# **Final Project Report**

**Student Name:** Heta Milan Shah

**Github Username:** hetashah21 (https://github.com/hetashah21)

**Semester:** Fall 2023

**Course:** CS5001 21148 Intensive Foundations of CS SEC 39

## **Project Description:**

This document encompasses the specifics of implementing a Minimum Viable Product (MVP) for an e-commerce platform designed to facilitate the exploration of furniture brands, categories, and products. The application incorporates fundamental functionalities tailored for administrators, such as the addition of brands, categories, and products. Likewise, it offers customers essential features, including product listing, viewing detailed product information, and adding items to the shopping cart, among others.

Technologies:

* Flask (python)
* Bootstrap
* HTML
* CSS

## **Key Features:**

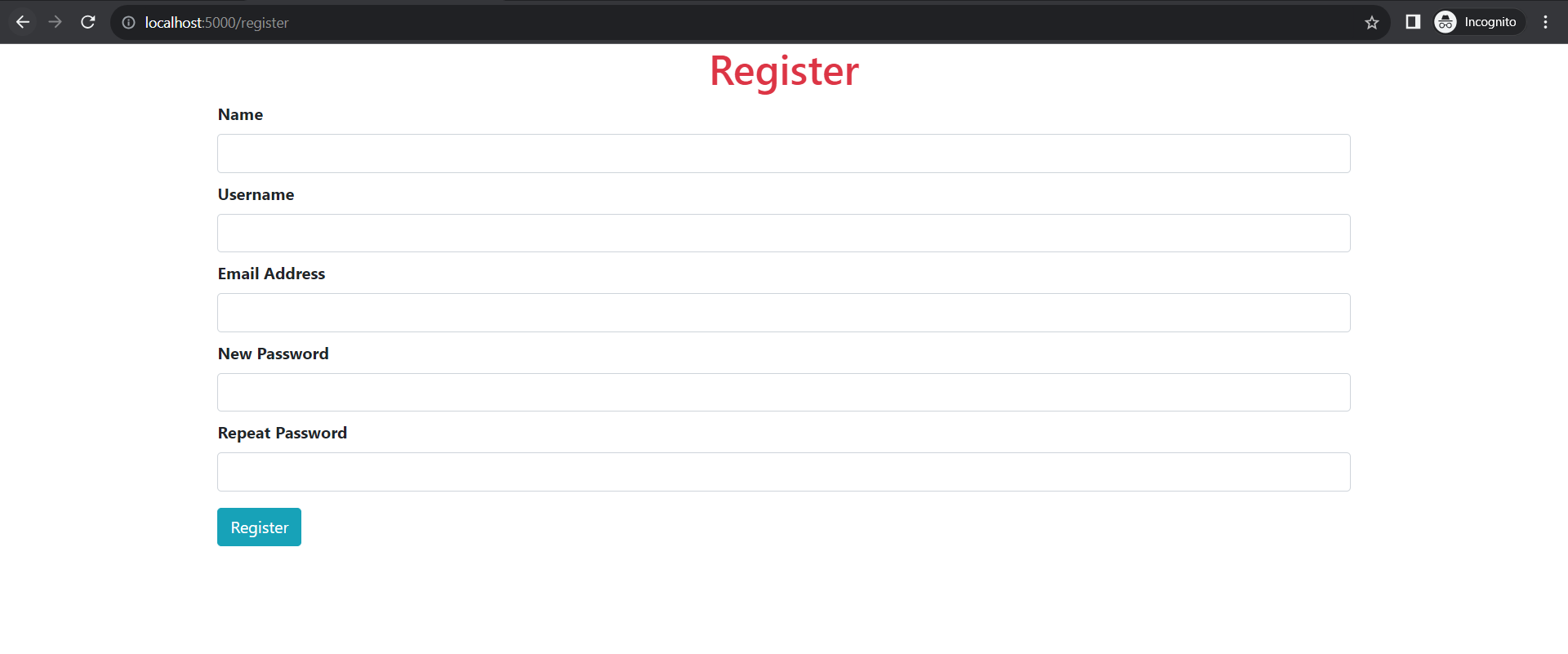
Some of the key features are as follows:

