



ADDIS ABABA UNIVERSITY
COLLEGE OF TECHNOLOGY AND
BUILT ENVIRONMENT
SCHOOL OF INFORMATION TECHNOLOGY
AND SOFTWARE ENGINEERING
TITLE: HEALTH NET

SOFTWARE REQUIREMENTS SPECIFICATION DOCUMENT

SECTION: 1

GROUP: 4

Name	ID No
1. Asanti Oluma	UGR/8165/16
2. Bekalu Addisu	UGR/9538/16
3. Fita Alemayehu	UGR/7071/16
4. Lemi Gobena	UGR/1589/16
5. Martha Tegegne	UGR/4457/16
6. Misganaw Habtamu	UGR/1707/16
7. Olit Oljira	UGR/8925/16
8. Selamawit Mulat	UGR/1033/16

Advisors: **Mrs. Nuniyat Kifle**

Date : **November 14 ,2025**

Table of Contents

Contents	Page
Document Approval	I
List of Tables	II
List of figures	II
Definitions, Acronyms, and Abbreviations	III
DECLARATION	IV
1. Introduction	1
1.1 Purpose	1
1.2 Scope	1
1.3 Overview	2
2. General Description	2
2.1 Product Perspective	2
2.2 Product Functions	3
2.3 User Characteristics	3
2.4 General Constraints	4
3. Specific Requirements	5
3.1 External Interface Requirements	5
3.1.1 User Interfaces	5
3.1.2 Hardware Interfaces	8
3.1.3 Software Interfaces	8
1. Web Browser	8
2. Database Management System	8
3. Backend Server / API	9
3.1.4 Communications Interfaces	9
3.2 Functional requirements	10
3.2.1 Introduction	10
3.3 Use Cases	13
3.4 Non-Functional Requirements	25
3.4.1 Performance	25
3.4.2 Reliability	25
3.4.3 Availability	26
3.4.4 Security	26
3.4.5 Maintainability	26
3.4.6 Portability	26
3.4.7 Usability	27
3.4.8 Scalability	27
3.4.9 Legal and Ethical Compliance	27
3.5 Inverse Requirements	27

3.6 Design Constraints	28
3.7 Logical Database Requirements	28
3.7.1 Logical Requirements	28
3.7.2 Schema Design	29
3.8 Other Requirements	30
4. Change Management Process	31
References	32

Revision History

Date	Description	Author	Comments
13/11/2025	First Version	All Group Members	First Revision

Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

Signature	Printed Name	Title	Date
	Mrs. Nuniat Kifle	FSE Instructor.	

List of Tables

<i>Table 1 : User Characteristics</i>	3
<i>Table 2 : Admin Login</i>	10
<i>Table 3 : Register Patients(mass registration)</i>	10
<i>Table 4 : Register Doctors and Assign Patients</i>	10
<i>Table 5 : Doctor Login</i>	11
<i>Table 6 : Add Diagnosis/ Test Results</i>	11
<i>Table 7 : Patient Login</i>	11
<i>Table 8 : View Medical Records(patients)</i>	11
<i>Table 9 : Insurance Status Record</i>	12
<i>Table 10 : Emergency Tab(patients)</i>	12
<i>Table 11 : QR Code Emergency Access</i>	12
<i>Table 12 : US-00 — User Login</i>	13
<i>Table 13 : US-01 — Register User</i>	14
<i>Table 14 : US-02 — Assign Patient to Doctor</i>	15
<i>Table 15 : US-03 — Manage User Profiles</i>	16
<i>Table 16 : US-04 — View Patient Record</i>	17
<i>Table 17 : US-05 — Update Patient Record</i>	18
<i>Table 18 : US-06 — Upload Lab Result</i>	19
<i>Table 19 : US-07 — Create Appointment</i>	20
<i>Table 20 : US-08 — View Appointments</i>	21
<i>Table 21 : US-09 — View Lab Results and Diagnosis</i>	22
<i>Table 22 : US-10 — QR-Based Record Sharing</i>	23
<i>Table 23 : Database Schema</i>	29

List of figures

<i>Figure 1 : Patient Dashboard</i>	5
<i>Figure 2 : Patient records</i>	6
<i>Figure 3 : Patient Lab records</i>	6
<i>Figure 4 : Patient Profile</i>	7
<i>Figure 5 : Patient Appointments</i>	7
<i>Figure 6 : Admin Usecase</i>	24
<i>Figure 7 : Patient Usecase</i>	24
<i>Figure 8 : Doctor Usecase</i>	25

Definitions, Acronyms, and Abbreviations

Term/Acronym	Definition
SRS	Software Requirements Specification – A document that describes the external behavior of a software system.
EHR	Electronic Health Record – A digital version of a patient's paper chart.
HealthNet	The name of the proposed national EHR system for Ethiopia.
UPI	Unique Patient Identifier – A code assigned to uniquely identify each patient within the HealthNet system.
QR Code	Quick Response Code – A machine-readable code used to store information, in this case, for quick patient identification and emergency access.
RBAC	Role-Based Access Control – A method of regulating access to computer or network resources based on the roles of individual users within an organization.
Admin	Administrator – A user role with the highest level of system access for managing users and assignments.
HTTPS	Hypertext Transfer Protocol Secure – The secure version of HTTP, the protocol over which data is sent between a browser and a website.
API	Application Programming Interface – A set of rules that allows programs to talk to each other.
JWT	JSON Web Token – An open standard for securely transmitting information between parties as a JSON object.
UI	User Interface – The means by which the user and a computer system interact.
SQL	Structured Query Language – A programming language used to manage and manipulate relational databases.

