

Django Food Service App Documentation

Overview

The Food Service app is a web-based application that connects customers with food vendors, allowing customers to order food online and vendors to manage their orders and inventory. The app is built using Django, a Python-based web framework.

Features

The Food Service app consists of three main tabs:

Customer Login Tab

- Login/Registration: Customers can log in or register to access their account.
- Order History: Customers can view their order history, including past orders and order status.

Vendors Tab

- Vendor Profile: Vendors can manage their profile, including their business information and menu items.
- Order Management: Vendors can view and manage their orders, including order status and customer information.

Admin Tab

- User Management: Admins can manage customer and vendor accounts, including account creation and deletion.
- Order Management: Admins can view and manage all orders, including order status and customer information.
- Vendor Management: Admins can manage vendor profiles, including approving or rejecting vendor applications.

Technical Requirements

- Backend: Django 3.2.5
- Database: PostgreSQL 13.2
- Frontend: HTML, CSS, JavaScript (using Bootstrap 4)
- API: RESTful API using Django Rest Framework

Supported Operating System

We can configure this project on following operating system.

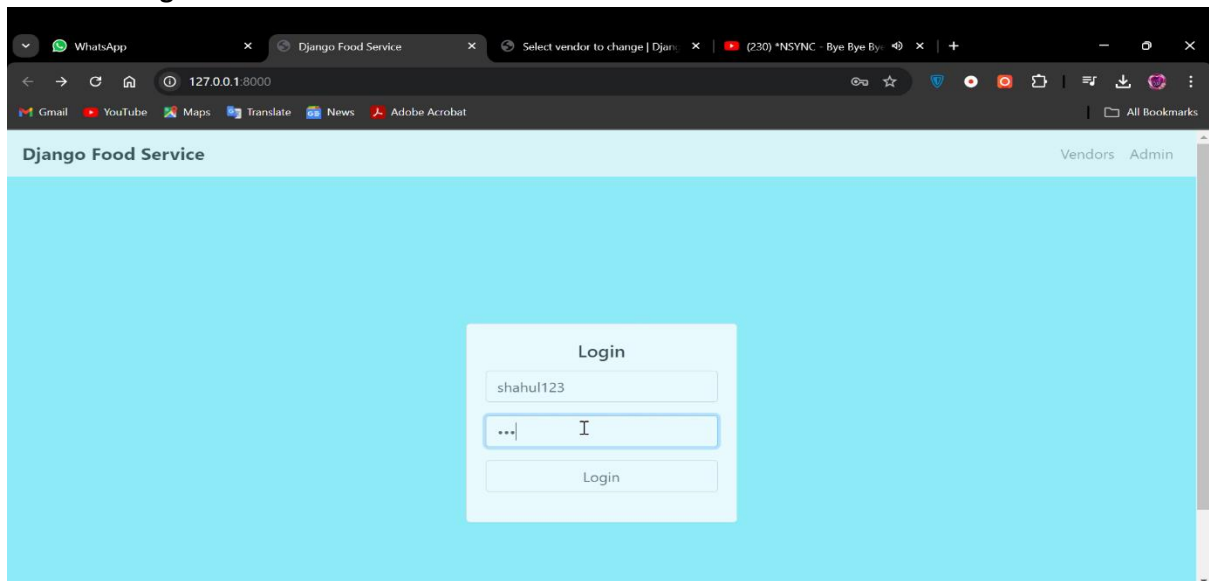
- Windows: This project can easily be configured on windows operating system. For running this project on Windows system, you will have to install
- Python 2.7, PIP, Django.
- Linux : We can run this project also on all versions of Linux operating systemMac : We can also easily configured this project on Mac operating system.

