CS319 Object Oriented Software Engineering Project

Final Report

Iteration 2

IQ PUZZLER PRO

GROUP - 3G

Fatih Çelik

Cenk Er

Enes Yıldırım

Eren Yalçın

Burak Bayar

Ebru Kerem

1. Introduction

Our solo part of the game is currently complete. All the levels are not prepared yet, but the functional requirements are met. A player can play the game like they could on its board version; the logic part of the game is successfully implemented. Additionally a player can also use hints in our application to ease the game. The user can also create a custom map by removing selected parts from an already complete puzzle; which is given randomly from a set.

We have solved issues from the first iteration too. We have solved some key issues with the GUI elements of the system. For example, we had a flickering problem while dragging our pieces, we have solved this by enabling double buffering for our application. The pieces are now much easier to drag and drop as intended. Additionally, the pieces are not leaving (broken) graphical tracks on the form.

What is left out?

The online part of the game is left out for the most part. There is currently no custom map browsing, custom map sizes, score system and custom piece selection for map creation. Regarding the solo part, the game is yet to be tested by others to make it more user friendly and the levels are limited. There are no levels for the second type of solo puzzles, as there are no inclined pieces yet.

2. Design Changes

Our menu and GUI design changed a lot since the first iteration. Our game in the first iteration does not work properly and we could not combine GUI and menu part. However,

now we changed the form structure make every form connected with each other. In GUI part, our game did not have some features like getting hint, keeping time or refresh the map. We added those features to make game playing more desirable.The hint option gives user to chance to see some puzzle pieces in inventory which selected as a hint in the map. According to difficulty of the level, we decided to give various number of hints.For example, if the user is in the easy level, we give them only 1 or none hint. More the level gets difficult, we give more hints. However, the maximum number of hint for even hardest level is three. Refresh map provides to restart the game. All the puzzle pieces which are initially in the inventory come back initial locations, hints are updated and the time restarts. In the business layer of the system, some design patterns were not possible to implement as intended, hence other kind of solutions are used in the implemented application. For example, our game manager class was supposed to be master controller class that deals with most logic functions. However, many functions like moving pieces, checking collisions, checking completion are implemented in the form classes, without interacting with the game manager. This leads us to new approaches and new ways of coding. Although we said that we will implement custom maps with user selected size, we decided that the map with constant size which contains initially all puzzle pieces in it is more easy to implement from the point of the player. Therefore, we changed the custom map creation logic. In the map builder classes, we implemented the map according to it’s array and we did not use puzzle pieces. Instead of puzzle pieces, we just drew the picture. However, we realized that if we implement the map with puzzle pieces, it will give us a lot of advantages like deciding whether game is over or not. Thus, we changed the design of the map classes and we used the puzzle pieces to implement map classes.

3. Lessons Learnt

This project was a great opportunity for us to experience the software engineering project stages. We have observed how some of the typical software engineering project management issues occured in our team. We failed to prepare for some of those issues and it decreased our momentum immensely. One of the bigger issues we have experienced is the importance of visualising the same solution pattern by every stakeholder in the project, which is satisfied by communication. We experienced how hard it is to discuss a solution or a problem without proper communication ways like using diagrams. Even for what it seems to be a simple idea to tell, it was very time and energy taking for everyone to understand it without a proper communication way. Some of the typical software engineering project management issues we have observed can be listed as:

Lack of Organizational Structure and Accountability: Even though we were a small group of 6, we failed to assess a proper organizational structure. Each of us had an unique skill set which fits to different parts of the project. But we failed to properly organize and assign duties between each other. We have experienced how hard it was to track functional requirement solutions by a result of this. We have learnt that while it is energy and time consuming to organize a proper structure for a project, it is essential for the project to satisfy the requirements.

Lack of a Stable System Design: We were inexperienced on how to design a software project, which led to unstable designs for our project. We also did not exactly know what the capabilities and incapabilities of C# for designing a game. We insisted to exclude game development frameworks at first, but we encountered some

problems in GUI development which led us to try using frameworks in our project. Hence, we were unable to stick to our system design. Without a documented system design, it was very hard for all of us to work together on the implementation. Thus, we learned that it is important to have a realizable and complete system design.

