Object Oriented Software Engineering

CS319

Term Project

Final Report

Kubitz

Group No: 3F

Group Name: Wasted Potentials

Mertkan Akkus 21602951

Yasin Alptekin Ay 21601849

Ahmet Furkan Bıyık 21501084

Ramazan Mert Cinar 21601985

Yaman Yagiz Tasbag 21601639

1. Introduction

We have started the implementation of Kubitz when we were done with the design phase. All the group members used the IntelliJ IDEA IDE to implement the different parts of the project. We split the work in basically three sections which are server-side, main logic of the game and graphic user interface. While these three main parts are being implemented by different group members, github has been used by us to track the work done. The implemented functionalities, design changes, what we have learnt and a simple user guide will be provided in the rest of this report. At the moment the implementation phase has ended. We managed to achieve the most of our promises in the implementation. In the second iteration we were thinking about adding more game modes than we did in the first iteration, but there wasn't enough time left because of making the other parts work perfectly. We were also supposed to add a spectator mode where the players will be able to watch other players' game, but it also hasn't been satisfied in time. However, even these unfortunate occasions come up, we have done finishing the multiplayer game modes, optimization of the game, adding more musics and sound effects, polishing game controls and adding alternative controls as promised in the second iteration.

2. Implemented Functionalities

We have aimed to implement the highest priority promises that we made and a smooth User Interface that will support these promises in the implementation. Right now, our system provides the functionalities below:

- Singleplayer Game Modes
- Multiplayer Game Modes

- Navigation through menus & all menu interactions
- Configuring graphic and sound settings. User specified settings are saved to a file and read from the file
- Tutorials for each of the game modes
- Lobby system with all functionalities including chat and invite feature
- Server controllers and database usage
- Added musics and the sound effects
- Exhibition of credits
- Automation of builds
- Game Themes
- Key Bindings

3. Design Changes

At first we planned how to design the project and then we moved on to write the design and analysis reports according to these reports. However, when we started to implement the project we saw that the design is not completely accurate. Therefore, we decided to redesign and improve the system. As the implementation continues, different ideas come up and this makes the code open for changes at every step. The changes which occurred after the design phase and during the implementation are stated in the sections below.

3.1 Low Level Changes

Some of the classes were not sufficient to implement the project so we have changed some of the classes and added new classes. For instance we did not have a lobby class at our structure design stage. However we needed to implement such a class so that

implementing the UI and would be easier. In addition, we changed some classes' constructors since we need some properties in objects for example we added Config parameter to SettingsScreen's to get or set configuration. We also added new methods in some objects in order to improve the code in terms of readability and reliability such as initializeResources method to screens to make constructor be shorter.

Other than the changes in the classes, we also decided to relocate the server side of the game from our systems to Azure which provides an online Virtual Machine service. This decision has improved the game performance, connection speed and communication between the server and client sides.

We have reworked the GUI management system. We extended the management system so that each screen now has a lifecycle.

We implemented a theme system for which we created a theme class and a theme manager class. We can easily add or remove color themes via theme manager.

The default UI was looking ordinary and boring for some components. So we extended the UI of some components such as tabbedPane and scrollBar to make them more appealing to the user.

We implemented a new feature, key bindings. For this we added a new screen and a config file for key bindings.

3.2 Gameplay Changes

Some of the game modes were not solid at the design state. Such as, how many game cards will be given to the users as objective in the classical game mode, how many seconds are needed to switch the boards in the switch game mode and how many extra

seconds will be given as a bonus to the players after a card has been solved in survival mode. Some minor changes have also been applied such as the theme song of the game.

4. Lessons Learnt

During both implementation and design phase, we have encountered some problems about being synchronized and learned how to overcome those by experiencing it. First of all, it was a challenge for us to be available at the same time to arrange a meeting and use that time efficiently. Therefore we have decided about the issues to tackle prior to meeting and also about the dedicated time for the meeting. It was another task for to divide the project into subtasks and distribute them amongst the team members. We have realized that there were some overlapping subtasks but still, we have managed to handle them and also learned to do better.

To be synchronized, we are using GitHub as a git service and using many functionalities of it. For instance, we are using the issue tracker and pull request system of the git service. We are adding the issues of any type, e.g. bug, error, to the system and mark them as resolved when it is fixed. We are creating a branch for major features and merge them with the master branch. We believe that those functionalities of git service increases the productivity.

Hence, we think that teamwork is much faster and professional when it can be managed. The team members should be communicating frequently for a better product.

5. Build Automation

We have constructed a build automation both for the deployment of the server and for the deployment of the executable jar. We are hosting a web server in which there is the

executable jar of Kubitz - it is going to be fully served from 26.12.2018 . The deployment scheme can be found in figure 1.1.

