Foodie ExpressA Django Real-Time Food Delivery & Booking Management System. Foodie Express is a real-time Django-based food delivery platform that enables role-based access, live order tracking, and chat communication between customers and delivery partners.

Table of Contents

[1. Project Overview 3](#_Toc211500272)

[2. Objectives 3](#_Toc211500273)

[3. Tech Stack 3](#_Toc211500274)

[4. User Roles & Key Features 4](#_Toc211500275)

[1. Admin 4](#_Toc211500276)

[2. Customer 5](#_Toc211500277)

[3. Delivery Partner 8](#_Toc211500278)

[4. Login/Signup 9](#_Toc211500279)

[5. System Architecture 11](#_Toc211500280)

[6. Implementation Summary 12](#_Toc211500281)

[7. Deployment Details 12](#_Toc211500282)

[8. Conclusion 12](#_Toc211500283)

# 1. Project Overview

Foodie Express is a food delivery and booking management system developed using Django.  
It allows three types of users — Customers, Delivery Partners, and Admins — to interact in real-time for booking management, order tracking, and chat communication.  
The project demonstrates full-stack Django development with role-based authentication, real-time updates using Django Channels, and deployment on Render.

# 2. Objectives

* To develop a role-based food delivery management system.
* To implement real-time chat between customers and delivery partners using Django Channels.
* To enable booking creation, assignment, and live tracking with simple, user-friendly interfaces.
* To demonstrate deployment of a fully functional Django app on a production platform (Render).

# 3. Tech Stack

| **Layer** | **Technology** | **Description** |
| --- | --- | --- |
| Backend | Django | Manages authentication, bookings, and chat logic. |
| Frontend | Django Templates + jQuery | Handles user interface and dynamic updates. |
| Real-Time Communication | Django Channels | Enables live chat between customer and delivery partner. |
| Database | SQLite | Stores user details, bookings, and chat messages. |
| Deployment | Render | Cloud platform used to host the live web application. |

