Team Member Full Name	NetID
Swindar Zhou	**kzhou3**
Samara Jacobo	**sjacobo3**
Kaia Damian	**kdamian**

Features Implemented for Phase 1

- Feature 1.1: Create New User Profile
- Feature 2.1: Create Listings
- Feature 3.1: View All Listings

Persistent Storage Design

We are using SQLite database to persist our data, located in the db.sqlite3 file in the outer course_project folder. Our database includes the tables shown in Figure 1. It contains 2 tables. The `User` table is implemented using Django's built-in User class, only using the attributes name, username, email, and password fields required for Feature 1.1. The 'first_name' built-in Django field acts as the 'name' field. Each user can have zero or many listings. The `Listing` table (which models each listing a user can make) is implemented using a custom model we made, which is a subclass of the Model class inside the models module from django. It includes the following attributes: created_by (a foreign key of the user who created the listing), title, description, price, condition (with choices New, Like New, Used, or Fair), photo, and status (with choices Available or Unavailable).

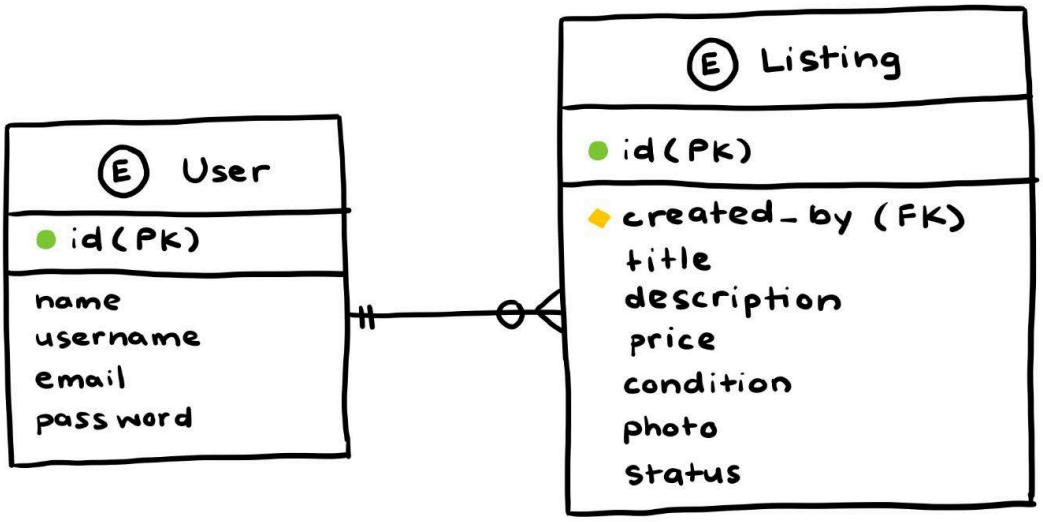

Figure 1. Database Schema with User and Listing Tables

Demonstration of the Features Implemented for Phase 1

Feature 1.1 Create New User Profile

Figure 1 demonstrates the user Register page where registered users can enter their name, email, username, and password to create a new account with those specified credentials. In our backend, we used django.contrib.auth.models API to create a user profile with their name, email, username, and password as displayed in Figure 2 for new users. When the user clicks the black text 'Register' button, a new user profile will be created and stored in the SQLite database.

