**Medley pharma**

**Introduction:**

Medley Pharma is a versatile and dynamic software tool designed to simplify the process of creating and managing online forms. With its intuitive interface and powerful features, Medley Forma empowers users to build customized forms for various purposes, such as surveys, registrations, feedback collection, and more.

Medley Forma offers a wide range of form field options, allowing users to gather specific information from respondents. These fields can include text inputs, multiple-choice questions, checkboxes, dropdown menus, file uploads, and many other types of interactive elements. Users can easily drag and drop these fields into their form, arrange them in a logical order, and customize their appearance and validation rules to suit their needs.

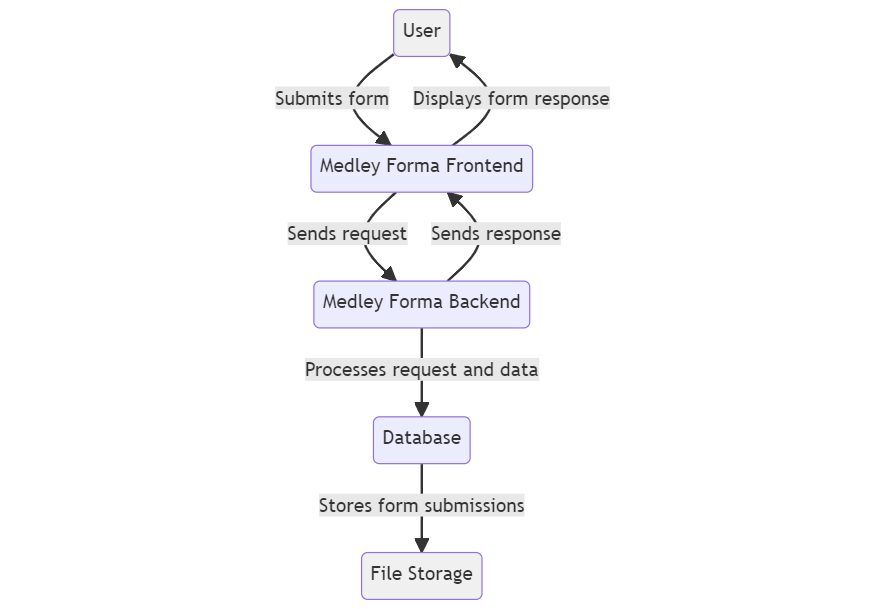
One of the key strengths of Medley Forma is its flexibility in form design. Users have full control over the visual aspects of their forms, including layout, colors, fonts, and branding elements. This enables organizations to maintain a consistent and professional look throughout their forms, aligning with their brand identity.

Medley Forma also prioritizes data security and privacy. It provides options for adding CAPTCHA verification, SSL encryption, and form submission limits to protect against spam and ensure the integrity of the data collected. Additionally, the platform offers seamless integration with popular third-party services and applications, enabling users to automate workflows, send notifications, and manage form responses effectively.

Furthermore, Medley Forma simplifies the process of analyzing and managing form submissions. It offers intuitive data visualization tools, allowing users to generate reports, charts, and graphs based on the collected responses. The platform also supports exporting form data in various formats, facilitating further analysis and sharing with team members or stakeholders.

Whether you're a business owner, event organizer, researcher, or educator, Medley Forma provides a user-friendly and feature-rich solution for creating and managing online forms. Its versatility, customization options, and robust functionality make it an ideal choice for individuals and organizations seeking an efficient and reliable tool to streamline their data collection and form management processes.

**Technical Architecture:**

****

In this architecture, the flow begins with a user (represented by the "User" node) interacting with the Medley Forma frontend (represented by the "Medley Forma Frontend" node). The user submits a form through the frontend interface.

The Medley Forma frontend then sends a request to the Medley Forma backend (represented by the "Medley Forma Backend" node). The backend processes the request and the associated form data.

The backend communicates with a database (represented by the "Database" node) to store the form submissions securely. The database stores the submitted form data, including the responses provided by the user.

