CONNEC+

Design Document

Connec+ (ConnectPlus)

Team 9

Cameron Glass, Calvin Henry, Victoria Oldson, Nirali Rai, and Joey Vinyard

Index

Purpose

Current social media platforms are designed around the concept of communicating with those you already know. While fine for communicating with friends on a day to day basis, it is not helpful for making new friends. This is especially apparent on college campuses, where thousands of students come from various backgrounds, and may find it difficult to find others similar to them. Our solution is an application that lets students find others nearby who may have similar interests, hobbies, and classes. By combining the worlds of technology and real life, we can expand students' social network while helping them to have a healthier college experience.

Our targeted users, college students, spend a lot of time on Facebook, Twitter, Instagram, Snapchat and other social media platforms. However, this method of communication often leaves students feeling alone, and with fewer meaningful friendships, contributing to high rates of loneliness. For example, many students coming from different parts of the world do not know any other students who attend their university, and this application can allow them to find those from the same country or region. Much of the interactions that these students experience occurs over phones and computers, and lacks the companionship that spending time in person can provide. Our goal as developers is to encourage face to face interaction between students and make new friends that they may have otherwise not met.

Design Outline

Connec+ will implement a client-server model to allow users to share their data with others in their area. Since users choose when to set themselves as visible, we will be using a Client-Server model to populate our database as well.

Server

1. The Server will provide a layer of abstraction for our database. It will facilitate communication between the
    a. Database and Client
    b. Client and Linked API's (Social Media, Blackboard, etc)
    c. Database and Linked API's
2. The server will accept requests from the client to load their data into the database
3. The client will make calls to the Linked API's based on requests from the client, and then will *temporarily* load that data into the database.
4. The client will accept requests from the client to remove their data from the database, and it will then be deleted until the client requests it be added again.
5. The server will accept requests from the client, query the database and retrieve the requested information.