### User registration/login

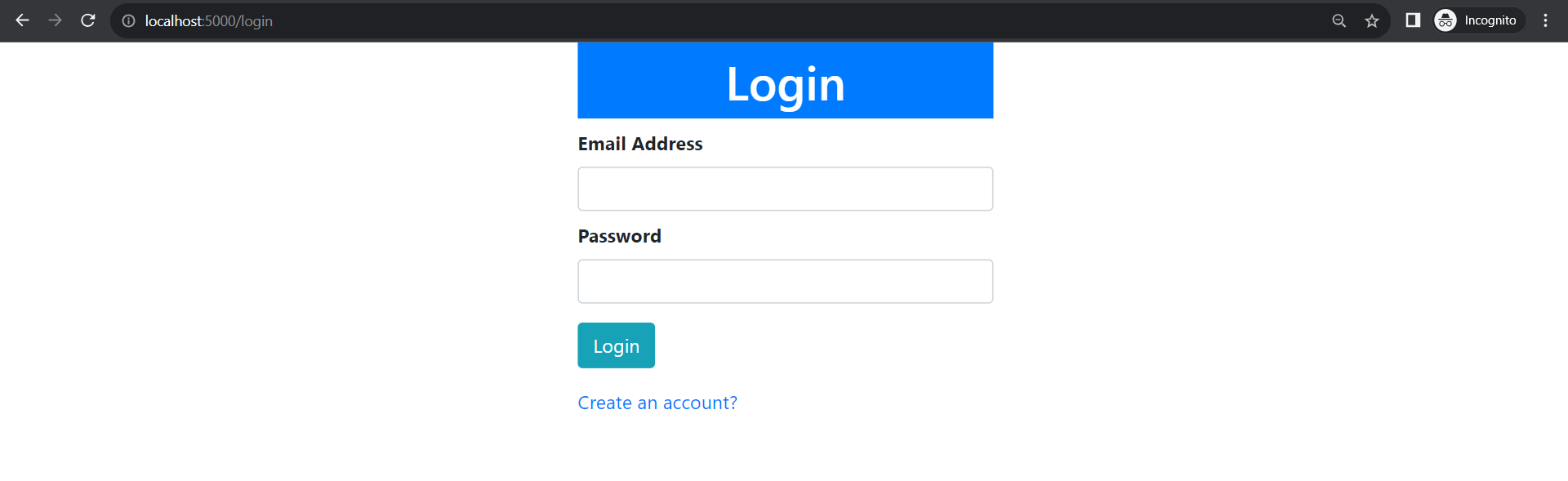
The first major feature in the application is the user registration page. This has been developed using the following packages:

1. Flask-WTF:

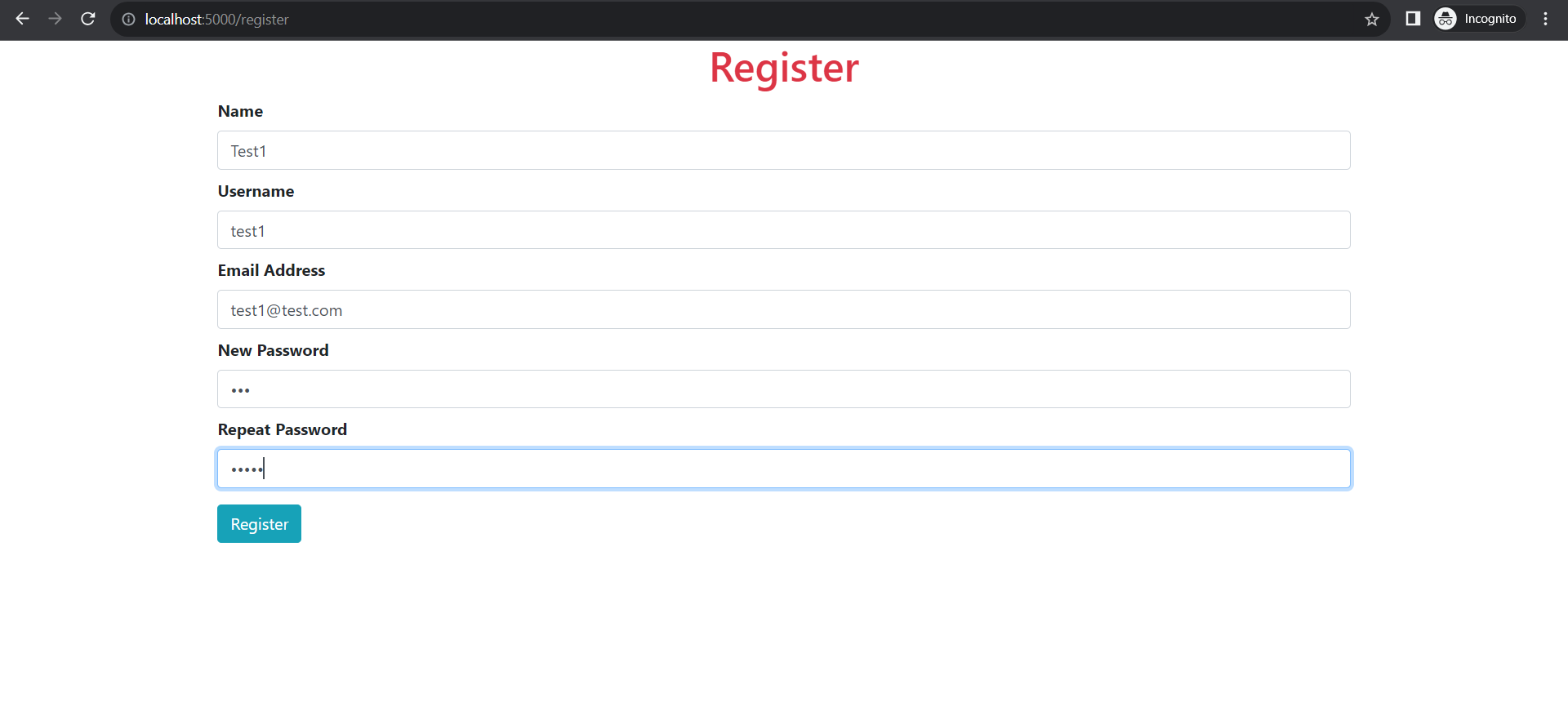
Flask-WTF is a Flask extension that integrates the WTForms library, which provides useful features for creating and handling forms in a simple way for a Flask web application. This has been used in the project to create forms like the registration page (<http://localhost:5000/register>) and login page (<http://localhost:5000/login>) . This also makes sure that the required fields are completed, the input value meets the character requirements and checks if the “New Password” and “Repeat Password” match.