Unstructured and hurried software development: Our group had great ideas for the project, we had a potential for a great game. However, most of these ideas did not stay alive until implementation. It is not necessarily bad for an idea to be scrapped, but we were also scrapping great ideas to hurry up our game development. We could implement at least some of these ideas with a much more organized project lifecycle.

4. User's Guide

4.1 System Requirements

- Windows 7 or better.
- 1,8 GHz or a faster cpu.
- 2 GB RAM or more.
- 4 MB disk space or more.
- At least 480p resolution with 24 fps screen.

4.2 Installation guide

To play IQ Puzzler Pro, the player has to download all the related folders and files from the provided source and execute the executable file from the "obj -> Debug -> IQPuzzlerProVer2.exe"

4.3 How to play

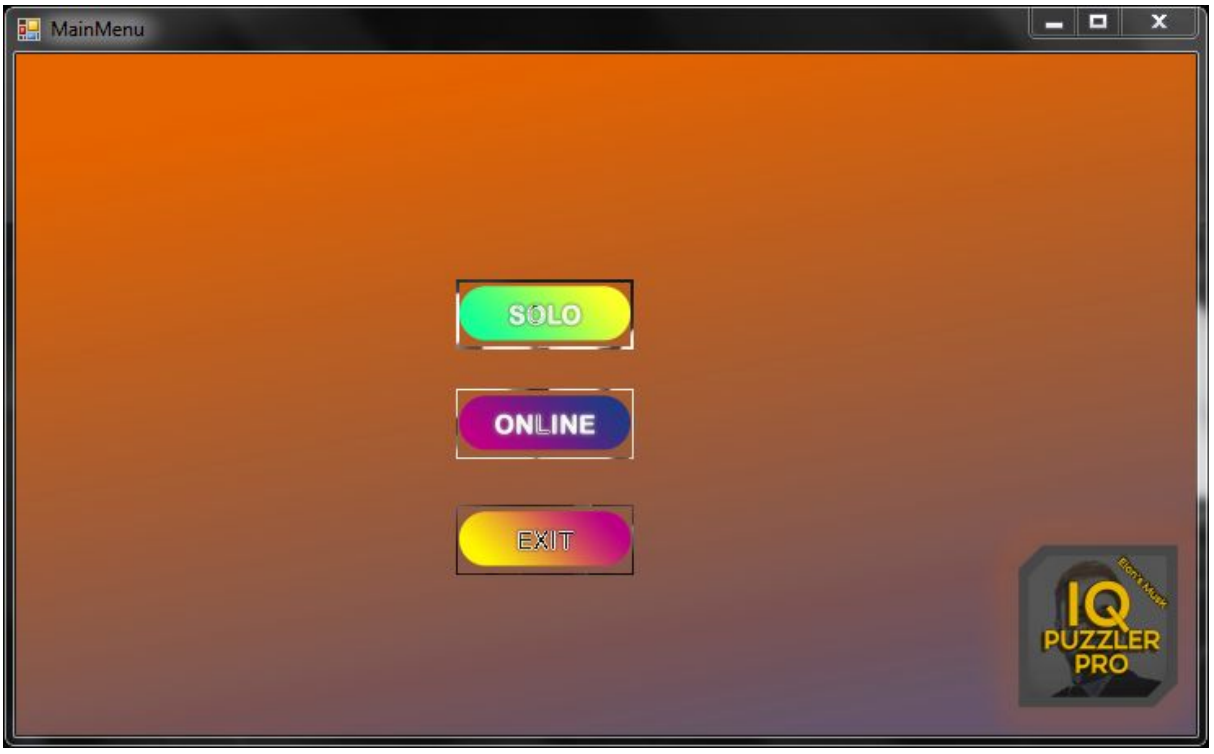

Figure 1: Main Menu Screen

Game starts with a Main Menu with 3 options. Two of them are for playing games and one of them is to exit. By clicking SOLO Button, user is directed to Difficulty Level Screen. On the other hand, if the user clicks ONLINE Button, login

panel will be shown. Also, user can exit by clicking EXIT Button.

Figure 2: Solo Difficulty Level Selection Menu

In this screen, user will have a chance to select among 3 different difficulty levels: easy, medium and hard. At the bottom, there is a return menu to go back to main menu screen.

