Team

| Name | UID |
|---|---|
| Dave Ho | 105-327-667 |
| Samuel Alsup | 805-371-633 |
| Rishab Jain | 005-409-041 |
| Matthew Ruiz | 205-400-681 |

## popfund: Project check-in

**Overall:** Tech stack has been decided. We will use React for the frontend client side services. Alongside, we will use Node JS + Express to handle backend/server services and handle API requests. For storage of dynamic data, including user, business, and item data, we will use MongoDB with Atlas as the cloud hosting service. The repository has been made with a created react-app that contains all of these components. Please refer to the repository linked below as well as the highlighted points on both Frontend side and Backend side.

**Link to GitHub Repo:** https://github.com/popfund/popfund

**Frontend**

- Switched frontend framework/language from Swift to React
  - Learning curve to learn new framework - many similarities however
- Started working on the main list-view page
  - List-view of local businesses
  - Including text areas for meta data for each list-view item
- Making an attractive UI for the business list-view page
  - Learning how to incorporate Figma and other UI design wireframes into React
- Next steps:
  - Create a search bar for the list-view
  - Implement a business page for each business (upon clicking via list-view item)
  - Work with backend to fetch business names/data to display on list-view
  - Start working on implementing the map/GPS page

**Backend**

- Installed express node module
- Set up server to listen to requests locally
- Set up storage
  - Create new storage cluster on MongoDB Atlas
  - Install mongodb node module
  - Successfully connected to MongoDB Cluster
    - Connect to client

        - Use client to access sample records
- Test API Routing
    - Create sample API routes
    - Successfully test calls to sample API via Postman
- Next Steps:
    - Create getBusinesses route to return proximate businesses objects to Frontend
        - A way to loop through businesses database
        - Function that calculates euclidian distance from businesses
        - A way to check if a business's euclidean distance is less than the threshold
        - Wrapping the list of businesses that meet the conditions in JSON format
        - Sending the JSON formatted list via res.send()