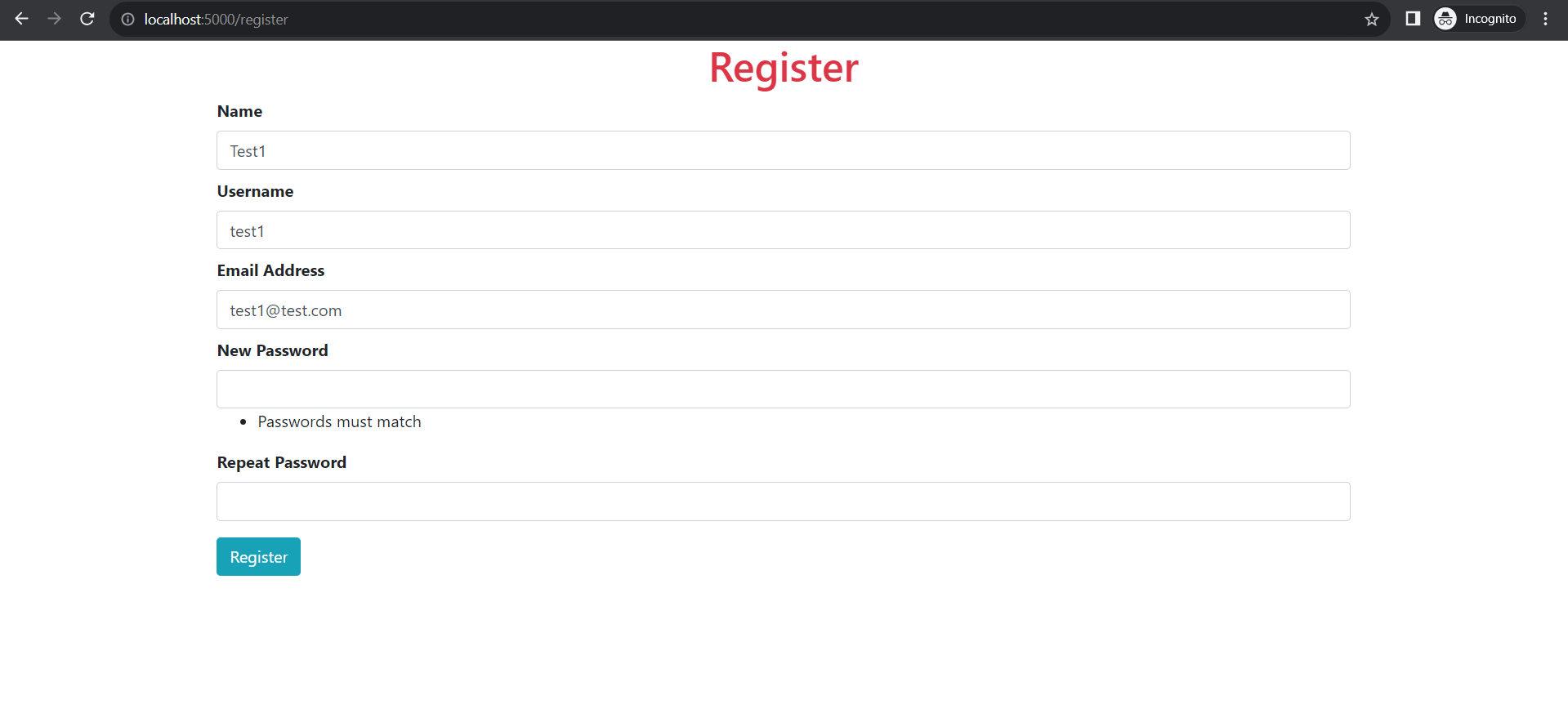
Registration form



Login Form



The password entered by the user are different

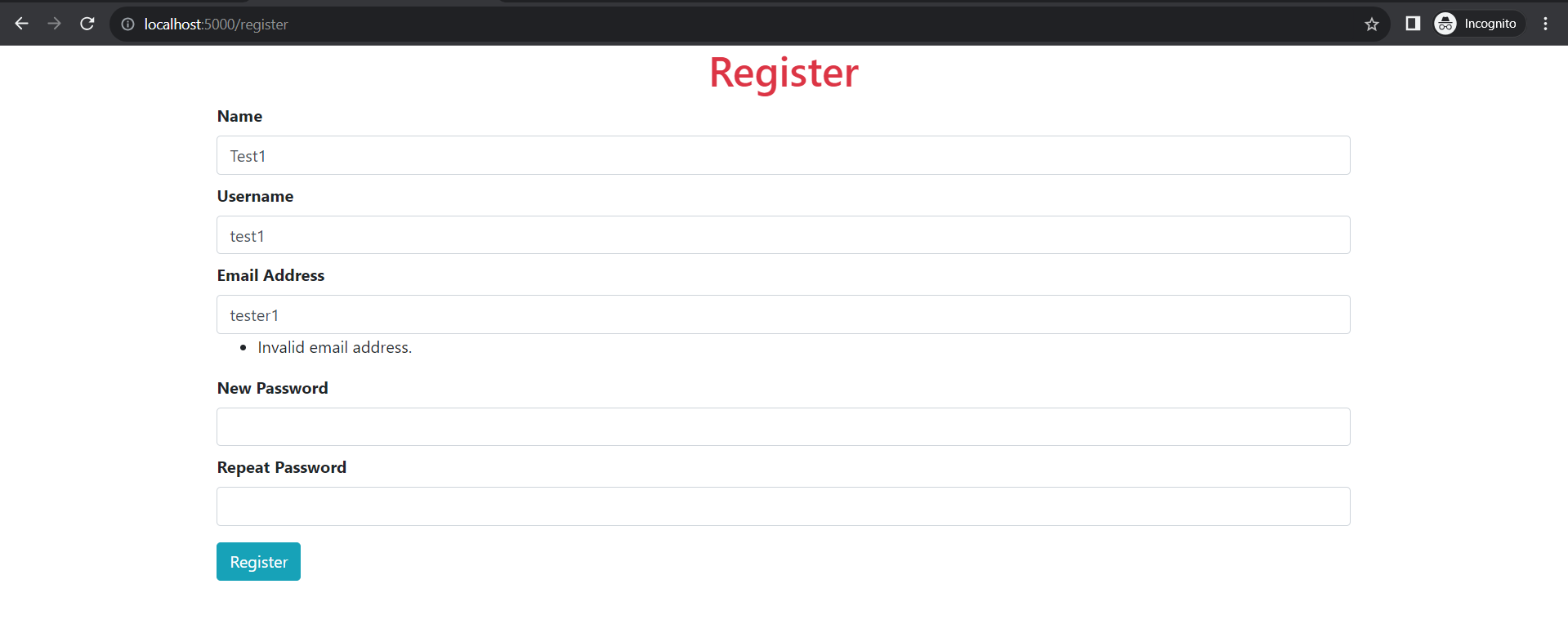


The user gets an error message that the “Passwords must match” when the user tries to register