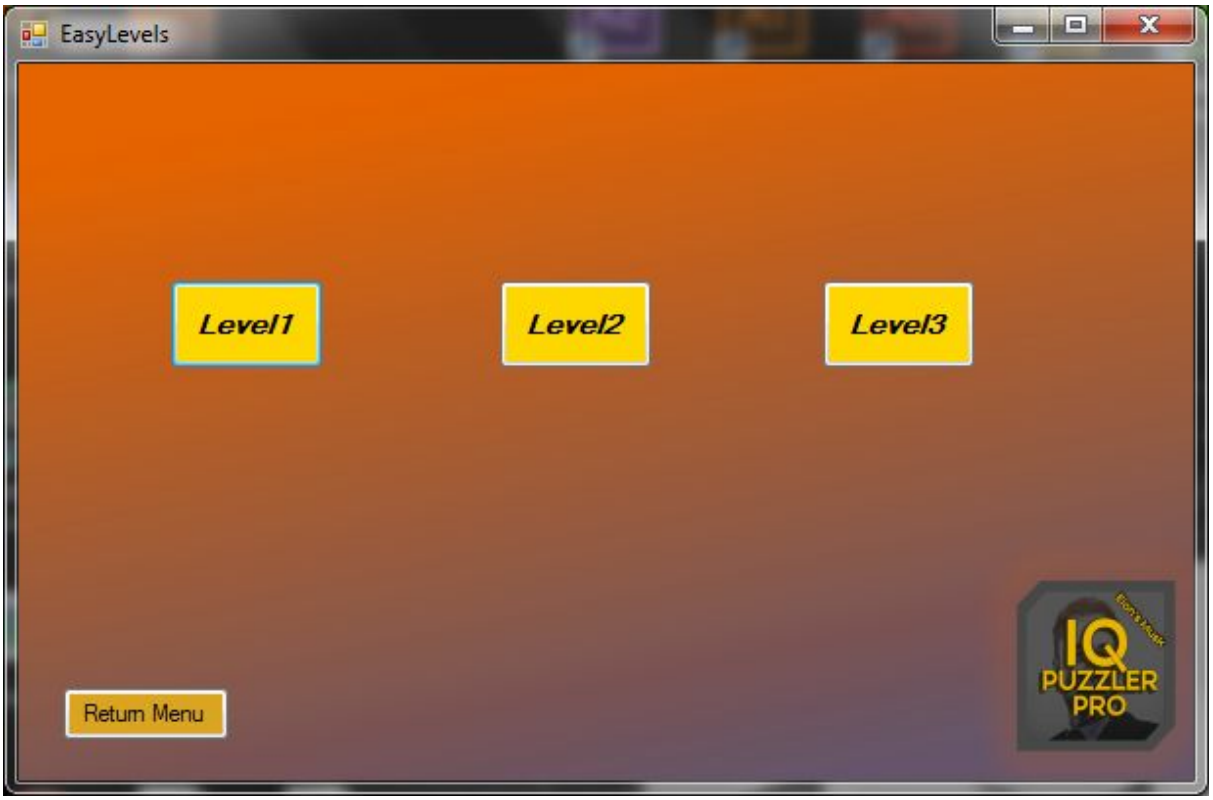

Figure 3: Easy Levels Screen

There are 3 levels available in this screen and user can reach which he/ she chooses. According to selection of the user, the level will be initialized and continue from the solo game screen with specified level.

