

Construction 1 for Team Project

Table of Contents

- [Overview](#)
 - [Users Guide](#)
 - [Use Case Model](#)
 - [Glossary](#)
 - [System Architecture](#)
 - [Working Application Code](#)
 - [Test Code](#)
 - [Contributions Summary](#)
-

Overview

This section give an overview of what was accomplished during this iteration

It should list the items that were completed as well as the items not completed

Users Guide

This section provide the first draft Users Guide for the software

Introduction

- The Puzzle Slider game is a fun way to kill some time. When you are feeling bored or just enjoy playing puzzle games why not play Puzzle Slider? This game is fun and easy to navigate, but in case you aren't quite sure what to do we have constructed this users guide so you can figure out those tricky questions.

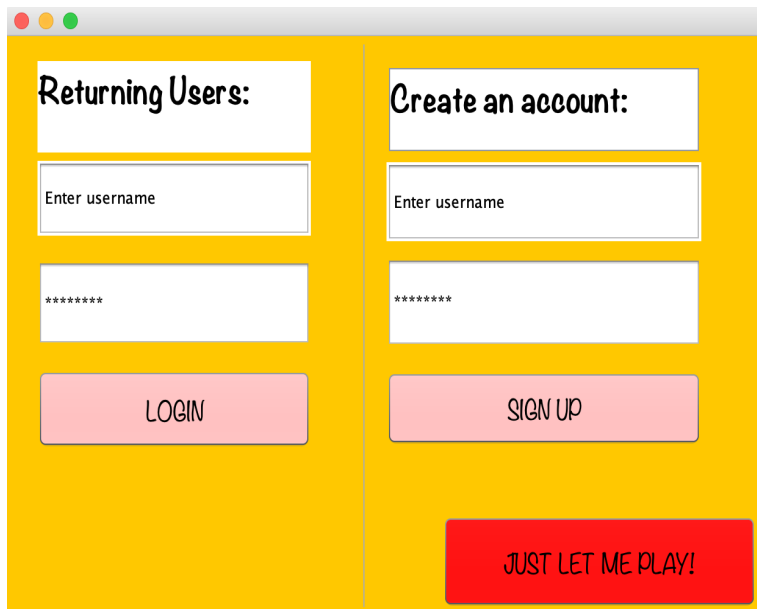
Getting Started

1. Quick Start

- Upon the login screen you will see three buttons: "Login", "Sign up", and "JUST LET ME PLAY!". If you are already registered, you may enter your username and password and login. If you would like to register you can enter in a username and password and then select the "Sign up" button. If you are not wanting to sign in or register a new account you can select the "JUST LET ME PLAY!" button. After logging in/signing up/or selecting just let me play you will be taken to the Main Menu screen. From this screen you have the options to play a new game, load an existing game, check your previous high-scores, access game settings, and logout. If a new game is selected, then the Game Setup Menu is displayed. This gives you the option of selecting your difficulty, choosing a photo or uploading a new photo to play the game with. Once you have selected your difficulty and chosen a photo (or uploaded one) you will then be directed to the Game Session, here you will play the game until you select one of the following options: Pause, Game Settings, Save and Exit, or Restart Game.

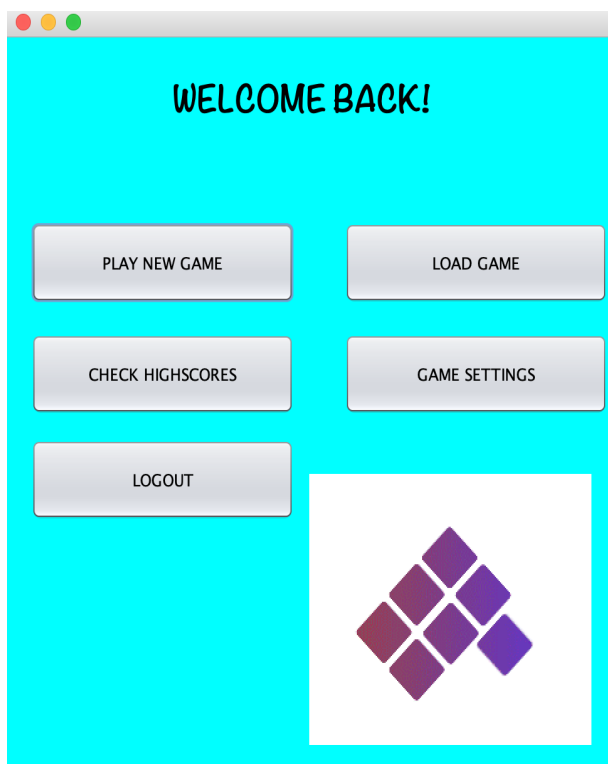
2. Launching the Application

Currently to launch the application, you must open the project in an IDE that supports Java. You must open "LoginScreen.Java" from the edu.winona.cs.app package. Running the file will open an interface that allows you to login, sign up, or play as a guest.



A user interface for login and registration. It features a yellow background with two main sections: 'Returning Users:' and 'Create an account:'. Each section has a text input for 'Enter username' and a password input with masked characters '*****'. Below the inputs are buttons labeled 'LOGIN' and 'SIGN UP' respectively. A red button labeled 'JUST LET ME PLAY!' is positioned at the bottom right.

Upon selecting one of the options from the Main Menu Screen, the next interface will automatically pop up and give an abundance of options. On this screen you will be able to select a new game, load an unfinished game, check your highscores, change some game settings and logout.



A main menu screen with a cyan background. At the top, it says 'WELCOME BACK!'. Below this, there are five buttons arranged in two columns: 'PLAY NEW GAME', 'LOAD GAME', 'CHECK HIGHSCORES', 'GAME SETTINGS', and 'LOGOUT'. At the bottom right, there is a logo consisting of several purple diamonds arranged in a larger diamond shape.

Use Case Models

This should contain most use case diagrams as well as the most of the text description of your use cases.

