✅ 1. Start with a Quick Summary:

> “We’re building a web application called **CareConnect**, a multi-role healthcare support platform for users (patients), caretakers, and doctors. It includes role-based dashboards, real-time chat, schedule/task handling, entertainment options, and doctor-patient-caretaker coordination. The app supports OTP-based login/registration, language preference, and offline functionality.”

✅ 2. Define the User Roles:

List the 3 main types of users:

- **User** (Care Seeker)
- **Caretaker** (Care Provider)
- **Doctor**

Each has different flows and dashboards.

✅ 3. Explain the Major Features / Modules:

A. Pre-Login Home Screen:

- Rotating messages (every 5 seconds)
- Language support (auto-detect via browser/device settings)
- Buttons:
  - “Sign In”
  - “Create Account”
- Extras: Help, contact info, language dropdown
- Guided tour for new users

- Offline mode (uses cached data)
- Auto-resume previous session if logged out

B. Registration Flow (Dynamic by Role):

Common Fields: Name, mobile, email, password (strong validation), security Q&A, OTP verification (user & emergency contact).

Users Only:

- Gender, age, location (hierarchical dropdowns), preferred languages, disabilities
- Ask: “Do you want a doctor?” → Yes/No
- Emergency contact (mandatory)
- Optional: Add relatives (requires caretaker approval)

Caretakers (in addition to user fields):

- Salary, work timing, experience, certifications (upload PDF/image)

Doctors:

- Name, mobile, email, password, specialty, clinic address, certifications, experience

→ After OTP, assign unique ID like `USR-1234567890` and redirect to login.

C. Login:

- By mobile/email/ID + password
- Face/fingerprint login (optional if browser supports WebAuthn)
- Forgot password: via OTP

- 5 wrong attempts = temporary lockout

D. Post-Login Dashboards:

1. User Dashboard:

- If no caretaker → emergency contact is default
- If hiring caretaker → see grid/list of caretakers with filters
- Features:
    - Chat/call (with caretaker, doctor)
    - Calendar (tasks from doctor, updated by caretaker)
    - Emergency button
    - Entertainment (approval system)
    - Edit care needs (with caretaker approval)
    - Add relatives (pending approval)

2. Caretaker Dashboard:

- See user requests
- Accept/reject pairing (only 1 paid user at a time)
- Schedule management (upload/update doctor reports)
- Confirm or deny user requests (e.g., entertainment, doctor chat)
- Chat/call with user and doctor
- Log missed sessions with reasons
- End/extend arrangements

3. Doctor Dashboard:

- Grouped patients
- Send medical reports (food, medicine, exercise) to caretakers
- Update records
- Schedule appointments
- Chat/call with user and caretaker
- Moderate disability inputs
- Resolve disputes
- Broadcast messages
- Download logs, analytics

E. Unpairing / Ending Arrangement:

- Triggers: User, caretaker, or contract expiry
- Notifications sent to all parties
- Option to rehire or select new caretaker
- Feedback requested
- Special handling for unresponsive renewal or disputes

✅ 4. Technical Features to Emphasize:

- **Responsive design** (mobile-first, grid-based)
- **Role-based access** and dashboards
- **OTP verification (email-based)** for all users

- **Offline support** (Service Workers, localStorage)
- **Language localization** (i18n)
- **Secure file upload** (PDF/image, max 5MB)
- **Real-time communication** (Socket.io or WebRTC for calls/chats)
- **Filtered search system** (caretakers, doctors)
- **Scalable backend** (Node.js + MongoDB recommended)
- **Role-based data linking** (user-caretaker-doctor relationships)
- **Security**: Strong password rules, 2FA for doctor login, session timeouts
- **Analytics/Logs dashboard** for doctors

✅ 5. Suggest a Tech Stack (if needed):

We suggest a **MERN stack**:

- **Frontend**: React + Tailwind CSS
- **Backend**: Node.js + Express
- **Database**: MongoDB (with role-based collections)
- **Authentication**: JWT + email OTP
- **Storage**: Cloudinary / Firebase for uploaded docs
- **Localization**: i18n-next
- **Push Notifications**: Firebase or custom server