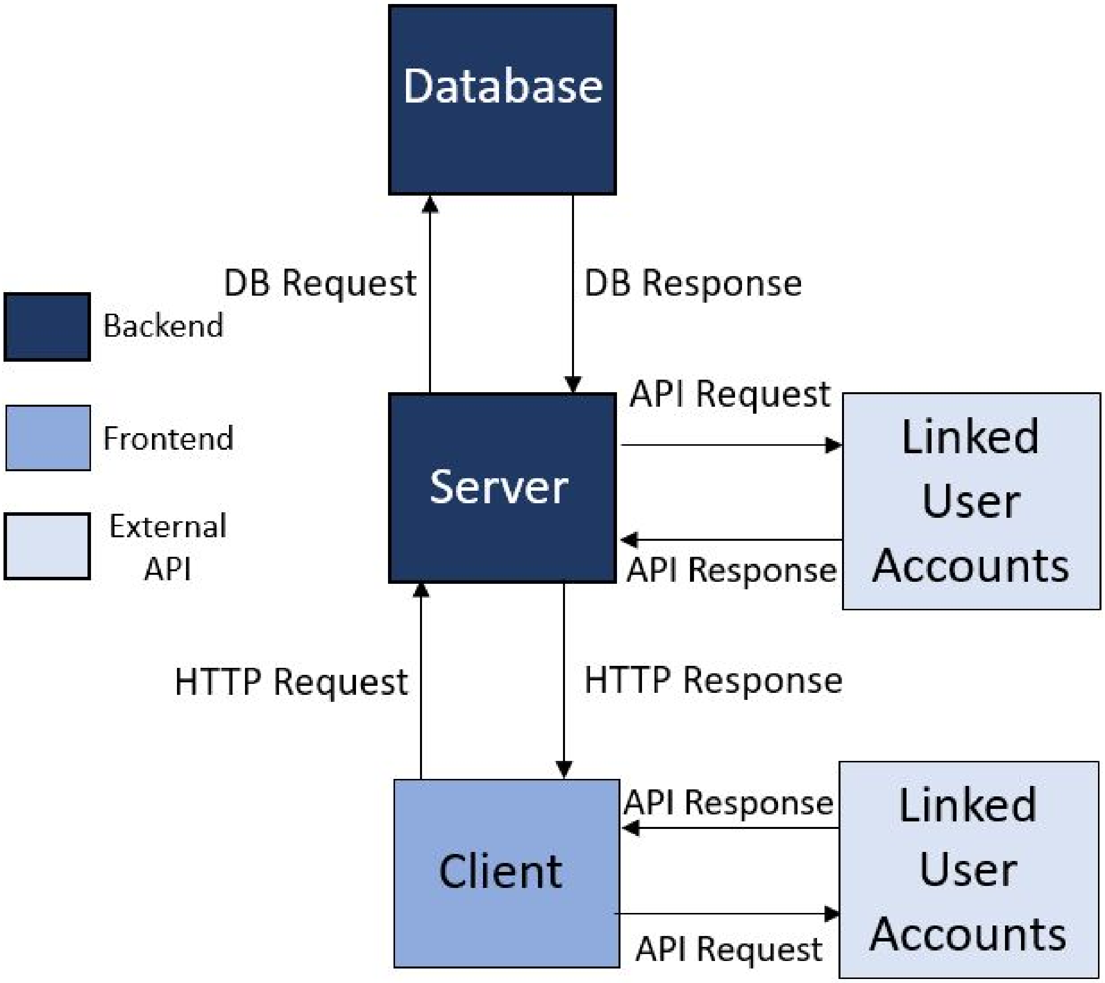
Client

1. Although we have three potential platforms that our client program may be able to run on, all three of them will interact with the rest of our project architecture in the same way.
2. The client, whether as a web page or a mobile application will be the interface that gives the users access to our system.
3. The client will retrieve and send data to our server.

4. The client will make requests to the Google Maps API to assist in displaying responses from the server.

Database

1. The database will store basic information about the user and their location.
2. The database will be populated by information provided by the server.
3. The database will fulfill requests made by client and server for information.
4. The database will not interact directly with either the client, or any of the external APIs. All database interaction will occur through the server.



Design Issues

Functional Issues

1. How should the user be able to create a new account?
   a. Choice from already existing accounts on other social media platforms
   b. Need to make a new account
   c. Both

Decision: When a user wants to create a new account, they must create an account that is specific to Connec+. There are three main reasons for this: First, Connec+ is independent and separate from other platforms, such as Facebook, LinkedIn, Blackboard, and Instagram. Second, this platform allows users to connect other accounts to their Connec+ account, allowing them to have multiple account attachments, instead on being forced to choose one. For example, logging in with a Facebook account allows for linking their Connec+ account with Facebook only. Lastly,, this allows for a separate account, and does not require the user to have additional social media accounts to have access to use this one.

2. How should potential connections be displayed to users
   a. Shown on a map
   b. Shown in a list
   c. Both

Decision: We decided to implement both a map and list view. The only downside to this option is that it will require slightly more work when implementing the functionality of the application. We decided that this was definitely worth giving the users an option to switch between an intuitive list view, and a more graphical map view depending on their preference.

3. Should messaging be a feature?
   a. Yes
   b. No
   c. In a limited capacity

Decision: We decided that messaging should be implemented in a limited capacity for two main reasons. First, this application's goal is to increase face to face communication and interaction. Having an unfettered messaging feature implemented in this application would defeat this goal. Second, some messaging features will be available to get location information, or create a meeting time and place.

4. How should location services update?
    a. As the user moves
    b. When the user opens the application and confirms current location

Decision: When the application is opened, it will confirm the location of the user, instead of updating as the user moves. This will decrease the amount of updates the user posts to the backend, as well as protect the user's privacy. By updating only when the user opens the application or chooses to update their location, we can prevent continuous data transfers from the user. This will also increase the accuracy of the app, because users have to confirm their location, in order to reduce inaccuracies in location services across devices.

5. How long should invisibility mode last?
    a. Until the user disables it
    b. Custom length
    c. Both

Decision: We decided that the user will be able to choose to either set their visibility manually or have it set on a timer. Both of these options have positive and negative aspects that depend on the preference of the individual user. With the choice of a manual toggle, the user will be able to select a state that will make them invisible for an indefinite amount of time. In addition to this, we will provide a custom length timer. When the user selects this, they can remain invisible for an allotted time to better match their daily schedule. Combining both aspects provides for a more well tailored end user experience.

Non-Functional Issues

1. What platform should we develop the application for first?
    - **a. Web application**
    - b. Android application
    - c. IOS application

Decision: The web application has been selected to be developed first for multiple reasons. First, our team has multiple members whose primary interests lie in web development. This means that the start up cost of developing the web application is much lower, because less time will have to be invested in learning new frameworks. Second, we wanted to build an application that can be used ubiquitously, regardless of what device the user owns. Web applications are able to be scaled and used on almost any modern device. This means that only a small percentage of the potential market will be unable to use the application. With having a web application as the initial platform, it allows everyone to have access, as long as they have internet connection.

2. What platform is appropriate for backend implementation?
    - a. Java
    - **b. Node.js**
    - c. Python

Decision: For the backend implementation of Connec+, we have chosen Node.js as our runtime environment for two reasons. First, Node.js provides a runtime with asynchronous code execution. This makes for a more productive server, as an intensive request does not cause the entire server to hang. Second, Node.js implements a two-way communication paradigm, where both the client and server can initiate communication. This becomes particularly useful when websockets are implemented, which enable real time communication between the server and client.