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included. We have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Group No Team Date: 13-11-2025

1. Introduction

1.1 Purpose

This document provides a detailed description of the Software Requirements Specification (SRS) for the *HealthNet* system, Version 1.0. The purpose of this SRS is to define the functional and non-functional requirements for the development of a secure, centralized **National Electronic Health Record (EHR)** system for Ethiopia. This document is intended for the project stakeholders, including the development team, project advisors, healthcare professionals, and the Ethiopian Ministry of Health, to serve as a definitive guide for the system's design, implementation, and validation.

1.2 Scope

The *HealthNet* system is a web-based application designed to create a unified national health information ecosystem. It will connect healthcare facilities—from health posts to specialized hospitals—to provide a single, secure source of patient health information.

In-Scope (Key Deliverables):

- Design of a secure, scalable system architecture for a National EHR.
- Development of a functional software prototype demonstrating:
 - ✓ A centralized patient registry with a Unique Patient Identifier (UPI) and QR code.
 - ✓ Secure web dashboards for Healthcare Providers (Doctors, Lab Technicians, Administrators) to view, update, and manage patient records.
 - ✓ An Emergency Access Module for first responders to retrieve critical patient information via QR code scan without a full login.
 - ✓ Role-Based Access Control (RBAC) to enforce data privacy.
 - ✓ Modules for diagnoses, medications, allergies, lab results, and insurance status.
- A comprehensive project report, including a System Requirements Specification and a strategic roadmap for national implementation.

Out-of-Scope (Exclusions):

- The actual large-scale national roll-out, deployment, and maintenance across all Ethiopian health facilities.
- Procurement and setup of nationwide hardware infrastructure (servers, computers, network hardware).
- The enactment of new national data protection laws or policies (though recommendations will be provided).
- Full-scale integration with all existing vertical health programs (e.g., HIV/TB databases) and private health facilities.

1.3 Overview**The remainder of this SRS is structured as follows:**

- Section 2: General Description – Provides an overview of the product, its functions, users, and constraints.
- Subsequent Sections: Will detail specific requirements, including external interfaces, functional requirements (as partially listed in the provided documents), non-functional requirements (as provided), and other supporting information.

2. General Description**2.1 Product Perspective**

HealthNet is envisioned as a new, self-contained system that will replace and modernize the existing fragmented, paper-based health record-keeping in Ethiopia. It is not a component of a larger system but is designed to become the central national health information infrastructure. It will interface with standard web browsers and leverage existing communication protocols (HTTPS/TCP-IP). The system is designed to be independent of specific medical hardware, though it requires camera access on devices for QR code scanning. Its success is closely tied to national health policy and future integration potential with other digital health initiatives.

2.2 Product Functions

The core functions of the HealthNet system are as follows:

1. **User Authentication and Authorization:** Secure login for Administrators, Doctors, Lab Technicians, and Patients, with access rights determined by RBAC.
2. **Patient Management:** Registration of patients by Administrators, including the generation of a Unique Patient Identifier (UPI) and an associated QR code.
3. **Healthcare Provider Workflow:**
 - ✓ **Doctors:** View assigned patient records, add diagnoses and treatments, create appointments, and upload lab results.
 - ✓ **Lab Technicians:** Upload and manage lab results for patients.
 - ✓ **Administrators:** Manage all user accounts (Doctors, Patients), assign patients to doctors, and update patient insurance status.
4. **Clinical Data Management:** Secure storage and retrieval of patient medical histories, including diagnoses, medications, allergies, and lab results.
5. **Emergency Access:** Provision of immediate, read-only access to a patient's critical health information (allergies, blood type, major conditions) via a scanned QR code, without requiring a login.
6. **Patient Portal:** Allows patients to view their own medical records and lab results, and manage their emergency contact and health information.
7. **Data Security and Auditing:** Ensures all data is encrypted and access to patient records is logged for audit trails.

2.3 User Characteristics

The intended user groups for the HealthNet system are:

Table 1: User Characteristics

User Class	Description	Key Characteristics & Assumptions
Administrator	Manages system users and patient-doctor assignments.	IT-literate, understands organizational structure. Has the highest level of system access.
Doctor	Provides medical care to patients; views and updates patient records.	Medical professional. Computer-literate but may require an intuitive interface. Only accesses patients assigned to them.

Lab Technician	Conducts tests and uploads results to patient records.	Technical and medical professional. Requires ability to upload and link files/data.
Patient	The subject of the health record; views their own information.	Varied computer literacy. Access is read-only for most medical data. Can manage their own emergency information.
First Responder / Emergency Personnel	Accesses critical patient data in emergency situations	Works under high-pressure conditions. Requires instant, simple, read-only access with minimal steps (QR scan).

2.4 General Constraints

1. **Technological:** The system must be developed using open-source technologies (e.g., React.js, Node.js, PostgreSQL) to minimize costs and must be accessible via standard web browsers on both desktop and mobile devices.
2. **Infrastructural:** The design must account for varying levels of internet connectivity across Ethiopia, potentially requiring considerations for offline functionality or data synchronization in low-bandwidth scenarios.
3. **Schedule:** The project has a strict development timeline of approximately 3-4 months for the prototype delivery.
4. **Regulatory:** The system must be designed to comply with the Ethiopian eHealth Strategy 2025 and WHO Digital Health Data Protection Guidelines, even though specific national laws may still be under development.
5. **Security:** The system must enforce stringent data security and patient privacy through encryption, RBAC, and audit logs, operating under the assumption that health data is highly sensitive.
6. **Economic:** The solution must be cost-effective, avoiding proprietary software licenses and expensive hardware requirements to ensure long-term sustainability and potential for national scale-up.

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

Some of the *Figma* designs for the **User Interface(UI)** has been presented as follow.