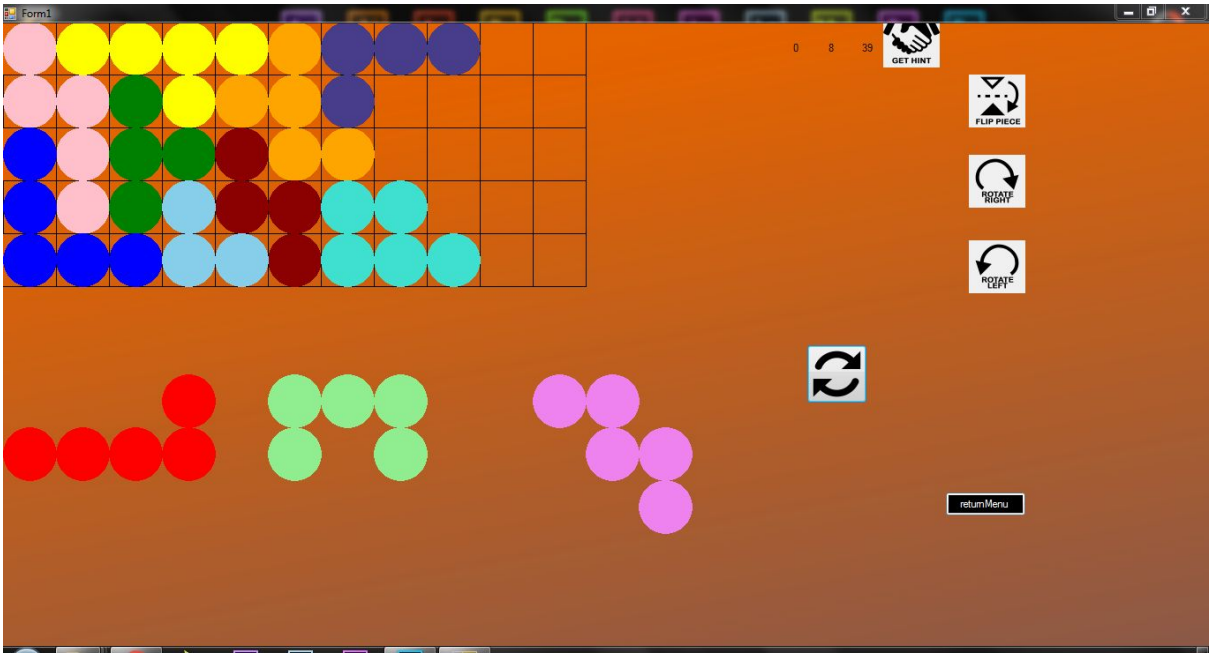

Figure 4: Solo Game Screen

This screen is the where the game will be played. The map is predefined and the puzzle pieces in the map are not changeable. The user can changes the position of puzzle pieces, can rotate and flip them in order to make puzzle complete. Refresh button refreshes all map. The time can be seen at the top of the screen. Furthermore, user can get hint by pressing hint button.

