We have 17 classes for this assignment however not all are currently being used, instead existing for future iterations.

First come a series of controllers, each of them is in charge of taking data from the user interface and conducting the other classes to manipulate the site to deliver an appropriate response.

SignUpController. This controller handles account creation. In terms of grasp patterns it is a controller, and an information expert. It will take their information and then through its association with the Account class create the user’s records in the database.

Login Controller This controller handles account login. In terms of grasp patterns it is a controller, and an information expert. This class will interact with account to access the user’s records from the database and set up session to allow the user to interact with the website.

Logout Controller This controller handles account logout. In terms of grasp patterns it is a controller, and an information expert. This class is in charge of ending sessions and clearing out any relevant cookies so that the user can be logged out.

PanelAccountController This controller handles account to panel connections. In terms of grasp patterns it is a controller, and an information expert. This controller will manage both the Account and Panel classes so that the user’s interactions with the Panel our successful.

PanelController This controller handles panel management. In terms of grasp patterns it is a controller, and an information expert. This class handles the panel, what it displays, what it pulls out of the database and what is submitted to the panel.

InterestController This controller handles interest selection. In terms of grasp patterns it is a controller, and an information expert. This will update the user’s selections on the interest’s page so that when selected their interests can be stored in the database.

Next is a series of classes dedicated to collecting methods used for managing the class’s details, for example the Account class doesn’t get instantiated instead it serves as a host for all account related methods.

Account. Has high cohesion and low coupling. Serves as an aggregation of methods used for Account interactions. There does not exist a location that instantiates it. Account will hold database interactions and be the reflection of the user.

Friend. Has high cohesion and low coupling. Serves as an aggregation of methods used for Friend interactions. There does not exist a location that instantiates it. Friend will represent the individual friend links and a collection of them serves as the friend list.

Interest. Has high cohesion and low coupling. Serves as an aggregation of methods used for Interest interactions. There does not exist a location that instantiates it. Serves as the collection of things that the user holds interest in.

Location. Has high cohesion and low coupling. Serves as an aggregation of methods used for Location interactions. There does not exist a location that instantiates it. Serves as where the user is in the world so to help form social links.

Panel. Has high cohesion and low coupling. Serves as an aggregation of methods used for Panel interactions. There does not exist a location that instantiates it. Serves as collections of posts and discussions on a topic.

Rating. Has high cohesion and low coupling. Serves as an aggregation of methods used for rating interactions. There does not exist a location that instantiates it. Serves to indicate ratings and reviews of a user or panel.