# 4. User Roles & Key Features

## 1. Admin

* Login and view all bookings.  
  A screenshot of a computer

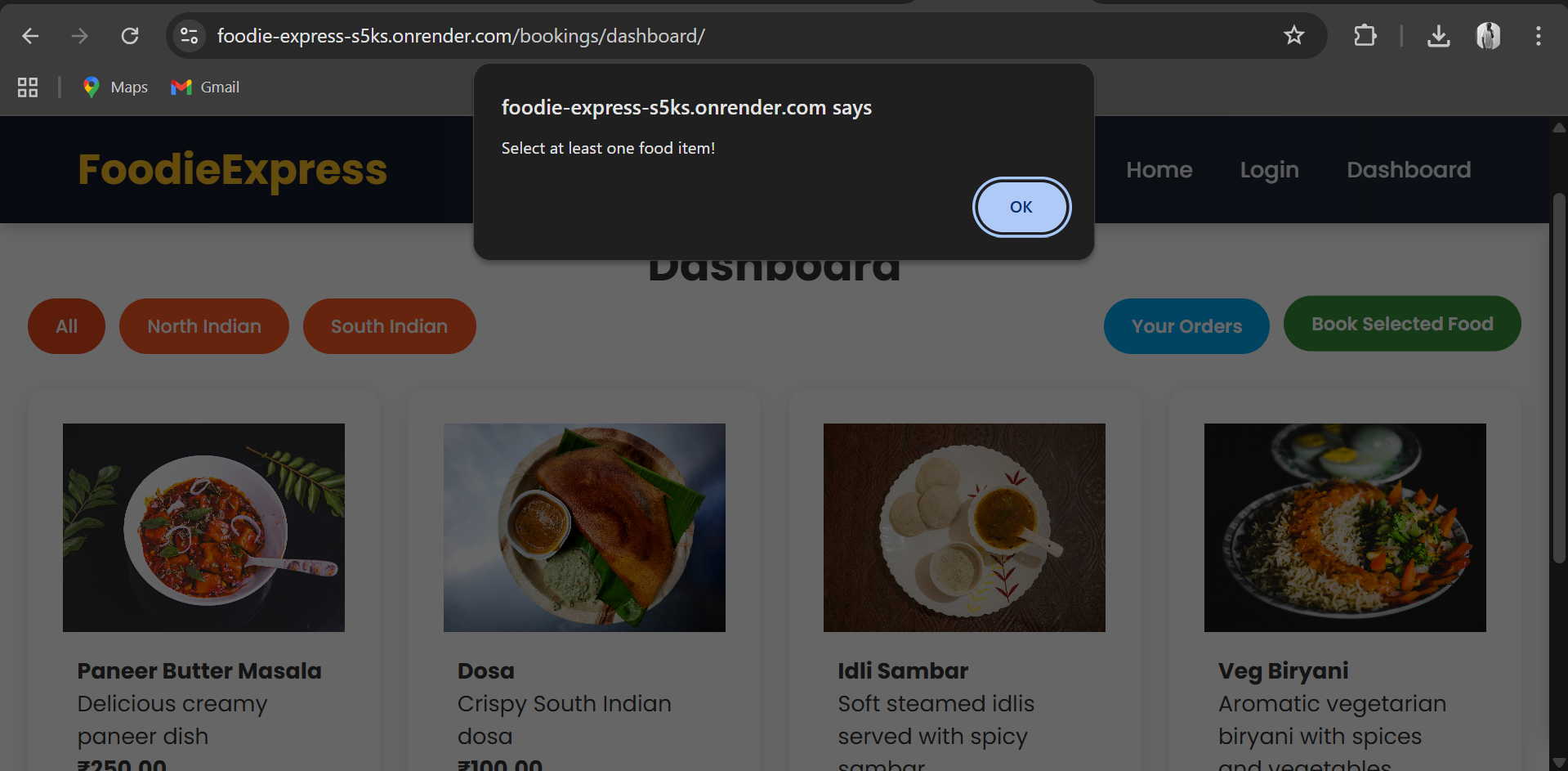