These pages are selected randomly from the interfaces designed. Some changes might be applied during the implementation process but it has been agreed by group members for now to use these. *These pages are taken out of the patients UI.*

Patient Dashboard: It has some quick access features for usage simplification.

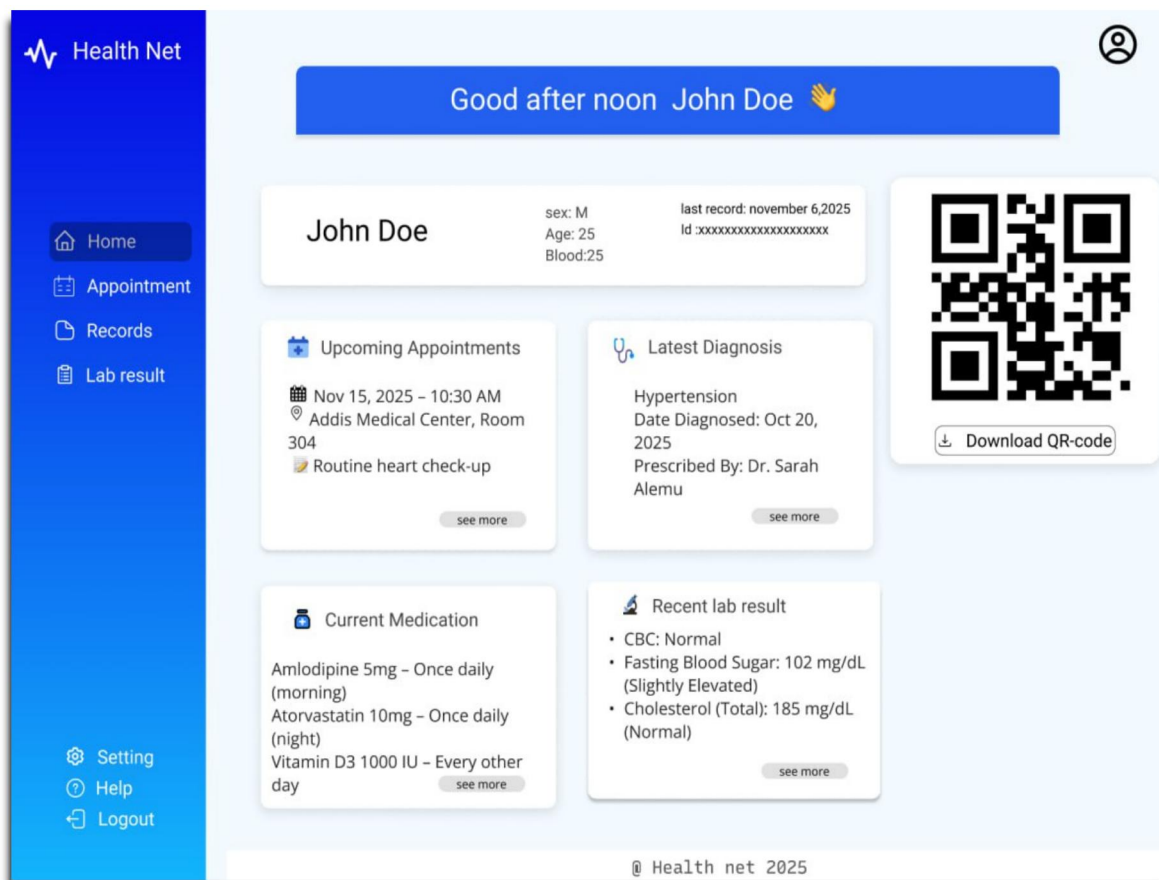


Figure 1: Patient Dashboard

Patient Health records Tab: Show's the Shrunk version of the patient records.

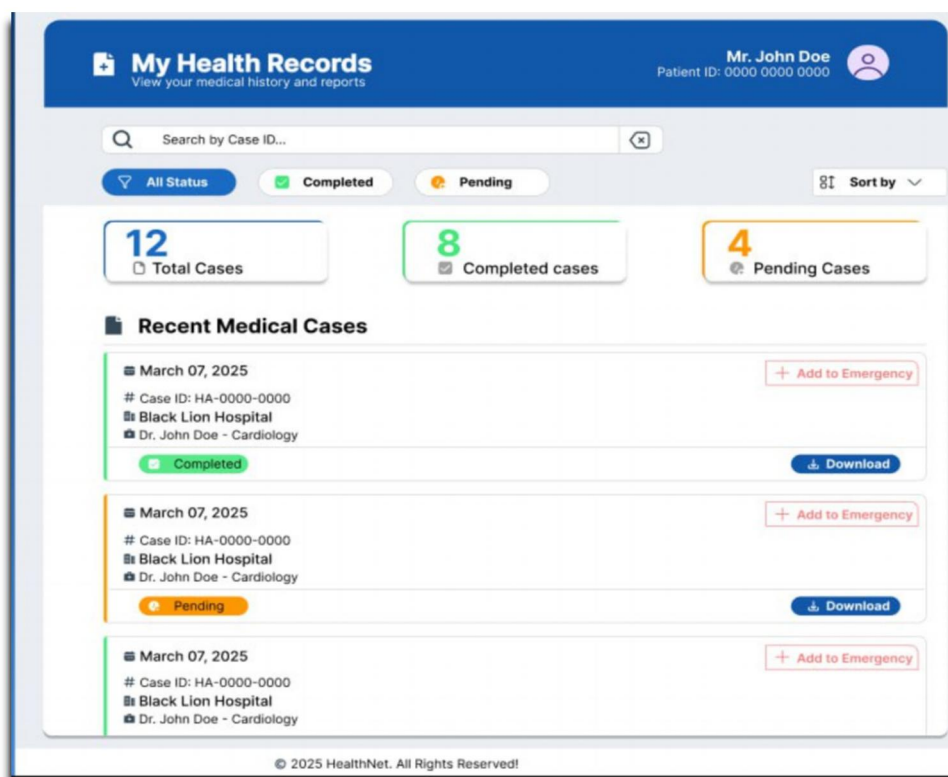


Figure 2: Patient records

Patient Laboratory records Tab: Show's the laboratory results of patient.

