Team 5 - Purdue Planner Design Document

Milia Enane

Jake Losin

Brandon Nice

Michael Rollberg

Kenneth Tam

Purpose

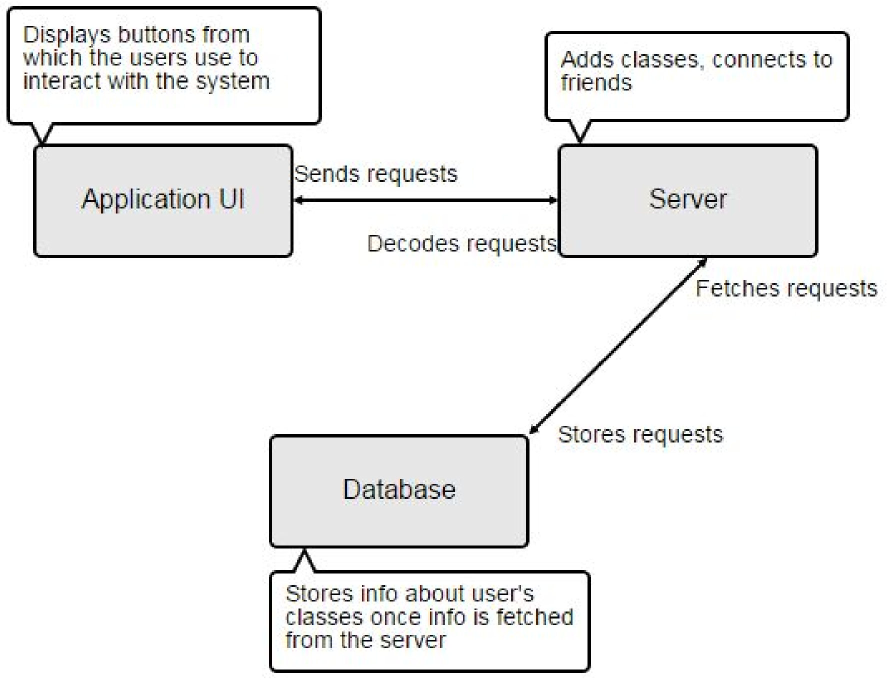
We will be assembling and organizing a database of Purdue's classes in order to create a system of user profiles capable of planning schedules and syncing with each other.

Design Outline

Our project will be a mobile application that will be available on Android and iOS that allows a user to view their schedule of classes. They will be able to add classes, events, and see other friends schedules. We will be using a client-server model where each individual user can connect to add classes, generate their map of classes, or see their friends. They will also be able to view their schedule and map while they are offline, but will not be able to add classes.

1. **Application UI**
   a. The user will be able to access the schedule, add classes to their schedule and see a map with all of the buildings that their classes are in pointed out.
   b. The user will be able to connect with friends, see other friends schedule and compare their schedule with friends.
   c. The user will be able to add events to their schedule which are listed based on organization and also will be recommended to the user based on their Facebook likes.
   d. The user will be able to change their settings, including which information about their schedule they share.
2. **Server**
   a. The server will interact with the Application UI and the database.
   b. The server will query the Purdue API when the user wants to add a class.
   c. The server will query the Facebook API when the user logs in.
   d. The server will get user's schedule, map, friends, and settings from the Database when the user logs in.
3. **Database**
   a. The database will store each user's schedule, map, friends, and settings.
   b. In order to access the database a user needs to be online to send requests through the server.

4. High-level UML layout of our system


Displays buttons from which the users use to interact with the system
Adds classes, connects to friends
Application UI
Sends requests
Server
Decodes requests
Fetches requests
Stores requests
Database
Stores info about user's classes once info is fetched from the server

Our application will be connected to multiple remote servers that are responsible for connecting to friends and adding the user’s classes. Depending on what type of request is sent (e.g. a class or a friend’s schedule), the server will then decide what action to perform and will communicate with the database to store information that the user needs.

Design Issues

- **Platform Compatibility**
  - Option 1: Xamarin and C# (Multiplatform)
  - Option 2: Android (Java) - Team has more experience in Java
  - Option 3: iOS (Swift / Objective C)

  Decision: Our group will need to familiarize itself with new software tools such as C# and Xamarin in order to make our application capable of functioning on multiple platforms.

