### **Roles and Responsibilities:**

### **1. Admin (Superior Role)**

**Primary Responsibilities:**

* **Account Creation and Management:**
  + **Create accounts** for patients, nurses, and caretakers.
  + **Approve or reject** nurse and caretaker registration based on their details (e.g., qualifications, experience).
  + Assign **patients to a nurse** and a **caretaker** (one patient can have only one nurse and one caretaker).
* **Assign Patients to Caretakers and Nurses:**
  + After registering users and reviewing profiles, the admin will assign patients to a specific nurse and caretaker.
* **System-Wide Management:**
  + View and generate reports (e.g., task completion rates, patient health reports).
  + Manage high-level configurations of the system, like access control and task permissions.
* **Full Access to Health Records and Tasks:**
  + View and update health records.
  + Admin can perform CRUD (Create, Read, Update, Delete) operations on all entities (tasks, care plans, patient records).

**Data Management:**

* Admin can see all patients' records, health history, and care plans.
* CRUD access to all health records, task data, and care plans.

### **2. Nurse (Mid-Level Role, Focus on Medical Oversight)**

**Primary Responsibilities:**

* **Patient Data Management:**
  + **Manage health records** for assigned patients.
  + **Update vital signs**, and **medical records** (limited to medical data, such as blood pressure, heart rate, etc.).
  + Upload or manage **care plans** (detailed in the next section).
* **CRUD Operations for Tasks:**
  + Nurses can create, edit, update, and delete **tasks** assigned to **caretakers**.
  + Nurses oversee **task completion**, tracking what the caretaker has completed and what's still pending.
* **Interaction with Patients:**
  + **Chat** with patients for communication related to medical care or task clarification.
  + Nurses provide **medical guidance** and communicate the status of treatment.
* **Limited Health Record Management:**
  + Nurses can **view** and **update health records**, but with fewer permissions than an admin. For example, they might not be able to modify certain locked fields like core diagnoses or irreversible medical data.
* **Daily Reporting:**
  + Nurses receive daily reports on patient health and can respond with new tasks or care instructions.
* **Care Plan Management:**
  + Nurses upload and manage **care plans** based on the patient’s condition.
  + The care plan includes treatment instructions, daily routines (feeding, bathing), medical checkups, and reminders.

### **3. Caretaker (Entry-Level Role, Focus on Daily Care)**

**Primary Responsibilities:**

* **Task Management:**
  + **Complete tasks** assigned by the nurse (e.g., bathing, feeding, assisting with mobility).
  + **Mark tasks** as "done" or "in progress," similar to **Trello-like task boards**.
* **Vital Sign Recording:**
  + Caretakers can **record vital signs** (e.g., temperature, heart rate) and update them in the system.
  + The system will record who added the vital signs for traceability.
* **Daily Report Submission:**
  + After completing tasks, caretakers can submit daily reports summarizing their activities and the patient's condition (e.g., "Patient ate all meals, slept well").
* **Limited Health Record Access:**
  + Caretakers can only view and update **non-medical parts** of the patient’s record, such as their activity level, feeding habits, etc.
  + Caretakers **cannot modify** core medical records like diagnoses or prescriptions.

### **4. Patient (Limited Role, Indirect Interaction)**

**Primary Role:**The patient will have **limited direct interaction** with the system. Most interactions will happen through nurses or caretakers.

* **Communication with Nurses:**Patients can chat with nurses via the app for health updates, instructions, or questions.
* **Patient Health Records and Care Plans:**Nurses and caretakers will update and manage the health records, care plans, and task lists on behalf of the patient.

### **Care Plan – What It Is and Its Components:**

* **Definition:**A **care plan** is a personalized document that outlines a patient’s daily and medical care routine. It is primarily created by the **nurse** and may be updated as the patient’s condition changes.
* **Example of Care Plan Components:**
  1. **Medical Instructions:**
     + Specific treatments or medications to be given.
     + Health measurements like blood sugar checks or blood pressure monitoring.
  2. **Daily Routine:**
     + Bathing, feeding, mobility assistance, toileting, etc.
  3. **Scheduled Checkups:**
     + Any scheduled doctor or therapy appointments.
  4. **Physical Exercises:**
     + Specific exercises or therapy routines for the patient, such as stretching or walking.
  5. **Dietary Requirements:**
     + Any special dietary needs, restrictions, or schedules for meals.
* Any special dietary needs, restrictions, or schedules for meals.