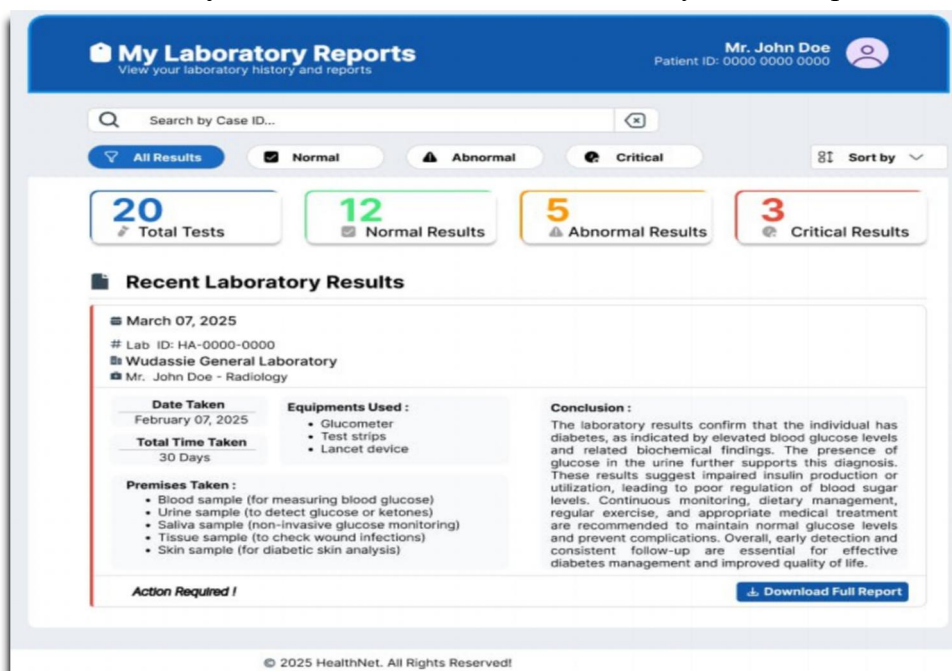
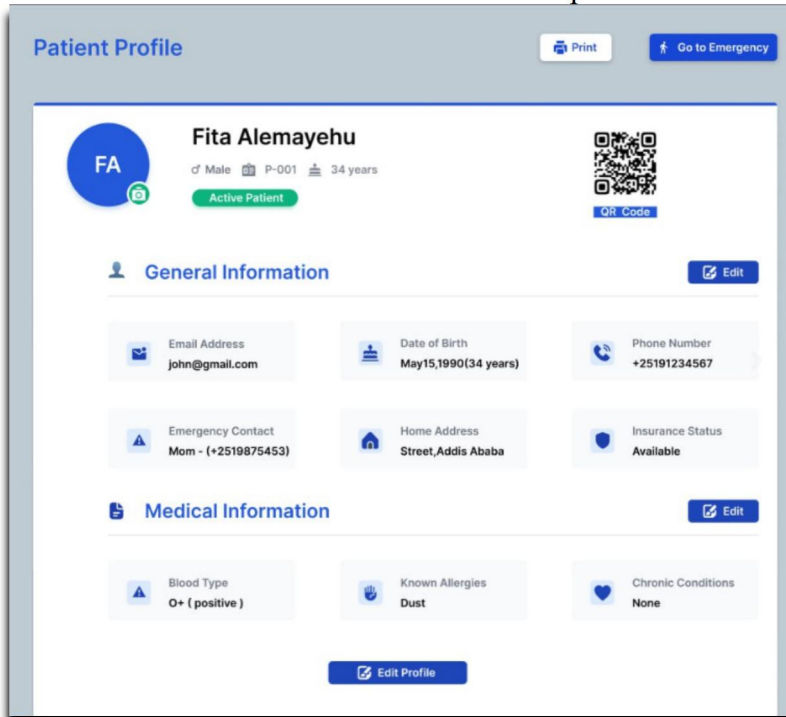


Figure 3: Patient Lab records

Patient Profile Tab: Show's Information about the patient.



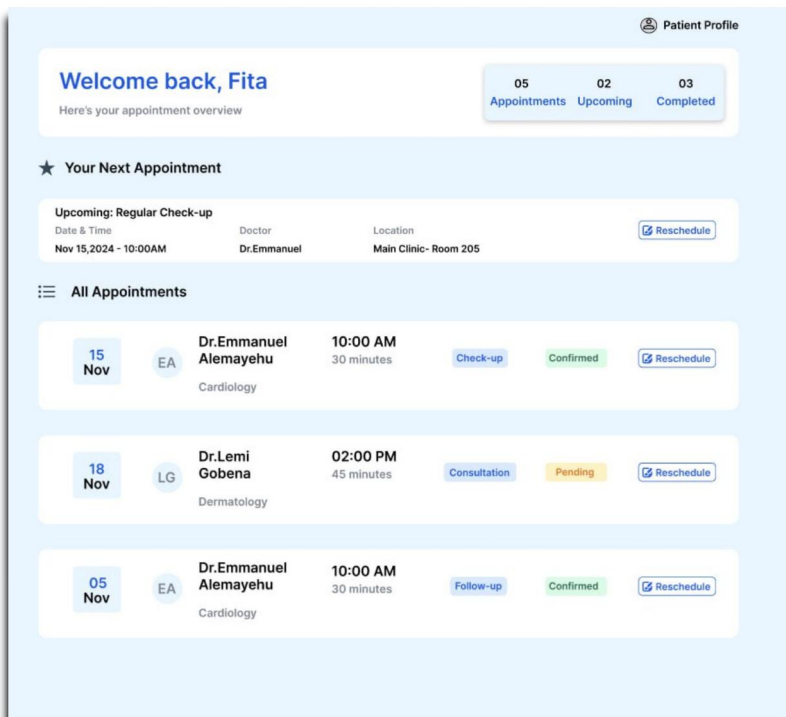
The screenshot shows the 'Patient Profile' tab for a patient named Fita Alemayehu. The profile includes a circular avatar with the initials 'FA', a QR code, and a status of 'Active Patient'. The 'General Information' section contains fields for Email Address (john@gmail.com), Date of Birth (May 15, 1990, 34 years), Phone Number (+25191234567), Emergency Contact (Mom - (+2519875453)), Home Address (Street, Addis Ababa), and Insurance Status (Available). The 'Medical Information' section includes Blood Type (O+ (positive)), Known Allergies (Dust), and Chronic Conditions (None). There are 'Print', 'Go to Emergency', 'Edit', and 'Edit Profile' buttons.