3. <u>What framework is appropriate for the frontend implementation?</u>
    a. **Angular**
    b. React
    c. Static HTML/JS

Decision: Each of the above frameworks are well supported, and could be used to build Connec+. However, we decided to implement our frontend with the latest version Angular. There are two main reasons for this choice. First, several of the group members have experience with Angular, which will decrease the cost of developing the first platform of the application. Second, Angular brings in TypeScript and Components. This will allow us to structure our web application in a much more meaningful way.

4. <u>What platform should be chosen when moving the web application to mobile?</u>
    a. **Android application**
    b. iOS application

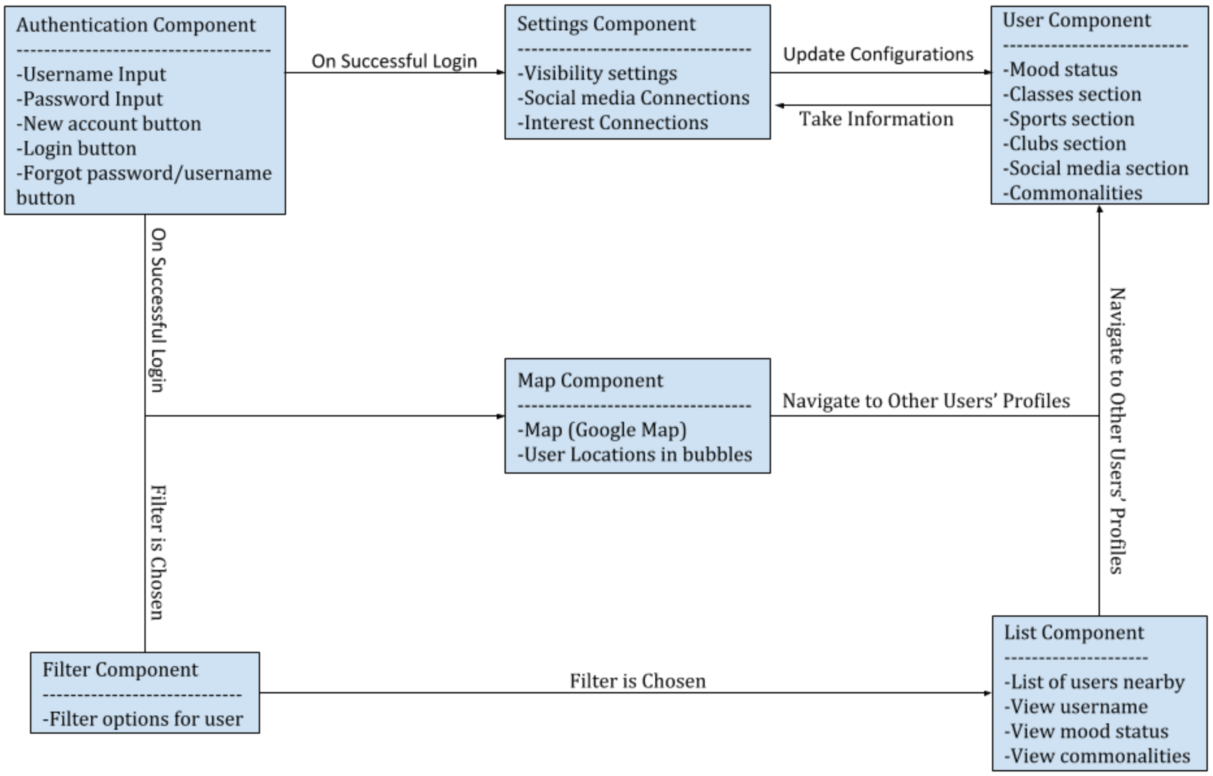
Decision: For the second choice of platform, we decided to develop an Android application, for two main reasons. First, the cost of entry for our team to develop an iOS application would be much greater than an android application. While the majority of the team lacks experience in both android and iOS development, the entirety of the team has experience with Java, whereas there is no Swift experience amongst our team. Second, we will be able to more rapidly prototype the app, due to the fact that the Android SDK is an environment that has existed much longer than Swift. This means that design issues and bugs will be more easily resolved as there are more resources readily available.

5. <u>What database should be used?</u>
    a. Firebase
    b. MongoDB
    c. SQL

Decision: Although all three of the options are well documented, cross-platform, and reliable, Firebase is the most logical choice. By utilizing Firebase in our backend, we decrease the demand on the hardware of our server by lowering the amount of work that the backend is in charge of. Also, by not integrating our database with our backend, we mitigate the risk of a total system failure. If our backend API goes down, there is no risk of the data in our Firebase database being damaged.

Design Details

Component Level Design of the System



Description of Components and their Interaction

The main web components of Connec+ are as follows: Authentication, Map, List, Filter, User, and Settings. The interaction between these components is described below:

- Authentication Component
    - Description
        - The first page a user will see when opening the website or application. It will ask them to either login with an existing account, or create a new account with proper username and password.

