**TABLE BOOKING APP**

**Introduction :**

Welcome to our Table Booking App! Our platform is dedicated to creating a seamless online reservation experience for all your dining needs. Whether you're planning a romantic dinner, a family gathering, or a casual night out with friends, our app offers a user-friendly interface to help you secure the perfect dining spot.

With an intuitive design, our e-commerce app allows you to effortlessly explore various restaurants, view their cuisines and menus, and make reservations with just a few clicks. You can browse through a diverse range of dining establishments, from cozy cafes to elegant fine dining, ensuring you find the ideal ambiance for your occasion.

Our priority is to ensure customer satisfaction by providing a hassle-free booking journey. From selecting the right restaurant to specifying your seating preferences, our app is designed to make your reservation experience both enjoyable and convenient. Whether it's a special celebration or a casual outing, our app is here to make your dining experience memorable.

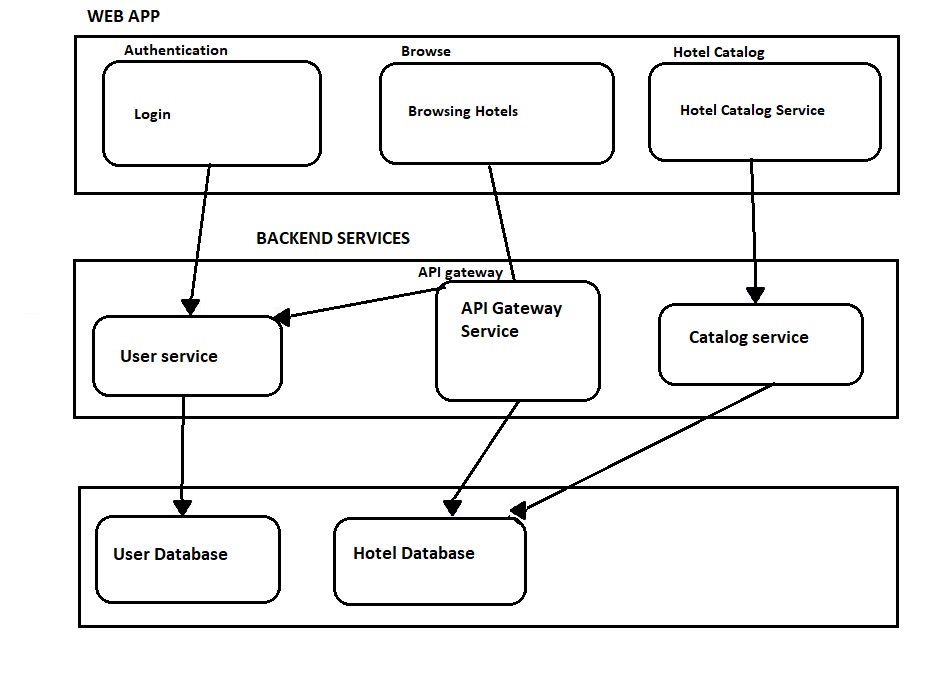
**Discover the Best Dining Experiences:** Welcome to the ultimate Table Booking App, where culinary delights await your exploration. With a commitment to offering exceptional dining experiences, our platform lets you discover a diverse array of restaurants, each with its own unique charm and menu offerings.

**Effortless Reservation Process:** Say goodbye to waiting in long queues or making endless phone calls. Our app streamlines the reservation process, allowing you to secure your desired table at your favorite restaurant within minutes. Just a few taps and you're on your way to a delightful dining experience.

**Seamless Group Reservations:** Planning a gathering? Our app simplifies group reservations, making it easy to coordinate with friends, family, or colleagues. Specify your group size, preferences, and let us take care of ensuring everyone has a seat at the table.

**Your Dining Companion:** Our Table Booking App isn't just a tool; it's your dining companion. Whether you're a culinary explorer, a food enthusiast, or someone who simply enjoys sharing meals with loved ones, our app is here to enhance your dining journey in every way possible.