General Information	
Email Address	john@gmail.com
Date of Birth	May 15, 1990 (34 years)
Phone Number	+25191234567
Emergency Contact	Mom - (+2519875453)
Home Address	Street, Addis Ababa
Insurance Status	Available

Medical Information	
Blood Type	O+ (positive)
Known Allergies	Dust
Chronic Conditions	None

Figure 4: Patient Profile

Patient Appointment Tab: Show's records of past and future appointments.



The screenshot shows the 'Patient Appointment' tab for Fita Alemayehu. It features a 'Welcome back, Fita' message and a navigation bar with '05 Appointments', '02 Upcoming', and '03 Completed'. The 'Your Next Appointment' section shows an upcoming 'Regular Check-up' on Nov 15, 2024, at 10:00 AM with Dr. Emmanuel. The 'All Appointments' section lists three appointments: a 'Check-up' on Nov 15, a 'Consultation' on Nov 18, and a 'Follow-up' on Nov 05. Each appointment entry includes the doctor's name, time, duration, status, and a 'Reschedule' button.

Your Next Appointment	
Upcoming: Regular Check-up	
Date & Time	Nov 15, 2024 - 10:00 AM
Doctor	Dr. Emmanuel
Location	Main Clinic- Room 205
Reschedule	

All Appointments	
15 Nov	Dr. Emmanuel Alemayehu
10:00 AM	30 minutes
Check-up	Confirmed
Reschedule	
18 Nov	Dr. Lemi Gobena
02:00 PM	45 minutes
Consultation	Pending
Reschedule	
05 Nov	Dr. Emmanuel Alemayehu
10:00 AM	30 minutes
Follow-up	Confirmed
Reschedule	

Figure 5: Patient Appointments

3.1.2 Hardware Interfaces

The HealthNet system has minimal hardware interface requirements. It does not directly control or interact with external medical hardware devices. However, the system requires access to the following standard device components:

- ✓ Mobile or computer device hardware
- ✓ Smartphones, tablets, or computers are used to access the system.
- ✓ Camera / QR Scanners

Required to scan patient QR codes for emergency access. No specialized medical equipment (such as biometric readers or hospital machines) is connected to the system. Therefore, the system has no direct hardware interface requirements beyond camera access for QR scanning.

3.1.3 Software Interfaces

The HealthNet System interacts with the following software components. Each interface is defined in terms of its purpose, supported version, and data exchange format. No external customer-mandated systems are required; all other software selections are considered design decisions

1. Web Browser

- ✓ Name: Web Browsers
- ✓ Mnemonic: WB
- ✓ Specification Number: N/A
 - Version: "Google Chrome" (Chrome version 138 or above)
 - "Mozilla Firefox" (Firefox version 140 or above)
 - "Safari" (Safari browser version 18 or above)
 - "Microsoft Edge" (Microsoft Edge browser version 110 or above)
 - "Opera" (Opera version 93 or above)
- ✓ Source: Google Play Store / Apple App Store / Public Web Browsers
- ✓ Purpose: Runs the HealthNet application and supports user interaction.
- ✓ Interface: The interface with web browsers is based on standard web technologies (HTML5, CSS, and JavaScript). The browser shall capture user inputs (keyboard, mouse, or touch events) and send requests to the backend server using HTTPS. Responses from the server shall be displayed dynamically on the web interface.

2. Database Management System

- ✓ Name: Relational Database Management System (RDBMS)
- ✓ Mnemonic: DB

- ✓ Specification Number: N/A
- ✓ Version: Any version specified by the institution
- ✓ Source: Open-source or licensed database software
- ✓ Purpose: Stores patient data, medical records, login credentials, and QR info.
- ✓ Interface: Communication is through secure queries and server-side scripts
- ✓ Data returned in structured database format (e.g., SQL rows)

3. Backend Server / API

- ✓ Name: REST API
- ✓ Mnemonic: API
- ✓ Purpose: Handles communication between client (app/website) and database
- ✓ Interface Format: JSON formatted requests and responses
- ✓ Example Request: { "patient_id": "P0021" }
- ✓ Example Response: { "blood_type": "O+", "allergies": "None" }
- ✓ No customer-specified external system must be integrated.
- ✓ All other software choices are design decisions, not requirements.

3.1.4 Communications Interfaces

The **HealthNet** system uses *standard internet communication* to exchange data. There are no custom communication protocols.