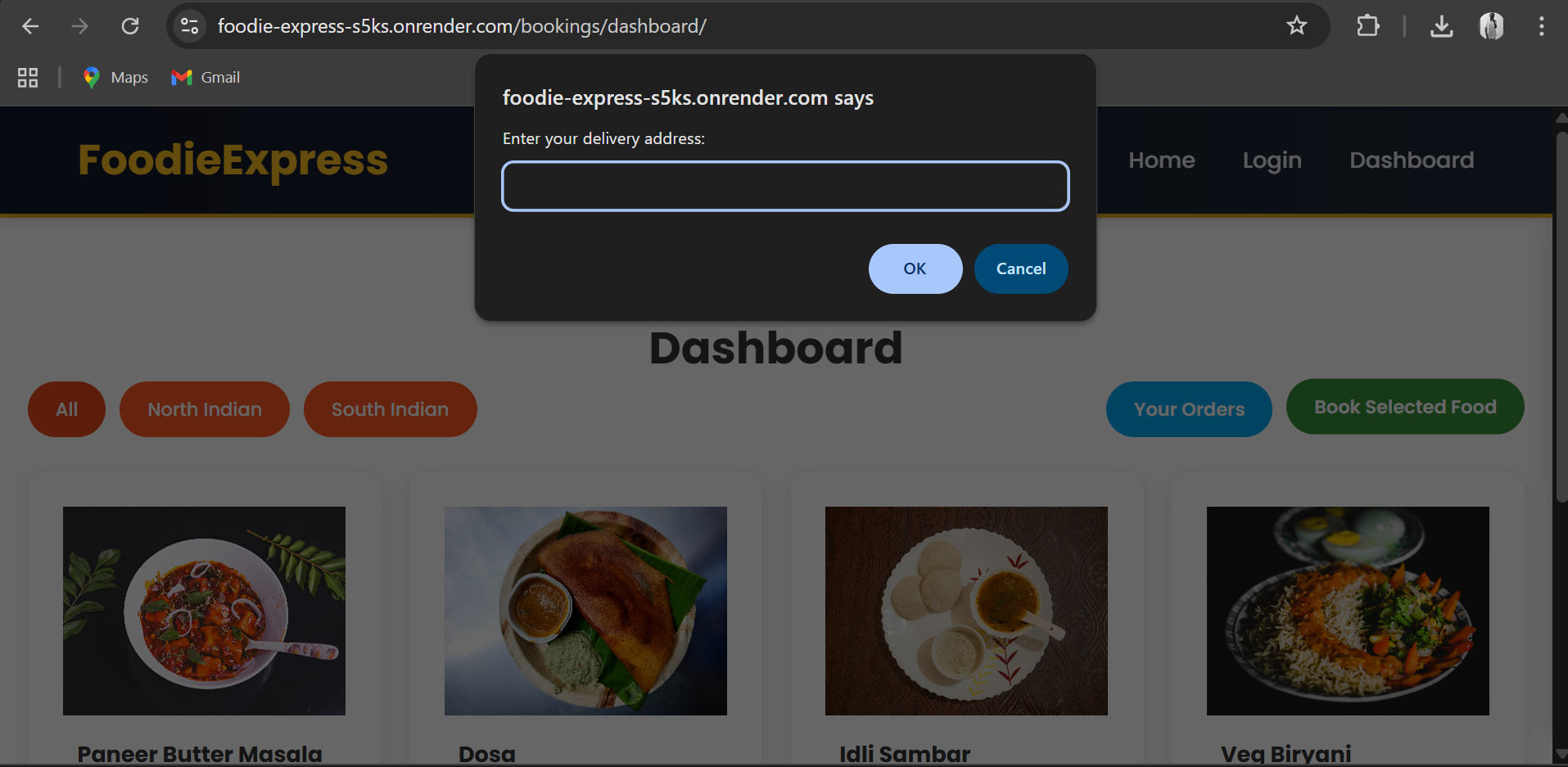
  AI-generated content may be incorrect.
* Assign delivery partners to customer bookings.  
  A screenshot of a computer