- Interaction
    - On successful login, the authentication component will move the user from the current component to the Map Component
    - On successful login, the authentication component will populate the Settings Component with the user's basic information. A user can then add more information after successful login

- Map Component
  - Description
    - A graphical view of the user's currently surroundings. Included in this map will be bubbles showing the location of other nearby users that are available to be connected with. The map is able to be scrolled to show more area, and user bubbles can be clicked to show more details and any similarities.
  - Interaction
    - The user is able to navigate to other users' profiles by selecting their bubbles on the map. This will navigate the user to the User Component.
- List Component
  - Description
    - A tabulated view of nearby users that can be connected with. Included in this is component is general information about the user such as: mood, distance, and similar interests. The user of the application will be able to scroll through the list and navigate into other users' profiles.
  - Interaction
    - The user will be able to select other connections, which will take the user to the User Component displaying the data of the selected connection.
- Filter Component

- Description
  - A component where the user can configure filters to change what types of users are displayed in the List Component and Map Component.
- Interaction
  - Once a filter is chosen, the Map Component will be updated to show only the user bubbles of those who fit within the filter. This will allow users to see only those around them that have similarities.
  - Once a filter is chosen, the List Component will be updated to display only the user accounts of those who fit within this filter. This will allow users to see only those around them that have similarities.

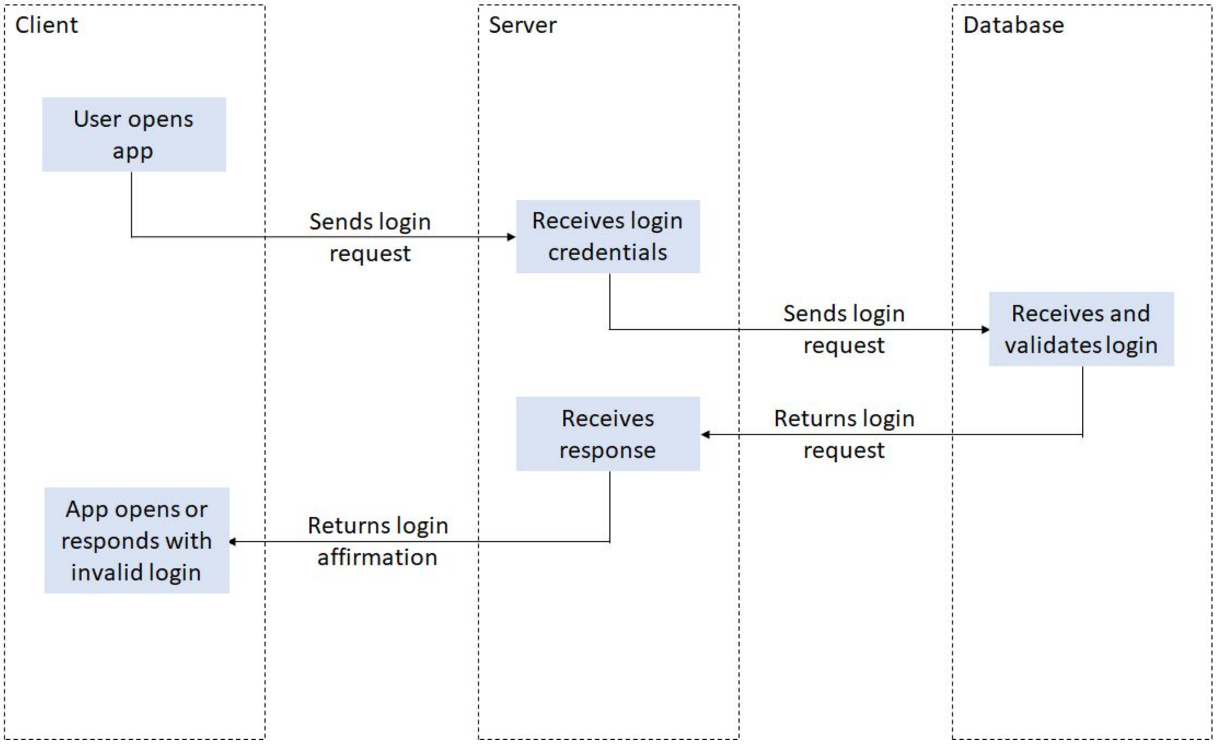
- User Component
  - Description
    - A view of the selected user that displays all public information on their account. This may include mood status, classes, clubs, sports, and social media connections. The user will be able to see commonalities between themselves and the selected user, and give them the ability to contact them.
  - Interaction
    - The User Component will take a user's information from the Settings Component, where a user can choose what information is publicly available.
- Settings Component
  - Description
    - A list view of configurations that the user can tweak to their liking. This would include, but not be limited to: visibility,

interests, social media connections, and location services. These would be saved to the user settings schema in