**TECHNICAL ARCHITECTURE:**

****

The technical architecture of an table booking app typically involves a client-server model, where the frontend represents the frontend and the backend serves as the server. The frontend is responsible for user interface, interaction, and presentation, while the backend handles data storage, business logic, and integration with external services like bookings and databases. Communication between the frontend and backend is typically facilitated through APIs, enabling seamless data exchange and functionality.

### ER DIAGRAM:

### 

The Entity-Relationship (ER) diagram for a Table booking app visually represents the relationships between different entities involved in the system, such as users, admin, restaurants, and reviews. It illustrates how these entities are related to each other and helps in understanding the overall database structure and data flow within the application.

### KEY FEATURES:

**Product Catalog:** Our Flower and Gift Delivery App provides an extensive product catalog with various categories and subcategories. Users can easily search, browse, and filter products based on their preferences, making it effortless to find the desired items.

**Shopping Cart and Checkout:** The app includes a shopping cart feature that enables users to add products, review their cart, and proceed to checkout. The checkout process offers multiple payment options, ensuring a smooth and secure transaction experience.

**Product Reviews and Ratings:** Customers can provide feedback and rate products, helping other users make informed purchasing decisions. This feature fosters a sense of community and trust among users.

**Order Tracking:** Once an order is placed, users can track its status in real-time. They receive updates on order processing, shipping, and delivery, providing transparency and peace of mind.

**Admin Dashboard:** For administrators, our Flower and Gift Delivery App offers a comprehensive dashboard to manage products, inventory, orders, and customer information. It provides insights into sales performance, stock levels, and customer analytics, enabling efficient business operations.

**Order Management:** The app manages the order lifecycle, including order placement, tracking, and status updates. Users can view their order history, track shipments, and request returns or cancellations.

**Search and Filtering:** Users can search for products using keywords and apply filters to narrow down the search results based on criteria such as price range, brand, or customer ratings.

**PRE REQUISITES:**

To develop a full-stack table booking app using AngularJS, Node.js, and MongoDB, there are several prerequisites you should consider. Here are the key prerequisites for developing such an application:

**Node.js and npm:** Install Node.js, which includes npm (Node Package Manager), on your development machine. Node.js is required to run JavaScript on the server side.

* Download: <https://nodejs.org/en/download/>
* Installation instructions: <https://nodejs.org/en/download/package-manager/>

**MongoDB:** Set up a MongoDB database to store hotel and booking information. Install MongoDB locally or use a cloud-based MongoDB service.

* Download: <https://www.mongodb.com/try/download/community>
* Installation instructions: <https://docs.mongodb.com/manual/installation/>

**Express.js:** Express.js is a web application framework for Node.js. Install Express.js to handle server-side routing, middleware, and API development.

* Installation: Open your command prompt or terminal and run the following command: **npm install express**

**Angular:** Angular is a JavaScript framework for building client-side applications. Install Angular CLI (Command Line Interface) globally to create and manage your Angular project.

Install Angular CLI:

* Angular provides a command-line interface (CLI) tool that helps with project setup and development.
* Install the Angular CLI globally by running the following command:

npm install -g @angular/cli

**Verify the Angular CLI installation:**

* Run the following command to verify that the Angular CLI is installed correctly: ng version You should see the version of the Angular CLI printed in the terminal if the installation was successful.

Create a new Angular project:

* Choose or create a directory where you want to set up your Angular project.
* Open your terminal or command prompt.
* Navigate to the selected directory using the cd command.
* Create a new Angular project by running the following command: ng new cl Wait for the project to be created:
* The Angular CLI will generate the basic project structure and install the necessary dependencies.

**Navigate into the project directory:**

* After the project creation is complete, navigate into the project directory by running the following command**: cd frontend**

###### Start the development server:

* To launch the development server and see your Angular app in the browser, run the following command: **ng serve / npm start**
* The Angular CLI will compile your app and start the development server.
* Open your web browser and navigate to http://localhost:4200 to see your Angular app running.

You have successfully set up Angular on your machine and created a new Angular project. You can now start building your app by modifying the generated project files in the src directory.