1. email\_validator

In addition to the forms, the app uses email\_validator package to ensure that the information provided in the forms follows the predefined rules for instance, email should contain 6 characters at the minimum and should be in valid format.

When the user tries to register with an invalid email by clicking on the register button, the user gets “Invalid email address” message



1. flask-SQLAlchemy

The register/login forms are also connected to the sqlite3 database table “user” using flask-SQLAlchemy. Whenever a user registers, the entry is added to the database.

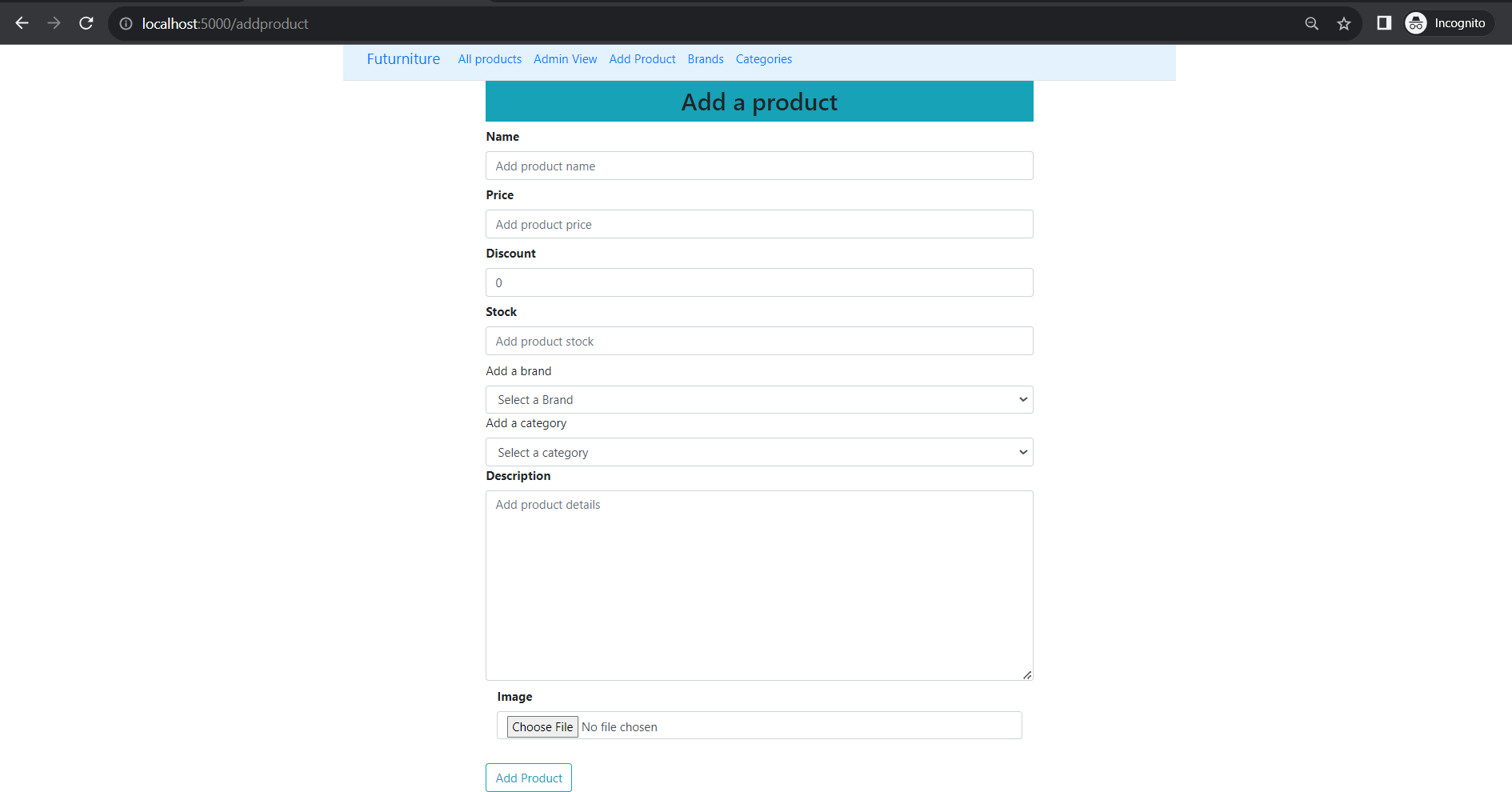
1. flask-bcrypt

All the passwords stored in the database are hashed using flask-bcrypt. Whenever a user tries to login, the password from the input is hashed using the same key and is compared with the password in the database. The user is only allowed to login if the user exists in the database.