Installation and Setup

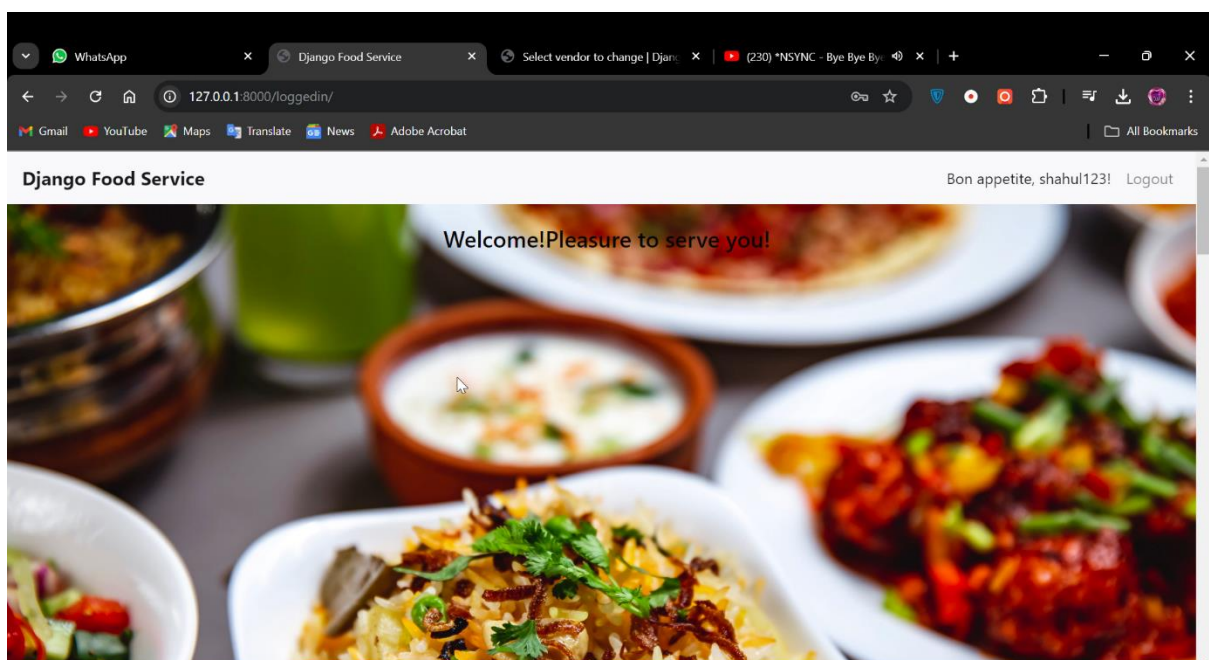
- Install Python 3.7 Or Higher
- Install Django version 2.2.0
- Install all dependencies cmd - **python -m pip install --user -r requirements.txt**
- Finally run cmd - **python manage.py runserver**
- After Creating an server , you can create a superuser for Login In cmd, Type python manage.py createsuperuser

