

## PROJECT DOCUMENTATION

### 1. Introduction

#### ***Purpose:***

The purpose of this system is to create an online food ordering and delivery platform which is similar to UberEats that allows customers to browse restaurants, view menus, add items to their cart, place orders, and manage their profiles. For restaurant owners, the platform helps manage their profiles, dishes, and orders. This system also supports user authentication, delivery address management, and order tracking to both customers and restaurants.

#### ***Goals:***

- The system is designed to achieve the following goals:
- Better experience for users to browse, order, and track food deliveries.
- Enable restaurant owners to manage menus and orders effectively.
- Support scalability for a potentially large number of users and restaurants.
- Ensure secure, reliable data handling, especially for user profiles, orders, and sensitive information.

### 2. System Design

#### ***Architecture Overview:***

This system is built on client-server architecture using **Django REST Framework** for the backend and **React** for the frontend. The backend handles core functionalities such as user management, order processing, restaurant management, and cart handling, while the frontend provides user-friendly interface for user interactions. The system leverages the following architectural components:

- **Django REST API** for backend services
- **React** for the client-side interface
- **SQLite** (for development) as the primary database
- **Token-based Authentication** for secure login sessions
- **AWS EC2** for scaling and deployment.

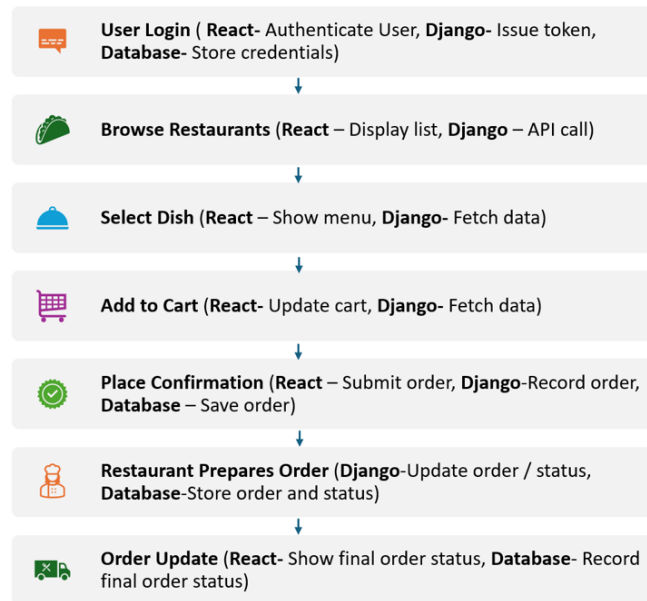
#### ***Components and Data Flow:***

- **Authentication:** Utilizes Django's User model for both customers and restaurants, with token-based authentication ensuring secure access to different user routes.
- **Order Processing:** Orders are stored in the Order model, with each order containing multiple OrderItems representing individual dishes. Customers add dishes to their cart, which can later be converted into orders. These orders can be viewed on the Order Management dashboard of the
- **Profile Management:** Customers and restaurants manage their profiles with separate models, allowing each user type to view or update their details securely.

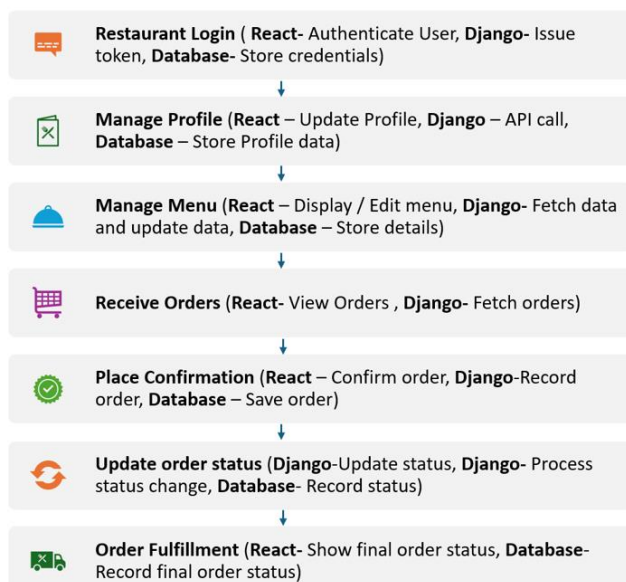
- **Dish and Menu Management:** Restaurant owners manage dishes through Dish models associated with Restaurant, with options to add and edit dishes from their menus.
- **Favorites and Cart:** Customers can save favorite restaurants and manage cart items, which are linked to orders upon checkout.
- **Order Tracking:** Customers and restaurant owners track orders using a series of status updates (new, preparing, on the way, etc.). Once there is an update from Restaurant, eventually the order status gets updated at the user end.