Please note that these instructions provide a basic setup for Angular. You can explore more ad- vanced configurations and features by referring to the official Angular documentation: https://angular.io

**HTML, CSS, and JavaScript:** Basic knowledge of HTML for creating the structure of your app, CSS for styling, and JavaScript for client-side interactivity is essential.

**Database Connectivity:** Use a MongoDB driver or an Object-Document Mapping (ODM) library like Mongoose to connect your Node.js server with the MongoDB database and perform CRUD (Create, Read, Update, Delete) operations.

**Front-end Framework:** Utilize Angular to build the user-facing part of the application, including products listings, booking forms, and user interfaces for the admin dashboard.

**Version Control**: Use Git for version control, enabling collaboration and tracking changes throughout the development process. Platforms like GitHub or Bitbucket can host your repository.

* Git: Download and installation instructions can be found at: [https://git-](https://git-scm.com/downloads) [scm.com/downloads](https://git-scm.com/downloads)

**Development Environment:** Choose a code editor or Integrated Development Environment (IDE) that suits your preferences, such as Visual Studio Code, Sublime Text, or WebStorm.

* Visual Studio Code: Download from<https://code.visualstudio.com/download>
* Sublime Text: Download from <https://www.sublimetext.com/download>
* WebStorm: Download from <https://www.jetbrains.com/webstorm/download>

To Connect the Database with Node JS go through the below provided link:

* Link: [https://www.section.io/engineering-education/nodejs- mongoosejs-mongodb/](https://www.section.io/engineering-education/nodejs-%20mongoosejs-mongodb/)

###### **To run the existing Table Booking App project downloaded from github: Follow below steps:**

1. **Clone the Repository:**
   * Open your terminal or command prompt.
   * Navigate to the directory where you want to store the e-commerce app.
   * Execute the following command to clone the repository:

**git clone: https://github.com/Namitha2002/TableBooking-using-MEAN.git**

###### **Install Dependencies:**

* + Navigate into the cloned repository directory:

cd BookingApp

* + Install the required dependencies by running the following command:

npm install

###### **Start the Development Server:**

* + To start the development server, execute the following command:

npm start

* + The Table booking app will be accessible at http://localhost:5100 by default. You can change the port configuration in the .env file if needed.

###### **Access the App:**

* + Open your web browser and navigate to http://localhost:5100.
  + You should see the e-commerce app's homepage, indicating that the installation and setup were successful.

**Project Repository Link: https://github.com/Namitha2002/TableBooking-using-MEAN.git**

Congratulations! You have successfully installed and set up the table booking app on your local machine. You can now proceed with further customization, development, and testing as needed.

###### **Roles and Responsibilities:**

The project has two types of users –Customer and Admin. The roles and responsibilities of these two types of users can be inferred from the API endpoints defined in the code. Here is a summary:

**CUSTOMER:**

1. Create an account and log in to the system using their email and password.
2. Browse and search for hotels available on the platform.
3. View detailed hotel information, including description and availability.
4. Book hotel as per your need.
5. Proceed to booking page and place your booking for selected hotels.
6. Make sure your bookings have been successful.
7. Track the status of your bookings.
8. Manage their profile information, including personal details.
9. Provide feedback and reviews for products and sellers.
10. Access customer support for any queries or issues related to their orders.