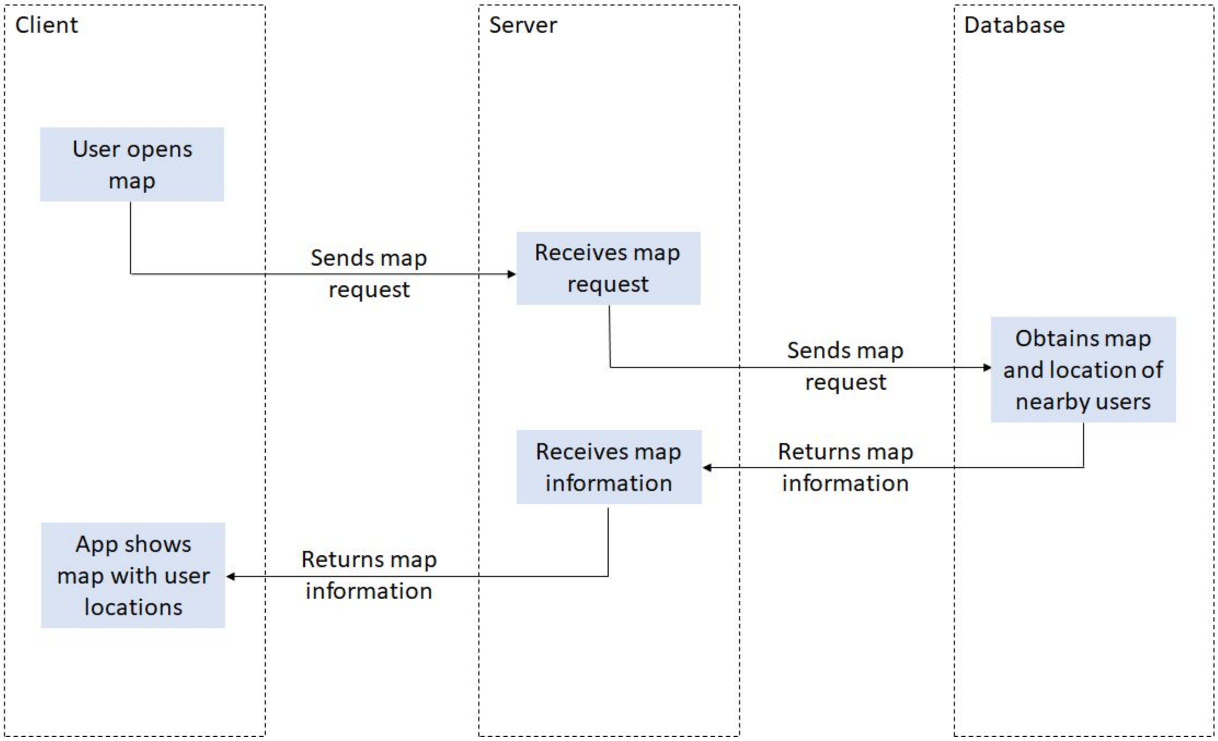
- Interaction
    - The Settings Component will communicate with the backend to fetch and update the user's configurations in the database. When finished, it will navigate back to the User Component, and update the relevant fields in the profile.