#### Uber eats prototype - Process chart (User)

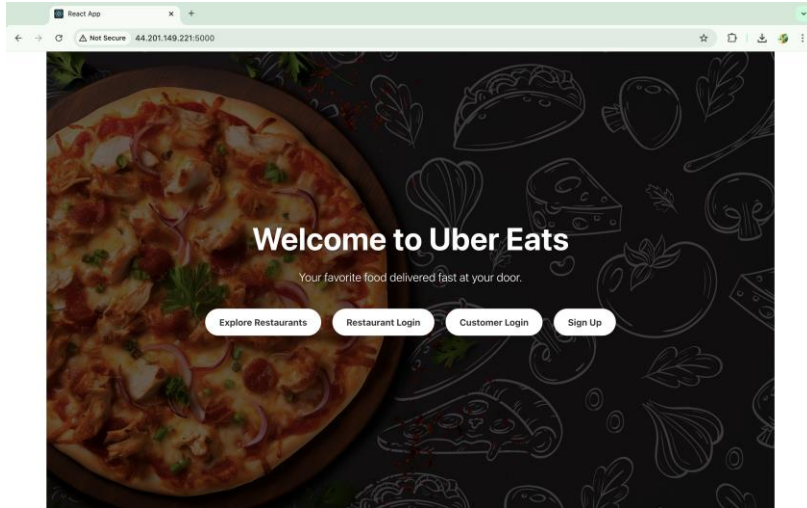


#### Uber eats prototype - Process chart (Restaurant)

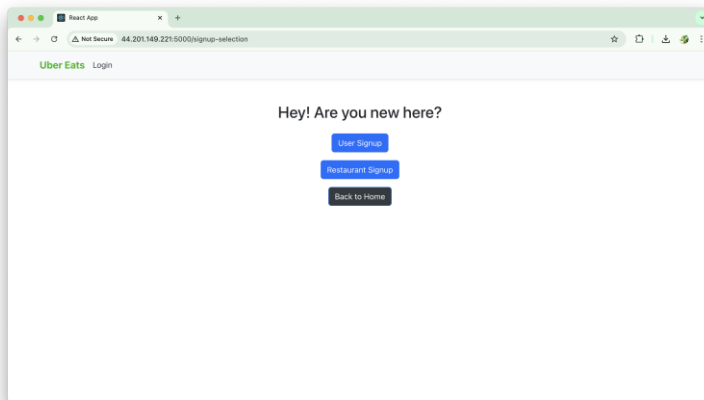


### 3. Results:

#### 1. Homepage with buttons

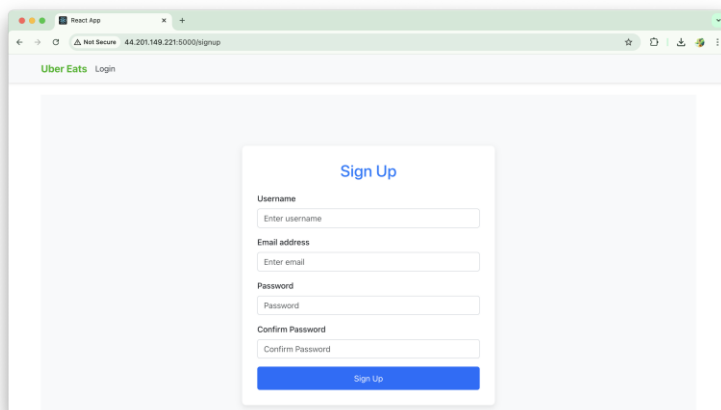


#### 2. Signup Selection

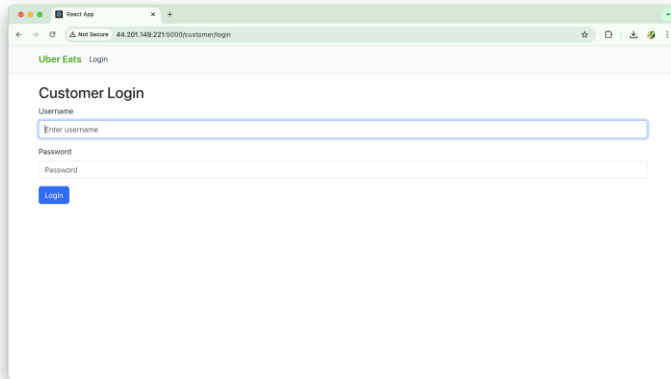


### **PART 1: On the Customer / User side:**

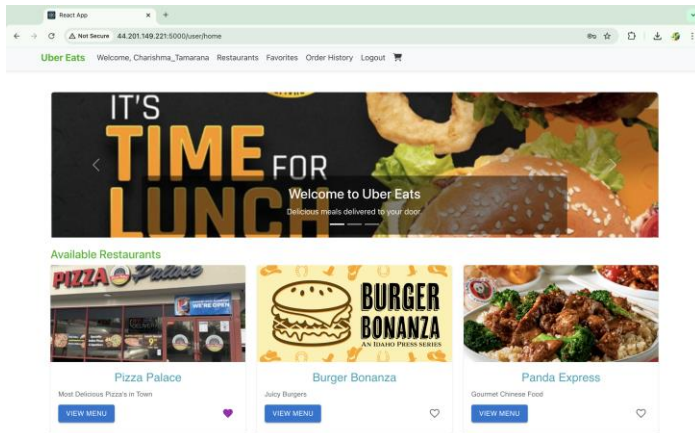