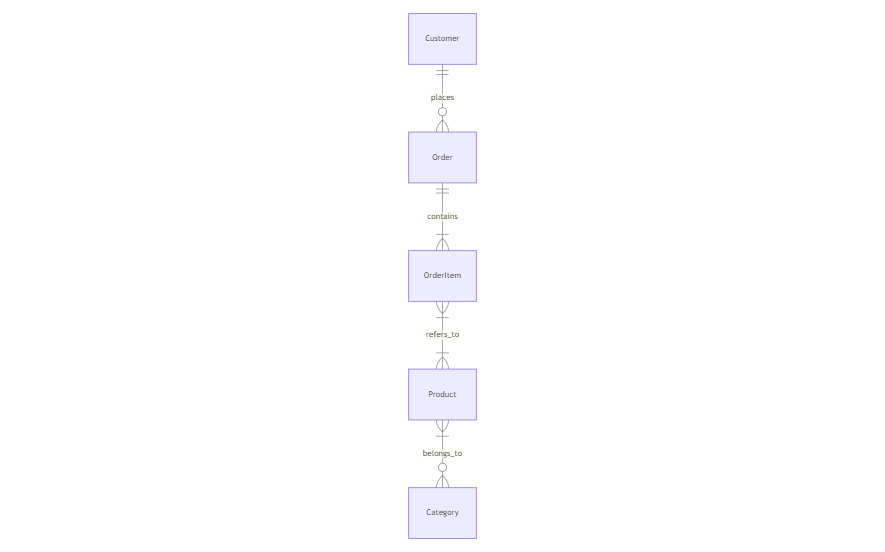
Additionally, the backend may utilize a file storage system (represented by the "File Storage" node) to store any files or attachments associated with the form submissions. This can include uploaded images, documents, or any other files relevant to the form.

Once the backend has processed the request, it sends a response back to the Medley Forma frontend. The frontend then displays the form response to the user.

The "style" statements at the bottom are optional and can be used to customize the appearance of specific nodes. In this example, the "User" and "File Storage" nodes have a light gray background.

Please note that this is a simplified representation of the technical architecture of Medley Forma. The actual implementation may involve additional components, such as authentication, form rendering, API gateways, load balancers, and more, depending on the specific requirements and design choices of the system.

**Er –Diagram:**

****

In this example, we have four entities: "Customer," "Order," "OrderItem," and "Product." The relationships between these entities are defined using various notations:

|| represents a one-to-many relationship, indicating that a customer can place multiple orders.

|--|{ represents a many-to-many relationship, indicating that an order can contain multiple order items, and an order item can belong to multiple orders.

}|--o{ represents a one-to-many relationship with optional participation, indicating that a product can belong to multiple order items, but an order item refers to a single product.

}|--o{ represents a one-to-many relationship with optional participation, indicating that a category can have multiple products, but a product belongs to a single category.

Feel free to modify this code to fit your specific requirements. You can try it out in the Mermaid Live Editor to visualize the ER diagram.

**Key Features:**

The key features of a Medley Forma booking app can vary depending on its specific functionality and target audience. However, here are some common key features that a booking app like Medley Forma might include:

User Registration and Profiles: Allow users to create accounts, set up profiles, and manage their personal information.

Booking Management: Allow users to search, browse, and book services based on their preferences, including selecting specific dates, times, and service providers.

Payment Integration: Facilitate secure online payments, supporting various payment methods, such as credit/debit cards, mobile wallets, or digital payment platforms.

Reviews and Ratings: Enable users to provide feedback and rate their experience with the booked services, helping others make informed decisions.

Booking History: Maintain a record of past and upcoming bookings for users to reference and manage.

Remember, these are just general features that a booking app like Medley Forma might have. The actual feature set can be customized based on the specific needs and requirements of the platform.

**PRE REQUISITES:**

To develop a full-stack expense tracker app using React js, 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**

**React**: React is a JavaScript library for building user interfaces. To create and manage your React project, you can use Create React App, a popular tool for bootstrapping React applications.

**Install Create React App:**

React offers a command-line tool called Create React App that simplifies project setup and development.

**To install Create React App globally, run the following command:**

npm install -g create-react-app

**Create a new React project:**

Choose or create a directory where you want to set up your React project.

Open your terminal or command prompt.

Navigate to the selected directory using the cd command.

**Create a new React project by running the following command:**

npx create-react-app my-react-app

Replace my-react-app with your preferred project name. Wait for the project to be created.