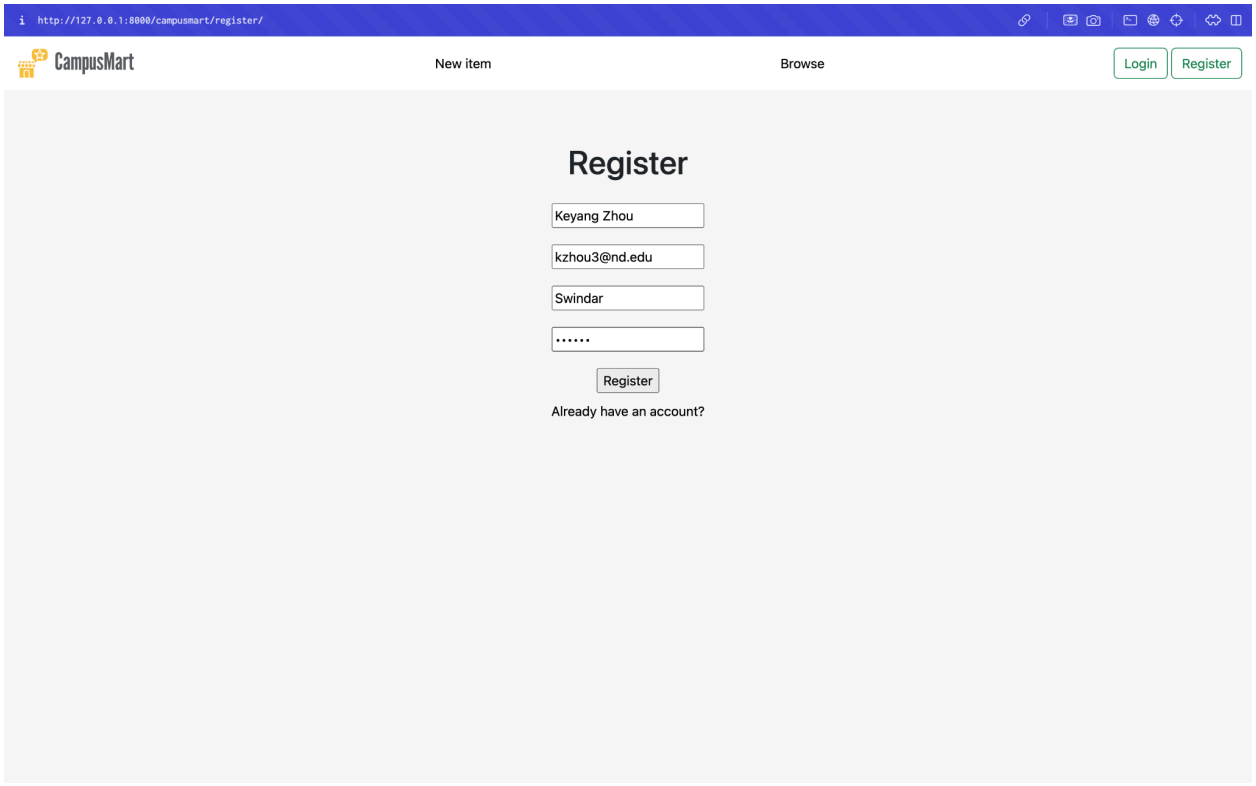

Figure 2. Create User Profile (by clicking 'Register' in the Navigation Bar)

Feature 2.1 Create Listings

Figure 3 demonstrates how the user creates a listing. The user will enter title, description of the listing product, price, and product condition (New, Like New, Used, Fair). The user can also upload an image file to display. The data will be stored in sqlite database based on our Listing class created inside the models.py for our app in django.