- ✓ Communication is supported through: HTTPS (Hypertext Transfer Protocol Secure)
- ✓ Used to send medical and login data securely between client and server
- ✓ TCP/IP
- ✓ Underlying transport protocol used by internet-connected devices Wi-Fi or Mobile Data
Required for mobile app or web access

Purpose:

- ✓ Patient login
- ✓ Doctor login and data entry
- ✓ Admin data entry
- ✓ Viewing and updating pending medical records
- ✓ Emergency access via QR
- ✓ No special local network configurations or proprietary communication methods are required.
- ✓ Standard protocols ensure compatibility with hospitals, phones, and browsers.

3.2 Functional requirements

3.2.1 Introduction

This section describes the specific functional requirements of the *HealthNet* system. The system allows authorized administrators to register and manage patient information while ensuring strict confidentiality of health data. Patients can log in to view their records and manage emergency information, but they cannot alter medical details. The system also supports emergency QR access for limited, life-saving data visibility.

Table 2: Admin Login

ID	FR-01
Name	Admin Login
Introduction	The administrator shall be able to log into the system to manage patients and doctors.
Input	Admin username and password
Processing	The system shall verify the username and password to ensure the user is a registered admin.
Output	Successful login redirects to the admin dashboard.
Error Handling	Displays “Invalid username or password” if login fails.
Reference	UC-00 (User Login)

Table 3: Register Patients(mass registration)

ID	FR-02
Name	Patient Registration
Introduction	The admin shall be able to register new patients and create their accounts.
Input	Patient personal information, contact details, insurance status
Processing	The system stores data, generates patient username, password, and QR code.
Output	Patient account created and QR code generated.
Error Handling	Displays error if required fields are empty or duplicate patient exists.
Reference	UC-01 (Register User)

Table 4: Register Doctors and Assign Patients

ID	FR-03
Name	Register Doctors & Assign Patients
Introduction	Admin shall register doctors and assign specific patients to each doctor.
Input	Doctor information and selected patient IDs
Processing	System creates doctor account and links selected patients to that doctor.
Output	Doctor account created and patients assigned.
Error Handling	Displays message if assigned patient does not exist or username is taken.
Reference	UC-01 (Register User), UC-02 (Assign Patient to Doctor)

Table 5: Doctor Login

ID	FR-04
Name	Doctor Login
Introduction	Doctors shall be able to log in and access only assigned patients.
Input	Doctor username and password
Processing	System validates credentials and loads assigned patients.
Output	Doctor dashboard with patient list.
Error Handling	Displays “Invalid login” if credentials are incorrect.
Reference	UC-00 (User Login)

Table 6: Add Diagnosis/ Test Results

ID	FR-05
Name	Add Medical Records
Introduction	Doctors shall add diagnoses, treatments, or test results for assigned patients.
Input	Diagnosis details, lab results, patient ID
Processing	System attaches medical updates to patient record.
Output	“Record updated successfully” message.
Error Handling	If doctor accesses an unassigned patient, system shows “Access Denied.”
Reference	UC-05 (Update Patient Record)

Table 7: Patient Login

ID	FR-06
Name	Patient Login
Introduction	Patients shall be able to log in to view their medical records.
Input	Username and password
Processing	System validates credentials and loads patient profile.
Output	Patient dashboard with medical information.
Error Handling	“Invalid credentials” if username or password is wrong.
Reference	UC-00 (User Login)

Table 8: View Medical Records(patients)

ID	FR-07
Name	View Medical Record
Introduction	Patients shall view their medical record but cannot edit anything.
Input	Patient login session
Processing	System retrieves stored medical data belonging to logged-in patient.
Output	Displays treatments, visits, medications, and diagnoses.
Error Handling	If no record is found → “No record available.”
Reference	UC-09 (View Lab Results and Diagnosis)

Table 9: Insurance Status Record

ID	FR-08
Name	Insurance Status Record
Introduction	Admin shall record or update insurance status for each patient.
Input	Insurance provider, policy number, coverage validity
Processing	System stores and updates insurance details in patient record.
Output	Insurance information successfully updated.
Error Handling	Missing or invalid inputs → “Enter valid insurance information.”
Reference	UC-03 (Manage User Profiles) - Admin high-privilege extension

Table 10: Emergency Tab(patients)

ID	FR-09
Name	Emergency Information
Introduction	Patients shall add medical info they want visible in emergencies.
Input	Blood type, allergies, chronic diseases, emergency contacts
Processing	System saves info in an Emergency Tab linked to QR code.
Output	“Emergency information updated.”
Error Handling	If required fields are missing, system shows an error.
Reference	UC-03 (Manage User Profiles)

Table 11: QR Code Emergency Access

ID	FR-10
Name	QR Emergency Access
Introduction	Anyone scanning QR shall view ONLY the patient’s emergency information.
Input	Scanned QR code
Processing	System fetches and displays emergency tab without login.
Output	Emergency information display page.
Error Handling	Invalid QR → “QR not recognized.”
Reference	UC-10 (QR-Based Record Sharing)

3.3 Use Cases

Table 12: US-00 — User Login

Use Case ID	US-00
Actors	Admin, Doctor, Patient
Trigger	The user clicks the “ Login ” button after entering credentials (email/username and password).
Precondition	The user already has an account registered in the system.
Main Flow	<ol style="list-style-type: none"> 1. User opens the login page. 2. Enters valid username and password. 3. System checks credentials against stored data in the users table 4. If credentials match, the system creates an or issues a JWT token. 5. User is redirected to their dashboard based on their role (Admin, Doctor, or Patient).
Alternative Flows	<p>A1: Invalid Credentials</p> <ol style="list-style-type: none"> 3a. Credentials do not match. 3b. System displays: “Incorrect email or password.” and prompts retry.
Extension	<p>E1: Forgot Password</p> <p>E2: Remember Me</p>
Exception	Database Connection Error
Postcondition	User session is created, and access rights are granted based on user role.
Failure Scenario	Repeated invalid attempts (e.g., 5 times) → System temporarily locks the account or logs failed login attempts for security monitoring.

