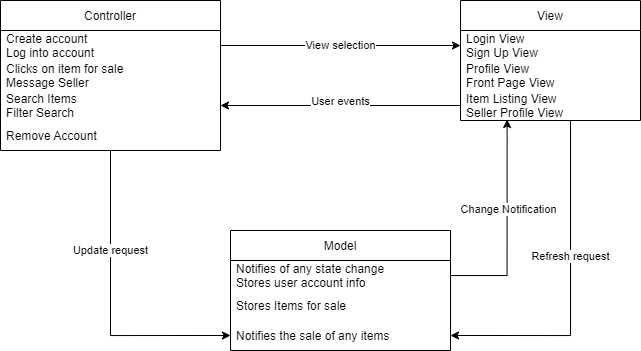
**Group:** CSwap

**Architecture Design:**

For designing the architecture of CSwap we designed a Model-View-Controller architecture. With an MVC architecture it is easy to represent and interact with the data in multiple ways, therefore because we will have the interchangeability between seller and buyers it will be easier to make these changes. The application will have various representations of the same data making it advantageous to use a MVC design. Lastly, because of the nature of the application, it is difficult to predict the future needs of the users therefore having a MVC will make the changes to the way people interact easier.

**Architecture Design Diagram:**

**Design Description:**

The architectural pattern consists of three components, the view, the controller and the model.

The view for CSwap handles the UI of the web application. This is mostly handled by React and Bootstrap. The application will have many views that can possibly be displayed. The Login View will display when the users first open the application. It will give them the option to enter their email address & password, sign in with either google or facebook, or navigate to the Sign Up View to create an account. Once they sign in, they will be presented with the Front Page View. From here, they will have the ability to view items for sale. Any interaction with the UI can trigger user events sending information to the controller to handle the inputs.

The controller handles the various inputs from the user. These inputs can be a simple button press to further inspect an item for sale, login or out of an account, to start a conversation with a seller, or to create a new listing. When an input is detected the controller then notifies the view of any changes if necessary and then notifies the model of any state changes.

The model for CSwap handles the data from the app. Here the view can access any data for specific elements in the app to update the UI. Any state changes notified by the controller will be stored in Firebase and a notification of a change if necessary can be sent out by the model to the view controller. This will allow for the view to be updated when a new item has been added or a new message has been sent out. This communication between the database and the view controller is imperative in updating information on the application.