  AI-generated content may be incorrect.

## 2. Customer

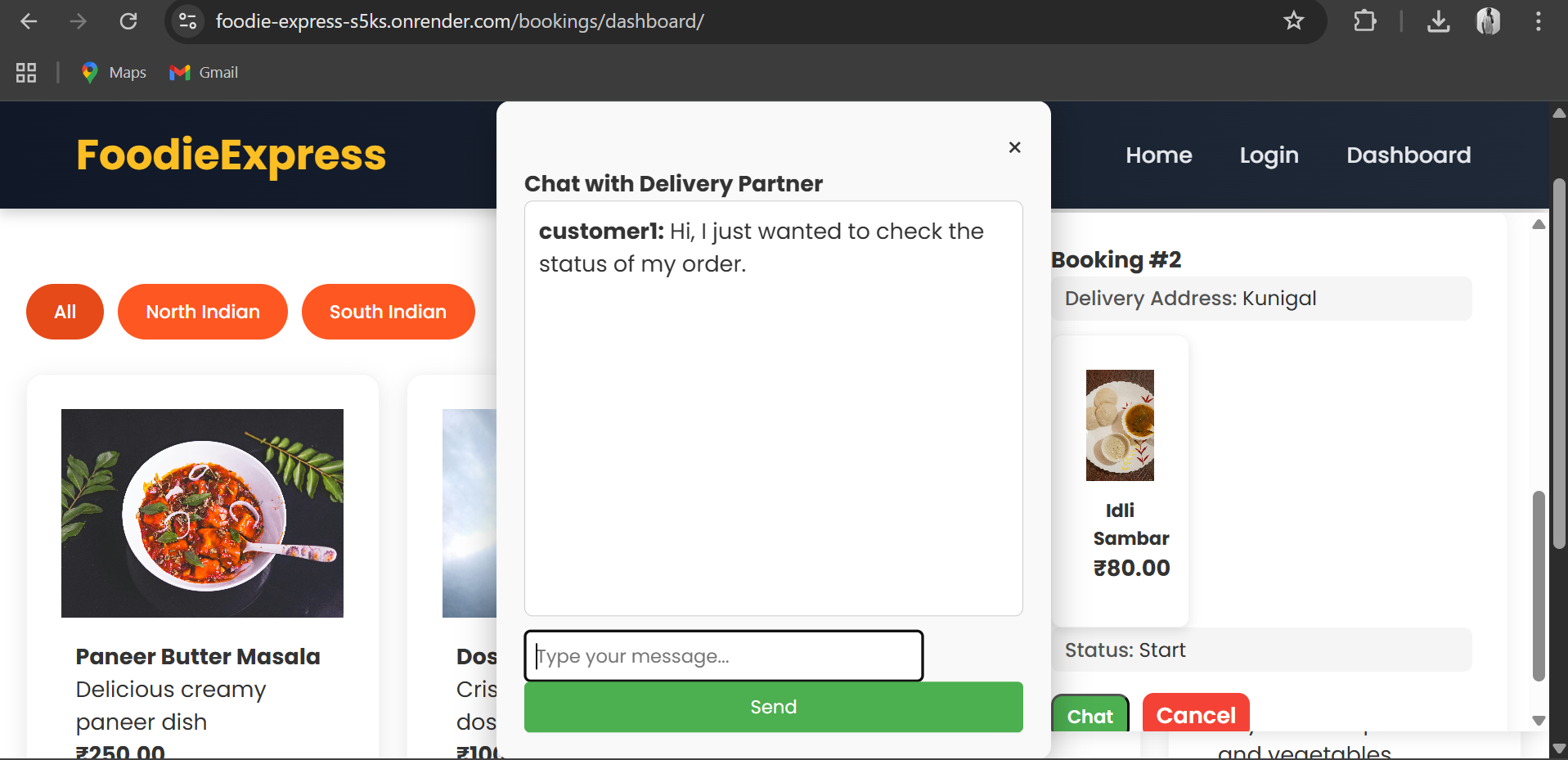
* Login using mobile number and OTP (1234).  
  A screenshot of a login screen