#### 1. Customer Signup



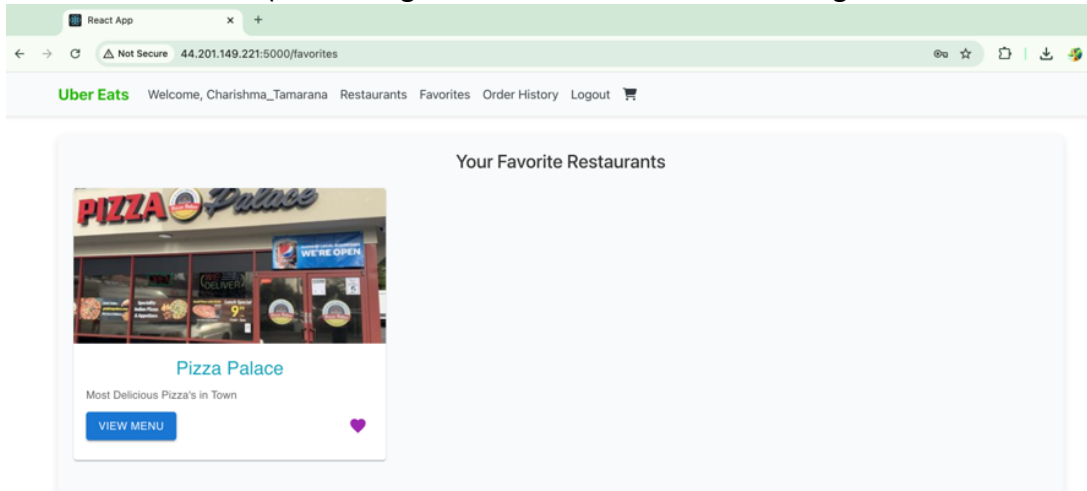
## 2. Customer Login



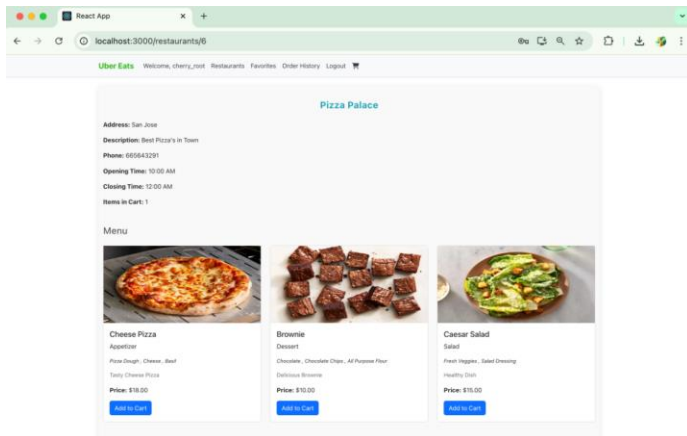
## 3. Restaurants Details



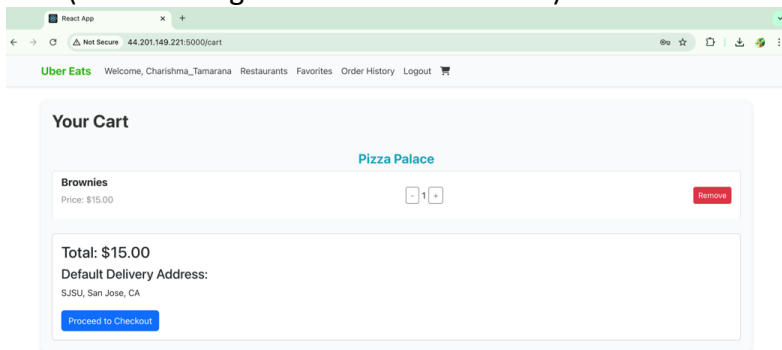
## 4. Favorite Restaurant (On clicking Favorite on Nav bar after clicking heart button on restaurant card)



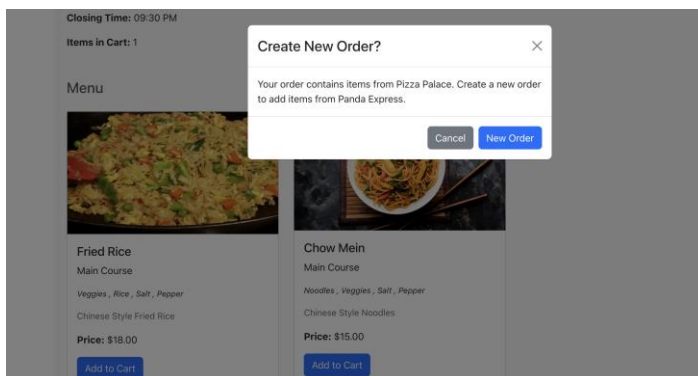
## 5. Restaurant Menu with dishes



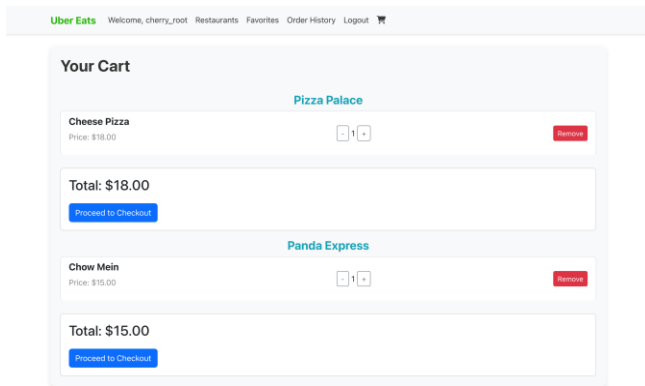