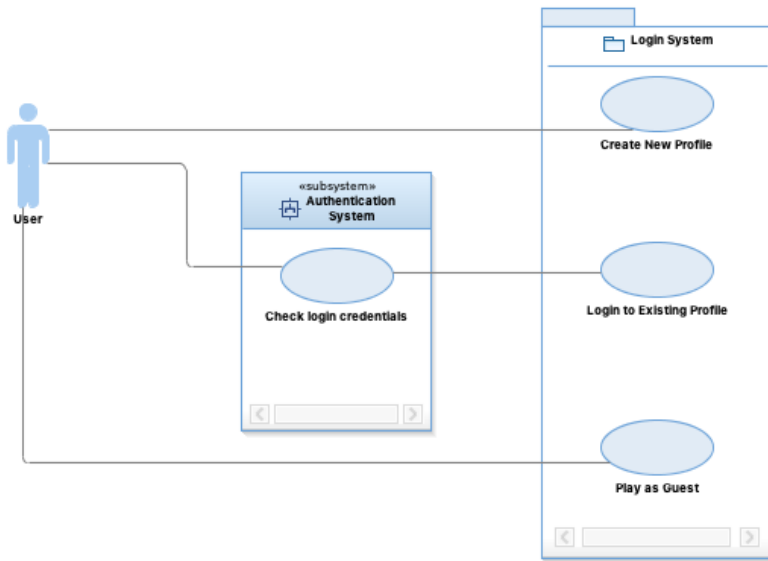
Most use cases should be identified and 60% of use case descriptions should be completed

Casual Login Model

A user that wants to create a new profile will be able to do so through the Login system by choosing the "Create a New Profile" option. Upon choosing this option the user will be prompted to create a new username and password. Once a username and password is created it will be saved to a persistent storage system.

A user that has previously created a profile and login to their profile through the Login system by choosing the "Login to Existing Profile" option. Upon choosing this option the user will be prompted via the authentication subsystem to enter their username and password. The authentication subsystem will check the user's credentials. If the authentication is successful, the user will be brought to the Main Menu. If the authentication fails, the user will be notified and will need to re-enter their username and password.

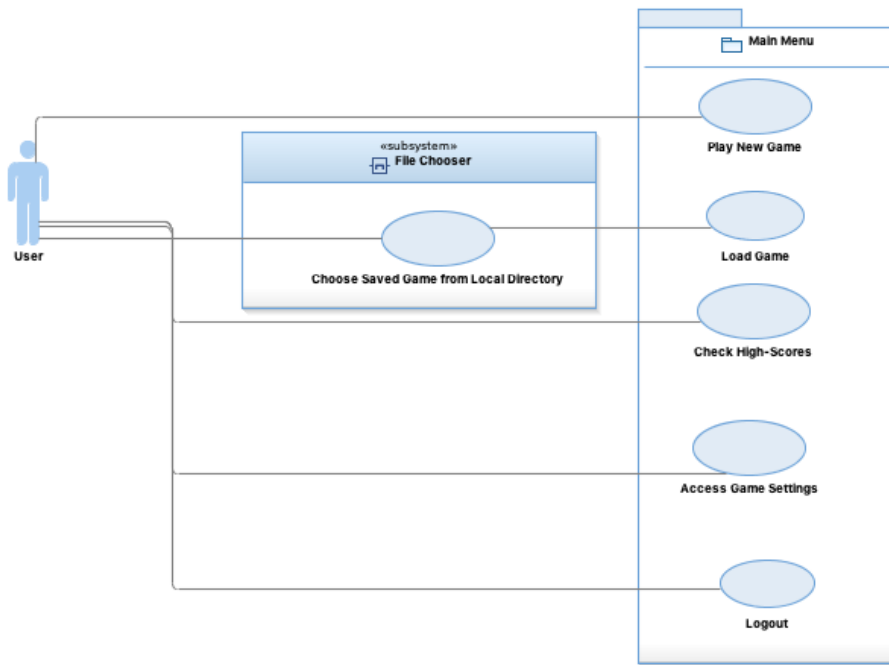
A user can choose to play as a guest. Upon choosing this option the system will notify the user that their progress and high scores will not be saved. The user will be brought to the Main Menu.



Main Menu Casual Use Case

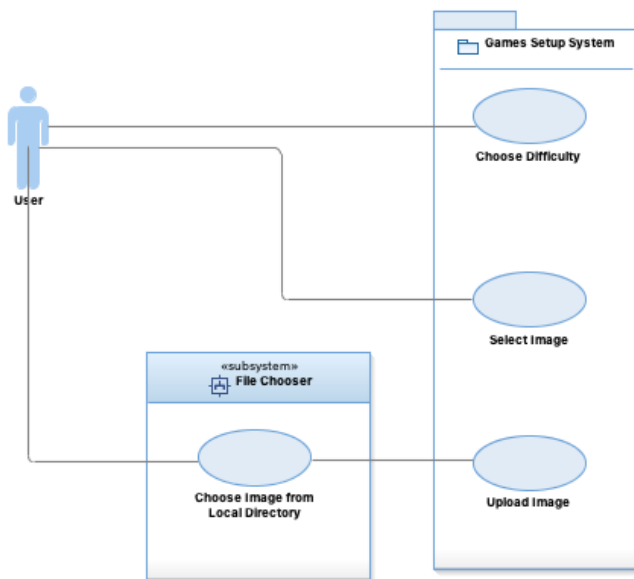
Upon logging in, the user will be given five different options to choose from.

1. The first option is to play a new game. After selecting this, the Game Setup screen (reference the Game Setup Use Case) will be opened.
2. A second option is to load a previously played and saved game. If the user chooses to load a preexisting game file, the File Chooser subsystem will prompt the user to choose a file from their local file system. If the user cancels the File Chooser subsystem, the user will be taken back to the main menu. If the user chooses a file that not in an acceptable save format, the user will be asked to choose another file. If the user chooses a file that is in an acceptable format, the File Chooser subsystem will then load the previously played game.
3. A third option is to check high-scores. This will result in a pop-up window appearing on the screen displaying various statistics regarding previously played games. If the user is logged in as a player, the high-scores will only be from their previously played games. If no previous scores exist, then the high-scores list will be empty. If the user is logged in as a guest, then fictional stats will be provided.
4. A fourth option is to access game settings. These are various settings that alter features that aren't critical to gameplay. These may include background display themes, music, or sound effects.
5. The fifth option is to logout. This option simply takes the user back to the initial Login screen (reference the Login Use Case).