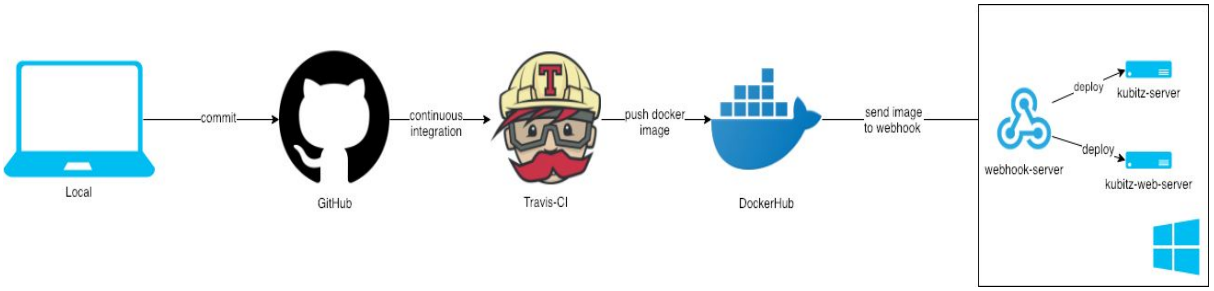

Figure 1.1

The scheme above explains the build automation. We have used Travis-CI, Docker, WebHook and Microsoft Azure to automate build. When one makes a commit on GitHub, the script that we have written executes on Travis CI servers and it builds a Docker image, then pushes it to the DockerHub. Once it is pushed, it sends a POST request to the WebHook server that we are hosting in Microsoft Azure. Finally the WebHook server in Azure pulls the image from DockerHub and builds the executable jar and hosts the Kubitz server along with the executable jar.

6. User's Guide

6.1 System Requirements & Installation

There are 3 ways to obtain the Kubitz executable. One is building the jar file automatically and the other one is downloading it directly from web - second one will be available from 26.12.2018 and the URL will be provided, then.

To download the executable file from web, one should go to the Kubitz download page. We are going to be providing this service by 26.12.2018. A user will be able to obtain the link inside README.md file on GitHub repository

(https://github.com/cinarmert/WastedPotentials-319). For the users who do not have Java installed on their system can follow this link (https://www.java.com/tr/download/) to download Java Runtime Environment (JRE).

Another way to get the executable file for Kubitz is download directly from the Github Releases (https://github.com/cinarmert/WastedPotentials-319/releases). We will be versioning the releases there. Again one has to have the JRE on the system to execute the file. Follow (https://www.java.com/tr/download/) for download instructions.

Last way to build the Kubitz executable file manually, one has to download Maven and Java Runtime Environment (JRE). Since our game occupies so little space, players shouldn't have concerns about memory. To install maven the instructions on the Apache's website should be followed (https://maven.apache.org/install.html). For the users who do not have Java installed on their system can follow this link (https://www.java.com/tr/download/) to download Java. Once a user has both of them on his/her system, he/she should download the source code from the GitHub repository (https://github.com/cinarmert/WastedPotentials-319). Finally the user should run the command "mvn package" from the command line. After the command finishes running, the only thing that users need to do is executing the "*Kubitz.jar*" file by double clicking it which lies in the target directory.

6.2 How to Use

6.2.1 Selecting Multiplayer Game Mode

To play multiplayer game modes, a user needs to navigate to play menu by clicking the "Play" button in the main menu. This will take the user to play menu in which player will choose multiplayer or singleplayer. After clicking "Multiplayer" button, the user will be

taken to the Lobbies screen. After this point, the user can either choose the multiplayer game mode by choosing the lobby with the game mode from the lobby list or create a lobby. In lobbies screen, player can also filter lobbies. By pressing “Filter” button, the player will be taken to lobby filter page in which he/she can specify desired lobby features. Lobbies screen can be seen in the figure 1.1.

