Sprint 1

Sanah Imani, Jainam Gala, Snigdha Tiwari

Web App Purpose

- A dynamic web application that intends to serve as a student-friendly platform for buying/selling items between students.
- Students will be able to more likely find school-specific items, usually at cheaper resell prices with quicker delivery/pickup cycles!
- The intended clients for this project are web app users of CMU.

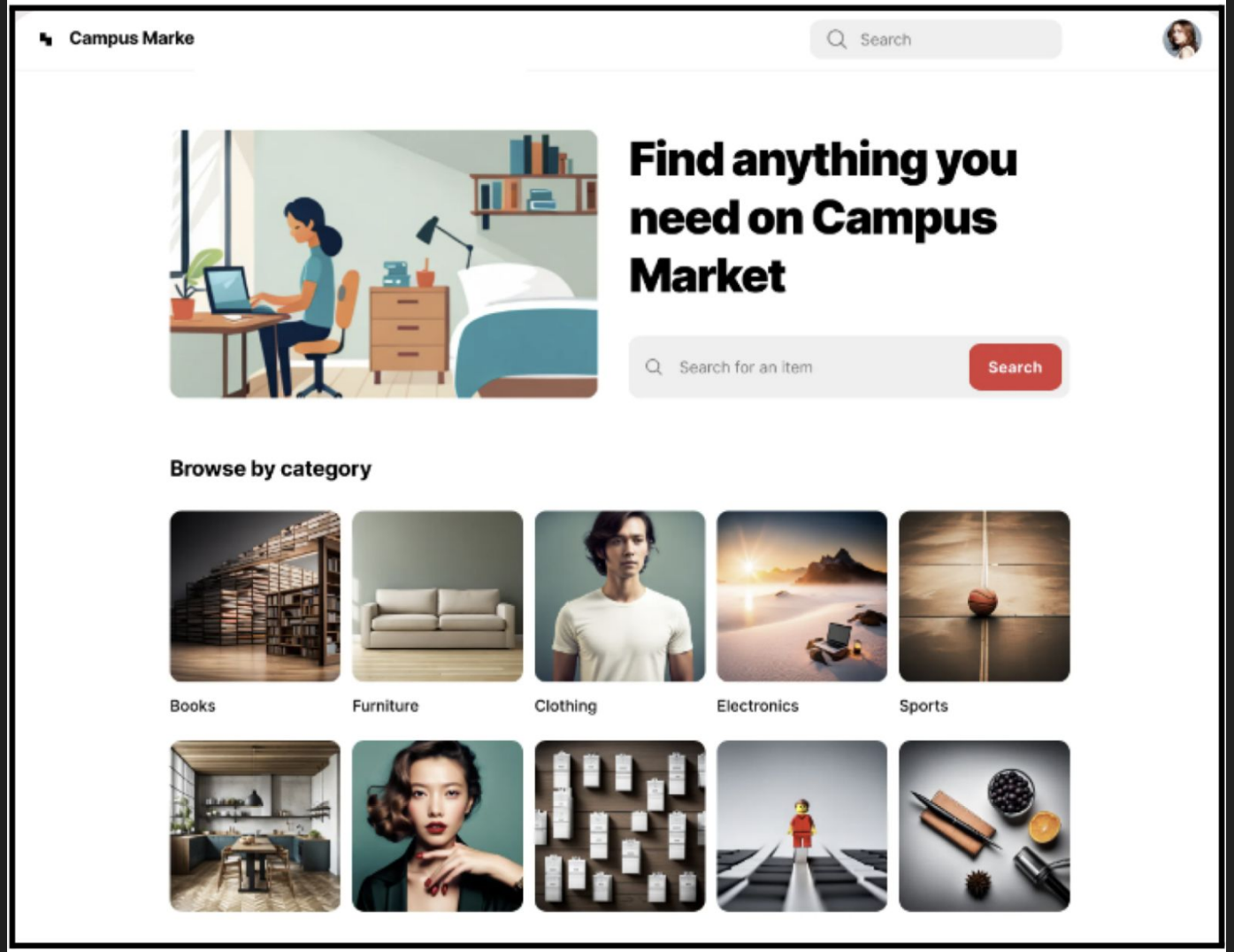
Product Features

- User can publish an item to sell
- User can search for an item in the marketplace by keyword and category
- User can buy an item
- User can pay through the payment gateway