### **Backend Architecture – Core Entities and Relations:**

#### **1. Admin**

* **Attributes:**
  + id
  + name
  + email
  + password
* **Responsibilities:**
  + Can create/manage users (nurses, caretakers, patients).
  + Admins are responsible for approving/rejecting caretaker and nurse profiles upon their signup.
  + Assign patients to nurses and caretakers.
  + Admins have CRUD access to all patients, tasks, health records, and care plans.

#### **2. Nurse**

* **Attributes:**
  + id
  + name
  + email
  + password
  + assignedPatients: List of Patients (array of patient IDs) under the nurse's care.
* **Responsibilities**:  
  + Nurses manage patient records, vital signs, and care plans.
  + They perform CRUD operations on tasks for their assigned caretakers.
  + They can generate reports and chat with patients.
* **Relationships:**
  + **Many-to-One with Admin:** The nurse’s profile is created and approved by the admin.
  + **One-to-Many with Patients:** One nurse is assigned to multiple patients.
  + **One-to-Many with Tasks:** Nurses create and manage tasks for caretakers.

#### **3. Caretaker**

* **Attributes:**
  + id
  + name
  + email
  + password
  + assignedPatients: List of Patients under the caretaker's care (usually one at a time).
* **Responsibilities**:  
  + Caretakers execute tasks assigned by the nurse and mark tasks as completed or in progress.
  + They record vital signs and other non-medical data.
  + Caretakers submit daily reports on patient care.
* **Relationships:**
  + **Many-to-One with Admin:** Caretaker's profile is created and approved by the admin.
  + **One-to-One with Patients:** One caretaker can take care of one patient at a time.
  + **One-to-Many with Tasks:** Caretakers can mark tasks as complete or in progress.

#### **4. Patient**

* **Attributes:**
  + id
  + name
  + age
  + medicalHistory
  + carePlan: List of care plan items
* **Responsibilities**:  
  + Patients don't interact with the app directly.
  + Nurses and caretakers will update the patient's records and care plan.
* **Relationships:**
  + **One-to-One with Nurse:** A patient is assigned to one nurse.
  + **One-to-One with Caretaker:** A patient is assigned to one caretaker.
  + **One-to-Many with Health Records:** The patient has multiple health records (created by nurses or caretakers).

#### **5. Task**

* **Attributes:**
  + id
  + title
  + description
  + assignedTo: Either a nurse or caretaker ID.
  + status: (e.g., Done, In Progress, Pending)
  + dueDate: Deadline for the task (optional).
* **Responsibilities:**
  + Tasks are created by the nurse and assigned to caretakers.
  + Caretakers mark the task's status and update the nurse on progress.
* **Relationships:**
  + **Many-to-One with Patient:** Tasks are associated with one patient.
  + **Many-to-One with Caretaker:** Tasks are executed by a caretaker.
  + **Many-to-One with Nurse:** Tasks are created and managed by nurses.

#### **6. Health Record**

* **Attributes:**
  + Id
  + patientId: The patient to whom the health record belongs.
  + vitalSigns: Blood pressure, heart rate, etc.
  + recordedBy: Nurse or Caretaker
  + timestamp
* **Relationships:**
  + **Many-to-One with Patient:** Each health record is tied to a patient.
  + **Many-to-One with Nurse/Caretaker:** Health records are created by either a nurse or caretaker.
  + A log of who entered the data is maintained.

**7. Care Plan**

* **Attributes:**
  + id: Unique identifier for the care plan.
  + patientId: The patient for whom the care plan was created.
  + createdBy: The nurse who created the care plan.
  + tasks: Array of daily/weekly routines (related to the task model).
* **Responsibilities:**
  + Nurses create care plans based on the patient's medical condition.
  + The care plan outlines daily tasks for the caretaker, with specific instructions for each task.

### **Complete API Routes (Node.js)**

Now, let’s look at the key API routes that will be implemented to support the roles and operations mentioned above. We will be using **RESTful APIs** with JSON responses for data communication.