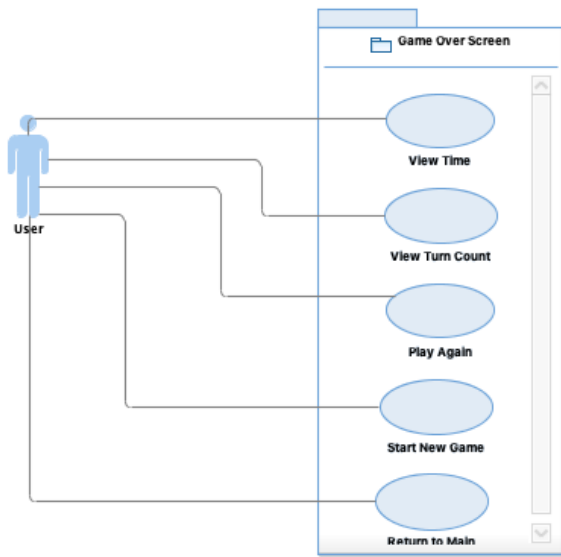
Casual Game Setup Use Case Model

A user that has chosen to start a New Game from the Main Menu or from the Win Game Menu will be prompted to set up their game options. A user can choose the difficulty of their New Game which will determine the number of squares the chosen image will be split. Upon choosing a difficulty, a user will be able to choose an image from a pre-defined set or upload their own image. If the user chooses a pre-defined image they will be brought to the Game Screen. If the user chooses to upload an image the File Chooser subsystem will prompt the user to choose a file from their local file system. If the user cancels the File Chooser subsystem, the user will once again be prompted to choose an image from a pre-defined set or upload their own image. If the user chooses a file that not in an acceptable image format, the user will be asked to choose another file. If the user chooses a file that is in an acceptable image format, the File Chooser subsystem will then upload the image and make it a choosable option from the pre-defined list.



Win Screen Casual Use Case

In the event the user successfully completes a game, they will be presented with the victory screen which contains several options to choose from. If the user selects the "View Time" button, the app will display the amount of time it took the user to complete the game as text. If the user selects the "View Turn Count" button, the app will display the number of turns (or moves) the user took to complete the game. If the user selects the "Play Again" button, the victory screen will disappear and the app will restart the user's current game as if it were never completed, allowing the user to play the same game again. If the user selects the "Start New" button, the victory screen will disappear and the game setup screen will appear, allowing the user to adjust the settings of their new game. If the user selects the "Return to Main" button, the victory screen will disappear and the app will take the user to the main menu screen.

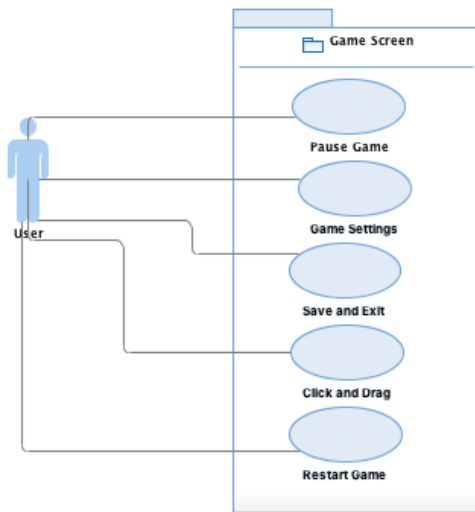


Casual Game Screen Model

A user that has started a new game will see many options on this game screen. First the user will have the option to pause the current game. When the user pauses the game, the timer will stop counting and the game will temporarily stop until "Resume" is clicked. The user also has the option to save and exit the current game. Upon this selection, the game will save the current state and will exit the game screen. This will then return the user to the main menu screen.

A user has the option to select a "Game Settings" option from the Game Screen. This option will prompt the user options to change the theme of the game screen (red, blue, purple, or pink). This will only change the settings of the game screen and not the login, main menu, or win condition screen. The theme selected from a user will include a save state for the current theme selected, meaning if they exit the game, log out and then log back in the theme will be the same as when they exited.

A user has the option to restart the current game on the Game Screen. Upon this selection the user will be prompted with a yes or no button that will ask "Are you sure you want to restart? Your progress will be lost.". If the user selects "yes", the current game will restart and they will not be able to undo this action. If the user should select "no", the prompt will exit and the current game on the Game Screen will continue as normal until the puzzle is solved.

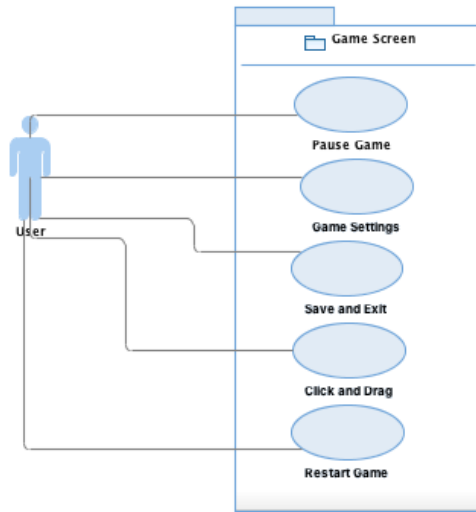


Casual Game Screen Model

A user that has started a new game will see many options on this game screen. First the user will have the option to pause the current game. When the user pauses the game, the timer will stop counting and the game will temporarily stop until "Resume" is clicked. The user also has the option to save and exit the current game. Upon this selection, the game will save the current state and will exit the game screen. This will then return the user to the main menu screen.

A user has the option to select a "Game Settings" option from the Game Screen. This option will prompt the user options to change the theme of the game screen (red, blue, purple, or pink). This will only change the settings of the game screen and not the login, main menu, or win condition screen. The theme selected from a user will include a save state for the current theme selected, meaning if they exit the game, log out and then log back in the theme will be the same as when they exited.

A user has the option to restart the current game on the Game Screen. Upon this selection the user will be prompted with a yes or no button that will ask "Are you sure you want to restart? Your progress will be lost.". If the user selects "yes", the current game will restart and they will not be able to undo this action. If the user should select "no", the prompt will exit and the current game on the Game Screen will continue as normal until the puzzle is solved.