Sample Screenshots

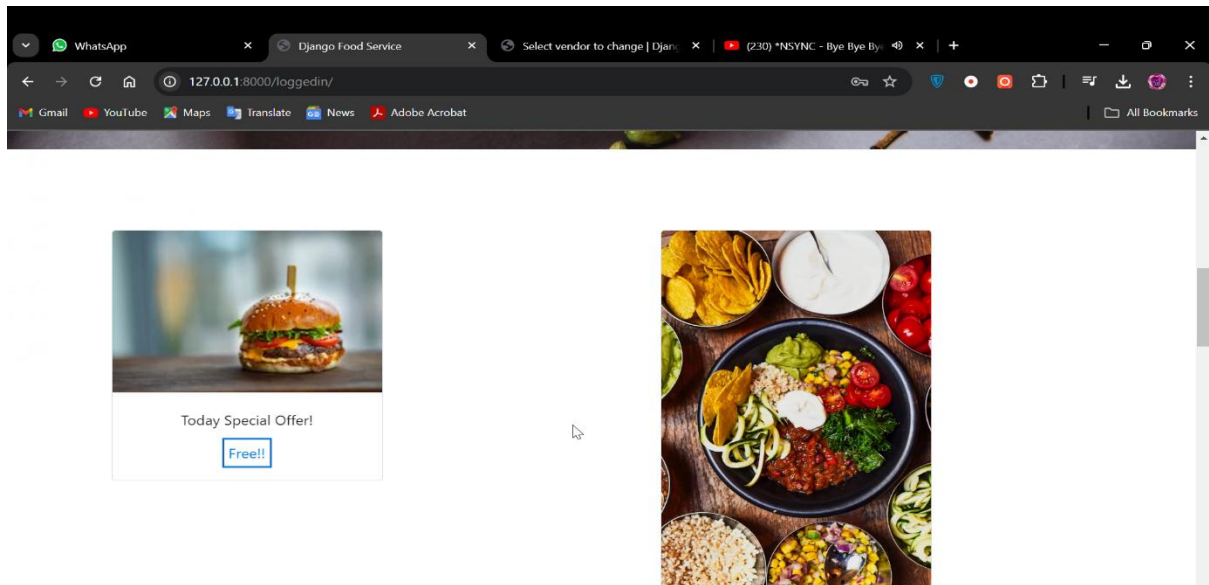
Customer Login



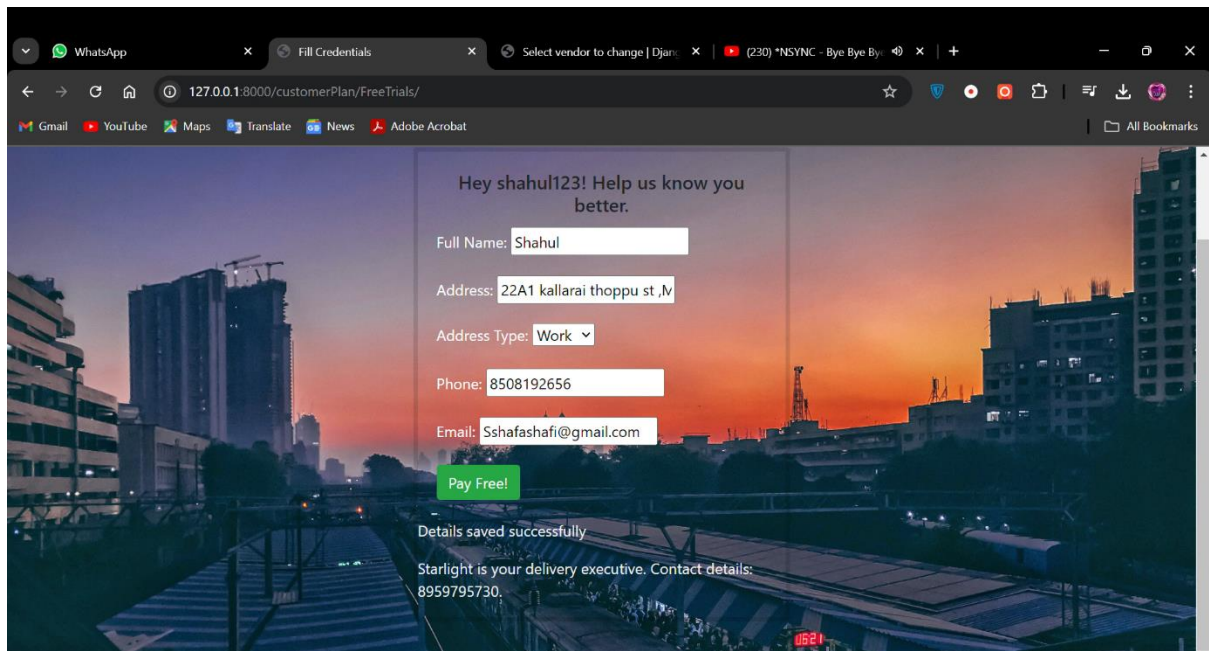
Customer Order Page



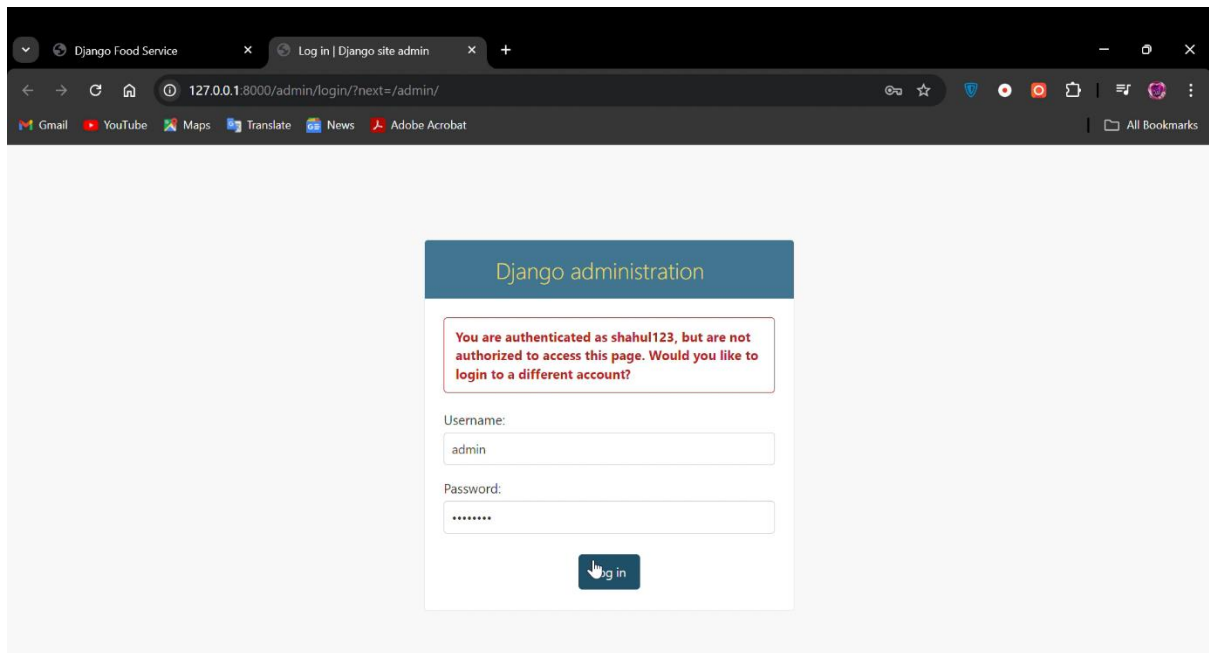
Food Packages



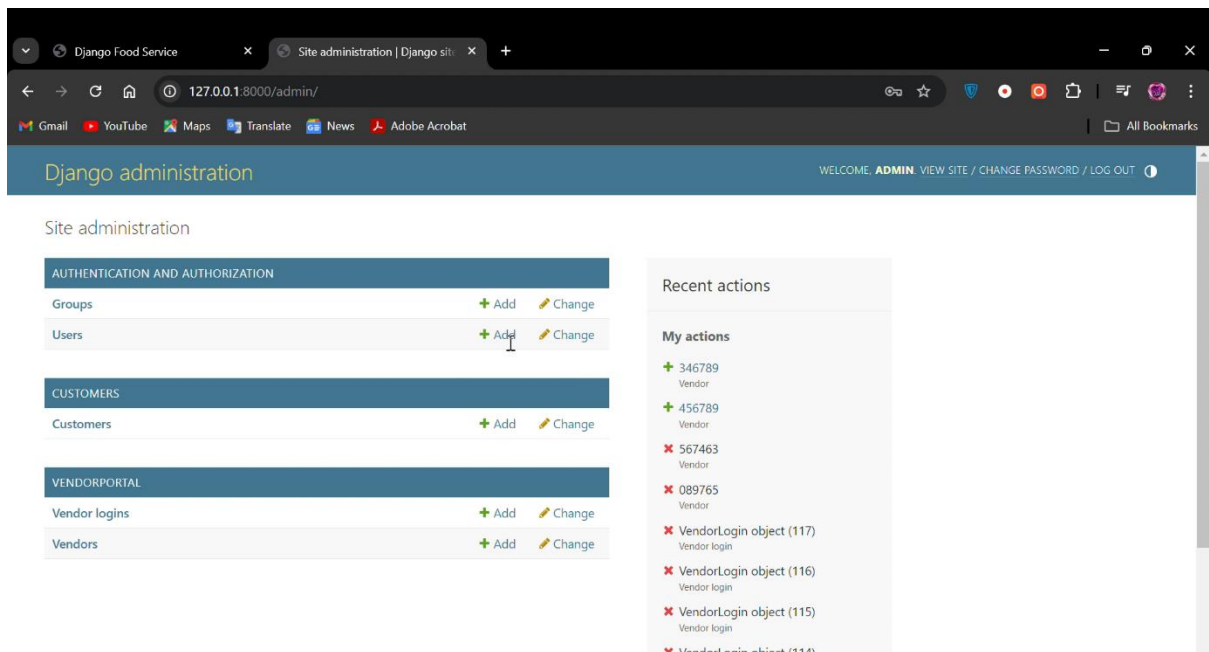
Customer Details For Order the Food



Admin Login Page



Edit/Delete/Add Page for Customer, Vendor



Vendor Add Page

The screenshot shows the Django administration interface for the Vendor Add Page. The browser address bar displays the URL `127.0.0.1:8000/admin/vendorPortal/vendor/15/change/`. The page title is "Django administration" and the user is logged in as "ADMIN". The breadcrumb trail is "Home > Vendorportal > Vendors > 346789".

