**Advanced Programing Project**

**Server Side:**

Includes all the classes of the server.

1. MyServer – General Server's class, we use it for the game's server. Has the ability to construct a Server and get requests from the clients.
2. MyHostServer – class for host Server, has the socket's list and the ability to communicate with the clients.
3. Handlers:

GusetHandler – class that used for handling the players's requests.

BookScrabbleHandler – class that used for handling the host's requests (proper words, checking in the dictionary).

1. HostManager – this class is responsible for the game managemnet.
2. dicManager - this class is responsible for the dictinary managemnet.
3. We also have classes the used for the management of the calender and the tiles form the game.

**Client side:**

Built in MVVM architicture.

**Model Layer:**

1. PlayerModel: our Main class of this layer, contains all the data and methods for each player of the game.
2. HostModeModel – extention for PlayerModel class, this class is intended for the hosting player.
3. GuestModeModel - extention for PlayerModel class, this class is intended for the other players.
4. MenuModel – contains methods that relevant to the menu screen which opens at the starting of the game.
5. Service – class that has service methods which help with casting.

**View Layer:**

This layer represents the display and the UI of the application.

We have two parts in this layer:

FXML files – this files includes the display that the clients see.

ViewController classes – these classes includes the methods that are activated after something is activated in the UI.

**ViewModel Layer:**

The ViewModel layer works as a mediator between the Model and the View. It contains the logic and has data and methods the the View is binded to. The ViewModel also gets data from the Model layer.

The flow of the communication in the MVVM architecture:

The View layer is binded to field and methods exposed by the ViewModel layer. The ViewModel is interactig with the Model to export and update data.

The Model tells the ViewModel everytime the data changes.

The ViewModel update the fields the View is binded to and activates the interface updates.

The View model gets the activaties of the user in the View and proccesed.