Navigate into the project directory:

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

cd my-react-app

**Start the development server:**

To launch the development server and see your React app in the browser, run the following command:

npm start

Create React App will compile your app and start the development server.

Open your web browser and navigate to http://localhost:3000 to see your React app running.

You have successfully set up React on your machine and created a new React 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 React. You can explore more advanced configurations and features by referring to the official React documentation: https://reactjs.org

**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 React 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-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](%20https:/code.visualstudio.com/download%20)
* 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 grocery-web 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 grocery-webapp app.
* Execute the following command to clone the repository:

**git clone** <https://github.com/Bharath136/Medley-Farma>

**2. Install Dependencies:**

* Navigate into the cloned repository directory:

cd grocery-webapp

* Install the required dependencies by running the following command:

npm install

**3. Start the Development Server:**

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

npm run dev or npm run start

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

**4. Access the App:**

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

**Video Tutorial Link to clone the project: -** <https://drive.google.com/file/d/1KTGK0XZj0XWOiDeNKJVRKQHXLyVWZYLM/view?usp=sharing>

**Project Repository Link:** <https://github.com/Bharath136/Medley-Farma>

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

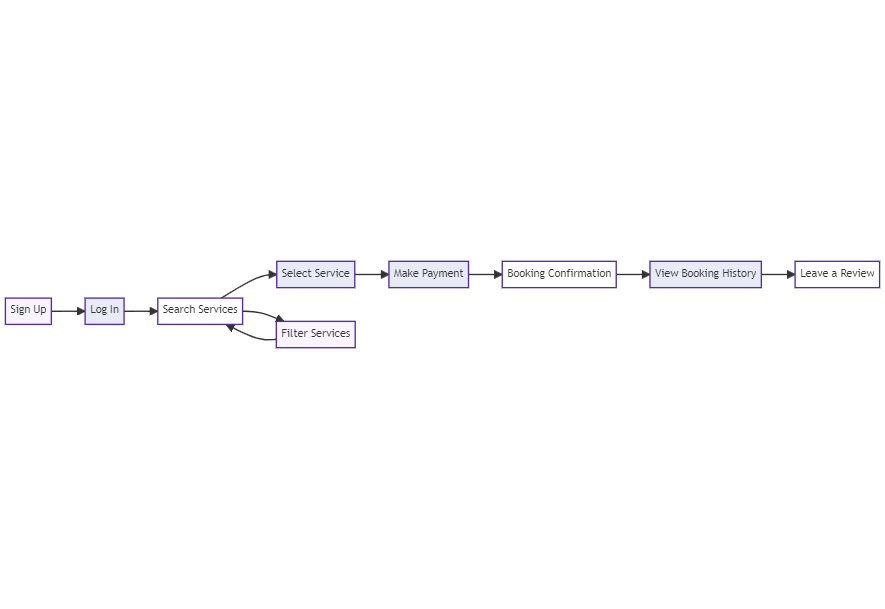
**Customer Role and Responsibilities:**

* Registration and Account Management: Customers are responsible for creating an account, providing accurate personal information, and managing their account details, including profile updates and password security.
* Service Search and Booking: Customers can search for available services based on their preferences, such as location, date, time, and type of service. They are responsible for selecting the desired service, booking it, and providing any necessary information for the booking.
* Payment: Customers are responsible for making payments for the booked services through the available payment methods provided by the app. They should ensure the accuracy and security of their payment details.
* Communication: Customers may need to communicate with service providers or the app's customer support team for inquiries, changes, or cancellations related to their bookings. They should utilize the provided communication channels appropriately and adhere to any specified guidelines.
* Feedback and Reviews: Customers have the responsibility to provide honest and constructive feedback on the services they have availed. They can submit reviews, ratings, and comments to help other users make informed decisions.

**Admin Role and Responsibilities:**

* System Management: The admin is responsible for overall management and administration of the Medley Forma booking app. This includes monitoring system performance, ensuring data security and privacy, and managing app settings and configurations.
* User Management: The admin is responsible for managing user accounts, including registration, verification, and account moderation. They may handle tasks such as account approvals, suspensions, or terminations when necessary.
* Service Provider Management: The admin oversees the registration, verification, and approval process for service providers who offer services through the app. They ensure the accuracy and quality of the service providers listed on the platform.