Table 13: US-01 — Register User

Use Case Name	Register User ID (US-01)
Actors	Admin
Trigger	Admin clicks “Register User” and selects user type (Doctor or Patient).
Precondition	Admin must be authenticated (logged in and verified).
Main Flow	1. Admin opens registration form. 2. Fills in required details (name, email, password, role, contact info). 3. System checks PostgreSQL for existing user with same email. 4. If unique, system inserts new record into users table. 5. Confirmation message ("User registered successfully") is displayed.
Alternative Flow	A1: Duplicate user credential 3a. System finds an existing user with the same email address. 3b. System prompts: “ User already exists. ” 3c. Admin is returned to the registration form.
Extension	E1: Role Selection Error.
Exception	Database/Network Failure
Postcondition	A new user record is successfully saved in the PostgreSQL users table with the correct role (Doctor or Patient).
Failure Scenario	Database constraint error or network issue prevents the insertion (covered in Exception X1).

Table 14: US-02 — Assign Patient to Doctor

Use Case Name	Assign Patient to Doctor (US-02)
Actors	Admin
Trigger	Admin selects “Assign Patient” in the dashboard.
Precondition	Both doctor and patient records must exist in the database.
Main Flow	<ol style="list-style-type: none"> 1. Admin selects a specific Patient ID and Doctor ID. 2. System validates IDs and creates a new entry in the database. 3. Confirmation message ("Assignment successful") is displayed.
Alternative Flow	A1: Invalid IDs <ol style="list-style-type: none"> 2a. One or both submitted IDs do not exist in the database. 2b. System displays: “User not found.” 2c. Admin is returned to the selection step.
Extension	Existing Assignment
Exception	Database Error
Postcondition	The patient-doctor relationship is successfully stored in the database
Failure Scenario	Foreign key constraint violation or SQL error .

Table 15: US-03 — Manage User Profiles

Use Case Name	Manage User Profiles (US-03)
Actors	Admin, Doctor, Patient
Trigger	User clicks “Edit Profile.”
Precondition	User is authenticated. Authorization Rule: Users are authorized to edit their personal contact/login details only . Only Admin and specific roles (e.g., Doctor) are authorized to edit sensitive record fields (like lab results, role, status).
Main Flow	<ol style="list-style-type: none"> 1. User accesses their profile page. (Display logic: Sensitive fields are read-only for Patient/Doctor self-service.) 2. Updates <i>editable</i> fields (name, email, phone, address). 3. User clicks "Save". 4. System validates input. 5. System executes UPDATE 6. Confirmation message displayed.
Alternative Flow	A1: Invalid Data Format <ol style="list-style-type: none"> 4a. System finds invalid data (e.g., non-unique email). 4b. System prompts: “Invalid input. Please correct and resubmit.” 4c. User corrects the error and returns to step 3.
Extension	Admin Edits Restricted Fields (High Privilege)
Exception	Unauthorized Access Attempt
Postcondition	The user's profile (or related records) is updated successfully, adhering to the specified authorization level.
Failure Scenario	Database error or explicit authorization failure.

Table 16: US-04 — View Patient Record

Use Case Name	View Patient Record (US-04)
Actors	Doctor
Trigger	Doctor selects patient from assigned list.
Precondition	Doctor is logged in and the patient is officially assigned to the doctor.
Main Flow	<ol style="list-style-type: none"> 1. Doctor selects a patient from their list. 2. System validates the Doctor-Patient assignment for authorization. 3. System executes a query joining patient_records, lab_results, and diagnoses. 4. System fetches and displays the complete clinical history and lab results.
Alternative Flow	A1: No Medical Data 3a. The query returns an empty set (no medical data found). 3b. System displays: “ No medical data available. ”
Extension	Filter/Search Records 4a. Doctor applies a filter (e.g., date range, or record type like 'only show lab results').
Exception	Authorization
Postcondition	The patient's medical record is successfully displayed on screen, or a relevant status message is shown.
Failure Scenario	Unauthorized query or SQL fetch failure (covered in Exception X1).

Table 17: US-05 — Update Patient Record

Use Case Name	Update Patient Record (US-05)
Actors	Doctor
Trigger	Doctor clicks “Add Diagnosis” or “Update Record.”
Precondition	Doctor is logged in and authorized (patient is assigned to the doctor).
Main Flow	<ol style="list-style-type: none"> 1. Doctor opens the authorized patient file. 2. Doctor enters or modifies diagnosis details (e.g., ICD codes, notes, treatment plan). 3. Doctor clicks "Save." 4. System validates the input data against required format and constraints. 5. System executes INSERT INTO diagnoses (...) (for new) or UPDATE diagnoses SET ... (for existing). 6. System automatically logs timestamps and the Doctor's User ID for auditing. 7. Confirmation message ("Record updated") is displayed.
Alternative Flow	<p>A1: Invalid Entry</p> <ol style="list-style-type: none"> 4a. Validation fails due to missing required fields or incorrect data format. 4b. System highlights the problematic field(s) and displays: “Invalid entry. Please correct and resubmit.” 4c. Doctor corrects the error and returns to step 3.
Extension	<p>Adding a New Lab Result Link/File</p> <ol style="list-style-type: none"> 2a. Doctor uploads a document or adds a URL link to external lab results.
Exception	Data base Update or Authorization Failure
Postcondition	Patient medical record is successfully updated with the latest information, and the entry is timestamped .
Failure Scenario	Data base update fails or unauthorized access