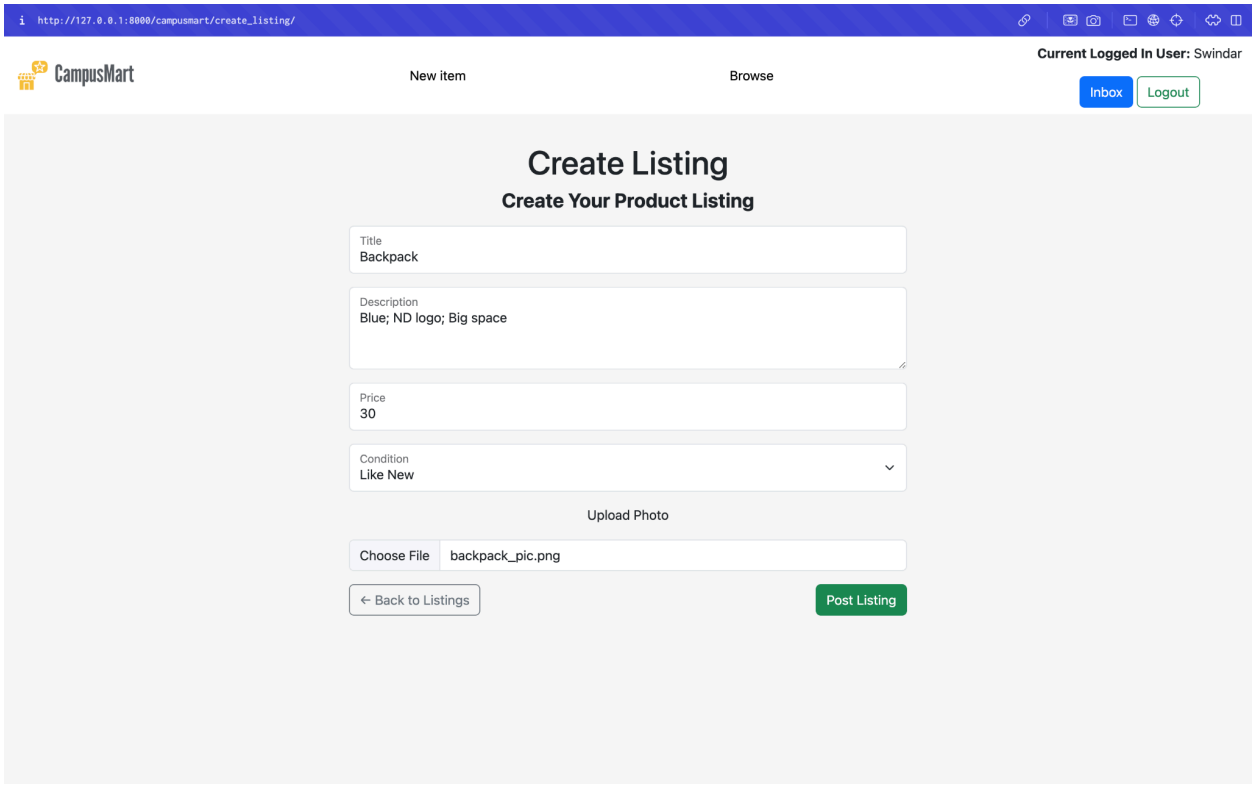

Figure 3. Create Listing (by clicking 'New item' on the Navigation Bar)

Feature 3.1 View All Listings

Figure 4 shows a screenshot of the implementation of Feature 3.1. This feature takes all the listings with the status 'Available' and renders them to the Browse tab. This feature then shows the user the image of the listing, the name and the asking price. These listings are also clickable, taking the user to a detailed view of the listing with additional information like a description and its condition. While not part of feature 3.1, the user can also search an item based on its description. If the page were to have more than twenty listings, there would be multiple pages that the user could go to, following the directions of 20 listings per page.

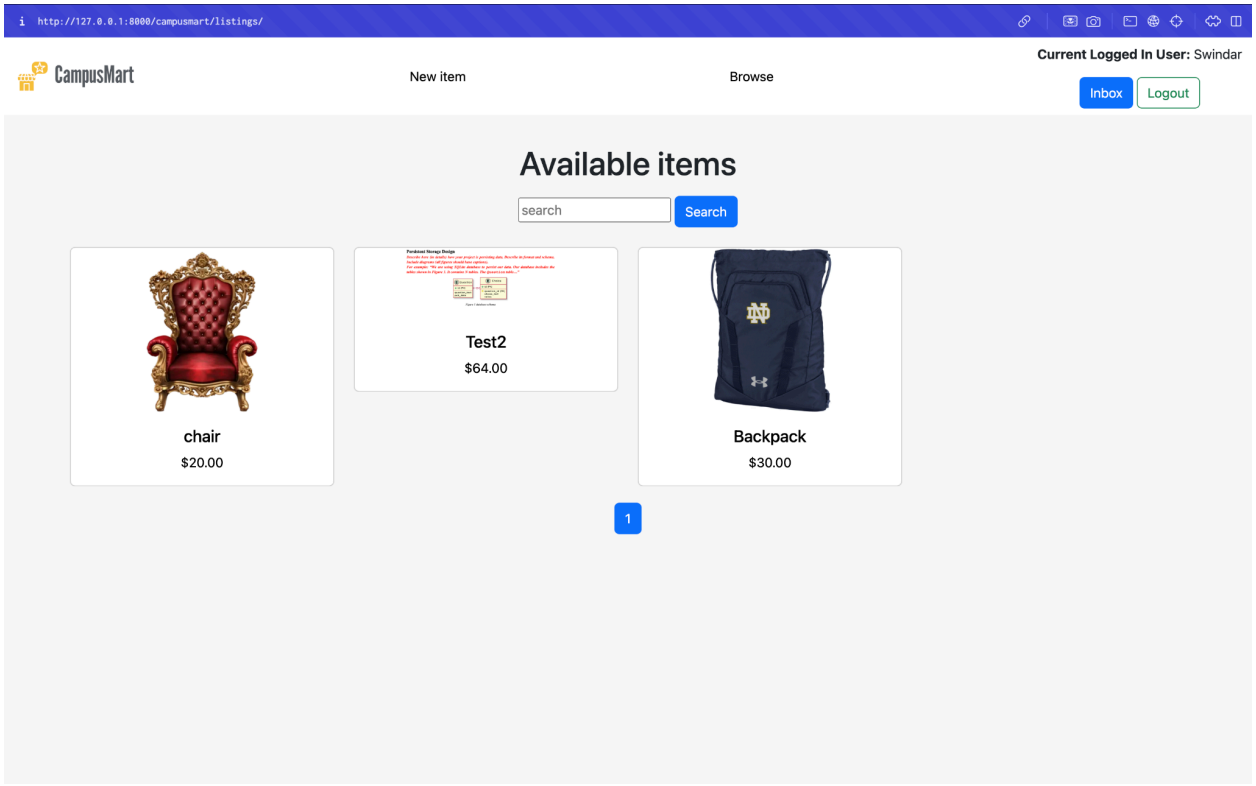

Figure 4. View Listing (by clicking 'Browse' in the Navigation Bar)