- **Accessing every Purdue Class for the application**
  - Option 1: Obtain them from an official Purdue staff member
  - Option 2: Query the Purdue classes API
  - Option 3: Extract the classes from the official class roster

  Decision: We chose option 2 because it is an already implemented way to get all the instructors, sections, times, etc. for every class that Purdue has for each semester.

- **Server Hosting:**
  - Option 1: Renting a dedicated server
  - Option 2: Using a Raspberry pi
  - Option 3: Using an old computer

  Decision: We chose option 2 because it is a cheap and viable option. The only problem we may have with this is the hardware limitations on raspberry pi.

- **Login Implementation:**
  - Option 1: Implement the Facebook API to access a login service that many people use and are familiar with
  - Option 2: Use the Twitter API in order to login
  - Option 3: Create our own login API to store and create user profiles

  Decision: We believe that the Facebook login API is the easiest to use and most stable regarding login security. Having our own database of user IDs and passwords would probably make many users uncomfortable and would be more time-consuming to implement.

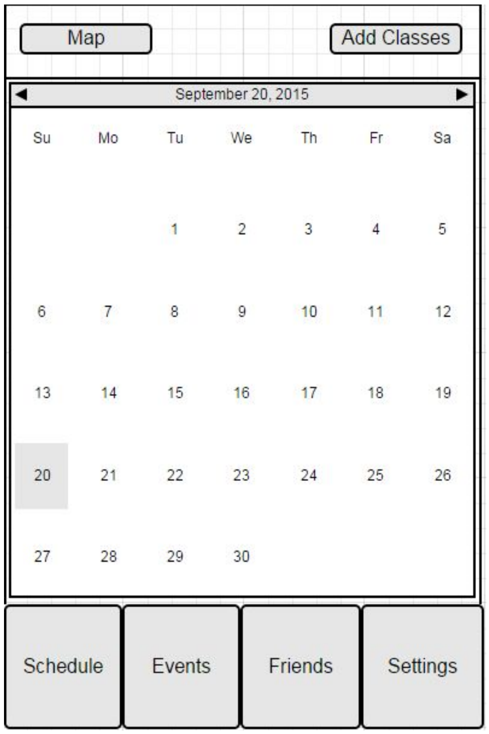
- **UI Design:**
  - Option 1: Tab system on the bottom similar to Facebook iOS app.
  - Option 2: Side-swipe similar to GroupMe app.
  - Option 3: Home page with links to each different screen similar to the Purdue app.

  Decision: We chose option 1 because we think it is the most user friendly way for the app to be implemented.