The left sidebar contains the following menu items:

- AUTHENTICATION AND AUTHORIZATION
 - Groups + Add
 - Users + Add
- CUSTOMERS
 - Customers + Add
- VENDORPORTAL
 - Vendor logins + Add
 - Vendors + Add

The main content area is titled "Change vendor" and displays the following form fields:

- Vendor Id: 346789
- Vendor Name: Jason
- Vendor Phone: 9688381800
- Customers Delivering: Def
- Total Customers: 2

At the bottom of the form, there are four buttons: "SAVE", "Save and add another", "Save and continue editing", and "Delete".

Vendor Id Page For Viewing the Orders to Deliver

The screenshot shows a web page with a scenic background image of a lake and mountains. A modal dialog box is displayed in the center of the page, titled "Please Enter The Vendor ID". The dialog box contains the following text and form elements:

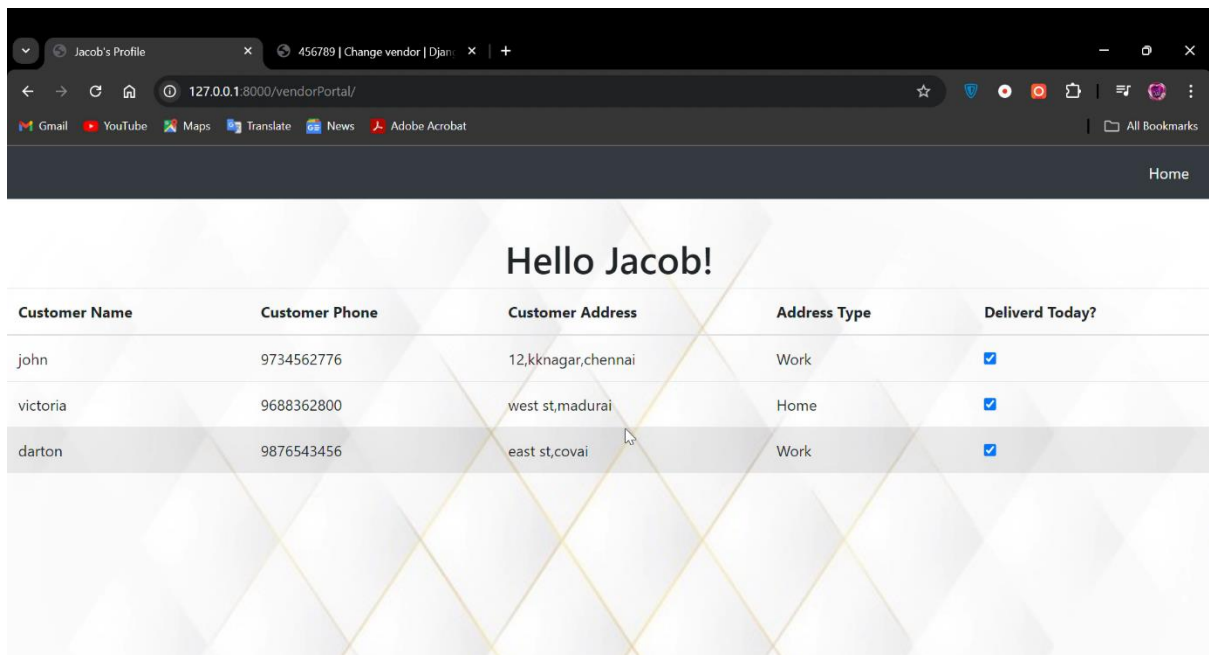
Please Enter The Vendor ID

Vendor ID:

456789

Submit

Vendor Page to view Order



Customer Name	Customer Phone	Customer Address	Address Type	Delivered Today?
john	9734562776	12,kknagar,chennai	Work	<input checked="" type="checkbox"/>
victoria	9688362800	west st,madurai	Home	<input checked="" type="checkbox"/>
darton	9876543456	east st,covai	Work	<input checked="" type="checkbox"/>

Note : You can create multiple Vendor ID , Based on the quantity of the vendor then the order can be placed in to various vendor ID

Security

- **Authentication:** Customers, vendors, and admins are authenticated using Django's built-in authentication system.
- **Authorization:** Access to API endpoints is restricted based on user role (customer, vendor, or admin).
- **Data Encryption:** Data is encrypted using SSL/TLS.