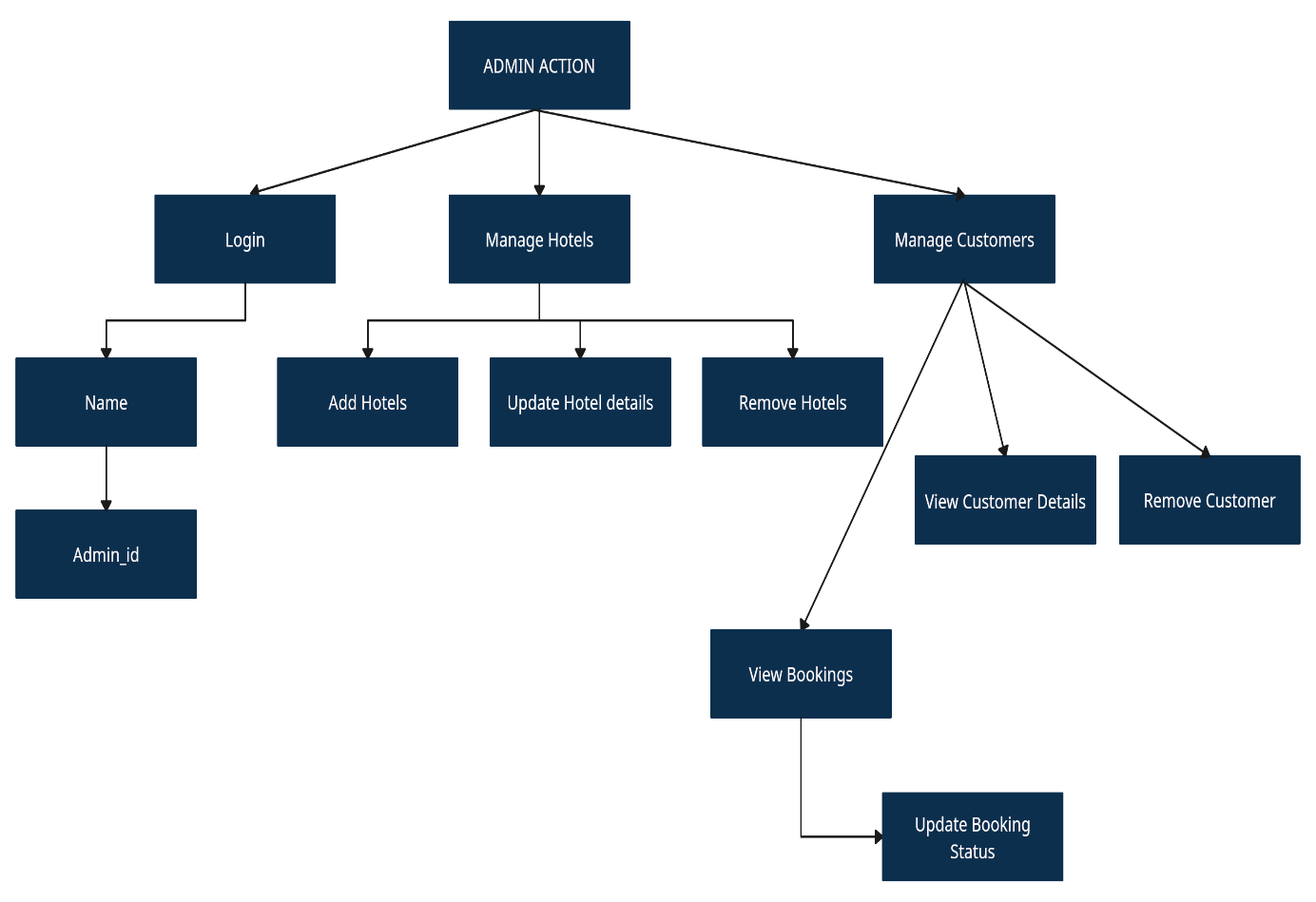
**ADMIN:**

1. Manage and monitor the overall operation of the table booking platform.
2. Approve and update new hotels.
3. Monitor and moderate hotel listings, ensuring compliance with guidelines and policies.
4. Handle customer disputes and resolve issues.
5. Manage user accounts, including customer and hotel profiles.
6. Analyze platform performance and generate reports on hotels, customer behavior, and application popularity.
7. Implement and enforce platform policies, terms of service, and privacy regulations.
8. Continuously improve the platform's functionality, user experience, and security measures

These roles and responsibilities are aimed at ensuring a smooth and efficient operation of the table booking app, providing a seamless experience for customers, sellers, and administrators.