Design Details

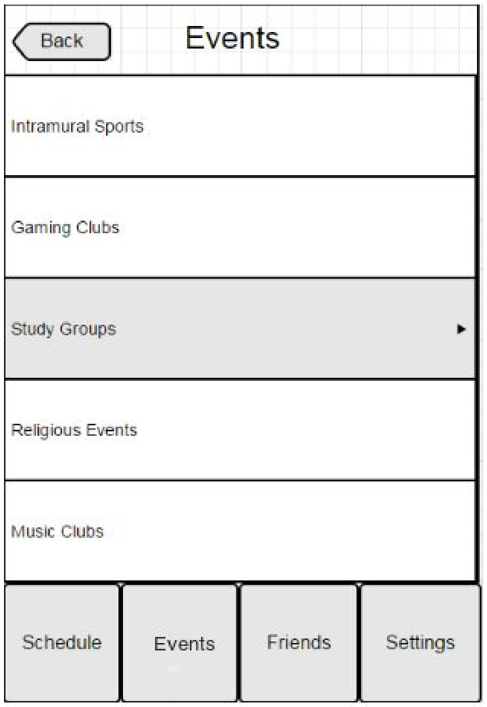
- **UI Mockups:**

Main view



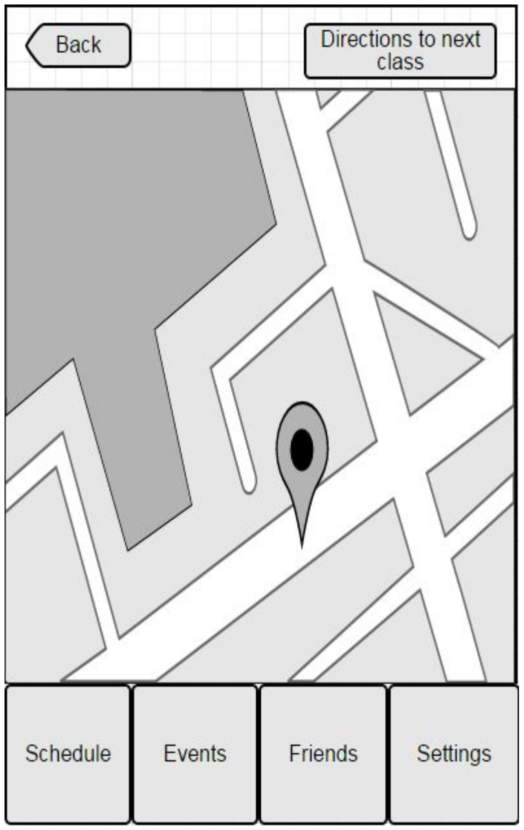
This is the screen that the user will first see when opening the application. Our Main View UI features a calendar view along with two buttons on the top (Map and Add Classes), along with a tab list of the main features on the bottom (Schedule, Events, Friends, and Settings). From here, the user can select which feature to use first.

Event view



Displayed here is our Events UI mockup. Here, we wanted to feature a menu where the user can select the type of event and can see the related events from the option selected.