Sequence Diagrams

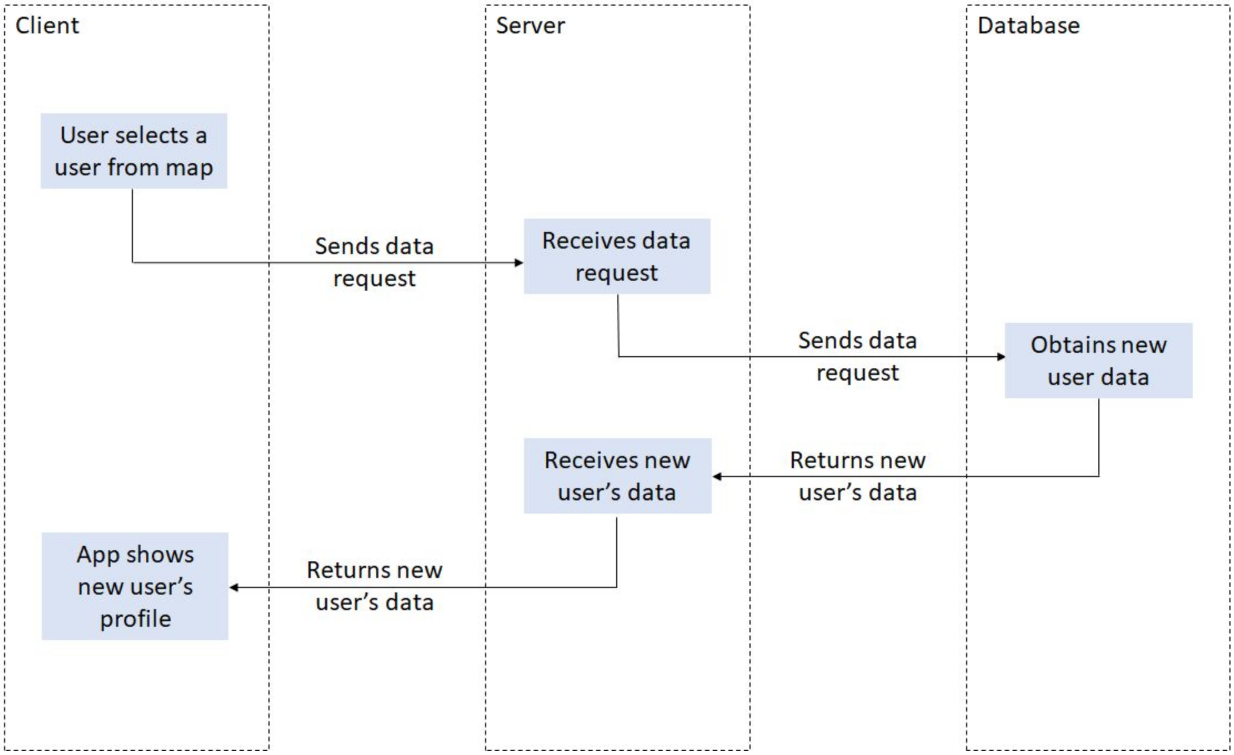
Login


Client
Server
Database
User opens app
Sends login request
Receives login credentials
Sends login request
Receives and validates login
Receives response
Returns login request
App opens or responds with invalid login
Returns login affirmation

