BLINT(Online Grocery Stores Application)

Team Members:
SaiKumar Reddy Atluri(22m0745)
Ravindra M S S R (22m0764)

Abstract:

In today's world e-commerce giants like Instamart and amazon fresh are dominating the grocery markets. As a result of this, small scale vendors and grocery store owners are losing their business. They are not able to compete with these big players in terms of resources and digitalization. Our website "blint" helps these neighborhood grocery store owners to digitize their grocery information and make it reachable to each and every customer in their surroundings. Customers can directly place the order and it's up to the store owner how to deliver the order. This curtails the monopoly of big e-commerce websites and helps in sustenance of the neighborhood grocery store ecosystem.

What We have achieved from this project:
Customer Functionalities

- View Nearby Grocery Stores: Using this functionality, users can see all the stores nearby. As of now, we have kept the nearby stores as static. Also populated only the first 2 stores for demo purposes.
- View available items in selected store: Whenever user clicks on the desired store, he/she can access the items from that store. We are retrieving the items from the database, whenever this happens. We have also provided a functionality that allows users to add more than one item of the same kind, instead of adding the same item multiple times.

- Add items to the cart: Users can add items from the desired store using the interactive UI provided and can click on the cart option in the navigation bar to see all the items that are added to the cart.
- Place an order:After users select the required items, they can place the order by clicking on the Place Order button. Order will be placed and sent to the respective grocery shop owner as well
- Status of current order: Users after placing their order, can check their order status in the myorder section. There are 2 states in order, i.e accepted and pending, represented by A and P letters. Along with the current order, they can also check their previously placed orders and their respective status.

Grocery Store Owner Functionalities:

- Can view all the items present in his inventory. As soon as he logs in with his user id and password, they will be displayed on the home page.
- Accepting Pending Orders: Store owner can see all the pending orders by clicking on pending orders option on navigation bar and he/she will be redirected to a page where they can see all the pending orders. They will be given an option to accept the offer or leave it.
- As soon as they accept the order, they can no longer see that order in the pending orders section and it is also updated on my orders section in the user's interface as accepted.

Possible future work pending:

- We can make nearby grocery stores as dynamic. It will scale up and down with adding and removing the new stores.
- Associate a secure payment gateway or define a robust payment method. So when a user clicks place order, he will be safely redirected to that gateway.
- Because of no reliable delivery system, we are only sticking to order status as accept and pending. But can include Reject and Delivered status with appropriate assistance.

- Currently the store owners can only see their menu. But in the future, we create access to the store owners to add and delete items from their menus, without affecting the consistency of the database.
- Adding cookies to user sessions.

Technologies Used: HTML/CSS, BOOTSTRAP, DJANGO, POSTGRES, PYTHON

Directory Documentation:

BLINT_DJANGO: It is the master directory for the whole application. It contains manage.py file

templates: This directory contains all the template code for UI. Our Html files are present here.

blint: This directory contains all urls for files when application loads, urls.py and settings.py file. It also has a static folder to store all the images and plugins in our application.

Clients: This folder contains all the urls for the client and vendor interfaces. It also contains a view.py file which has all the views to handle operations.

Compiling instructions:

- Start Postgres server using command:
  pg_ctl -D $PGDATA start
- In setting, connect to the POSTGRES.
- Start django server using : python3 manage.py runserver
- Open localhost:8000 and you will be redirected to our UI. Login or register with your Details.