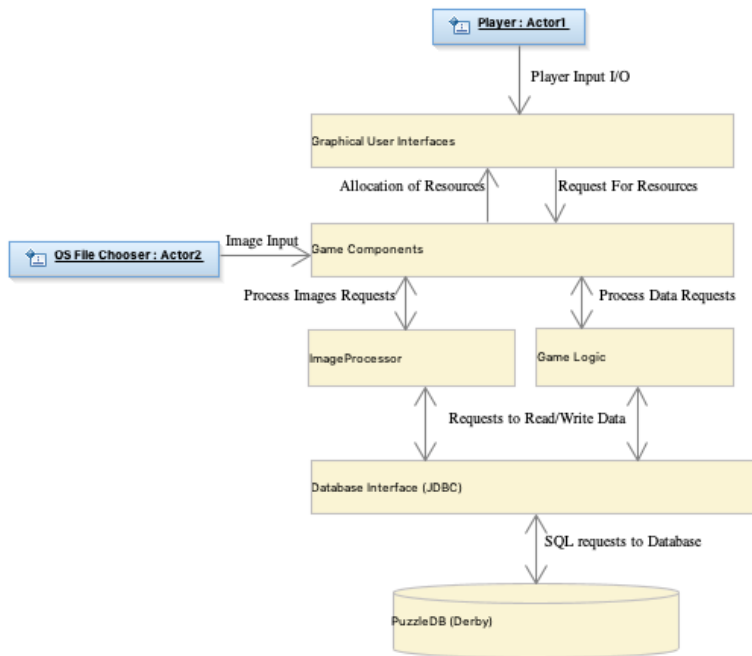
Glossary

This section should contain the glossary for your project.

Key Words:

- Sliding tile puzzle: A combination puzzle that requires the user to slide flat pieces in a specific order to accomplish an end configuration.
 - Video game: A game played by electronically manipulated images on a computer screen.
 - User: A person who is operating the video game.
 - Progress: The development toward a complete state of the current game.
 - Save: An option to keep your progress in the game, unaffected by exiting the program.
 - Settings: The option to change preferences of the game.
 - Difficulty: How easy or challenging the game may be.
 - High score: The highest score the user has gotten so far.
-

System Architecture



Our *Puzzle Slider* game will use both a **Layered Architecture** and a **Repository Architecture**. Our program will be layered in the following way:

1. **Graphical User Interface (GUI):** This is the first layer which will directly interact with the user and send requests down to the Game Components layer. Responses from the Game Components layer will then update the GUI.
2. **Game Components:** This layer will handle requests from the GUI layer above and return the results of these requests to the GUI layer. Some examples of requests accepted by this layer include:
 - Start or load games.
 - Create new user or login to existing user.
 - Check win conditions.
3. **Game Logic:** handles requests from the Game Components layer that require data processing of some sort. Once processing as been completed, this layer returns data back to the Game Componets layer.
4. **Image Processing:** this layer is on the same level as the Game Logic layer. This layer handles request from the Game Components layer that require image processing and returns completed requests back to the Game Componets layer.
5. **Database Interface** this layer provides the Game Logic and Image Processing layers the ability to make read/write requests to the underlying database which acts as a **Repository** for all of our game's data.
6. **Puzzle Database** the derby database that acts as our repository is the final layer to our architecture. It will respond to SQL statements from the Database Interface layer.

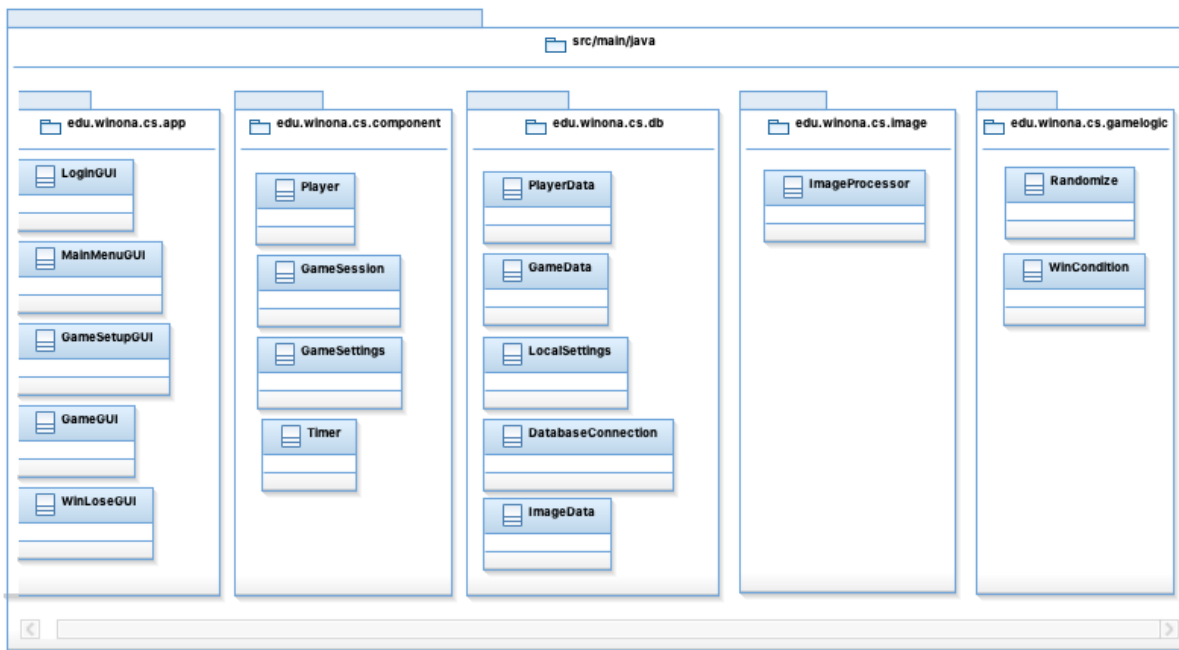
The **System Architecture** above will be supported by a Maven build architecture. Maven will build our project into a excutable .JAR program with automated JUnit testing.

UML Diagrams

This should contain and describe the class, activity, and sequence diagrams in your design.