**User Flow:**

****

This code represents a basic user flow for the Medley Forma booking app, including sign-up, login, service search, booking, payment, confirmation, viewing booking history, leaving a review, and password reset functionality.

**Sign Up**

* Users can create a new account by providing their email address, choosing a password, and completing the registration form.
* After signing up, users will receive a verification email to confirm their account.

**Log In**

* Registered users can log in to their Medley Forma accounts using their email and password.
* If users forget their password, they can initiate a password reset process.

**Search Services**

* Users can search for various services offered on Medley Forma, such as appointments, classes, events, or reservations.
* The search functionality allows users to filter services based on their preferences, such as location, date, time, or service type.

**Select Service**

* Once users find a service of interest, they can view detailed information about it, including descriptions, pricing, and availability.
* Users can select the desired service from the search results to proceed with the booking process.

**Make Payment**

* Users are prompted to provide payment details and complete the payment process securely.
* The app supports various payment methods, such as credit/debit cards, mobile wallets, or digital payment platforms.

**Booking Confirmation**

* After successful payment, users receive a booking confirmation with all the relevant details, including the service, date, time, and location.
* The confirmation serves as proof of the booked service and provides any additional instructions or requirements.

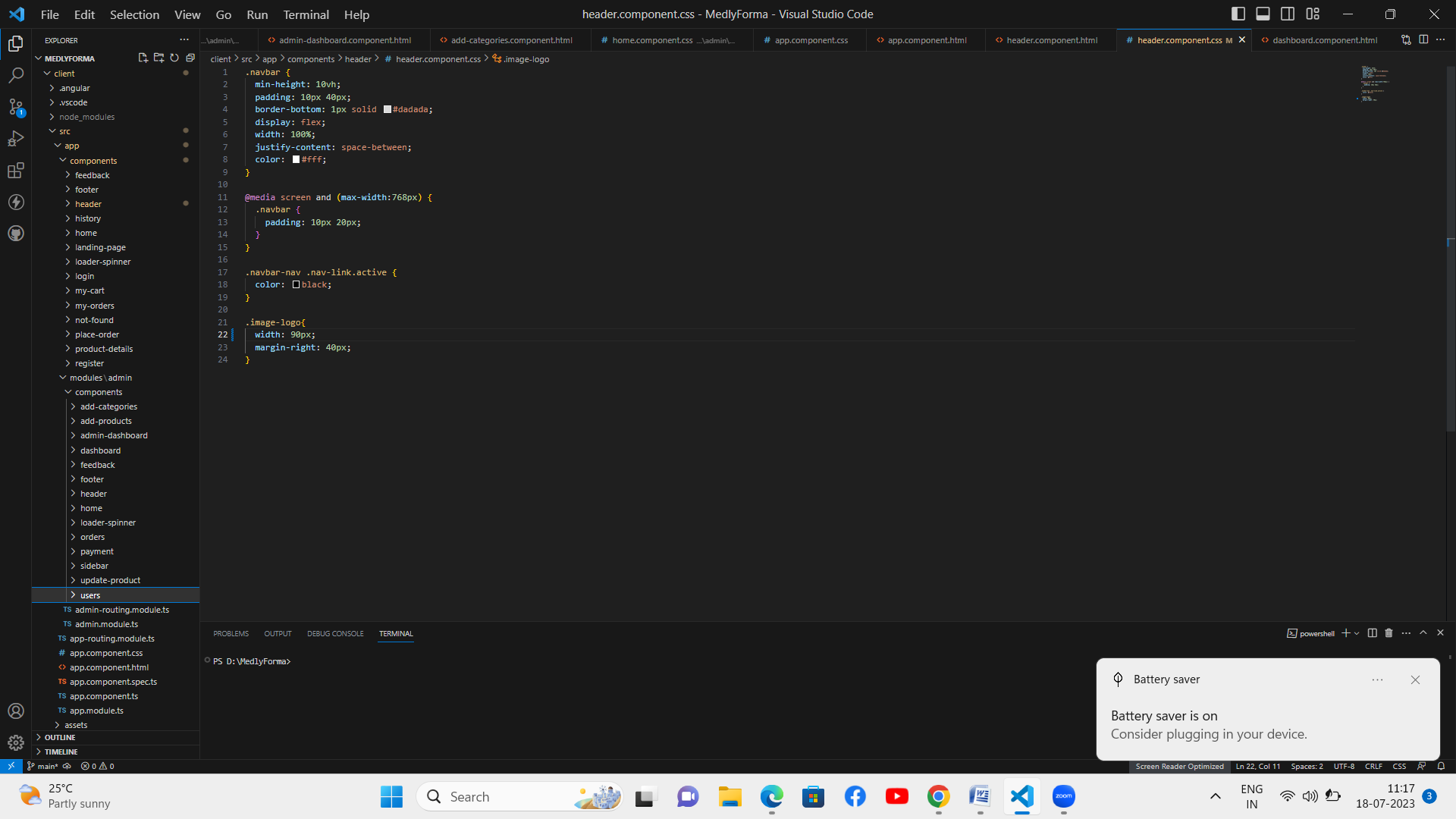
**View Booking History**

* Users can access their booking history to review past and upcoming bookings.
* The booking history displays information such as service details, dates, times, and any associated notes.

**Leave a Review**

* Users have the option to leave reviews and ratings for the services they have availed.
* Reviews help other users make informed decisions and provide valuable feedback to service providers.

**PROJECT STRUCTURE:**



The project structure may vary depending on the specific framework, programming language, or development approach used. It's essential to organize the files and directories in a logical and consistent manner to improve code maintainability and collaboration among developers.

app/app.component.scss, src/app/app.component.spec.ts: These files are part of the main AppComponent, which serves as the root component for the Angular app. The component handles the overall layout and includes the router outlet for loading different components based on the current route.

**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:**

* Client folders.
* Server 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.

**Milestone 3: Web Development:**

**1. Setup Angular Application:**

* Create Angular application using angular CLI.
* Configure Routing.
* Install required libraries.

**2. Design UI components:**

* Create Components.
* Implement layout and styling.
* Add navigation.