### **1. Authentication & User Management Routes:**

1. **Authentication (Common for All Roles):**

POST /auth/login:  
**Description:** Login for all users (admin, nurse, caretaker).  
**Request Body:**json  
Copy code  
{

"email": "user@example.com",

"password": "password123"

}

* + **Response:**
    - Success: Returns a **JWT** token and user details based on their role.

json  
Copy code  
{

"token": "jwt\_token\_here",

"user": {

"id": "userId",

"role": "admin/nurse/caretaker",

"name": "John Doe",

"email": "user@example.com"

}

}

* + - Failure: Returns an error if login credentials are invalid.

POST /auth/register:  
**Description:** Register a user (caretaker or nurse), pending admin approval.  
**Request Body:**json  
Copy code  
{

"role": "nurse/caretaker",

"name": "John Doe",

"email": "user@example.com",

"password": "password123",

"otherDetails": "Specific details for nurse or caretaker"

}

* + **Response:**
    - Success: Returns the new user's details.
    - Failure: Returns validation errors (e.g., if email is already taken).
  + POST /auth/logout:  
    **Description:** Invalidate the user's token (optional based on session management strategy).
  + GET /auth/me:  
    **Description:** Get the current authenticated user’s details. Useful for checking role-based access or user info.  
    **Response:**
    - Success: Returns the currently logged-in user's details.

json  
Copy code  
{

"id": "userId",

"role": "admin/nurse/caretaker",

"name": "John Doe",

"email": "user@example.com"

}

### **2. Admin Routes:**

POST /admin/register:  
**Description:** Create a new admin account (for super admin to create other admin accounts).  
**Request Body:**json  
Copy code  
{

"name": "Admin Name",

"email": "admin@example.com",

"password": "admin\_password"

}

* GET /admin/nurses:  
  **Description:** Retrieve a list of all registered nurses.
* GET /admin/caretakers:  
  **Description:** Retrieve a list of all registered caretakers.
* POST /admin/approve-nurse/:nurseId:  
  **Description:** Approve a nurse's registration.  
  **Params:** nurseId - ID of the nurse to approve.
* POST /admin/approve-caretaker/:caretakerId:  
  **Description:** Approve a caretaker's registration.  
  **Params:** caretakerId - ID of the caretaker to approve.

POST /admin/assign-patient:  
**Description:** Assign a patient to a specific nurse and caretaker.  
**Request Body:**json  
Copy code  
{

"patientId": "patient\_id\_here",

"nurseId": "nurse\_id\_here",

"caretakerId": "caretaker\_id\_here"

}

### **3. Nurse Routes:**

* POST /nurse/register:  
  **Description:** Nurse registration (requires admin approval).  
  **Request Body:** Similar to the auth/register endpoint.
* GET /nurse/patients:  
  **Description:** Retrieve the list of patients assigned to the nurse.

POST /nurse/patients/:patientId/health-record:  
**Description:** Add or update a health record for a specific patient.  
**Request Body:**json  
Copy code  
{

"vitalSigns": {

"bloodPressure": "120/80",

"heartRate": 72,

"temperature": 98.6

},

"notes": "Patient is stable"

}

POST /nurse/tasks:  
**Description:** Create tasks for the assigned caretakers.  
**Request Body:**json  
Copy code  
{

"patientId": "patient\_id\_here",

"caretakerId": "caretaker\_id\_here",

"task": {

"title": "Check patient vitals",

"description": "Record blood pressure and heart rate",

"dueDate": "2024-09-15"

}

}

PUT /nurse/tasks/:taskId:  
**Description:** Update a task’s status or details.  
**Request Body:**json  
Copy code  
{

"status": "In Progress" // or "Done"

}

* GET /nurse/reports:  
  **Description:** Generate a patient report (e.g., task completion rate, health status).  
  **Query Params:** Can include filters like patientId, dateRange, status, etc.