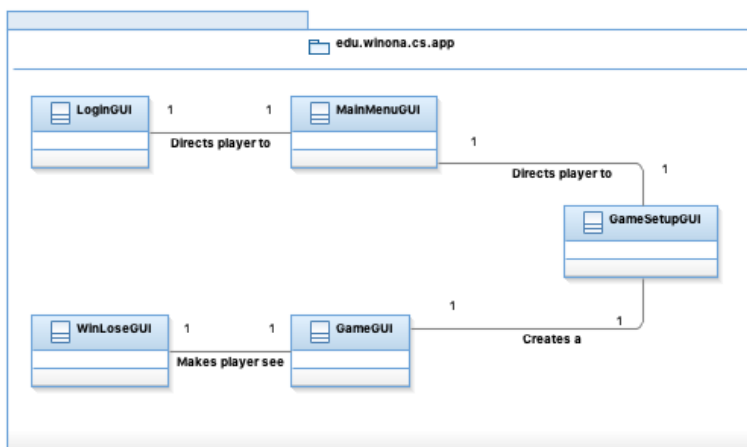
Package Diagrams

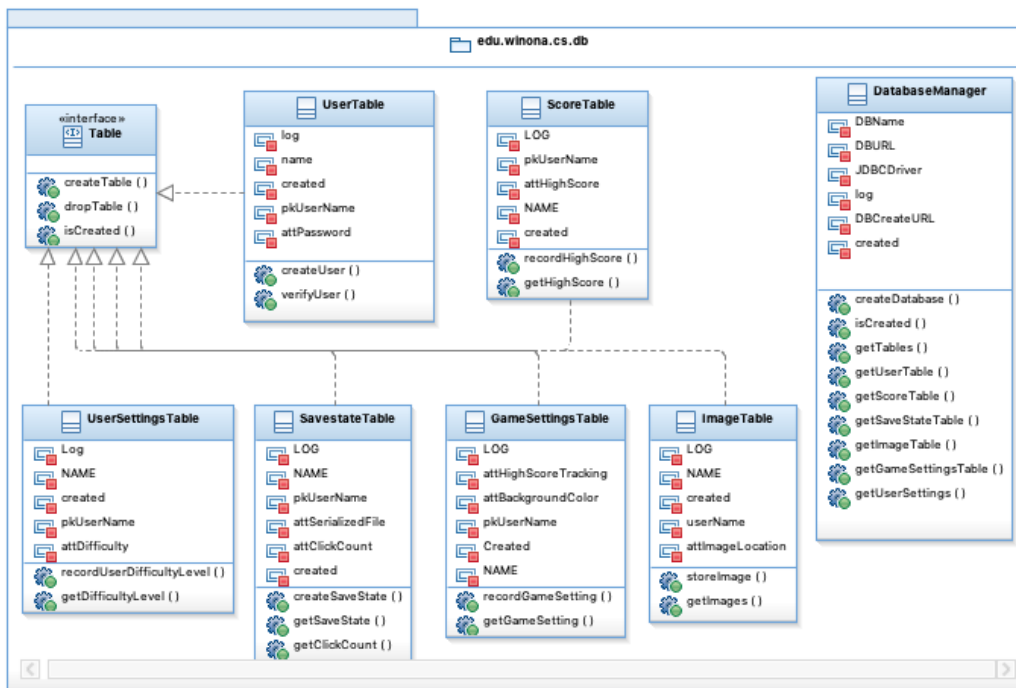
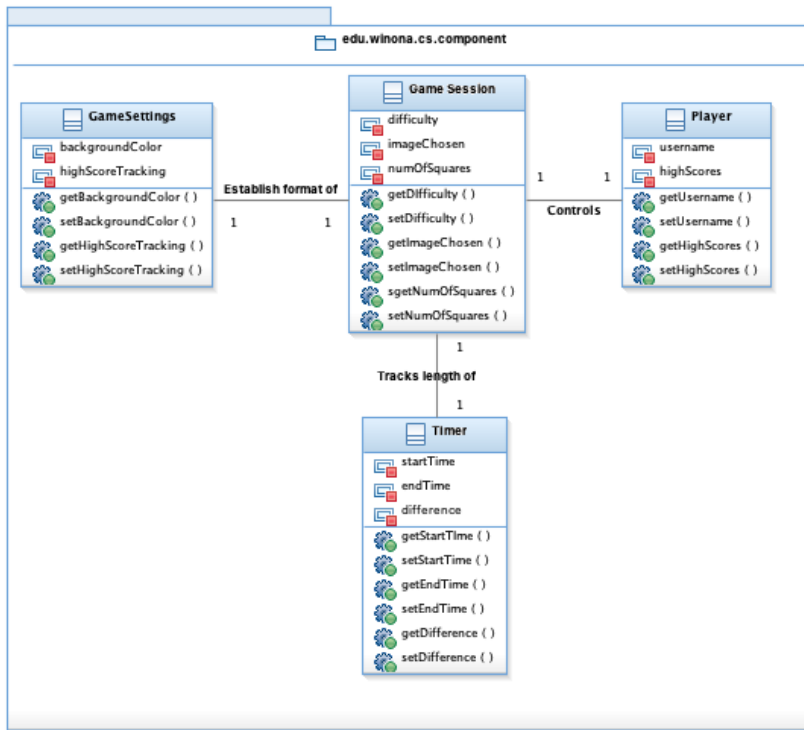
The package overview below shows the layers of our system architecure in package-class form. This is the overview of our UML diagraming. Each package will be separtely diagrammed with full class diagrams below.

