Cse299 project

Here's a list of pages for the Mental Health Counseling Center Web Application:

1. **Home Page**
2. **Login Page**
3. **Registration Page**
4. **Patient Dashboard**
5. **Counselor Dashboard**
6. **Admin Dashboard**
7. **Patient Profile Page**
8. **Counselor Profile Page**
9. **Admin Profile Page**
10. **Appointment Request Page**
11. **Appointment Management Page (Counselor)**
12. **Appointment Approval Page (Admin)**
13. **Visit History Page (Patient)**
14. **Patient Records Page (Counselor)**
15. **Payment History Page (Patient)**
16. **Payment Management Page (Admin)**
17. **Counselor Availability Page**
18. **User Management Page (Admin)**

1. Project Structure and Environment Setup

- **Frontend (React.js):**
  - Set up a React project with Next.js or Create React App. Use react-router-dom for navigation.
  - Organize components by user roles: /components/patients, /components/counselors, /components/admin.
  - Use CSS-in-JS libraries like styled-components or Tailwind CSS for responsive and modular design.
- **Backend (Node.js with Express or PHP Laravel):**

    - Set up Express for a REST API structure or Laravel for MVC-based routing.
    - Structure folders by functionality: /controllers, /models, /routes, /middleware.
    - Ensure consistent API response formats for handling frontend integration smoothly.
- **Database (Cloud Firestore or MySQL):**
    - **Cloud Firestore:** Ideal for fast data access without strict schemas. Suitable for patient records and appointment management.
    - **MySQL:** Opt for a relational structure if you have complex relationships (e.g., visit records and counselor-patient interactions).
- **Authentication (JWT and OAuth):**
    - Implement JWT for handling secure sessions. Consider OAuth if you want to allow users to sign in with social accounts (e.g., Google).

2. User Management and Authentication

- **Registration and Login:**
    - Create custom registration forms based on user role (patients, counselors, administrators).
    - For password security, use bcrypt for hashing (Node.js) or Laravel's native hashing.
- **Role-Based Access Control (RBAC):**
    - Set up middleware to restrict access to specific features based on roles (counselor, patient, admin).
    - Store user roles in JWT tokens, and implement role-checking middleware to protect routes.

3. Patient Records and Visit History

- **Database Schema:**
    - **Patients Table:** Store patient profile info, unique patient ID, contact details.
    - **Visits Table:** Each record should include visit date, counselor ID, session notes, mental health status updates, and next appointment.
- **Data Access:**
    - Allow counselors to add session notes to patient records.
    - Patients should have read-only access to their own visit history.

4. Appointment Scheduling and Management

- **Patients:**

    - Implement a calendar-based scheduling system where patients can view counselor availability.
    - Integrate a feature to book appointments and show pending requests.
- **Counselors:**
    - Create a dashboard where counselors can set availability slots, confirm or cancel appointments, and view their schedules.
- **Administrators:**
    - Build tools to approve or deny appointment requests, assign patients to counselors, and reassign counselors as needed.

5. Payment Tracking System

- **Database Structure:**
    - **Payments Table:** Track payment history with fields for patient ID, payment date, amount, and payment status.
- **Patient View:**
    - Display payment history, pending fees, and allow patients to make payments if needed.
- **Admin Tools:**
    - Enable admins to update payment status and view all patient payment histories.

6. Admin Dashboard

- **Overview Dashboard:**
    - Display summaries of appointments, user registrations, and outstanding payments.
- **Appointment and User Management:**
    - Provide a table view for quick access to user profiles, including tools to activate/deactivate accounts.
- **Reports and Analytics:**
    - Track KPIs, such as the total number of appointments per counselor, average wait times, and financial metrics.

7. Frontend Development

- **Component Layout:**
    - Use reusable components such as AppointmentForm, PatientProfile, and AdminDashboard.
- **State Management:**

- Use Context API or Redux to manage global state (e.g., authentication, appointments).

- **UI/UX Optimization:**
    - Design with accessibility in mind, using ARIA labels, high-contrast themes, and responsive design.
    - Ensure intuitive navigation, especially for first-time users seeking mental health support.

8. Testing and Quality Assurance

- **Unit Testing:**
    - Use Jest and React Testing Library to write tests for key frontend components.
    - Use Mocha and Chai (Node.js) or PHPUnit (Laravel) to test backend logic, ensuring each feature behaves as expected.
- **Integration Testing:**
    - Test flows like booking appointments, adding patient records, and tracking payments end-to-end.
- **User Testing:**
    - Run usability tests with sample users to identify UX issues and improve based on feedback.

9. Deployment and Monitoring

- **Frontend Deployment:**
    - Deploy on Vercel, Netlify, or a similar service for efficient CI/CD integration.
- **Backend and Database Hosting:**
    - Choose a cloud provider (AWS, GCP, or Azure) for backend deployment, with load balancers to handle scaling.
    - Use managed services like AWS RDS for MySQL or Firebase for Firestore.
- **Monitoring:**
    - Set up logging with a tool like LogRocket (frontend) or Winston (backend).
    - Integrate error reporting via Sentry or a similar platform for tracking runtime errors and crashes.

10. Future Features and Scaling Considerations

- **Real-Time Communication:**
    - Add chat or messaging for counselors and patients using WebSockets or Firebase real-time database.

- **Appointment Reminders:**
    - Set up automated email or SMS reminders using a third-party service like Twilio.
- **AI-Based Recommendations:**
    - Implement AI models to suggest mental health resources based on patient history.