Maps view

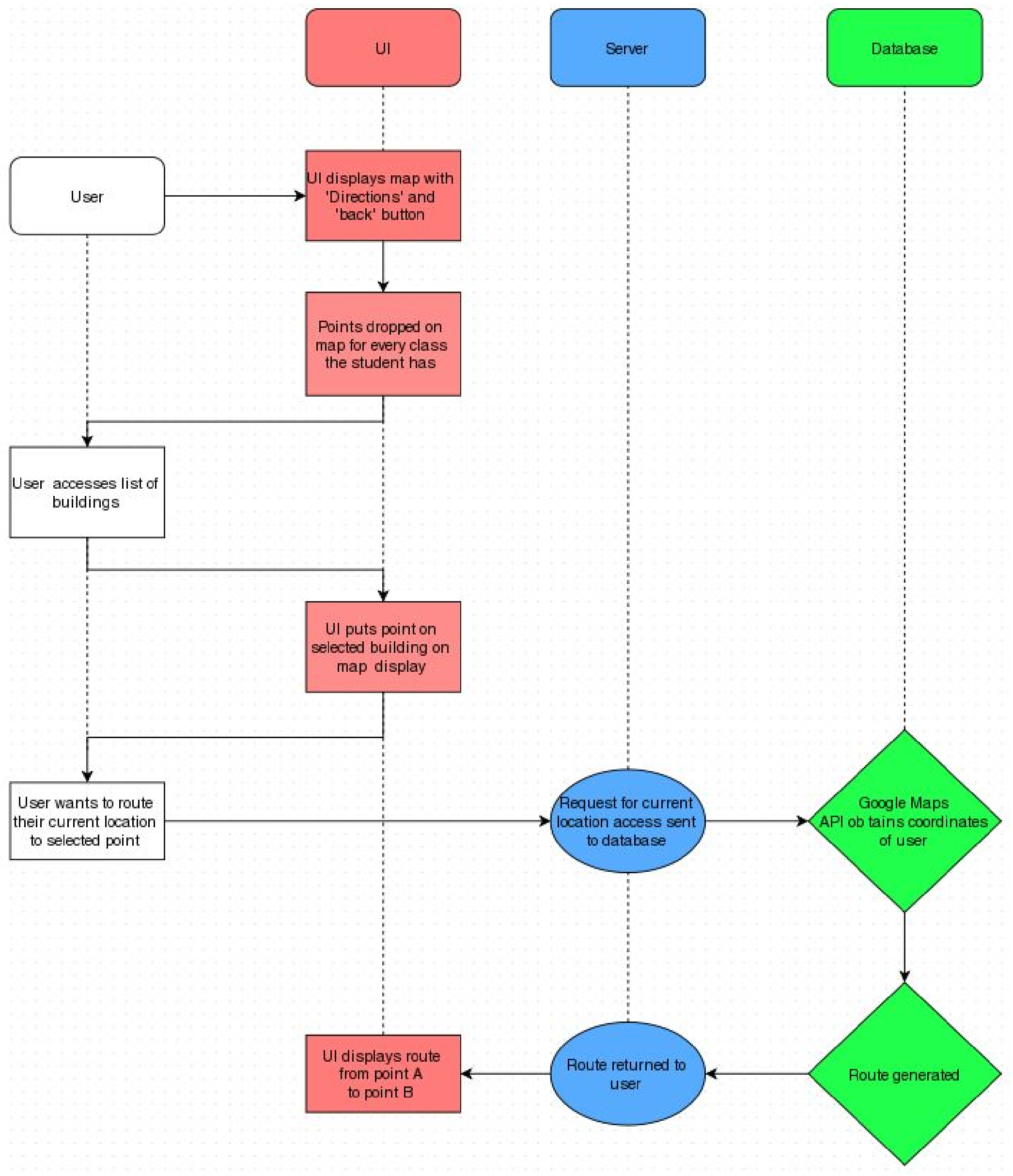


The picture here roughly estimates what the user will see when the user clicks on the “Maps” tab. The main object displayed is the map itself, and there is a button which allows the user to obtain directions to their next class.

- **Sequence Diagrams**

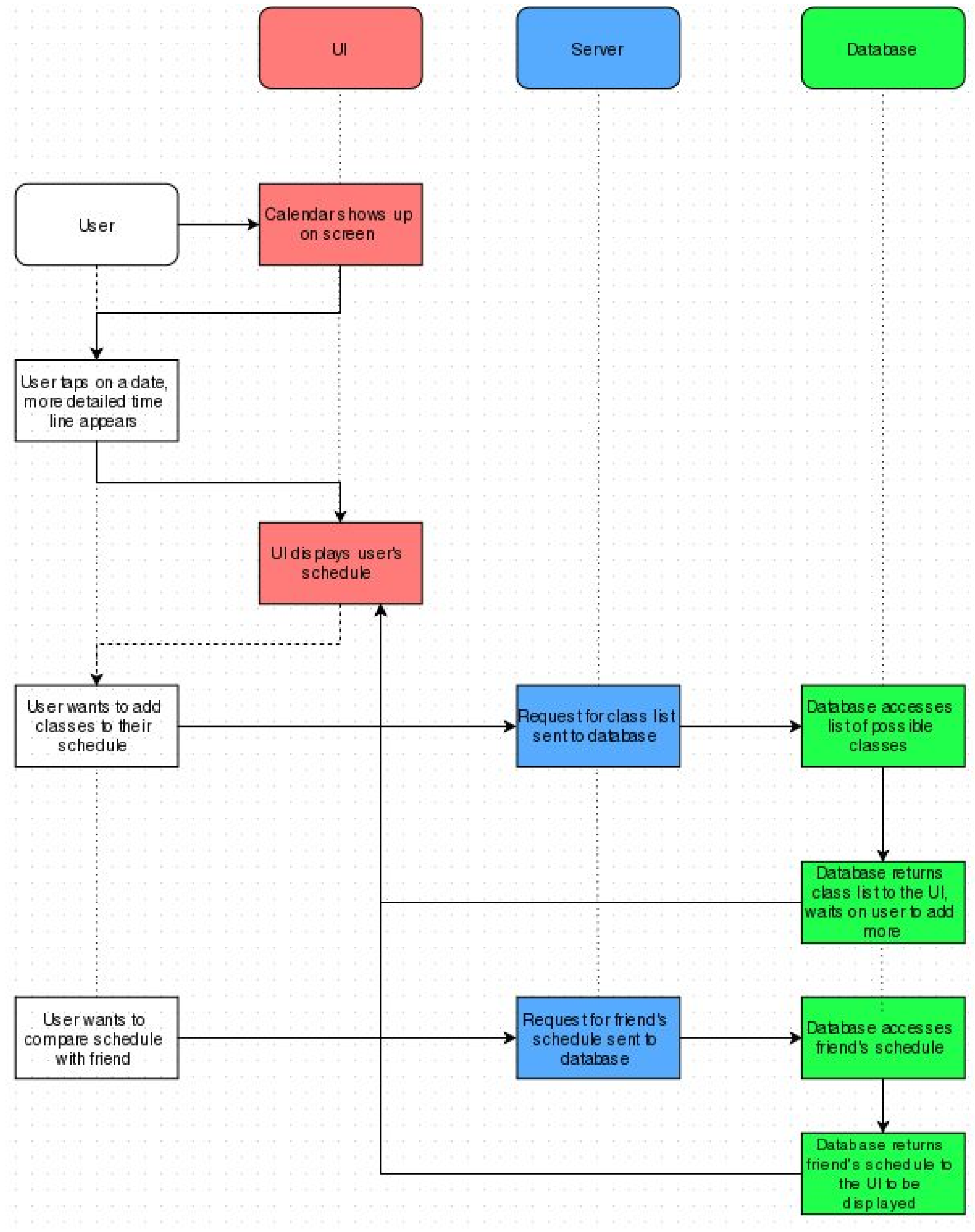
Map Sequence Diagram

The user will be able to interact with a map section of the app to see where their classes are and decide whether or not they want to route a path their via Google maps.