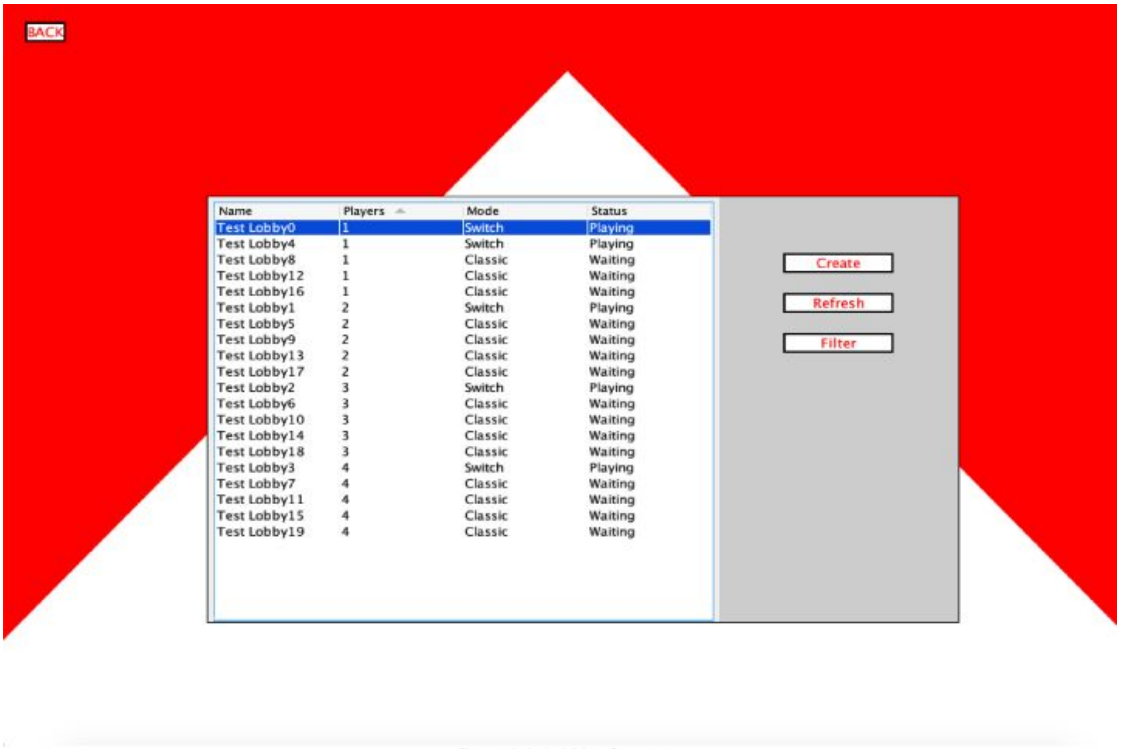

Figure 1.1: Lobbies Screen

6.2.1.1 Joining Lobby

In order to join a lobby the precondition is that the player must be in lobbies screen for which the instructions are given in section 5.2.1. For joining a lobby, the player should only double click on a lobby which he/she wants to join. This will take the player to the lobby screen.

6.2.1.2 Creating Lobby

In order to create a lobby the precondition is that the player must be in lobbies screen for which the instructions are given in section 5.2.1. To create a lobby the player

should click on “Create” button in the lobbies screen which will take her/him to lobby creation screen. In this screen, player must enter a name for the lobby, choose the game mode (from switch and classic game modes), set the lobby size (for switch mode maximum is 2, for classic it is 4). If the player checks the “Private Game” checkbox, other players will not be able to join the player’s lobby from the lobbies screen thus, players will be able to join the lobby only via invites that are sent by any player in that lobby. After clicking “Create Lobby” button, the user will be taken to the lobby screen

6.2.1.3 Lobby Screen

This screen is displayed after the players joins a lobby either by creating or selecting from the list. There are 3 functions that can be used by the lobby admin (creator of the lobby) and one function that every player can use. The lobby admin can start the game by clicking “Start” button and kick a player by choosing the player by clicking on his/her name and then clicking “Kick” button. The lobby admin can also change the lobby settings by clicking “Settings” button. This action will take the player to lobby settings screen which is similar to lobby creation screen (see 5.2.1.2 for lobby creation instructions). Each player in the lobby can invite another player by typing his/her name to the textfield. After clicking the “Invite button”, an invitation to the given player will be sent. There is also a simple chat system implemented in the lobby screen. Players can enter a message and send by pressing enter button on their keyboard.

6.2.2 Selecting Singleplayer (Survival / Daily Challenge) Game

To play singleplayer game modes, a user needs to navigate to play menu by clicking the “Play” button in the main menu. This will take the user to play menu in which player will choose multiplayer or singleplayer. A player should press the Survival or Daily Challenge

button to play a singleplayer game. The game screens appears immediately after pressing the related buttons. See section 5.2.3 for how to play the game.

6.2.3 Playing The Game

Players will be on the play screen after the game starts. At the top right corner of this screen there is a game card as the objective, a cube sample at the middle left and the grid at the middle right. Players should arrange the cubes in such an order that will match the game card above to complete the objective.

The arrangement of the cubes can be done by the Q, W, E, A, S, D keys from the keyboard. By hitting the W, A, S, D keys user can change the faces of the cube up, left, down, right respectively and by hitting the Q and E keys player can rotate the cube left and right respectively.