**3. Implement frontend logic:**

* Integration with API endpoints.
* Implement data binding.

**Create database in cloud video link:-** <https://drive.google.com/file/d/1CQil5KzGnPvkVOPWTLP0h-Bu2bXhq7A3/view?usp=sharing>

**To Setup the frontend development and to connect node.js with MongoDB Database Go through this video link: -**

<https://drive.google.com/file/d/1b5bMvnqmASXLnSZ74B2t3EzNjuWHj63g/view?usp=drive_link>

**Backend:**

The backend of the Medley Forma booking app would typically involve the server-side components responsible for processing user requests, managing data, and facilitating communication between the client-side (the app or website) and the server. Here are some aspects typically involved in the backend development:

Server and Hosting: Set up a server environment and select a suitable hosting provider to ensure the app is accessible and performs optimally.

API Development: Design and implement APIs (Application Programming Interfaces) to handle various functionalities such as user authentication, service search, booking creation, payment processing, and more.

Database Management: Choose an appropriate database system (e.g., MySQL, PostgreSQL, MongoDB) to store and manage user data, service information, booking records, and other relevant data.

User Authentication: Implement secure user authentication and authorization mechanisms, including features like registration, login, password hashing, and session management.

Service Management: Develop backend logic to manage services, including service creation, availability management, service provider details, and any additional attributes associated with services.

Booking Management: Build functionality to handle booking creation, modification, and cancellation. Store and retrieve booking details from the database, handle availability checks, and send relevant notifications to users and service providers.

Payment Integration: Integrate with payment gateways or APIs to facilitate secure and seamless online payment processing. Handle payment verification and transaction management.

Notifications and Messaging: Implement a system to send notifications and reminders to users and service providers regarding booking updates, confirmations, cancellations, or any other relevant information.

Analytics and Reporting: Incorporate tools and techniques to gather and analyze data related to app usage, bookings, user behavior, and other metrics. Generate reports to gain insights and improve the app's performance and user experience.

Security and Data Protection: Ensure robust security measures are in place to protect user data, such as data encryption, secure APIs, and adherence to relevant data protection regulations.

Scalability and Performance Optimization: Design the backend architecture to handle increasing user traffic and data volume. Employ caching mechanisms, database optimization techniques, and load balancing strategies to enhance app performance and scalability.

The specific technologies and frameworks used for the backend development may vary depending on the development team's preferences, programming languages, and the overall architecture of the Medley Forma booking app.