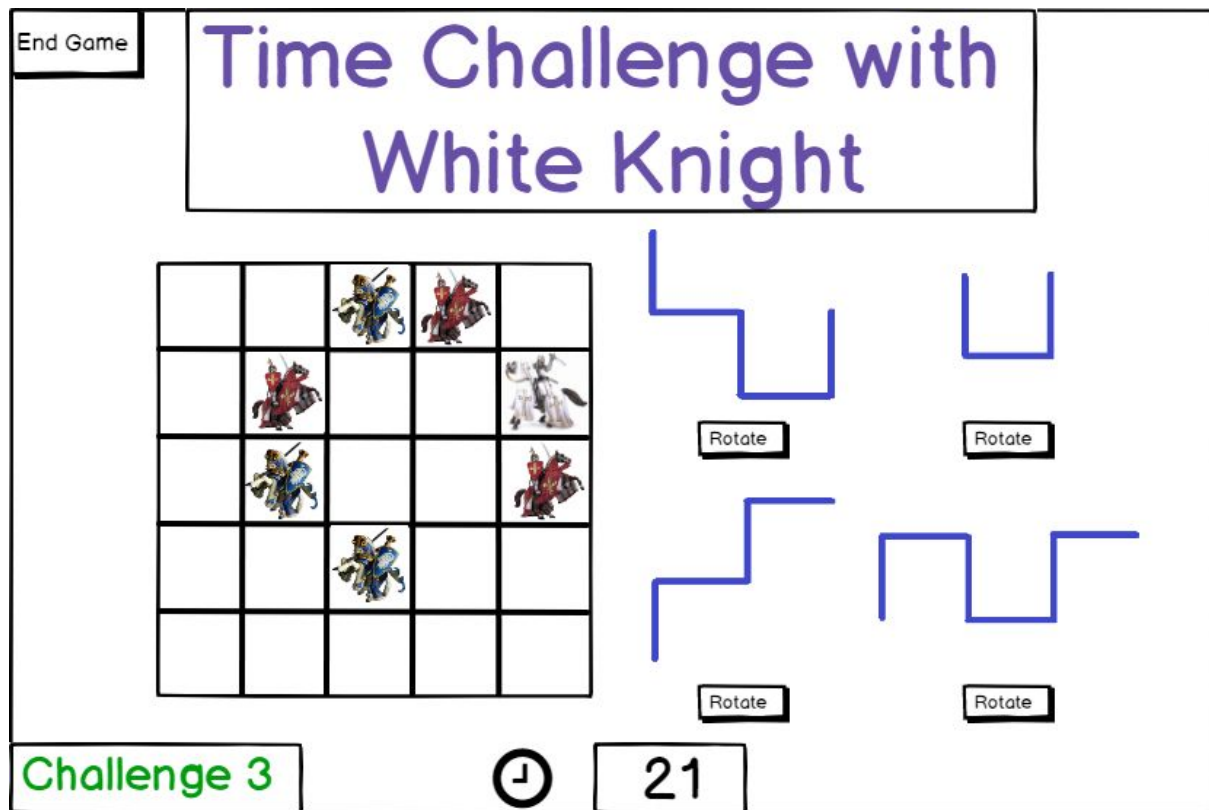
5.4.12 Play Arcade with White Knight Page

This is the gameplay page of arcade with white knight mode. When users click on the "Arcade with White Knight" button on the Play Menu Page, they are redirected to here and start playing. The game board is positioned on the left side of the screen. Game board consists the blue knights, red knights and an extra white knight. The walls are on the right side. Walls can be placed on the board by drag and drop. There is a "Rotate" button under every wall piece. Before placing the walls, players can rotate them in order to get the walls into correct positions. On the bottom left corner, players can see the challenge they are on.

There is also another button "Main Menu" on the top left corner of the page. It leaves the game and redirects to Main Menu Page.

5.4.13 Play Time Challenge with White Knight Page

This is the gameplay page of time challenge with white knight mode. When users click on the "Time Challenge with White Knight" button on the Play Menu Page, they are redirected to here and start playing. The game board is positioned on the left side of the screen. Game board consists the blue knights, red knights and an extra white knight. The walls are on the right side. Walls can be placed on the board by drag and drop. There is a "Rotate" button under every wall piece. Before placing the walls, players can rotate them in order to get the walls into correct positions. On the bottom left corner, players can see the challenge they are on. Next to it, there is a timer. Since this is a time challenge, players should solve the challenge before the timer reaches 0. When the timer reaches 0 before the challenge is solved, the game will be over.

There is also another button "Main Menu" on the top left corner of the page. It leaves the game and redirects to Main Menu Page.

5.4.14 Game Over Page

In the time modes, when the timer reaches 0 before the challenge is solved, the game will be over and players will be redirected to this page. This page displays the score. Using the "Back to Main Menu" button, users can go back to main menu and continue enjoying the application.

One thing to remember that, this page is not displayed after arcade mode games. Because these games never end unless the user wants to end.

6. Conclusion

In this analysis report, we analyzed our project for the first iteration. This report contains 5 main parts with detailed description in each of them. The first part is the introduction of the project. It describes the origin of our game Walls & Warriors and how we developed a new game from originating Walls & Warriors game.

Second part of the analysis report is the overview of the project. In this part, the detailed explanation of the improvements and the fundamental aspects of our version take place. What we have introduced with Lords in Halls, what is there to discover, what is expected and what is provided is described in every detail. The provisions of gameplay and user interactions are defined in this part.

In the third part, functional requirements are listed. In this part, we listed down all the functionalities a user can perform. Being understandable and crystal clear is number one priority here. Although we clarified all the functionalities that we are planning to implement here but there is always some left behind for improvement. Similarly, we might give up implementing or change the structure of some functionalities in the upcoming reports or demo. We will do our best.

In the fourth part, we described non-functional requirements of our projects. These requirements are basically non-technical but charming features of the projects. Performance, interface, user-friendliness and security issues are discussed here. This part is also open for development and might change in the upcoming reports or demo as well.

In the final part, we tried to explain the work chain of our application with visuals. All of the sequence and class diagrams are placed here. We explained the diagrams with example scenarios and how the user-application-server interactions will be done in terms of functionality. With object and class diagram, we also clarified the objects and abstract materials which took place in the game in terms of their structure and how they interact with game, user and server within themselves.

All in all, this report is the detailed description of what are we going to implement and preparations we have made so far.

7. References

(1) "Google Chrome Version History." *Google Chrome Version History - Wikipedia*, www.wikizeroo.com/index.php?q=aHR0cHM6Ly9lbi53aWtpcGVkaWEub3JnL 3dpa2kvR29vZ2xlX0Nocm9tZV92ZXJzaW9uX2hpc3Rvcnk.

(2) "The New, Fast Browser for Mac, PC and Linux | Firefox." *Mozilla*, www.mozilla.org/en-US/firefox/.

(3) Irons, Sam. *Opera Software AS - Opera Version History*, www.opera.com/docs/history/.

(4) "Balsamiq Cloud." *Balsamiq Cloud*, balsamiq.cloud/.