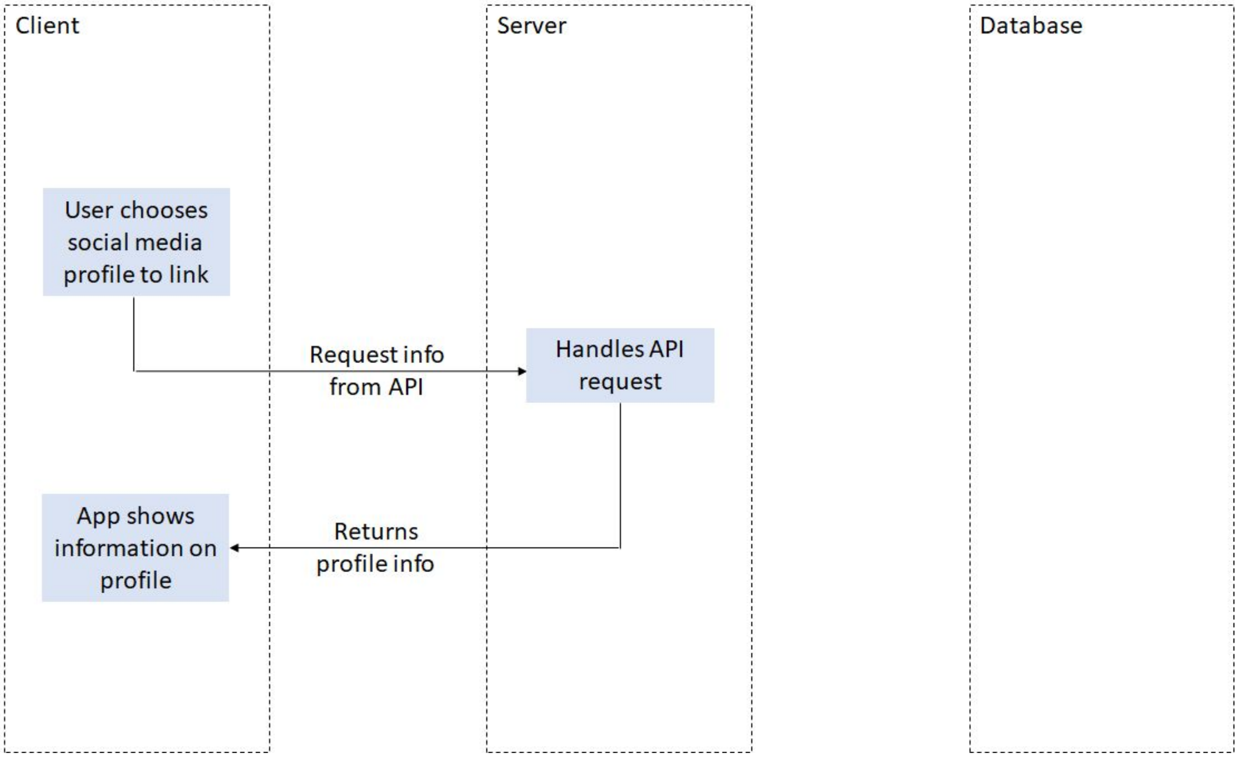
Open map



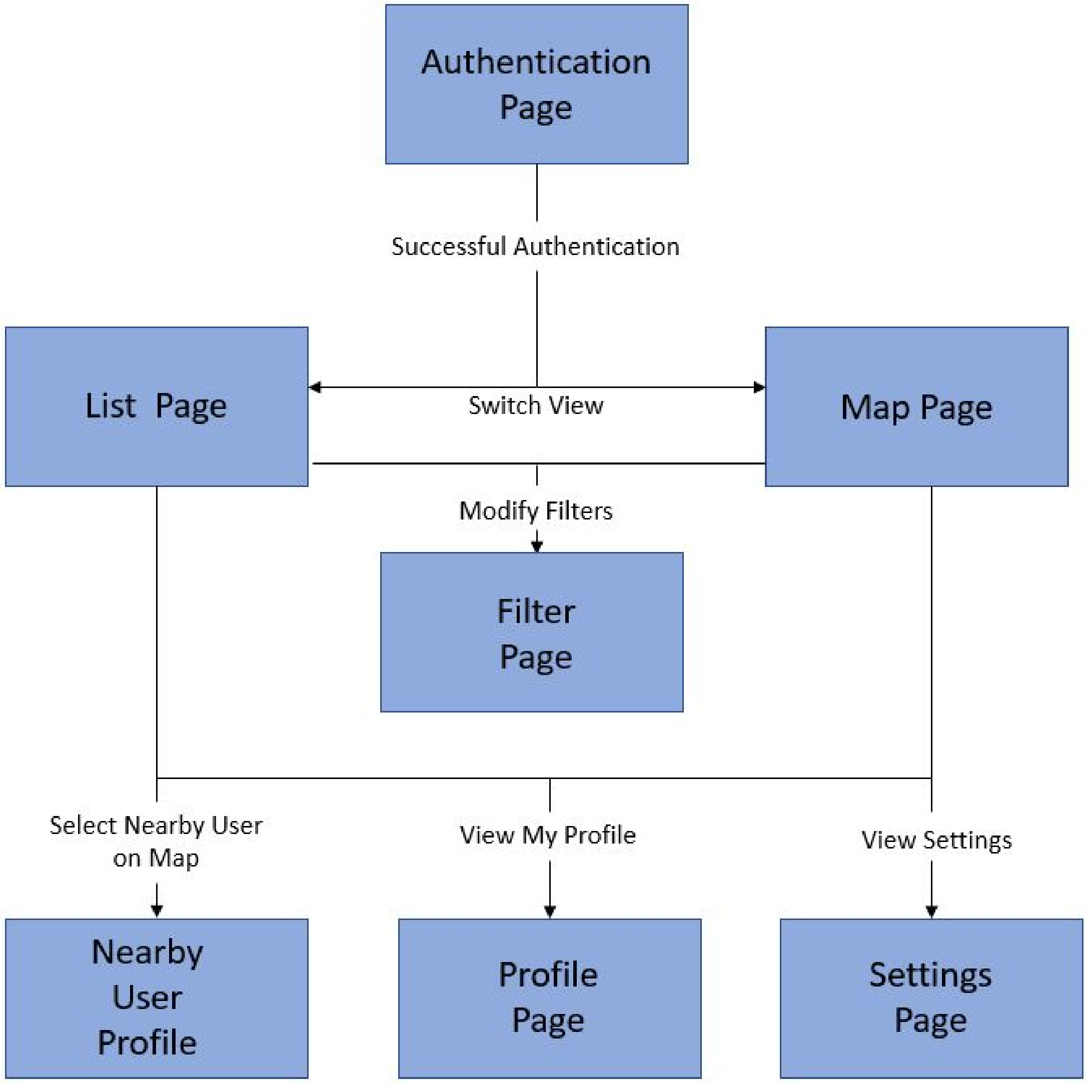
Open nearby user's profile



Link Social Media Profiles

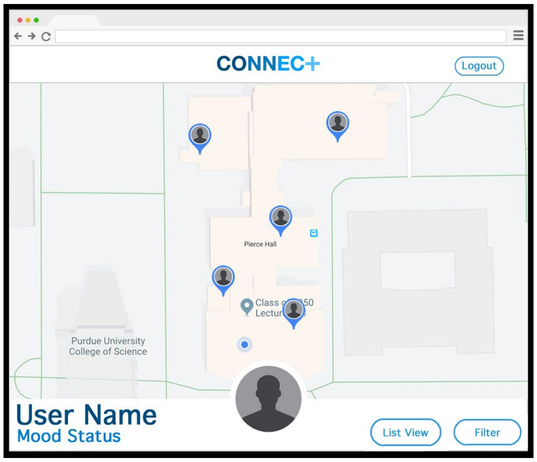
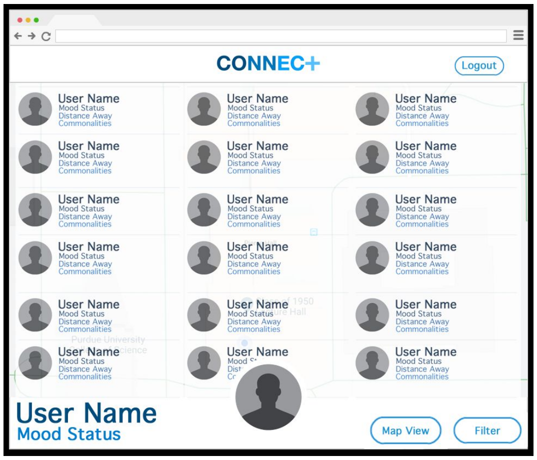
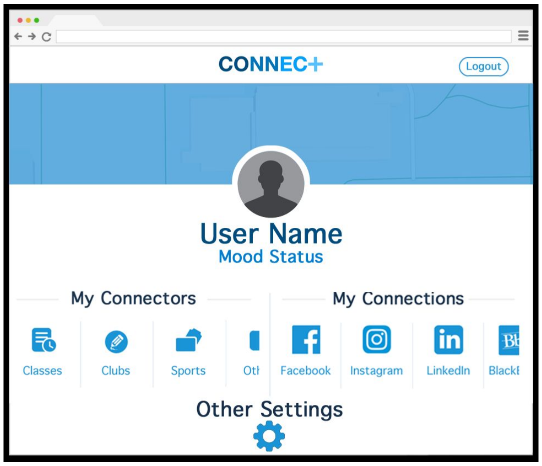


State Diagram

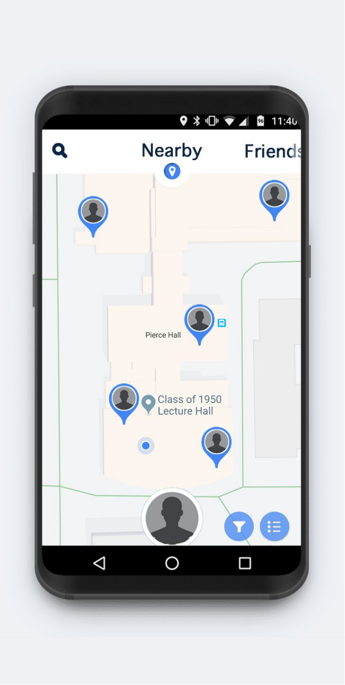


UI Mockups

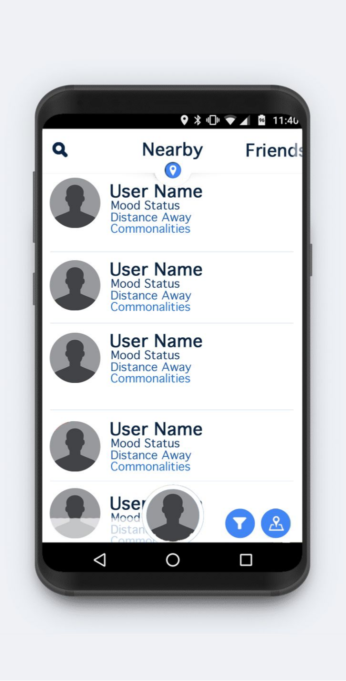
Website

