**MedicalZone Primary Requirements:**

**1. \*\*User Registration and Login:\*\***

- Users should be able to create accounts and log in.

- Differentiate between patient and doctor accounts.

**2. \*\*Hospital Search:\*\***

- Implement a search feature for hospitals.

- Display a list of hospitals with details such as location and available doctors.

**3. \*\*Doctor Search:\*\***

- Allow users to search for doctors by name.

- Enable searching for doctors based on the hospital they work in.

**4. \*\*Disease-based Search:\*\***

- Implement a search option where users can search for doctors based on the disease they specialize in treating.

**5. \*\*Appointment Booking:\*\***

- Patients should be able to schedule appointments with doctors.

- Doctors should be able to view their appointment schedules.

**6. \*\*Chamber Selection:\*\***

- Patients should choose a specific chamber when booking an appointment.

**7. \*\*Dashboard for Doctors and Patients:\*\***

- Doctors and patients should have individual dashboards displaying their appointments, profiles, and relevant information.

**8. \*\*Front-End Development:\*\***

- Use React JS for the front end.

- Design an intuitive and user-friendly interface.

**9. \*\*Database:\*\***

- Utilize an SQL database to store user profiles, appointment details, hospital information, and other relevant data.

**System Design:**

**1. \*\*Database Schema:\*\***

- Tables: Users (doctors and patients), Appointments, Doctors, Hospitals, Chambers, Diseases.

**2. \*\*Backend (Django):\*\***

- Create Django models for each table in the database schema.

- Implement views and controllers for user registration, login, hospital and doctor search, appointment scheduling, and chamber selection.

**3. \*\*Frontend (React JS):\*\***

- Develop components for user registration, login, hospital and doctor search, appointment scheduling, and dashboards.

- Implement a responsive and intuitive user interface.

**4. \*\*Authentication:\*\***

- Implement secure authentication for user accounts.

**5. \*\*APIs:\*\***

- Develop RESTful APIs for communication between the front end and back end.

**Environment Setup:**

**1. \*\*Django Backend:\*\***

- Install Django using `pip install Django`.

- Set up a new Django project and app.

- Define models, views, and controllers.

- Configure Django settings, including database connection.

**2. \*\*React Frontend:\*\***

- Install Node.js and npm.

- Create a new React app using `npx create-react-app`.

- Set up React components and routes.

**3. \*\*Database:\*\***

- Choose an SQL database (e.g., PostgreSQL, MySQL).

- Configure the database connection in Django settings.

**4. \*\*API Development:\*\***

- Use Django REST framework to create APIs for user authentication, hospital and doctor search, appointment scheduling, etc.

**5. \*\*Integration:\*\***

- Integrate the frontend and backend by making API calls from React components.

**6. \*\*Testing:\*\***

- Test the functionality of user registration, login, hospital and doctor search, and appointment scheduling.

**7. \*\*Deployment:\*\***

- Deploy the Django backend and React frontend on a hosting service (e.g., Heroku, AWS).

**Android App (Future Development):**

**1. \*\*API Integration:\*\***

- Develop an Android app using Java or Kotlin.

- Integrate the app with the existing APIs for user authentication, hospital and doctor search, and appointment scheduling.

**2. \*\*User Interface:\*\***

- Design an intuitive and mobile-friendly user interface for the Android app.

**3. \*\*Testing:\*\***

- Test the app's functionality on different Android devices.

**4. \*\*Deployment:\*\***

- Publish the app on the Google Play Store.