UI
Server
Database
User
UI displays map with 'Directions' and 'back' button
Points dropped on map for every class the student has
User accesses list of buildings
UI puts point on selected building on map display
User wants to route their current location to selected point
Request for current location access sent to database
Google Maps API obtains coordinates of user
Route generated
Route returned to user
UI displays route from point A to point B

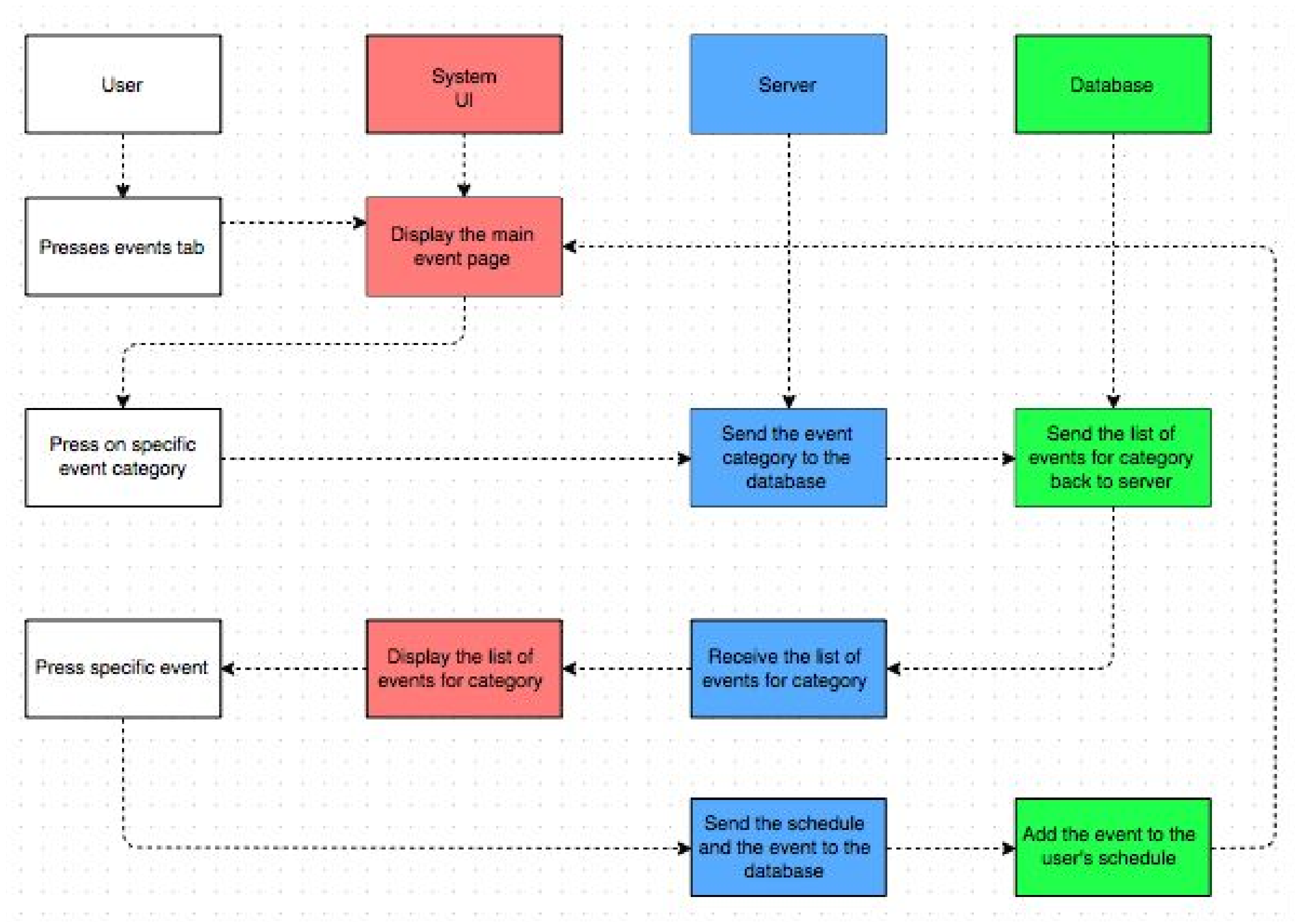
Schedule Sequence Diagram

The user will be able to view their schedule, add more classes to their schedule, and compare their schedule with friends.


