**Scenario Walk-through:**

In bullet-point form for clarity and ease of reading.

***Important Note:*** *In the scenario walkthrough, we often say a class calls another class, but that doesn’t necessarily mean it calls them directly. Sometimes it’s indirectly such as when AppManager contacts ViewAccountWindow, AppManager doesn’t actually depend on ViewAccountWindow, but on ViewAccountWindow’s superclass Window and the interface DisplayEntitiyInformation.*

#1: User is able to signs up, add some medication, and see a sample timetable

* User signs up:
  + We start at AppManager. We call StartScreenWindow to show the user their options of logging in or signing up.
  + StartScreenWindow gets user input on creating a new account.
  + We call a method to create a new account. This method calls CreateAccountWindow and CreateAccountWindow gets user input and returns it to AppManager.
  + We first add the username and password to the map containing usernames and passwords in AppManager.
  + We call ManagementSystem to add a new user and pass in the name and username.
  + In ManagementSystem, we call UserManager to add a new user. We pass in the name and username.
  + In UserManager, we create a new instance of User and pass in the name and username.
  + We go back to AppManager and call viewAccountWindow to show the user information.
  + We call ManagementSystem to get user information. We call UserManager from ManagementSystem to get the user’s name, username, and list of medicine they have.
  + We go back to AppManager and pass this information into viewAccountWindow so that the class can display the information.
  + From this window, the user can decide to add a new medicine, remove medicine or edit the existing medicine.

* User adds some medication to their account:
  + The user decides to add medication.
  + ViewAccountWindow returns user input of selecting edit medication back to AppManager.
  + AppManager calls the AddMedicineWindow. The user is able to enter the name of their medicine, its type, the method of administration, and any extra instructions.
  + The user also enters the specific times they need to take medication.
  + We get user input from AddMedicineWindow and return this information back to AppManager.
  + AppManager calls ManagementSystem to add a new medicine. It passes through the information about the medicine.
  + ManagementSystem passes the information about the medicine to UserManager, who passes that information to MedicineManager.
  + MedicineManager creates a new instance of Medicine with the information provided by the user and returns it. During the step, Medicine also creates an instance of MedicineSchedule and stores it as an instance attribute.
  + UserManager takes the returned instance of Medicine from MedicineManager and calls User’s add medicine.
  + User adds the instance of medicine into its list of medications.
  + We return to ViewAccountWindow.

* User is able to see a sample timetable:
  + The user selects that they want to see the timetable.
  + ViewAccountWindow returns the user input to AppManager.
  + We call ManagementSystem and ask to return a schedule.
  + ManagementSystem first calls UserManager to get a list of all medicine.
  + UserManager calls the getter for the list of Medicine from User and returns this list.
  + ManagementSystem calls ScheduleManager to compile a schedule and passes in the list of Medicine as a parameter.
  + ScheduleManager calls ScheduleCompiler to make a complete schedule and passes in a list of MedicineSchedules which it got from the list of Medicine.
  + ScheduleCompiler compiles all the MedicineSchedules and returns a Schedule object.
  + ScheduleManager takes the returned Schedule and returns this to ManagementSystem.
  + ManagementSystem returns the Schedule to AppManager.
  + AppManager calls TimeTableWindow to display the schedule.