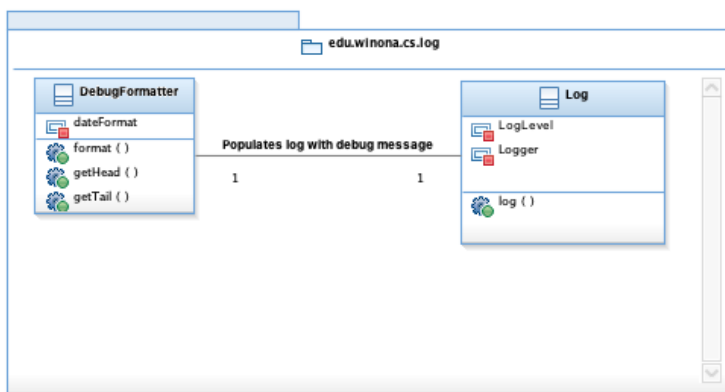
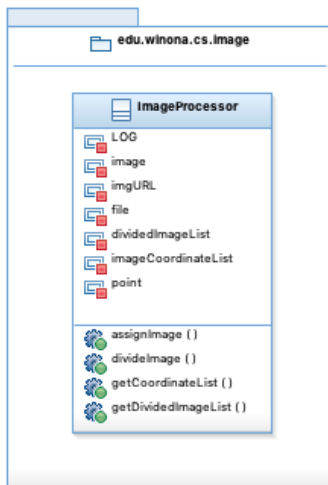
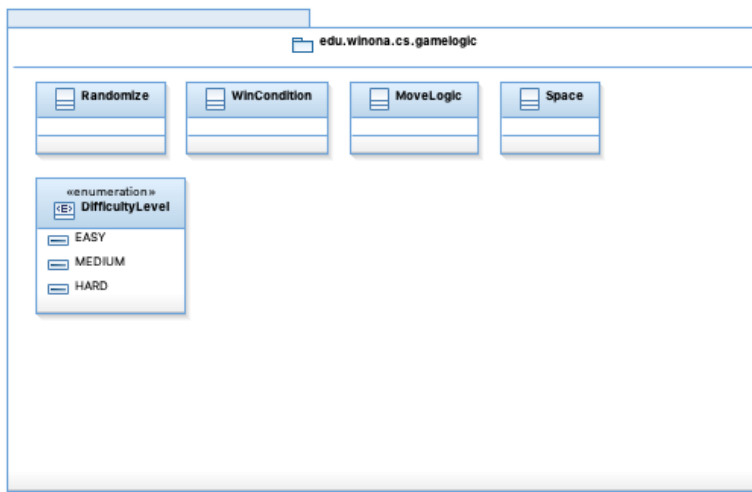


Class Diagram

The package overviews below show the layouts of our packages and the structure of the classes within them.