POST /nurse/care-plan/:patientId:  
**Description:** Create or update a care plan for a patient.  
**Request Body:**json  
Copy code  
{

"carePlan": [

{

"task": "Administer medication",

"frequency": "daily",

"notes": "Give after meals"

},

{

"task": "Monitor blood pressure",

"frequency": "twice daily",

"notes": "Notify if higher than 130/90"

}

]

}

* GET /nurse/chat:  
  **Description:** Access patient communication system (Chat with assigned patients).

### **4. Caretaker Routes:**

* POST /caretaker/register:  
  **Description:** Caretaker registration (requires admin approval).
* GET /caretaker/patients:  
  **Description:** Retrieve the patient assigned to the caretaker.

POST /caretaker/tasks/:taskId/complete:  
**Description:** Mark a task as complete or in progress.  
**Request Body:**json  
Copy code  
{

"status": "Done", // or "In Progress"

"notes": "Task completed successfully"

}

POST /caretaker/patients/:patientId/health-record:  
**Description:** Record basic health data for an assigned patient (vital signs).  
**Request Body:**json  
Copy code  
{

"vitalSigns": {

"bloodPressure": "115/75",

"heartRate": 70,

"temperature": 98.4

}

}

POST /caretaker/daily-report/:patientId:  
**Description:** Submit a daily report on patient care and condition.  
**Request Body:**json  
Copy code  
{

"report": "Patient had a good day, ate meals well and vital signs are stable."

}

### **5. Patient Routes:**

* GET /patients/:patientId:  
  **Description:** Retrieve patient details and medical history.

PUT /patients/:patientId:  
**Description:** Update patient information (only accessible by nurses or admins).  
**Request Body:**json  
Copy code  
{

"name": "John Doe",

"age": 65,

"medicalHistory": {

"diabetes": true,

"hypertension": false

}

}

* GET /patients/:patientId/health-records:  
  **Description:** Get all health records for the patient.
* GET /patients/:patientId/care-plan:  
  **Description:** Retrieve the patient’s care plan.

### **6. Task Routes:**

POST /tasks:  
**Description:** Create a new task (typically by a nurse or admin).  
**Request Body:**json  
Copy code  
{

"title": "Check patient vitals",

"description": "Record blood pressure and heart rate every morning",

"assignedTo": "caretaker\_id\_here",

"patientId": "patient\_id\_here",

"dueDate": "2024-09-15"

}

* GET /tasks:  
  **Description:** Retrieve all tasks for the logged-in user (filtered by role). Admins, nurses, and caretakers will only see their respective tasks.
* GET /tasks/:taskId:  
  **Description:** Get details of a specific task.

PUT /tasks/:taskId:  
**Description:** Update task status or content.  
**Request Body:**json  
Copy code  
{

"status": "In Progress", // or "Done"

"description": "Updated task details here"

}

* DELETE /tasks/:taskId:  
  **Description:** Delete a task (typically by nurses or admins).

### **7. Wi-Fi CSI Data Routes:**

POST /patients/:patientId/csi-data:  
**Description:** Upload Wi-Fi CSI data for a specific patient for analysis.  
**Request Body:**json  
Copy code  
{

"amplitude": 5.0,

"phase": 1.3,

"subcarrierIndex": 2

}

* GET /patients/:patientId/csi-data:  
  **Description:** Retrieve Wi-Fi CSI data for analysis (filtered by date, subcarrier index, etc.).