 CONNEC+ Logout Pierce Hall Class of 1950 Lecture Hall Purdue University College of Science User Name Mood Status List View Filter	Map View
 CONNEC+ Logout User Name Mood Status Distance Away Commonalities User Name Mood Status Map View Filter	List View
 CONNEC+ Logout User Name Mood Status My Connectors My Connections Classes Clubs Sports Otl Facebook Instagram LinkedIn Blackl Other Settings	User View

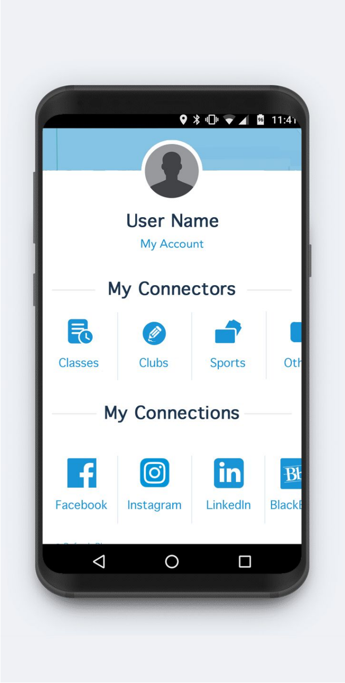
Android



Map View



List View



User Page