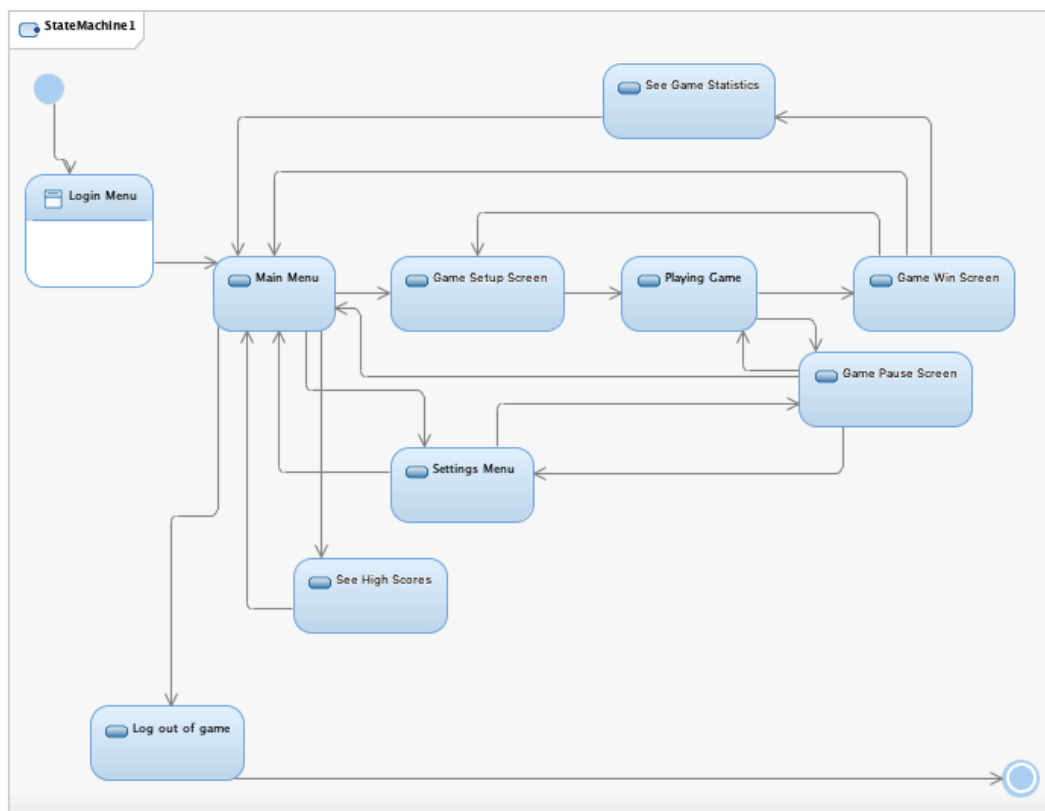






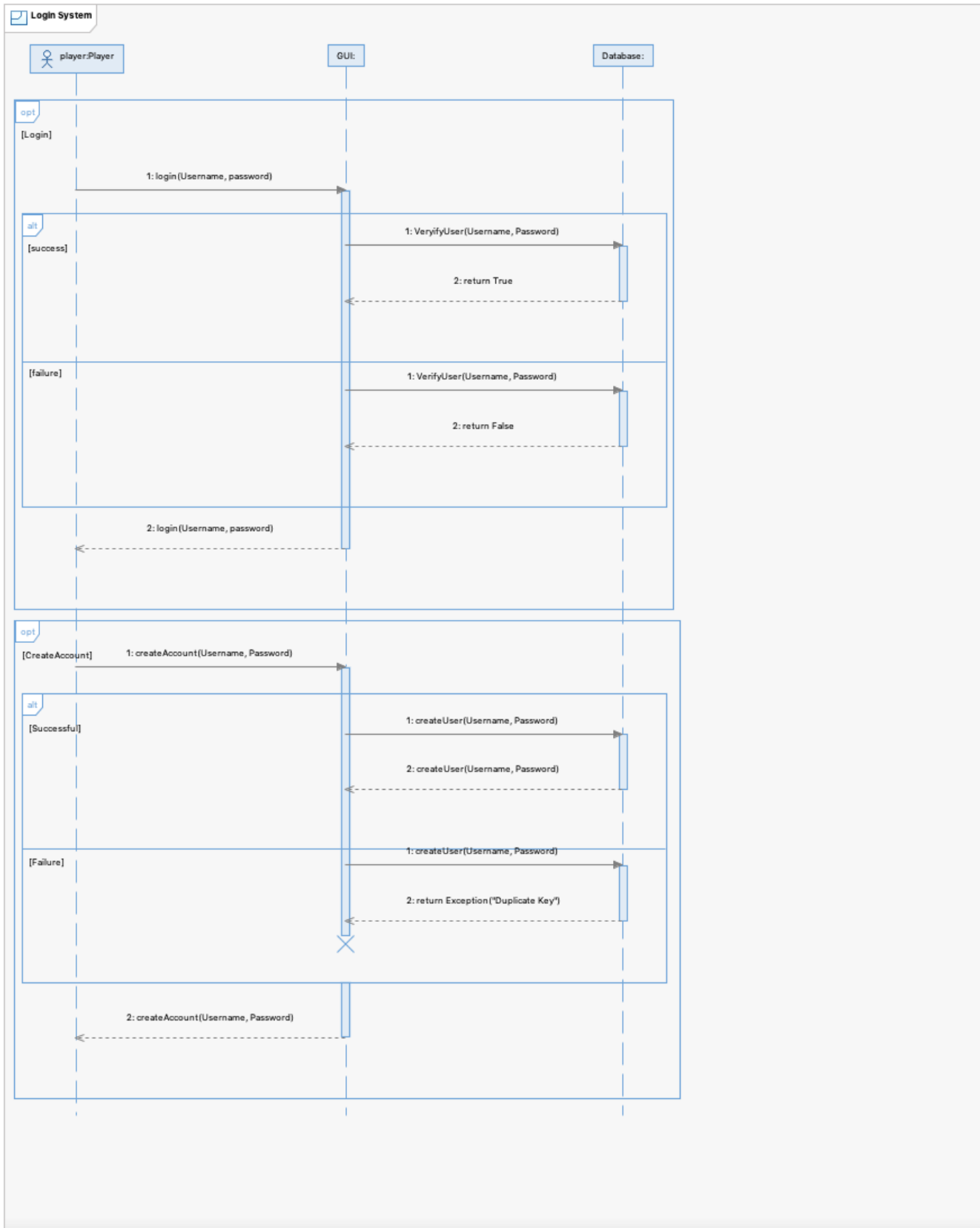
State Diagram

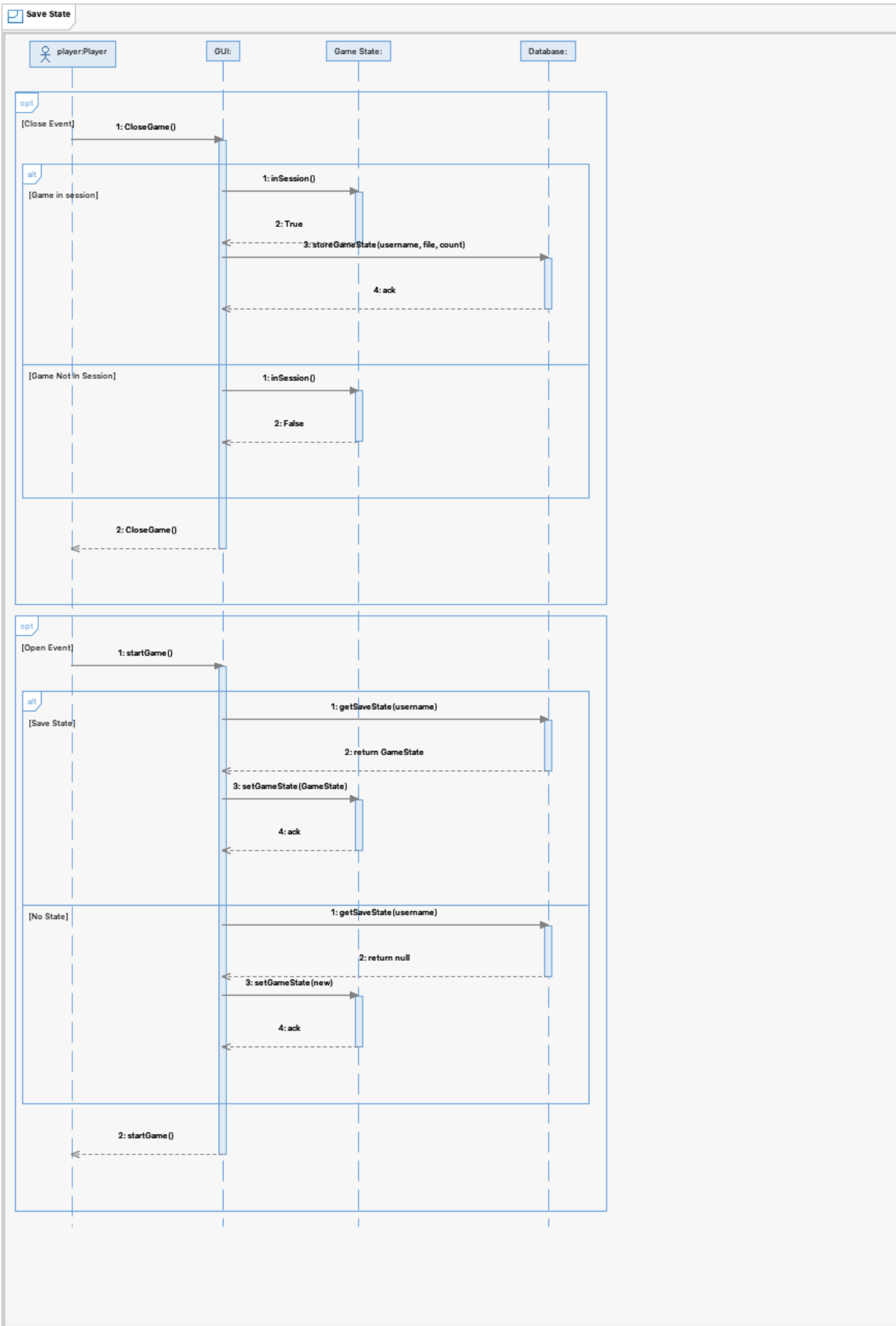
The image below shows the state diagram of our system.



Sequence Diagram

The diagrams below shows the sequence diagram of the login procedure and the save state procedure of the puzzle game.