  AI-generated content may be incorrect.  
    
  A screenshot of a computer

  AI-generated content may be incorrect.
* Create, view, and cancel bookings.  
  A food item must be selected when creating a booking, or an error will occur.  
    
    
    
    
  After selecting a food item, the user must fill in the address to complete the booking.  
  

Customers can see all details of their orders after booking.  
A screenshot of a menu

AI-generated content may be incorrect.

* Chat in real-time with assigned delivery partners.  
  

## 3. Delivery Partner

* Login and view assigned bookings.  
  A screenshot of a computer

  AI-generated content may be incorrect.
* Update order status step by step:  
  Start → Reached → Collected → Delivered.  
    
  updated the status from start to be reached.  
  A screenshot of a computer

  AI-generated content may be incorrect.
* Chat in real-time with the customer.  
    
  The assigned delivery person receives messages sent by the customer.  
  A screenshot of a computer

  AI-generated content may be incorrect.

## 4. Login/Signup

* Users can Login/Signup with phone number and OTP, unregistered numbers are redirected to Signup   
  A screenshot of a login screen

  AI-generated content may be incorrect.
* During signup, users select their role (Customer, Admin, Delivery) and Admin/Delivery enter an access key.

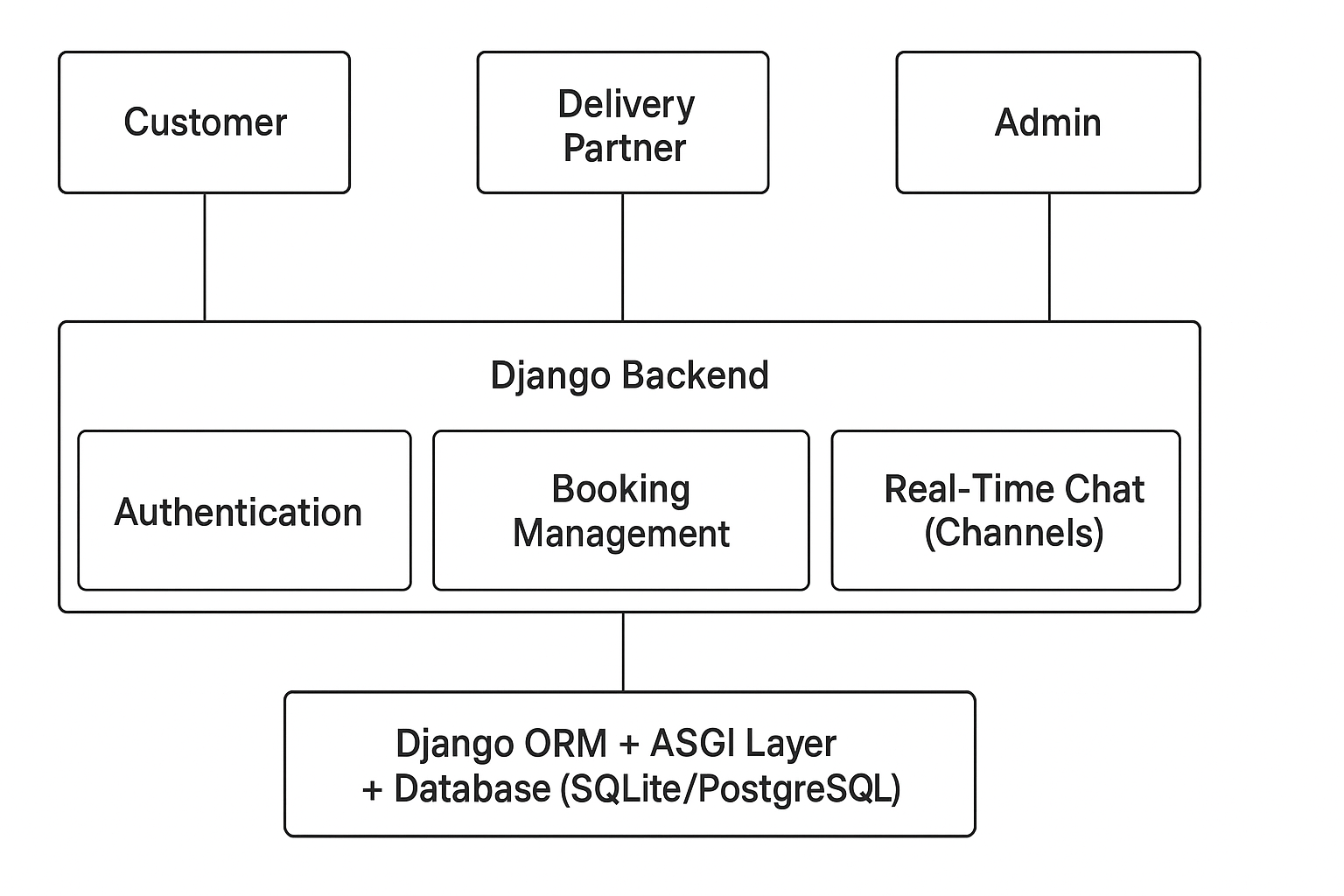
CustomerA screenshot of a computer

AI-generated content may be incorrect.AdminA screenshot of a computer

AI-generated content may be incorrect.

Delivery PartnerA screenshot of a computer

AI-generated content may be incorrect.

5. System Architecture ****

# 6. Implementation Summary

* Developed backend using Django MVC structure.
* Used Django Channels to handle WebSocket connections for live chat.
* Integrated role-based authentication using Django’s built-in user model and custom roles.
* Used jQuery and Ajax for dynamic front-end updates without reloading pages.
* Configured and deployed the app to Render for free public hosting.

# 7. Deployment Details

* Source code hosted on GitHub:[**https://github.com/Arpitha4/django\_food\_delivery**](https://github.com/Arpitha4/django_food_delivery)
* Deployed live using Render Web Service:[**https://foodie-express-s5ks.onrender.com**](%20https:/foodie-express-s5ks.onrender.com)
* **Steps included:**
  + Push project to GitHub.
  + Connect Render to GitHub repo.
  + Configure environment, run migrations, collect static files.
  + Start the server using Gunicorn and ASGI for Channels.

8. Conclusion

* Foodie Express demonstrates strong skills in Django backend development, real-time communication, and deployment management.
* It effectively combines role-based access control, booking workflows, and chat functionality into one interactive system.
* This project reflects the ability to build production-ready applications using modern Python web technologies and showcases readiness for full-stack Django development roles.