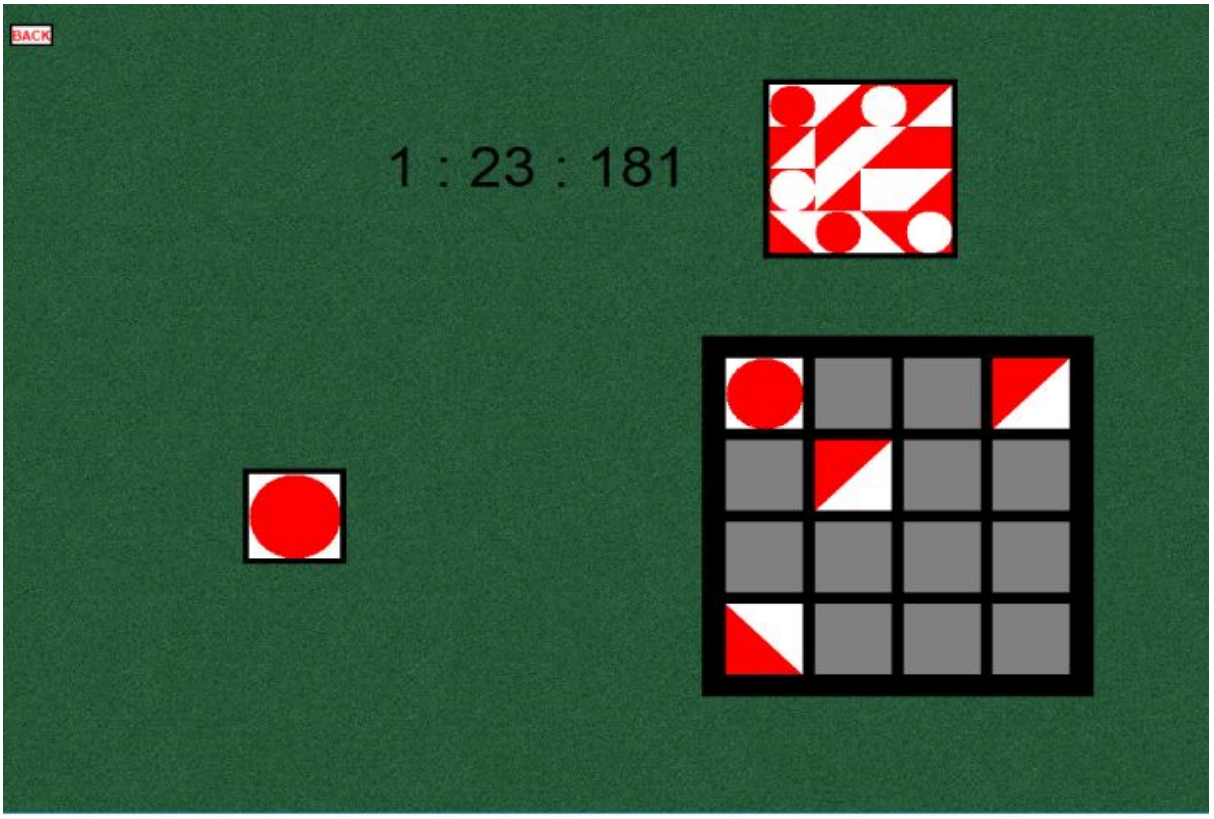

Figure 1.2:sample Gameplay

6.2.4 Guide for Settings

Players can access settings screen by clicking settings button in the main menu. In the settings screen, players can change the randomly generated nicknames by entering a nickname and clicking submit button after writing the desired nickname.

Another setting that players can arrange is the resolution of the game. Below the Graphics section player should select the suitable resolution from the drop down button, they can also select the Full Screen option.

The last part of the settings screen is the sound options, which includes the master volume, effects volume, music volume. Player who wants to configure the sound should scroll the desired volume bar - scroll right to volume up, left to volume down.

After completing configuring the settings, player must save the settings or click cancel to keep the settings as it was before.

7. Work Allocation

- Yaman Yağız Taşbağ
  - Implementation of logic
  - Design choices regarding GUI
  - Diagrams & Reports
- Mertkan Akkuş
  - Implementation of GUI
  - Extension of controllers
  - Game sounds
  - Diagrams & Reports
- Ramazan Mert Çınar
  - Implementation of server
  - GUI-Server communication
  - Continuous integration & deployment
  - Design choices regarding GUI
  - Diagrams & Reports
- Ahmet Furkan Bıyık
  - Implementation of GUI

    - Implementation of GUI - Logic Connection
    - GUI-Server communication
    - Diagrams & Reports

- Alptekin Ay
    - Requirement Analysis
    - Diagrams & Reports