OrderOnTheGo: Your On-Demand Food Ordering Solution

# Project Overview

Title: OrderOnTheGo – Your On-Demand Food Ordering Solution

## 1. ER-Diagram

Entities:  
- User  
- Restaurant  
- MenuItem  
- Order  
- Payment  
  
Relationships:  
- A User places an Order  
- An Order contains multiple MenuItems  
- A Restaurant offers many MenuItems

## 2. Pre-Requisites

- Python, Flask  
- HTML/CSS/JS  
- SQLite (or MySQL)  
- REST API knowledge

## 3. Application Flow

1. User Login/Register  
2. Browse Restaurants and Menus  
3. Place Order  
4. Make Payment  
5. Track Order

## 4. Project Structure

/orderonthego  
│  
├── /static # CSS, JS files  
├── /templates # HTML templates  
├── app.py # Flask app  
├── database.db # SQLite database  
└── /models # DB models

## 5. Project Flow

### 1. Project Setup and Configuration

Install Flask: pip install flask  
Set up virtual environment  
Create app.py and folders

### 2. Database Development

import sqlite3  
  
conn = sqlite3.connect('database.db')  
c = conn.cursor()  
  
c.execute('''CREATE TABLE users (id INTEGER PRIMARY KEY, name TEXT, email TEXT)''')  
c.execute('''CREATE TABLE restaurants (id INTEGER PRIMARY KEY, name TEXT)''')  
c.execute('''CREATE TABLE menu\_items (id INTEGER PRIMARY KEY, name TEXT, price REAL, restaurant\_id INTEGER)''')  
c.execute('''CREATE TABLE orders (id INTEGER PRIMARY KEY, user\_id INTEGER, item\_id INTEGER, quantity INTEGER)''')  
  
conn.commit()  
conn.close()

### 3. Backend Development (Flask)

from flask import Flask, render\_template, request  
import sqlite3  
  
app = Flask(\_\_name\_\_)  
  
@app.route('/')  
def home():  
 conn = sqlite3.connect('database.db')  
 c = conn.cursor()  
 c.execute("SELECT \* FROM menu\_items")  
 items = c.fetchall()  
 return render\_template('home.html', items=items)  
  
@app.route('/order', methods=['POST'])  
def order():  
 item\_id = request.form['item\_id']  
 quantity = request.form['quantity']  
 conn = sqlite3.connect('database.db')  
 c = conn.cursor()  
 c.execute("INSERT INTO orders (user\_id, item\_id, quantity) VALUES (?, ?, ?)", (1, item\_id, quantity))  
 conn.commit()  
 return "Order placed!"  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app.run(debug=True)

### 4. Frontend Development (home.html)

<!DOCTYPE html>  
<html>  
<head><title>OrderOnTheGo</title></head>  
<body>  
<h2>Menu</h2>  
<form method="POST" action="/order">  
 {% for item in items %}  
 <p>{{ item[1] }} - ₹{{ item[2] }}</p>  
 <input type="hidden" name="item\_id" value="{{ item[0] }}">  
 Quantity: <input type="number" name="quantity" value="1">  
 <button type="submit">Order</button><br><br>  
 {% endfor %}  
</form>  
</body>  
</html>

### 5. Project Implementation & Execution

Run the app: python app.py  
Visit: http://127.0.0.1:5000  
Try placing orders and check console/db

## Sample Output

Console Output:  
 \* Running on http://127.0.0.1:5000/  
 \* Order placed!

Web Output:  
Menu:  
Pizza - ₹120 [Quantity: 1] [Order]  
Burger - ₹90 [Quantity: 1] [Order]