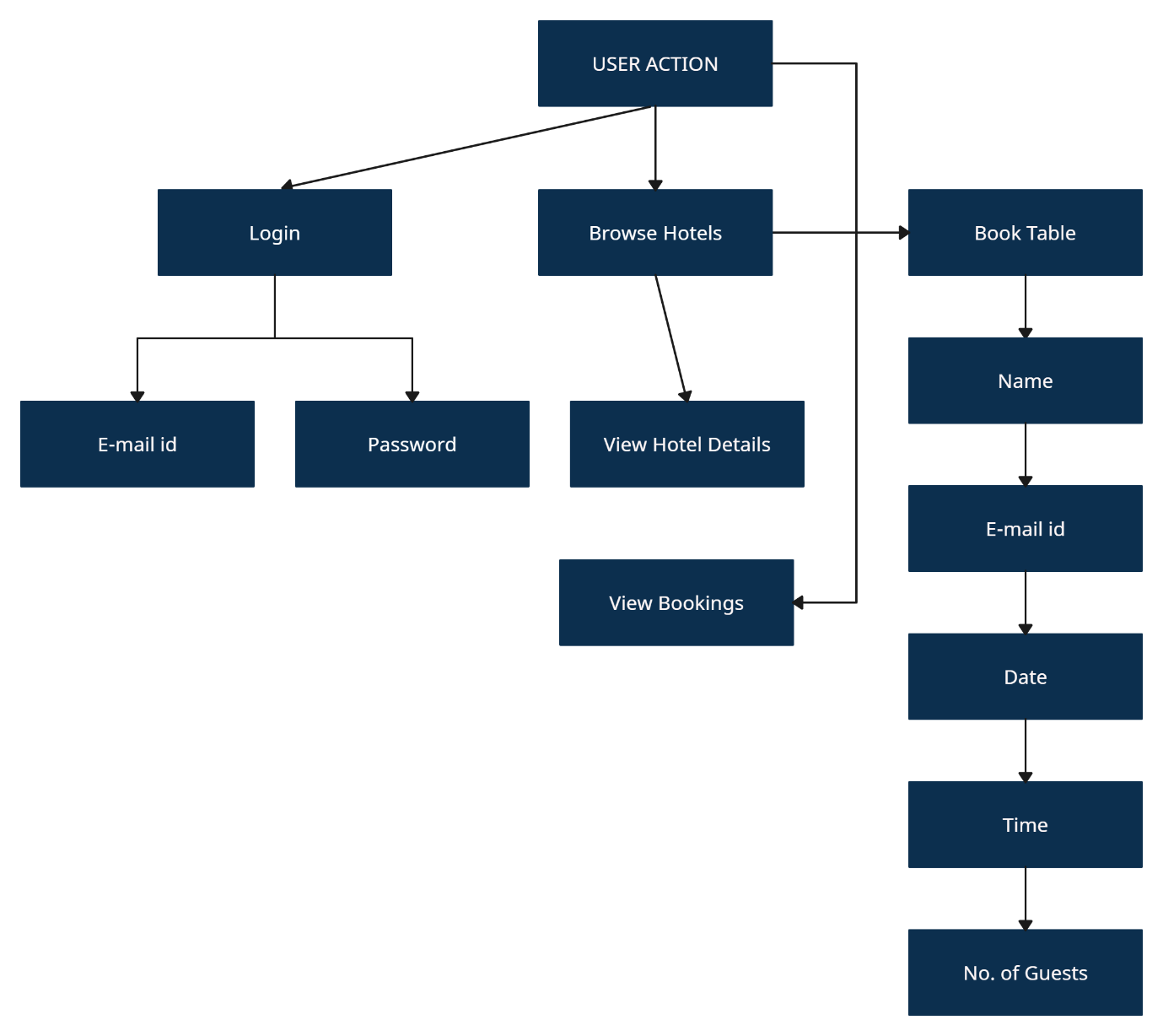
**Admin and User Flow:**

**Admin:**

****

UserFlow:

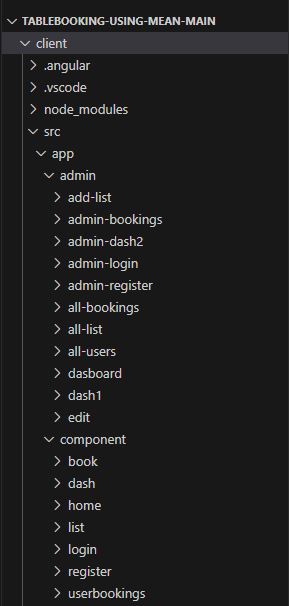
User:

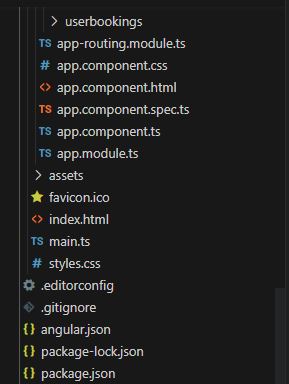


The project flow for a table booking app involves user actions such as login, browsing hotels, viewing bookings, viewing hotel details and receiving booking confirmation. Admin actions include login, managing hotels, viewing and processing bookings, managing customers, and updating hotel details and updating booking confirmation.

**PROJECT STRUCTURE:**

**Front End:**

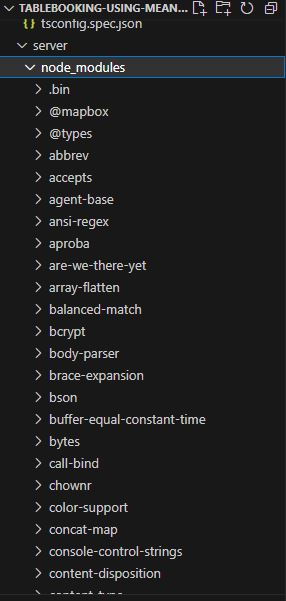
****

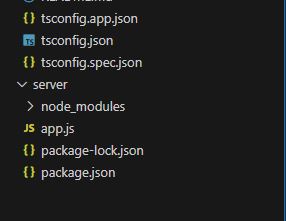
****

This structure assumes an Angular app and follows a modular approach. Here's a brief explanation of the main directories and files:

* + src/app/components: Contains components related to the customer app, such as register, login, home, booking, dashboard, hotel details, booking confirmation and more.
  + src/app/modules: Contains modules for different sections of the app. In this case, the admin module is included with its own set of components like add-category, add-hotel, dashboard, home, booking, update-hotel, users, and more.
  + src/app/app-routing.module.ts: Defines the routing configuration for the app, specifying which components should be loaded for each route.
  + src/app/app.component.ts, src/app/app.component.html, `src.

**Back End:**

****

****

###### **Project Flow:**

**Milestone 1: Project Setup and Configuration:**

1. Install required tools and software:
   * Node.js.
   * MongoDB.
   * Angular CLI.
2. Create project folders and files:
   * Frontend folders.
   * Backend folders.

###### Milestone 2: Backend Development:

Setup express server:

* + Install express.
  + Create app.js file.
  + Define API’s

Configure MongoDB:

* + Install Mongoose.
  + Create database connection.
  + Create Models.

Implement API end points:

* + Implement CRUD operations.
  + Test API endpoints.

**Backend:**

1. Set Up Project Structure:
   * Create a new directory for your project and set up a package.json file using npm init command.
   * Install necessary dependencies such as Express.js, Mongoose, and other required packages.
2. Database Configuration:
   * Set up a MongoDB database either locally or using a cloud-based MongoDB service like MongoDB Atlas.
   * Create a database and define the necessary collections for hotels, users, bookings, and other relevant data.
3. Create Express.js Server:
   * Set up an Express.js server to handle HTTP requests and serve API endpoints.
   * Configure middleware such as body-parser for parsing request bodies and cors for handling cross-origin requests.
4. Define API Routes:
   * Create separate route files for different API functionalities such as hotels, users, bookings, and authentication.
   * Define the necessary routes for listing hotels, handling user registration and login, managing bookings, etc.
   * Implement route handlers using Express.js to handle requests and interact with the database.

**Implement Data Models:**

* + Define Mongoose schemas for the different data entities like hotels, users, and bookings.
  + Create corresponding Mongoose models to interact with the MongoDB database.
  + Implement CRUD operations (Create, Read, Update, Delete) for each model to perform database operations.

