Architecture Design

by **RAVI KUMAR M**

Designing the architecture for your Restaurant Rating Predictor the project involves creating a structure that outlines how the different components interact and function together. Here's a step-by-step guide to help you conceptualise the architecture.

1. Architecture Overview

At a high level, the architecture involves the following components:

- Frontend: User-interactive HTML forms and templates.
- Backend (Flask: Handles user requests, processes the data, and returns predictions.
- Model: A trained machine learning model that makes predictions based on user inputs.
- Login and Session Management: Controls user authentication and access to the prediction feature.
- Data (Optional): If you choose to store or log user inputs or predictions.

2. Components of the Architecture

a. User Interface (UI) / Frontend

- Purpose: To provide an interface for users to log in, input data for restaurant ratings, and receive predictions.
- Technologies: HTML, CSS, and optionally JavaScript for enhanced user experience.

Key Features:

- Login Page: Allows admin to log in.
- Prediction Form: Allows users to select or input values like `Online Order`, `Book Table`, `Votes`, `Location`, etc.
- Prediction Output: Displays the predicted restaurant rating.

Interaction Flow:

- User visits the login page.
- After login, the user is redirected to the prediction page.
- The user fills out the form and submits the prediction request.

b. Backend / API Layer (Flask Application)

- Purpose: To handle incoming HTTP requests, process data, interact with the machine learning model, and render appropriate responses.
- Technologies: Flask (Python)

Key Features:

Routes:

- `/login`: Handles login authentication and redirects to the prediction page.
- `/predict`: Handles form data and returns predictions.
- `/logout`: Logs out the user and ends the session.

Prediction Logic:

- The backend fetches user inputs, transforms the data, and passes it to the trained model.
- The prediction result is returned and displayed on the frontend.

Interaction Flow:

- User data is passed from the form to the backend.
- The backend processes the inputs and sends them to the machine learning model.
- The model returns a prediction that the backend sends to the frontend.

c.Machine Learning Model

- Purpose: To predict restaurant ratings based on user inputs like `Online Order`, `Location`, etc.
- Technologies: Scikit-learn or any other library you've used to train the model.

Key Features:

- The trained model is stored as a serialized `.pkl` file.
- The model predicts based on input features and returns a numeric or categorical value (e.g., a rating).

Interaction Flow:

- The backend loads the model.
- Input data is preprocessed (label encoding, scaling, etc.).
- Predictions are generated based on the preprocessed data.

d. Session Management (Login System)

- Purpose: To secure the prediction page and ensure that only authorized users can make predictions.
- Technologies: Flask's built-in session management.

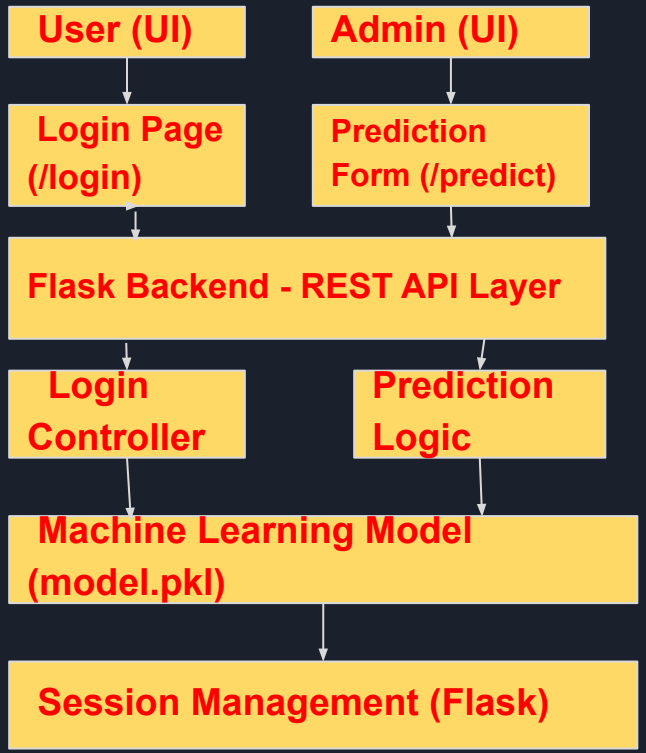
Key Features:

- Login Authentication: Verifies the username and password before allowing access to the prediction form.
- Session Handling: Stores the login state so that users don't need to log in repeatedly.

Interaction Flow:

- User provides login credentials.
- Flask session stores the user state as `logged_in`.
- Protected routes check if the user is logged in before allowing access.

3. Architecture Diagram



4. Flow of Actions

1. User Visits Login Page:

- The user enters their credentials (username and password).
- The backend checks the credentials and, if valid, starts a session.

2. After Login:

- The user is redirected to the prediction form page.
- The user selects options like `Online Order`, `Book Table`, and other features from dropdowns.
- User submits the form to request a prediction.

3. Backend Receives the Form:

- The backend fetches the form inputs.
- It preprocesses the data (e.g., encoding categorical variables).
- The backend passes the processed data to the trained model.

4. Model Prediction:

- The machine learning model generates a prediction based on the input data.
- The prediction result is returned to the backend.

5. Display the Prediction:

- The backend sends the prediction to the frontend.
- The user sees the restaurant rating prediction on the screen.

6. Session Handling:

- If the user logs out, the session is cleared, and the user is redirected to the login page.

5. Security Considerations:

- Since you are using session-based authentication (no database):
- Use a strong **secret key** for the Flask session to ensure sessions are secure.
- Protect sensitive routes by checking the session state.
- Make sure to clear the session on logout to avoid unauthorized access.

6. Scalability Considerations

While this is a simple project, consider the following for future scalability:

- ❏ **Database Integration: If you plan to store user data or prediction logs, introduce a database (e.g., SQLite, MySQL).**
- ❏ **API Versioning: If you intend to evolve the API, consider versioning the prediction API (`/api/v1/predict`).**
- ❏ **Load Balancing: For heavy traffic, use load balancers if you plan to deploy this to production.**

This architecture ensures that all components interact seamlessly, providing a good user experience while keeping the backend secure. Let me know if you'd like further help on any specific part!

Thank You