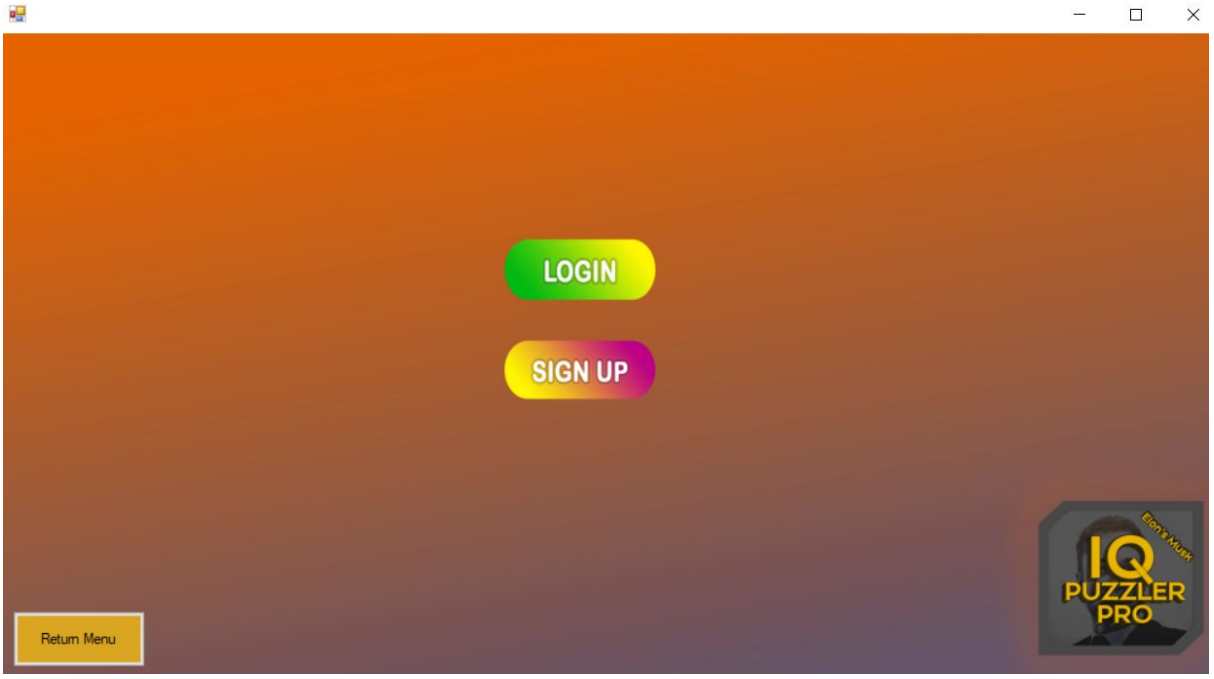

Figure 5 :Online User Screen

There are 3 buttons for the user who have chosen Online from Main Menu Screen. The player can create a new account or sign in the account he/she already

has.

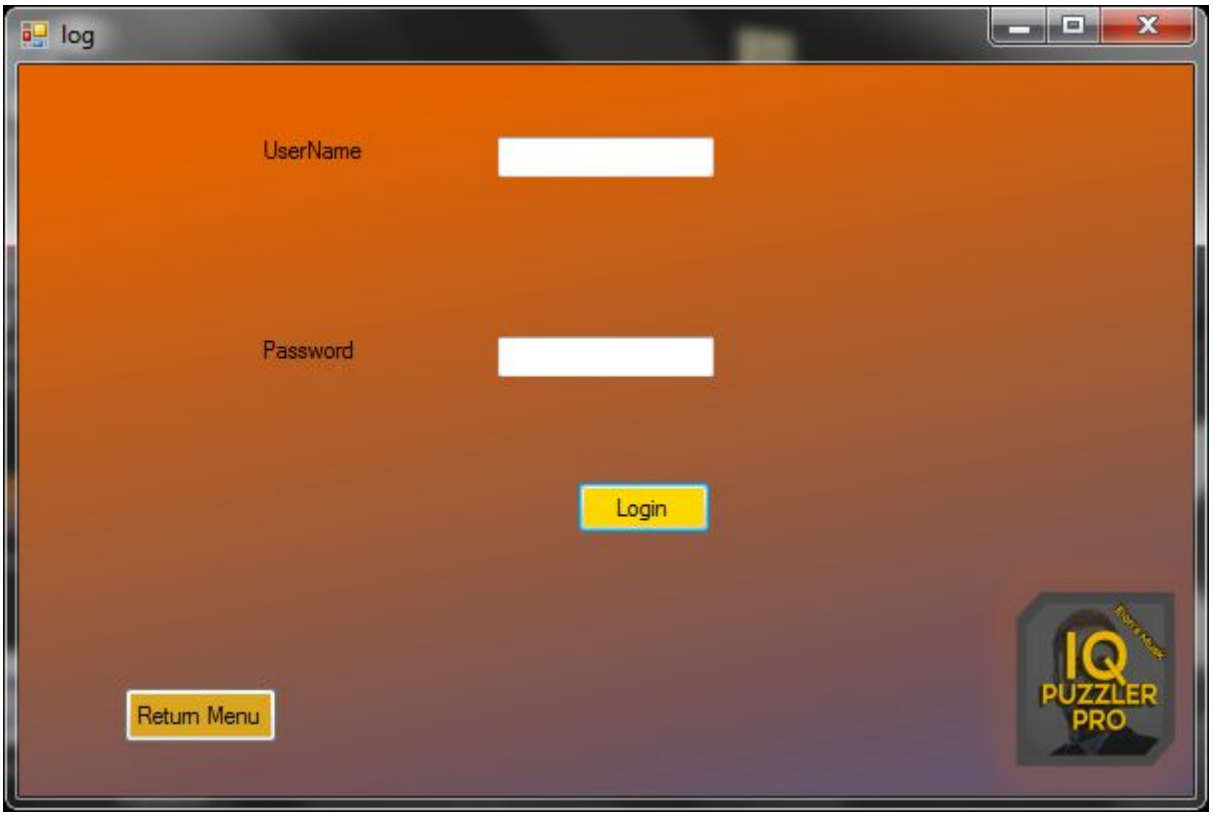

Figure 6: Login Screen

Users can login as a online user by entering UserNames and Passwords. Once the user clicks the login button, he/she creates own account and he/she can play online part of the game.