### **8. Report Generation:**

1. GET /reports:  
   **Description:** Generate various reports (e.g., task completion, patient health, daily summaries).  
   **Query Params:**
   * patientId
   * nurseId
   * caretakerId
   * dateRange
   * status
2. **Admin Routes:**
   * POST /admin/auth/login: Login to web dashbaord
   * GET /admin/nurses: List all registered nurses.
   * GET /admin/caretakers: List all registered caretakers.
   * POST /admin/approve-nurse/:nurseId: Approve a nurse's registration.
   * POST /admin/approve-caretaker/:caretakerId: Approve a caretaker's registration.
   * POST /admin/assign-patient: Assign a patient to a specific nurse and caretaker.
3. **Nurse Routes:**
   * POST /nurse/register: Nurse registration (admin approval required).
   * GET /nurse/patients: Retrieve the list of patients assigned to the nurse.
   * POST /nurse/patients/:patientId/health-record: Add or update a health record for a specific patient.
   * POST /nurse/tasks: Create tasks for the assigned caretakers.
   * PUT /nurse/tasks/:taskId: Update a task’s status or details.
   * GET /nurse/reports: Generate a patient report (e.g., task completion rate, health status).
   * POST /nurse/care-plan/:patientId: Create or update a care plan for a patient.
   * GET /nurse/chat: Access patient communication system (Chat with assigned patients).
4. **Caretaker Routes:**
   * POST /caretaker/register: Caretaker registration (admin approval required).
   * GET /caretaker/patients: Retrieve the patient assigned to the caretaker.
   * POST /caretaker/tasks/:taskId/complete: Mark a task as complete or in progress.
   * POST /caretaker/patients/:patientId/health-record: Record basic health data for an assigned patient (vital signs).
   * POST /caretaker/daily-report/:patientId: Submit a daily report on patient care and condition.
5. **Patient Routes (for viewing/editing by nurse/caretaker/admin):**
   * GET /patients/:patientId: Retrieve patient details and medical history.
   * PUT /patients/:patientId: Update patient information (only nurses or admin).
   * GET /patients/:patientId/health-records: Get all health records for the patient.
   * GET /patients/:patientId/care-plan: Retrieve the patient’s care plan.
6. **Task Routes:**
   * POST /tasks: Create a new task (usually by a nurse).
   * GET /tasks: Retrieve all tasks for the logged-in user (filtered by role).
   * GET /tasks/:taskId: Get details of a specific task.
   * PUT /tasks/:taskId: Update task status or content.
   * DELETE /tasks/:taskId: Delete a task.

### **Feedback/Support System for Users:**

### **Caretakers or Nurses may need a way to request help or report issues directly within the app (for instance, if they encounter technical problems with the system or have care-related questions).**

### **POST /support/ticket: Description: Allows nurses or caretakers to submit a support ticket for technical help or inquiries. Request Body: json Copy code {**

### **"userId": "nurse\_id\_here",**

### **"issue": "Unable to submit health record",**

### **"details": "I've been facing an issue where the health record submission fails."**

### **}**

### 

### **GET /support/tickets: Description: Retrieve a list of support tickets (admin view) to address incoming issues or questions from users.**

### 

### **Authentication and Role-Based Access Control (RBAC):**

The system will use **JWT (JSON Web Tokens)** for authentication and **role-based access control (RBAC)**. Each user (admin, nurse, caretaker) will have a token that grants them access to specific API endpoints based on their role.

* **JWT Strategy:**
  + Upon login, a JWT token is generated and sent to the user.
  + Each request to a protected route (e.g., tasks, health records) will require this token.
  + The token will contain information about the user’s role (admin, nurse, caretaker) and scope of access.
* **Access Control:**
  + Middleware will ensure that users are only accessing routes that align with their role. For example:
  + **Admin** can access all routes.
  + **Nurses** can only access routes related to patients assigned to them.
  + **Caretakers** can only see tasks and patient data for the patients they’re assigned to.

### **Backend Flow – CRUD Operations**

1. **Admin Workflow:**
   * **CRUD Operations:** Admin has full access to CRUD operations for users (nurses, caretakers), patients, tasks, and health records.
   * **Approval Process:** Admin reviews nurse/caretaker signup requests and either approves or rejects them.
   * Admin can later on disable or delete the profile of users.
   * Only admin can add patients
   * Admin can assign a nurse and a caretaker to a patient.
   * For more nurses and caretakers, patient will have to buy premium subscription.
2. **Nurse Workflow:**
   * **CRUD Operations:** Nurses can create tasks for caretakers, update patient health records, and upload care plans.
   * **Task Management:** Nurses track task completion, assign new tasks, or adjust care plans.
   * **Limited Updates to Health Records:** Nurses can update medical records but with fewer privileges than the admin.
3. **Caretaker Workflow:**
   * **Task Completion:** Caretakers can mark tasks as "In Progress" or "Done".
   * **Vital Signs Recording:** Caretakers can record basic health data (e.g., temperature, heart rate), but the system tracks who entered the information.
   * **View Patient Care Plan:** Caretakers follow the care plan provided by the nurse and execute the tasks.