UI
Server
Database
User
Calendar shows up on screen
User taps on a date, more detailed time line appears
UI displays user's schedule
User wants to add classes to their schedule
Request for class list sent to database
Database accesses list of possible classes
Database returns class list to the UI, waits on user to add more
User wants to compare schedule with friend
Request for friend's schedule sent to database
Database accesses friend's schedule
Database returns friend's schedule to the UI to be displayed

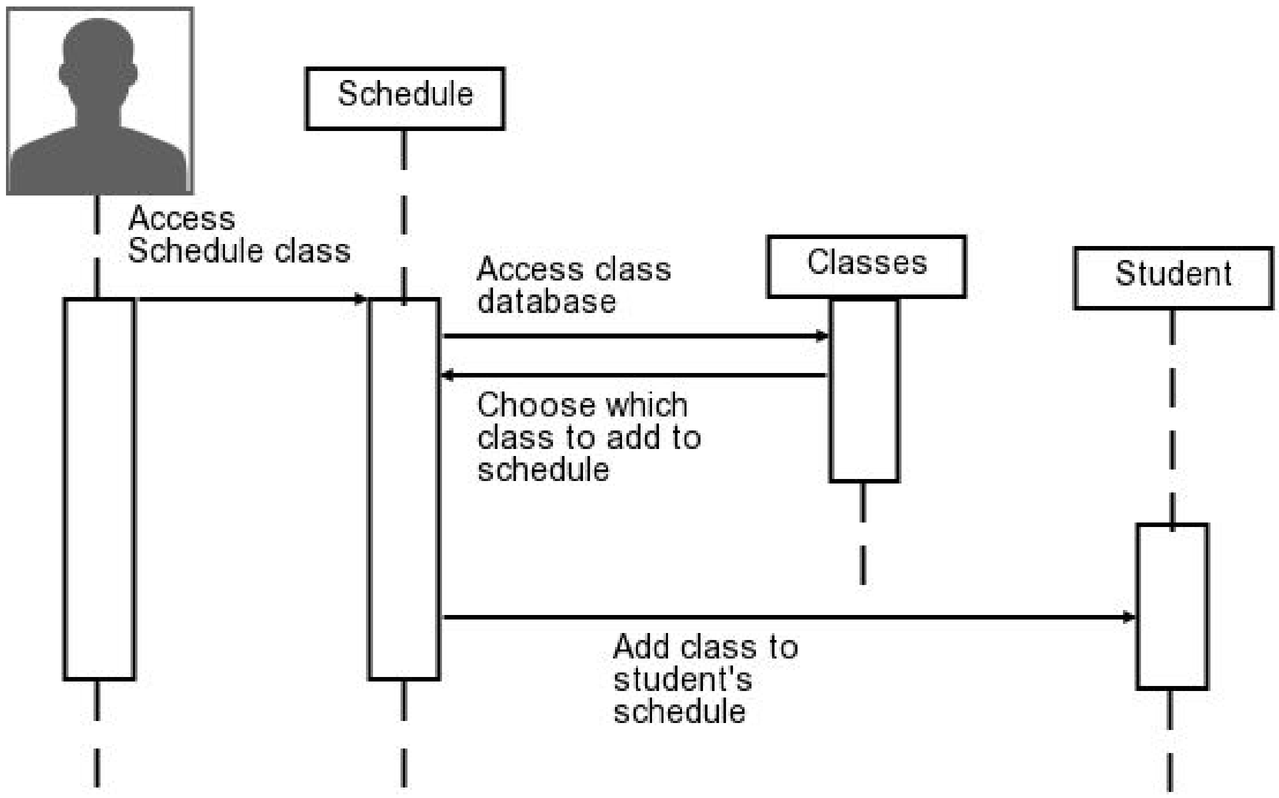
Event Sequence Diagram

The user will be able to access the events page and add any organization's events to their schedule.