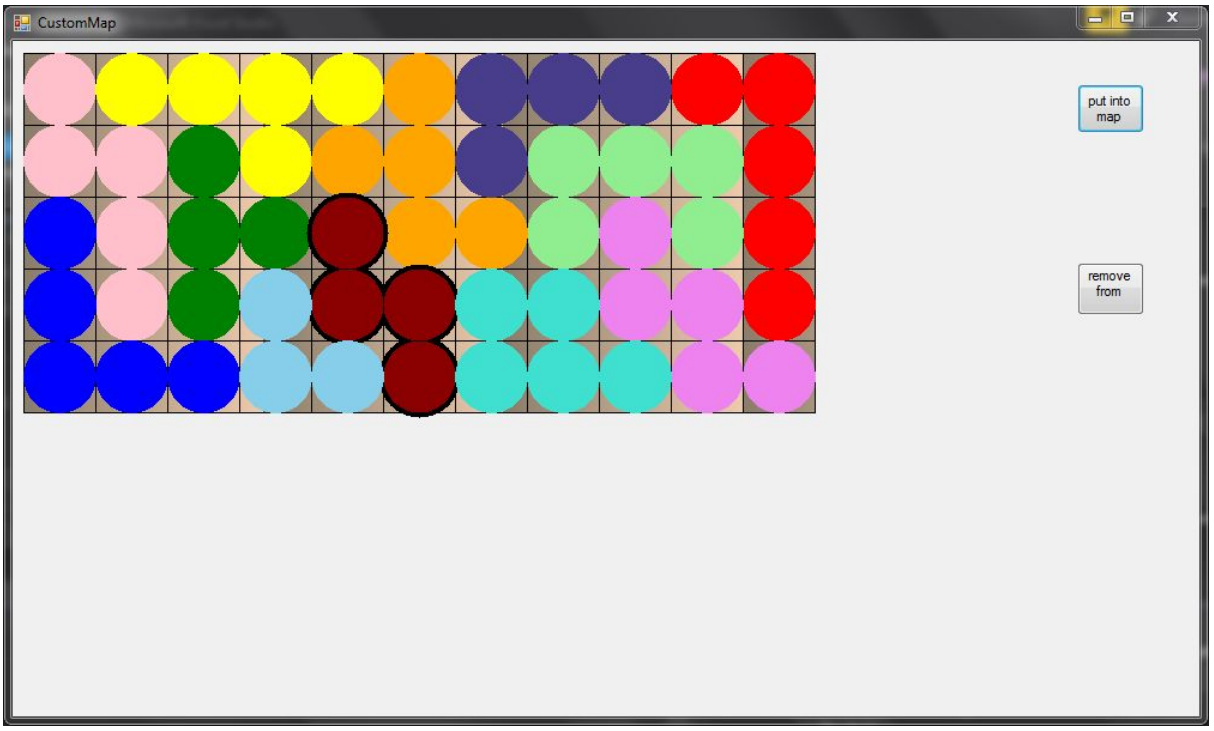

Figure 7: Create Custom Map Screen

In the create custom map screen, there is a default complete map. User can create a map by selecting the puzzle pieces which he/ she wants to be in the map.

5. Work allocation

- Manager of the project : Ebru
- Use case elicitation : Ebru
- Sequence Diagrams : Ebru
- Subsystem decomposition : Ebru
- Panel designs in design report : Ebru
- Map related implementation : Ebru, Cenk
- Database setup : Cenk
- Game logic implementation : Cenk
- Visual application design : Burak

- User interface mockups : Burak
- First iteration logic subsystem design : Burak
- Trailer video : Eren, Burak, Enes
- Activity Diagrams : Enes
- State Diagrams : Eren
- Design Trade-off documentation : Eren
- First iteration final report introduction : Eren
- Boundary conditions documentation : Enes
- Access control and security documentation : Enes
- Object and Class Model elicitation : Fatih
- Functional and nonfunctional requirements elicitation : Fatih
- Initial implementation of GameManager, Piece, Hint classes : Fatih