## 6. Cart (After clicking Add to cart from above)



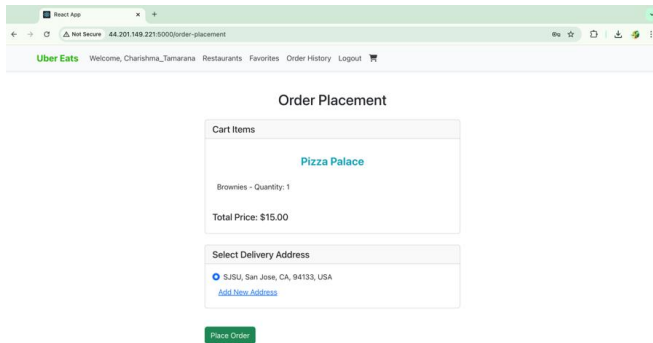
Dialog confirmation if order placed from another restaurant ( when add to cart is selected )



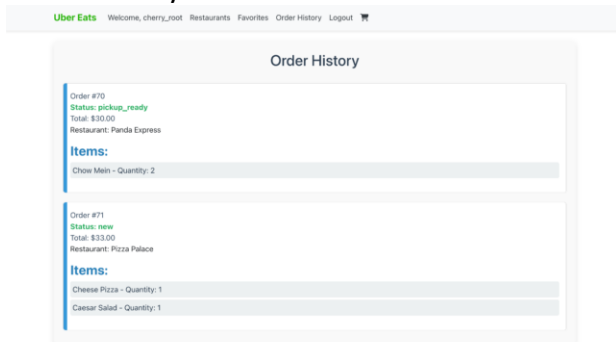
In Cart when New Order is Selected, It groups the new order with the restaurant and dish details but user can place order (proceed to checkout ) only from one restaurant :