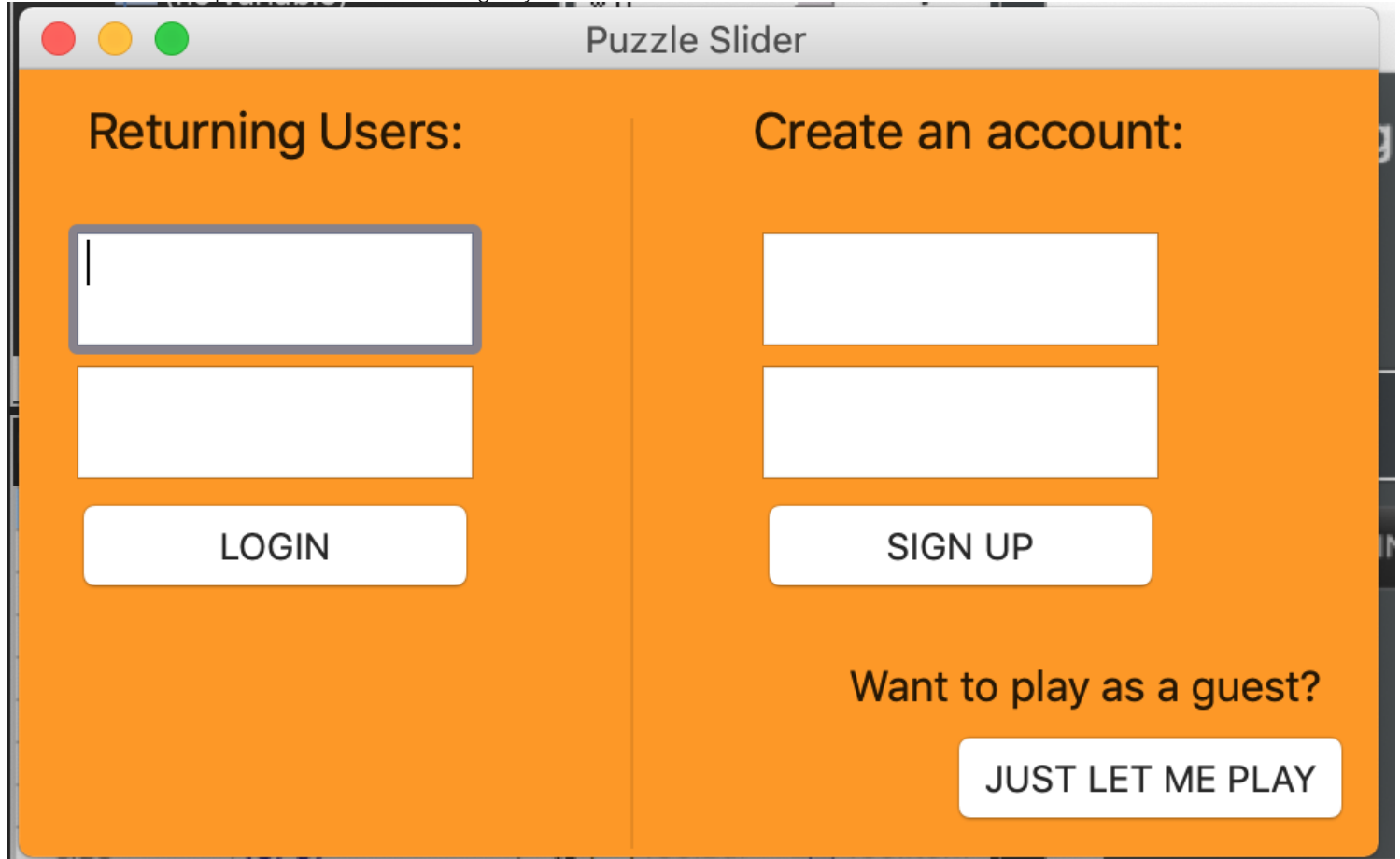
Working Application Code

This section describes your working code and provides links to it.

For elaboration 1 we have completed boilerplate application code to ensure that automated testing and build infrastructure is working. To see documentation for this working please refer to our [README.md](#) documentation.

For elaboration 2 we created boilerplate class code for each package's classes. The package package diagram that we used was developed as part of our [UML Diagramming](#).

We were able to complete our first GUI for the login system.



As a part of this system we also implemented the use of a Derby JDBC driver to store user login data in a Derby database. Output from automated testing shows the ability for us to connect to the database, create tables, and add/validate users. The output of this testing can be found [here](#).

Links to working application code:

APP:

[App.java](#)
[FileChooser.java](#)
[GameGUI.java](#)
[GameScreenEasy.java](#)
[GameSettings.java](#)
[GameSetupGUI.java](#)
[LoginScreen.java](#)
[MainMenuScreen.java](#)
[MenuMenuScreen.java](#)
[WinLoseGUI.java](#)

COMPONENT:

[GameSession.java](#)
[GameSettings.java](#)
[Player.java](#)
[Score.java](#)

UserSettings.java

DB:

DatabaseManager.java
DatabaseUtil.java
GameSettingsTable.java
HighScoreTable.java
ImageTable.java
SaveStateTable.java
Table.java
UserSettingsTable.java

GAMELOGIC:

DifficultyLevel.java
MoveLogic.java
Randomize.java
Space.java

IMAGE:

ImageProcessor.java

LOG:

DebugFormatter.java
Log.java

Test Code

This section describes your test code and provides links to it.

This section should also describe the testing that was not automated.

Automated Tests

- Test login: Verify user name and password when logging into an account. Throw exceptions for invalid username/password combination.
- Test new game: Select new game option as both a returning user or as a guest.
- Test load save game: test selecting a saved game from local directory, throw exception if a previous game is not found.
- Test upload image: Make sure image uploaded is in correct format and size. Throw exceptions if criteria is not met.
- Test pause: Validate the pause function is working and stops the timer.

Non-automated Tests

- Login GUI: Ensure that login GUI accurately notifies user of verification error, username already in use error, and closes correctly.
- Main Menu GUI: Ensure that all of the main menu options work as expected, and user is able to get back to the login screen if necessary.
- Game Setup GUI: Ensure that game set options display user preferences correctly, and that these settings are saved between sessions.
- Game GUI: Ensure that the game GUI displays output correctly and responds to user input in an acceptable fashion.
- Win Lose GUI: Ensure that the Win/Lose GUI correctly displays the high scores of the user, and the previous game session's results.

Contributions Summary

This section describes the contributions made by the members of your team

- Travis: Use case formatting, game randomization and win-condition logic
- Kyle: Database logic, sequence diagram
- Tristin: Image processing and authentication logic
- Erika: **Construction 1 leader** GUI design and development, user guide

This section intentionally left blank.

END OF THE DOCUMENT