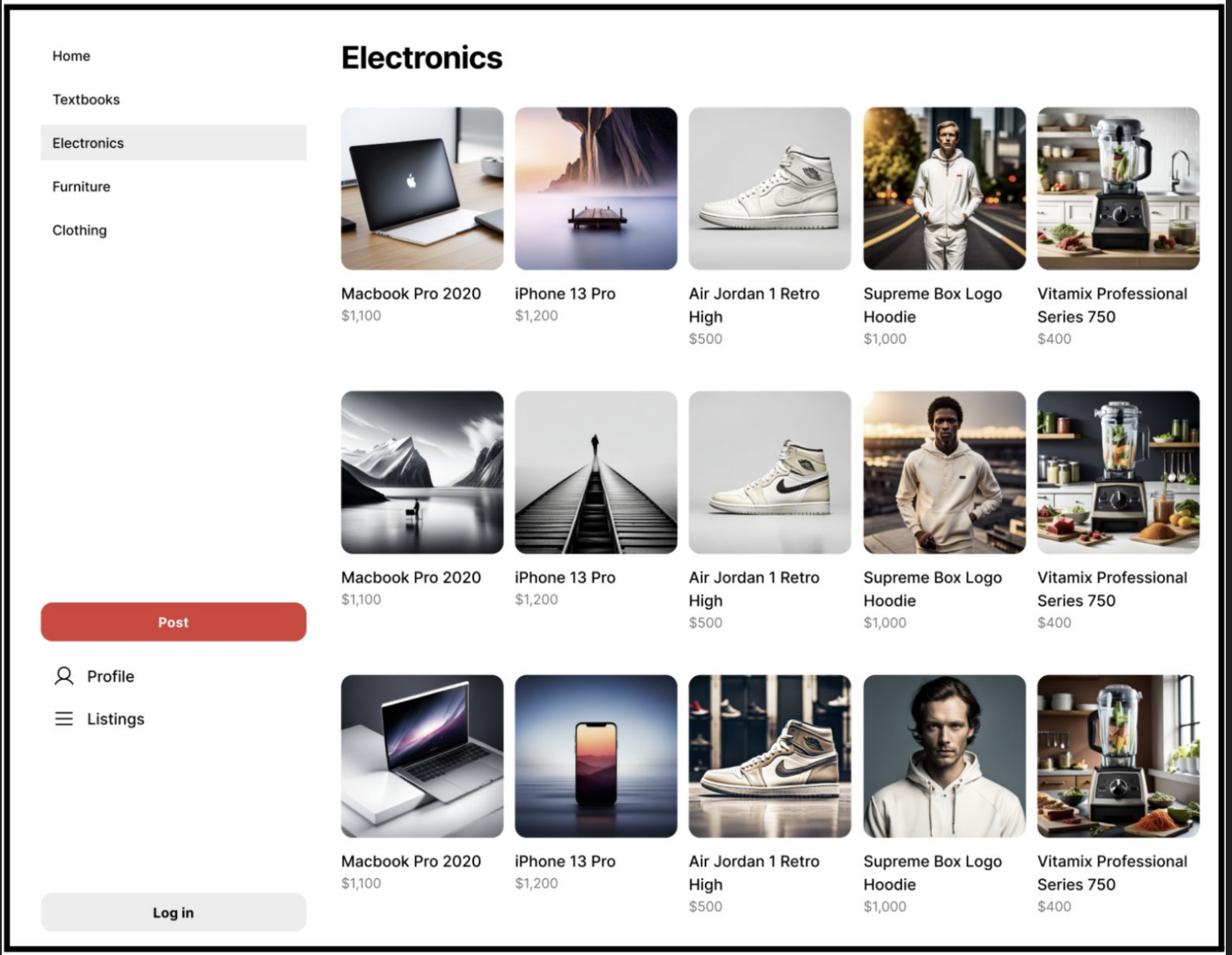
Unique Pages

- Homepage with Hero Section and Categories of Products
- Listings page with all products, search functionality
- Login Page for User
- Sign Up Page for User
- Item listing page where users can post items for sale
- Item detail page where users can view more information about an item
- Payment flow for item purchase
- User profile page with user-settings tab for account personalization, active-listings tab, archived-listings tab

Home Page


Campus Marke
Search
Find anything you need on Campus Market
Search for an item
Search
Browse by category
Books
Furniture
Clothing
Electronics
Sports

Category Wise page



List an Item pag

Campus

List an Item

Title

What are you selling?