## 7. Order Placement

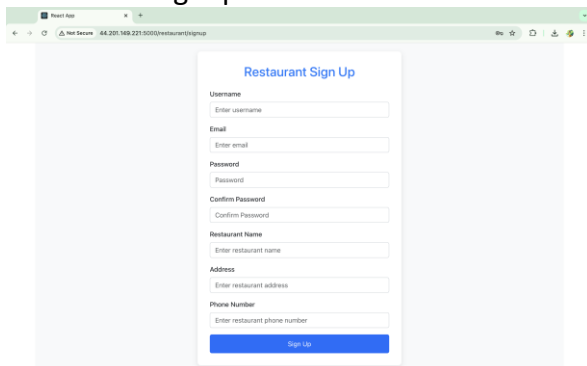


## 8. Order History

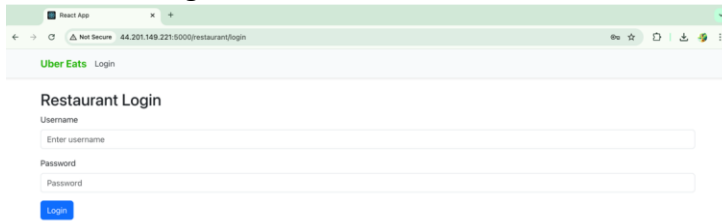


## Part 2: Restaurant Side:

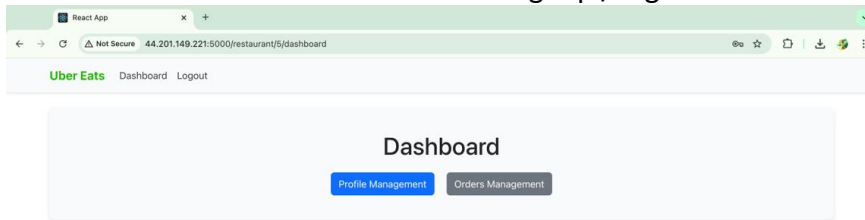
### 1. Restaurant signup



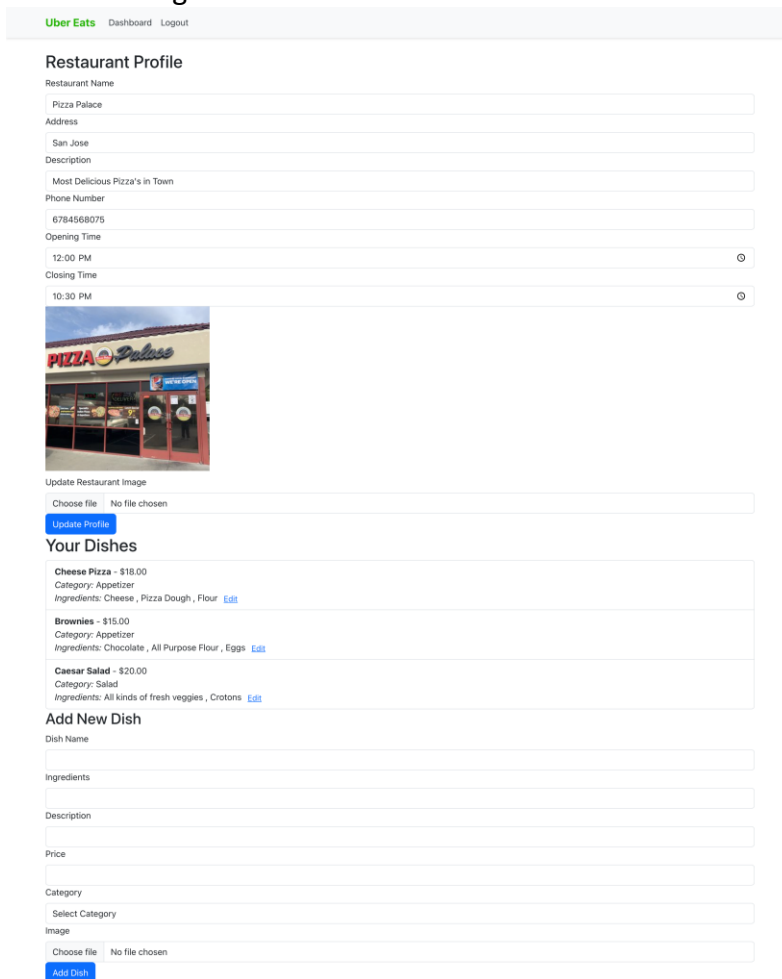
## 2. Restaurant Login



## 3. Restaurant Dashboard after successful signup / login



## 4. Profile Management




## 5. Edit Dishes (After clicking Edit from your Dishes )

Uber Eats Dashboard Logout

### Dish Details

Name Caesar Salad	Ingredients All kinds of fresh veggies , Croutons	Description Healthy tasty Salad	Price \$ 20.00
----------------------	--	------------------------------------	-------------------

Category  
Salad



Update Dish Image  
Choose file No file chosen  
[Update image](#)

## 6. Orders Management

Uber Eats Dashboard Logout

### Orders Management

All ▾

Order #71  
Status: NEW

Pick Up Re... ▾ [UPDATE STATUS](#)

Name: Cherry  
Email: cherry@gmail.com  
Phone: 776667776677  
Delivery Address: SJSU  
City: San Jose  
State: CA  
Country: USA

Status updated successfully

On user side, status updated (pickup\_ready On Pizza Palace)

Uber Eats Welcome, cherry\_root Restaurants Favorites Order History Logout 🛒

### Order History

Order #70  
Status: pickup\_ready  
Total: \$30.00  
Restaurant: Panda Express

**Items:**

Chow Mein - Quantity: 2

Order #71  
Status: pickup\_ready  
Total: \$33.00  
Restaurant: Pizza Palace

**Items:**

Cheese Pizza - Quantity: 1  
Caesar Salad - Quantity: 1



## 4.Performance:

In the Django Admin panel, a total of 40 users are managed, few associated with specific restaurant and dish information. This setup organizes data for user-specific operations, such as managing individual restaurants, updating menu items, and viewing orders, all through a centralized admin interface.

While the current setup has 40 users, Django Admin’s pagination and filtering features make the application scale further if the user base or associated data grows.

Django administration

WELCOME, CHERRY\_ROOT. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home » Authentication and Authorization » Users

Start typing to filter...