### Adding products:

The admin can use the Add Product form to add products to the database. The user needs to provide information like Name, Price, Discount offered (in %), Stock, Brand, Category, Item Description and upload a photo of the product. The Addproduct is in a relationship with the Brand and Category tables in the database using foreign keys brand\_id and category\_id. Thus, if a user adds any brands or categories to the database, those values are auto populated in the dropdown menu and whichever value is selected, the id for the value is mapped to the product.

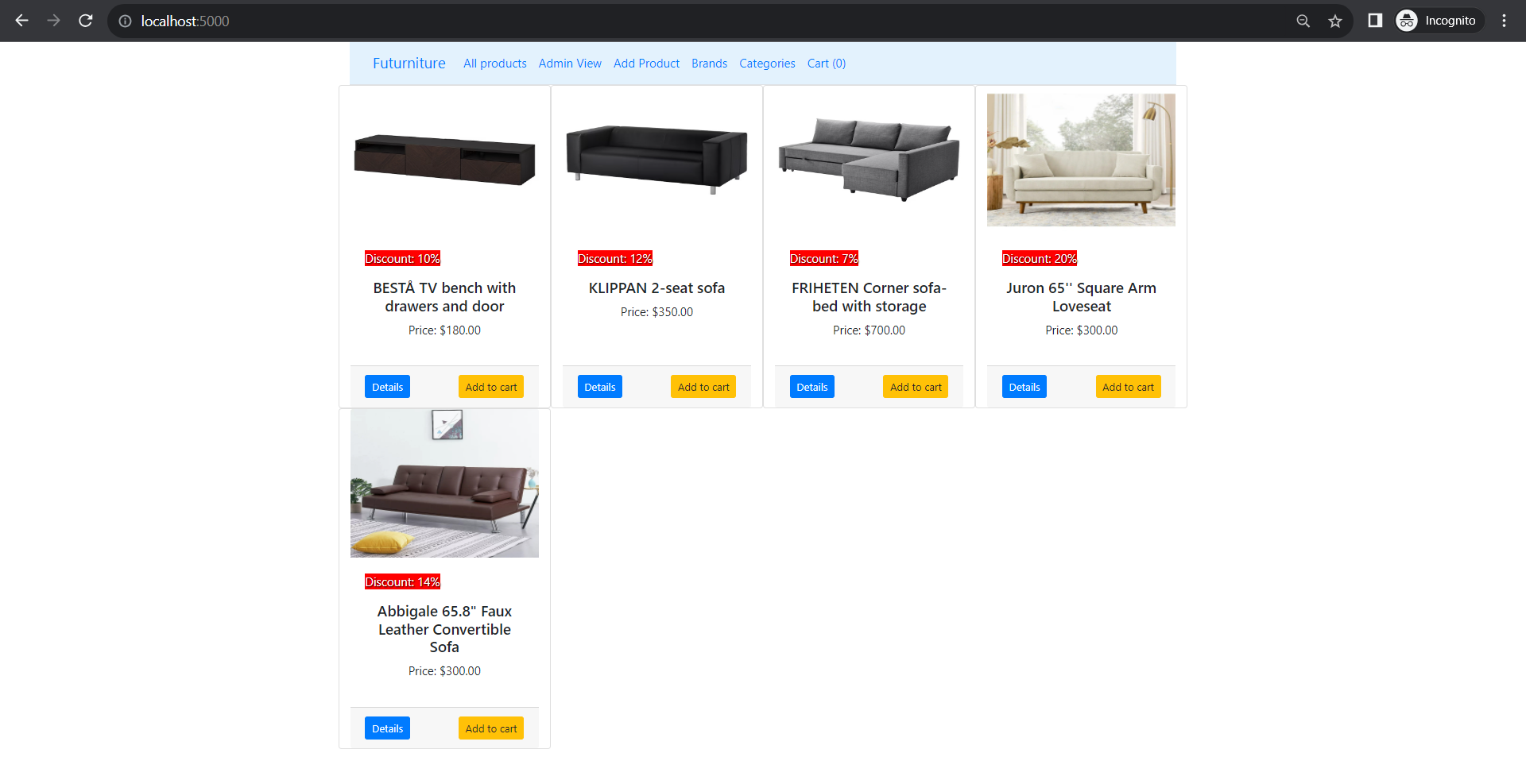
In addition to this, the admin also has an option to upload an image of the product. This has been implemented using “Flask-Reuploaded” package. The screenshot of the Add product form is as follows:



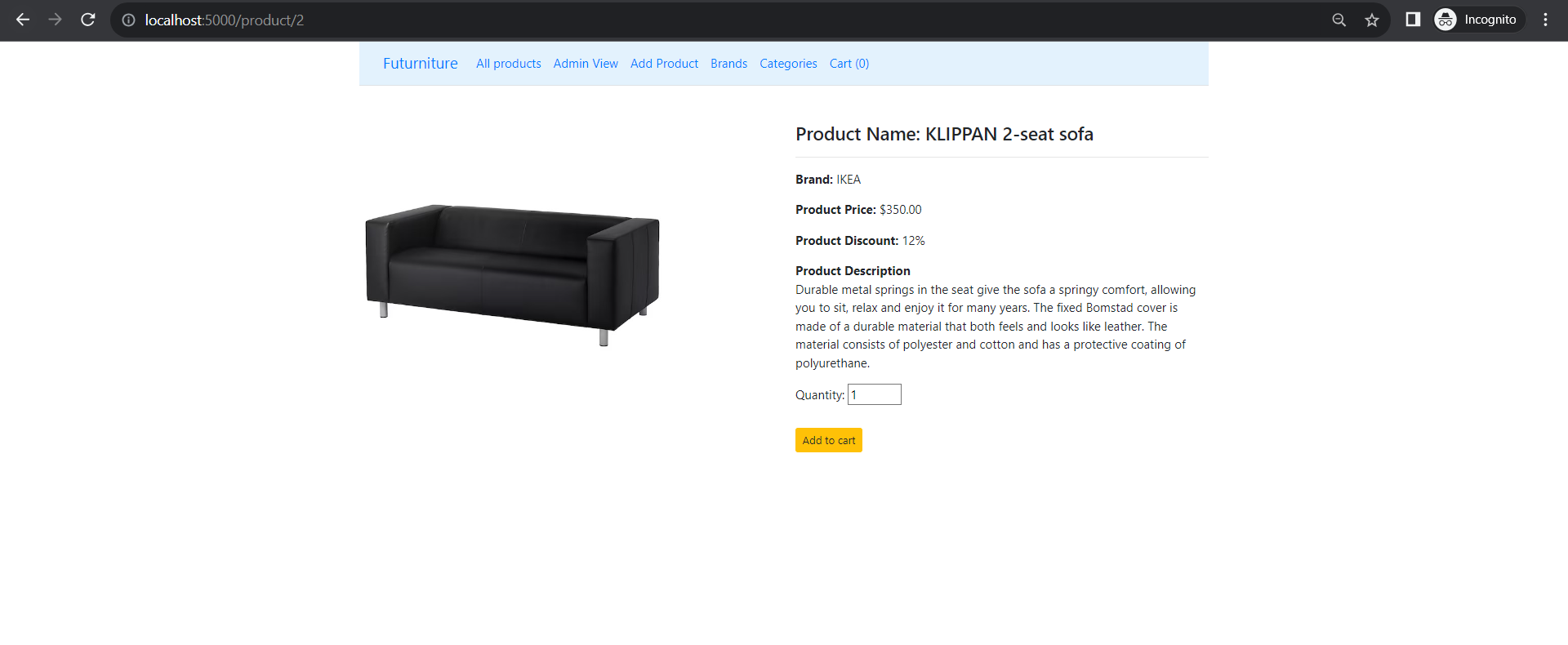
### Shopping cart:

The user can also add items to the shopping cart. There are two ways to do it:

1. Go to the “All Products” page and click on “Add to cart” button on any product. Only one quantity is added to the cart by default using this method

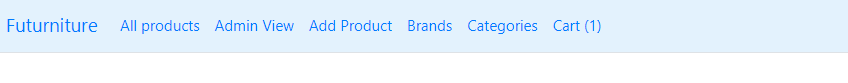


1. Go to the details page, select the quantity and then click on “Add to cart” button

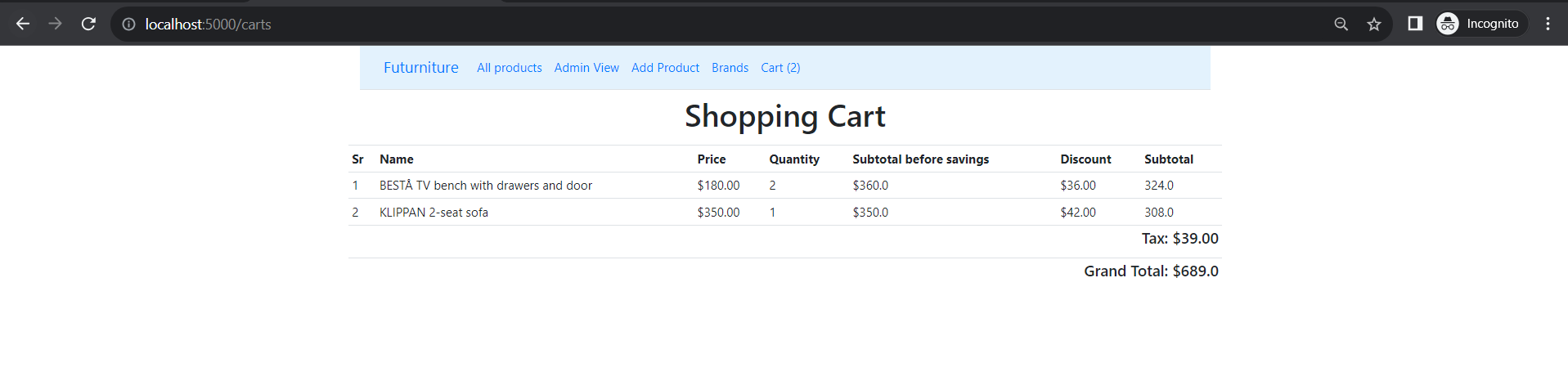


Note: Since this is an MVP, there are a few restrictions on the “Add to cart” button. The user can only add a product once.

Once the user adds an item to the cart, the count in cart increases in the navbar.



Once the user is done adding items to the cart, the user can click on Cart from the navbar and look at the items added in the cart.



This view provides all the details of the cart like the products, their price, quantity, subtotal before savings, discount applied on each items and their total after savings. In addition to this, the tax applied and the grand total is also displayed.

## **Installation Instructions:**

Make sure you have python installed on your machine.

Install virtualenv:

python3 -m pip install virtualenv

Create virtual environment:

python -m virtualenv venv

Activate virtual environment:

source venv/bin/activate

Install dependencies:

pip install -r requirements.txt

Run the application:

python run.py

Access the app on browser of your choice at <http://localhost:5000/>

## **Major Challenges:**

* Learning Flask Framework
* Had to reduce the scope since it was an MVP
* Identifying python packages that can be used to reduce the work required to be done from scratch
* Outdated documentation and incompatible packages
* Had to prefer manual testing to meet the deadline since automated UI testing had a learning curve

## **Future enhancements:**

Due to limited time, I had to reduce the scope of the project which can be added as future enhancements. They are as follows:

* Access Control: In future, I would like to add access control so that only admins can add products, brands and categories and users can only explore available products and order them.
* Updating and deleting brands, categories and products.
* Checkout feature: Adding payment methods and reducing the stock of the product once it is order.
* Filter by product name, brand and category

## **Final Reflection:**

This course has been a great learning exercise for me. From not knowing python at all to learning the fundamentals of python programming, learning a library like flask and building an MVP has been a major growth for me as an individual. The assignments and the coding exercises have helped me in developing my critical thinking skills. I also learnt working with a team and also taking the responsibility of developing an entire project on my own. I would like to continue learning python even after this course. Overall, I believe taking up this course has been the right decision for me and I would encourage others to enroll in the course in future.