Description

Describe your item

Category

Select a category

Price

$0.00

Quantity (optional)

1

Submit

Order Flow page

Campus Market

Search

Review your order

Nike Free Metcon 5 Women's Training Shoes
Size: 11

Subtotal	$120.00
Estimated Shipping	$0.00
Estimated Tax	$10.35
Total	$130.35

Email

First name

Last name

Address

City

State

United States

Save credit card for later use

Continue to Payment

Cancel

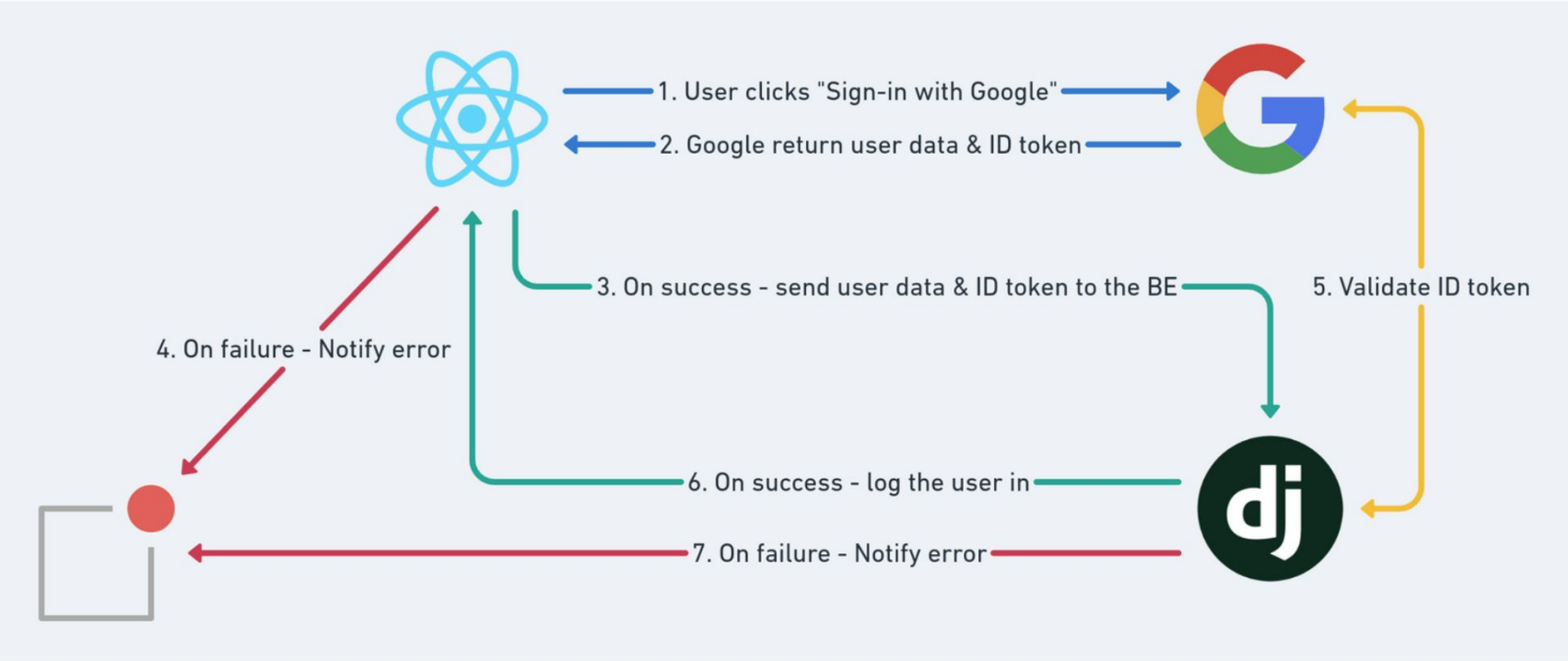
Sprint 1 Goals

Use Google Oauth 2.0 to allow students with an andrew email account to register and login in to their Stuket account.

The account will contain user settings, user profile, listed item history, and orders made through the account. It will also save payment information.

Google Oauth2 with React and Django

Here is a simple diagram that illustrates these steps:



Our Login Flow -

1. The user signs in using “Sign in with Google” button on the client side (React)

2. User provides login credentials for Google on their interface by selecting account and entering password (Google’s UI)

3. Google authenticates the user and returns an “access token” to the user (React)

4. We send this “access token” to our backend, and validate this token with Google (Django)

5. After validating the token, we fetch/create the respective User and return as response (Django)

6. The “access token”, in this case a JWT, is stored as a cookie in the browser for subsequent (post-login) requests (React)

Problems Encountered

Integrating React with Django

To send HTTP requests from React app to django backend, one has two main options.

- Integrated Fetch API (https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API)

- Axios (a promise-based HTTP library that lets developers make requests to either their own or a third-party server to fetch data.)

Storing JWT in a cookie/session storage.

Improvements to Tasks in Sprint 1

Using the cookie to skip login if user is already logged in.

Inclusion of the CSRF token for POST requests (security).

Managing the state of a React app using Redux.

- Redux is a library for managing and updating application state, using events called "actions".
- Avoids passing data through props within increasingly complex designs.

Sprint 2: Preview

- Complete Home page
- Category-wise page
- Item Details page
- Add an Item Form