AUTH TOKEN

Tokens

+

+

AUTHENTICATION AND AUTHORIZATION

Groups

+

+

Users

+

+

CUSTOMERS

Cart items

+

+

Customers

+

+

Favorite restaurants

+

+

Orders

+

+

RESTAURANTS

Dishes

+

+

Restaurants

+

+

Select user to change

ADD USER

Q

Search

Action: Delete selected users 38 of 40 selected

	USERNAME	EMAIL ADDRESS	FIRST NAME	LAST NAME	STAFF STATUS
<input checked="" type="checkbox"/>	121	121@b.c	-	-	
<input checked="" type="checkbox"/>	1211	1211@b.c	-	-	
<input checked="" type="checkbox"/>	12111122211ni1	1211112211ni1@b.c	-	-	
<input checked="" type="checkbox"/>	121111ni1	121111ni1@b.c	-	-	
<input checked="" type="checkbox"/>	121112221ni1	12111221ni1@b.c	-	-	
<input checked="" type="checkbox"/>	12111ni1	1211ni1@b.c	-	-	
<input checked="" type="checkbox"/>	121121ni1	121111ni1@b.c	-	-	
<input checked="" type="checkbox"/>	1211ni1	121ni1@b.c	-	-	
<input checked="" type="checkbox"/>	121ni1	12ni1@b.c	-	-	
<input checked="" type="checkbox"/>	Charishma_T	cherry_tree@gmail.com	-	-	
<input checked="" type="checkbox"/>	Charishma_cherry	cherry_tree@gmail.com	-	-	
<input checked="" type="checkbox"/>	Test	Test1267@gmail.com	-	-	
<input checked="" type="checkbox"/>	Test100	test100@gmail.com	-	-	
<input checked="" type="checkbox"/>	as	as@as.as	-	-	
<input checked="" type="checkbox"/>	as1	1as@as.as	-	-	
<input checked="" type="checkbox"/>	as2	as2@as.as	-	-	
<input checked="" type="checkbox"/>	asdsaf	asdas@asda.asda	-	-	
<input checked="" type="checkbox"/>	burger_bonanza	burger_bonanza@example.com	-	-	
<input checked="" type="checkbox"/>	cherry	cherry@gmail.com	-	-	
<input type="checkbox"/>	cherry_root	cherry@gmail.com	-	-	
<input checked="" type="checkbox"/>	cherry_test	cherrytest@gmail.com	-	-	
<input checked="" type="checkbox"/>	cherrymad	asdas@asdasd.efijnweof	-	-	
<input checked="" type="checkbox"/>	customer0	customer0@example.com	-	-	
<input checked="" type="checkbox"/>	customer1	customer1@example.com	-	-	
<input checked="" type="checkbox"/>	customer2	customer2@example.com	-	-	
<input checked="" type="checkbox"/>	customer3	customer3@example.com	-	-	
<input checked="" type="checkbox"/>	customer4	customer4@example.com	-	-	
<input checked="" type="checkbox"/>	customer5	customer5@example.com	-	-	
<input checked="" type="checkbox"/>	customer6	customer6@example.com	-	-	
<input checked="" type="checkbox"/>	customer7	customer7@example.com	-	-	
<input checked="" type="checkbox"/>	customer8	customer8@example.com	-	-	
<input checked="" type="checkbox"/>	customer9	customer9@example.com	-	-	
<input checked="" type="checkbox"/>	dslogin	dslogin@check.com	-	-	
<input checked="" type="checkbox"/>	nadae	qwe@qwe.w	-	-	
<input checked="" type="checkbox"/>	nikhil	a@v.b	-	-	
<input checked="" type="checkbox"/>	pizza_palace	pizza_palace@example.com	-	-	
<input checked="" type="checkbox"/>	rest1	rest1@test.com	-	-	
<input checked="" type="checkbox"/>	rest2	rest2@test.com	-	-	
<input checked="" type="checkbox"/>	root	root@gmail.com	-	-	
<input type="checkbox"/>	vaish	v@test.com	-	-	

40 users

FILTER

Show counts

By staff status

All

Yes

No

By superuser status

All

Yes

No

By active

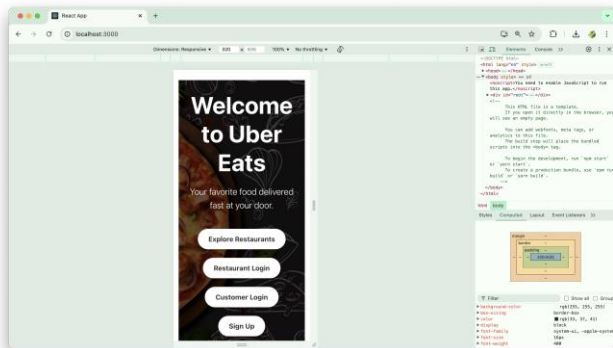
All

Yes

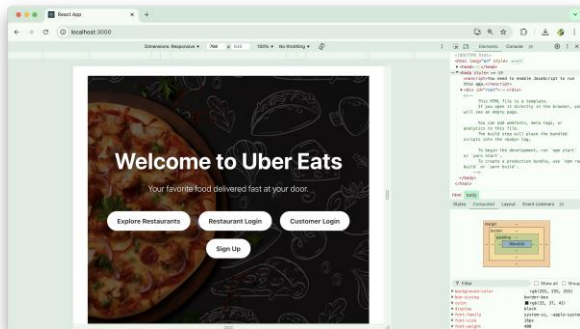
No

## Responsive Design:

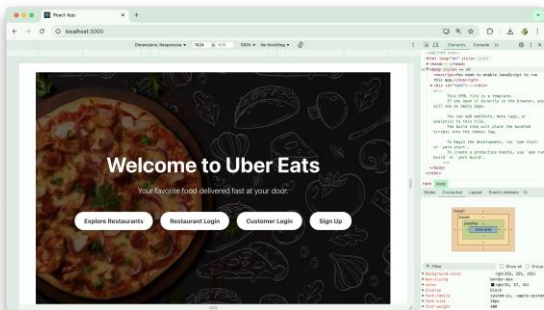
On Mobile view:



On Tablet view:



On Laptop view:



## Paired Programming (Group 7)

Vaishnavi Nanduri (017506489)

Charishma Tamarana (017549181)