**API Design and Development:**

* + Identify the necessary functionality and data required by the frontend.
  + Design a set of RESTful APIs using a framework like Express.js or Django REST Framework.
  + Define API endpoints for user management, product catalog, shopping cart, order management, payment gateway integration, shipping integration, etc.
  + Implement the API routes, controllers, and data models to handle the corresponding operations.
  + Ensure that the APIs follow best practices, are secure, and provide appropriate re- sponses.

**User Management and Authentication:**

* + Implement user registration and login functionality.
  + Choose an authentication mechanism like session-based authentication or token-based authentication (e.g., JWT).
  + Store and hash user credentials securely.
  + Implement middleware to authenticate API requests and authorize access to protected routes

**Catalog and Inventory Management:**

* + Design the database schema to store product details, pricing, availability, and invento- ry levels.
  + Create APIs to retrieve product information, update inventory quantities, and handle search and filtering.
  + Implement validations to ensure data integrity and consistency.

**Booking Management:**

* + Design the database to store customer details and booking information.
  + Create APIs to handle cart operations like adding hotels, modifying quantities, and booking tables.

**Database Integration:**

* + Choose a suitable database technology (e.g., MySQL, PostgreSQL, MongoDB) based on your application's requirements.
  + Design the database schema to efficiently store and retrieve e-commerce data.
  + Establish a connection to the database and handle data persistence and retrieval.

**External Service Integration:**

* + Identify third-party services like email service providers, analytics services, or CRM systems that are required for your application.
  + Utilize the APIs or SDKs provided by these services to exchange data and perform necessary operations.
  + Implement the integration logic to send order confirmations, track user behavior, or manage customer relationships.

**Security and Data Protection:**

* + Apply appropriate security measures like encryption techniques for secure data transmission and storage.
  + Implement input validation and sanitization to prevent common security vulnerabili- ties.
  + Implement access control to ensure authorized access to sensitive data.

**Error Handling and Logging:**

* + Implement error handling mechanisms to handle exceptions and provide meaningful error messages to the frontend.
  + Use logging frameworks to record application logs for monitoring and troubleshooting purposes.

**USE-CASE:**

1. **Users:**

* Model: 'User'
* Purpose: Represents model for user data, including information like name, email, password, and other relevant details. It is used for user registration, authentication, and managing user-related functionalities.

1. **Category:**

* Model: 'Category'
* Purpose: Represents model for hotel categories. It defines the structure for category data, such as name, description, and any other attributes related to categorizing hotels. It is used to manage and organize hotel categories within the e- commerce app.

1. **Hotel:**

* Model: 'Hotel'
* Purpose: Represents the model for individual hotels available in the e- commerce app. It includes attributes like name, description, images, and other details specific to hotel. It is used for product listing, details, and management within the app.

1. **Booking:**

* Model: 'Booking'
* Purpose: Represents model for customer booking placed in the e-commerce app. It includes details like booking\_id, user information, booking status, no.of guests, date, time and more. It is used for managing the booking lifecycle, tracking, and processing.

**Frontend:**

**User Interface (UI) Design:**

* + Create a visually appealing and consistent design using modern design principles.
  + Use a UI design tool like Adobe XD, Sketch, Figma, or InVision to create wireframes and mockups.
  + Pay attention to typography, color schemes, spacing, and visual hierarchy.
  + Use responsive design techniques to ensure the app looks great on different devices.

**Responsive Design:**

* + Utilize CSS media queries and responsive design frameworks like Bootstrap or Tail- wind CSS to create a responsive layout.
  + Test your app on various devices and screen sizes to ensure a seamless user experience.

**Hotel Catalog:**

* + Design and implement a hotel listing page that displays hotel images, titles, de scriptions, and other relevant details.
  + Implement search functionality to allow users to find hotels easily.
  + Include filters and sorting options to enhance the browsing experience.

**Browsing Hotels:**

* + Design and develop a browsing hotel component to allow users to browse hotels, view images of the hotels, and more.
  + Create a checkout process with certain information given by the user to book their table in the respective hotels.

**User Authentication and Account Management:**

* + Design and implement a user registration and login system.
  + Create user profile pages where users can view and edit their personal information, addresses, payment methods, and order history.
  + Implement authentication guards to restrict access to certain pages or features.