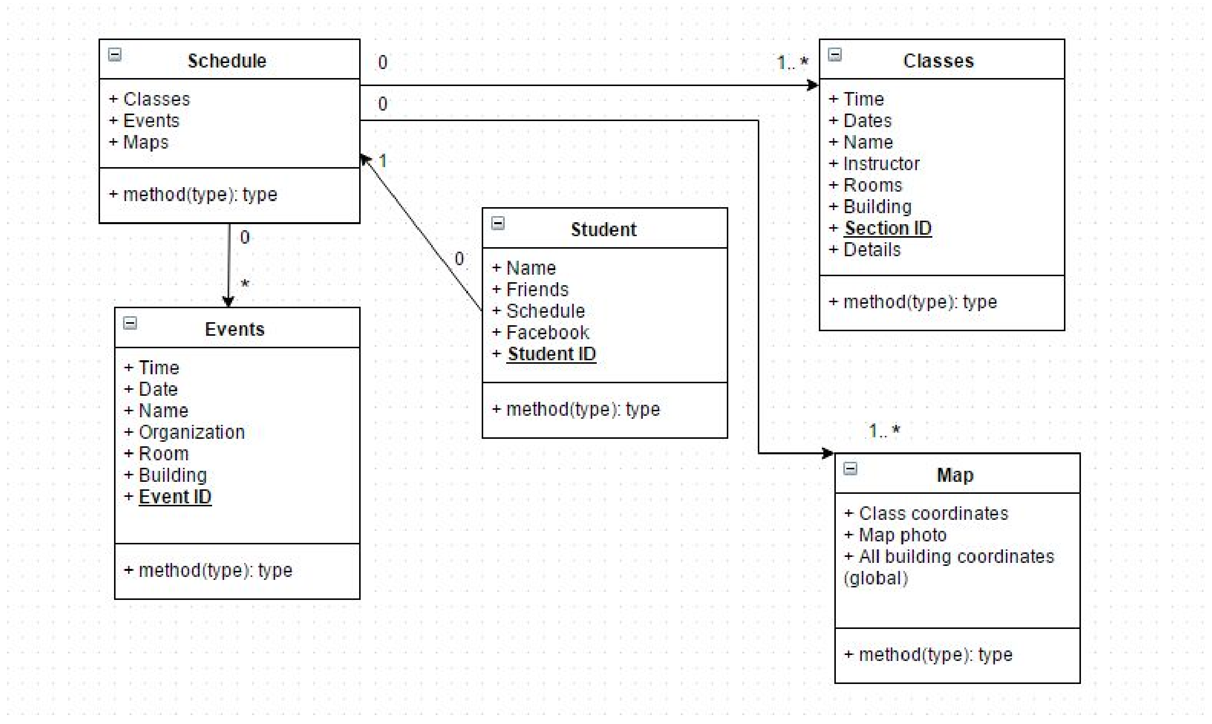

User
System UI
Server
Database
Presses events tab
Display the main event page
Press on specific event category
Send the event category to the database
Send the list of events for category back to server
Press specific event
Display the list of events for category
Receive the list of events for category
Send the schedule and the event to the database
Add the event to the user's schedule

Adding Classes Sequence Diagram

The user will be able to add classes to their schedule and make changes if need be.



UML Diagram



- We will have a “**Student**” class that contains information such as the student’s name, their friends, schedule, student ID, and Facebook login details.
- The “**Classes**” class will contain information about each class. This includes the name of the class, time and days it is held, the instructor, the building and room number the class is held in, and a short description of the class.
- The “**Schedule**” class will display a visual representation of the student’s schedule, including classes and other events.
- The “**Events**” class is needed to store information about extracurricular activities that a student may take part in. Variables will be used to store the name of the event, time and date it meets, the name of the organization that runs the event, and the room where the event is held.
- Finally, a “**Maps**” class will be used to store variables necessary for our map GUI. We will store the coordinates of every building and a map photo of Purdue.
- Our class called “Student” will use the “Schedule” class to access the courses the student is taking.
- The “Schedule” class includes instances of the “Classes” class in order to populate the schedule GUI and course list. It will also have access to any number of instances of the “Events” class so that a schedule can be compiled from the student’s extracurricular activities.
- The “Schedule” class will need to provide information to the “Maps” class so that it can paint the map of the Purdue campus for each building the student has classes in.