Table 18: US-06 — Upload Lab Result

Use Case Name	Upload Lab Result (US-06)
Actors	Doctor, Lab Technician
Trigger	Doctor or Lab Technician selects “Upload Lab Result.”
Precondition	A Patient record must already exist in the system. The Actor must be authenticated and authorized to add records for the specific patient.
Main Flow	<ol style="list-style-type: none"> 1. Actor selects the patient 2. Actor selects the lab result file (e.g., PDF, image) or enters structured data. 3. System performs file validation (type, size). 4. System saves the file to the designated storage. 5. System executes an INSERT operation 6. Confirmation message ("Lab result uploaded and linked") is displayed.
Alternative Flow	A1: Invalid File <ol style="list-style-type: none"> 3a. File validation fails (e.g., prohibited file type, file size too large). 3b. System prompts: “Invalid file. Please ensure the file is the correct format (e.g., PDF) and size.” 3c. Actor corrects the file selection and returns to step 2.
Extension	Automated Notification
Exception	Database Failure
Postcondition	The lab result file is securely stored, and a corresponding entry is created in the database , successfully linked to the Patient ID.
Failure Scenario	File upload or SQL insert error
Postcondition	A new appointment record is successfully saved in the PostgreSQL appointments table and is visible on both the Doctor's and Patient's dashboards.
Failure Scenario	Database insert fails

Table 19: US-07 — Create Appointment

Use Case Name	Create Appointment (US-07)
Actors	Doctor
Trigger	Doctor selects “ Create Appointment. ”
Precondition	Doctor is logged in and the patient is assigned to the doctor.
Main Flow	<ol style="list-style-type: none"> 1. Doctor fills in appointment details (date, time, reason, duration). 2. System checks the appointments table for time conflicts (with both the Doctor's and Patient's schedules). 3. If unique, the System executes an INSERT INTO appointments with doctor_id, patient_id, and other details. 4. Confirmation message ("Appointment created successfully") is displayed.
Alternative Flow	<p>A1: Conflict Detected</p> <ol style="list-style-type: none"> 2a. System finds a conflict with the Doctor's or Patient's existing schedule. 2b. System displays: “Conflict detected. Please select a different time or date.” 2c. Doctor is returned to the detail filling step (step 1).
Extension	E1:Notification
Exception	Database insert failure
Postcondition	A new appointment record is successfully saved in database and is visible on both the Doctor's and Patient's dashboards.
Failure Scenario	SQL insert fails (covered in Exception X1).

Table 20: US-08 — View Appointments

Use Case Name	View Appointments (US-08)
Actors	Doctor, Patient
Trigger	User clicks “ Appointments. ”
Precondition	User must be authenticated (logged in).
Main Flow	<ol style="list-style-type: none"> 1. System determines the user's role and ID. 2. System executes a query to fetch relevant rows from the database 3. System sorts and formats the results, separating them into upcoming and past appointments. 4. The list of appointments is displayed to the user.
Alternative Flow	A1: No Appointments Found <ol style="list-style-type: none"> 2a. The appointment query returns an empty set 2b. System displays: “No appointments found.”
Extension	Filtering and Sorting
Exception	Database Connection Error
Postcondition	Appointments relevant to the authenticated user are successfully displayed on the screen.
Failure Scenario	Database connection error (covered in Exception X1).
Technology Used	PostgreSQL (appointments table).

Table 21: US-09 — View Lab Results and Diagnosis

Use Case Name	View Lab Results and Diagnosis (US-09)
Actors	Patient
Trigger	Patient selects “ My Health Records ” (or equivalent menu item).
Precondition	Patient is logged in and the system has associated diagnoses or lab_results for the patient.
Main Flow	<ol style="list-style-type: none"> 1. System validates the Patient's ID. 2. System executes queries to fetch rows from database and diagnoses 3. System formats the raw data 4. The consolidated diagnosis and lab results data is displayed to the patient in a secure, readable format.
Alternative Flow	A1: No Results Available <ol style="list-style-type: none"> 2a. The combined query returns an empty set (no lab results or diagnoses found) 2b. System displays: “No results available.”
Extension	Download Option
Exception	Database Fetch Failure
Postcondition	The Patient's medical data (lab results and diagnoses) is successfully viewed securely on the screen.
Failure Scenario	Unauthorized query (prevented by initial validation) or SQL fetch

Table 22: US-10 — QR-Based Record Sharing

Use Case Name	QR-Based Record Sharing (US-10)
Actors	Patient, External Viewer (Doctor/Lab/Institution)
Trigger	Patient clicks “Generate QR.”
Precondition	Patient is logged in and has authorized data available to share.
Main Flow	<ol style="list-style-type: none"> 1. Patient selects specific record(s) and sets access parameters (e.g., expiry time, one-time view). 2. System validates selections and creates a secure, tokenized link. 3. System inserts the token, Patient ID, and expiry timestamp into the database 4. System uses QRious.js (or similar) to generate a QR image pointing to the tokenized link. 5. External Viewer scans the QR code. 6. System validates the token against qr_shares and checks the expiry timestamp. 7. If valid, the system fetches only the authorized fields and displays them to the viewer.
Alternative Flow	A1: QR Expired <ol style="list-style-type: none"> 6a. External Viewer scans the QR, but the system finds the expiry timestamp has passed. 6b. System displays an “Access Expired” message to the viewer. 6c. System prompts the Patient to log in and regenerate a new QR code.
Extension	Revoke Access <ol style="list-style-type: none"> 1a. Patient accesses a list of active shares (tokens).
Exception	Invalid Token or Unauthorized Scan
Postcondition	An external user is able to view the authorized, shared data temporarily before the token expires or is revoked.
Failure Scenario	Token invalid, unauthorized scan, or database failure (covered in Exception X1).

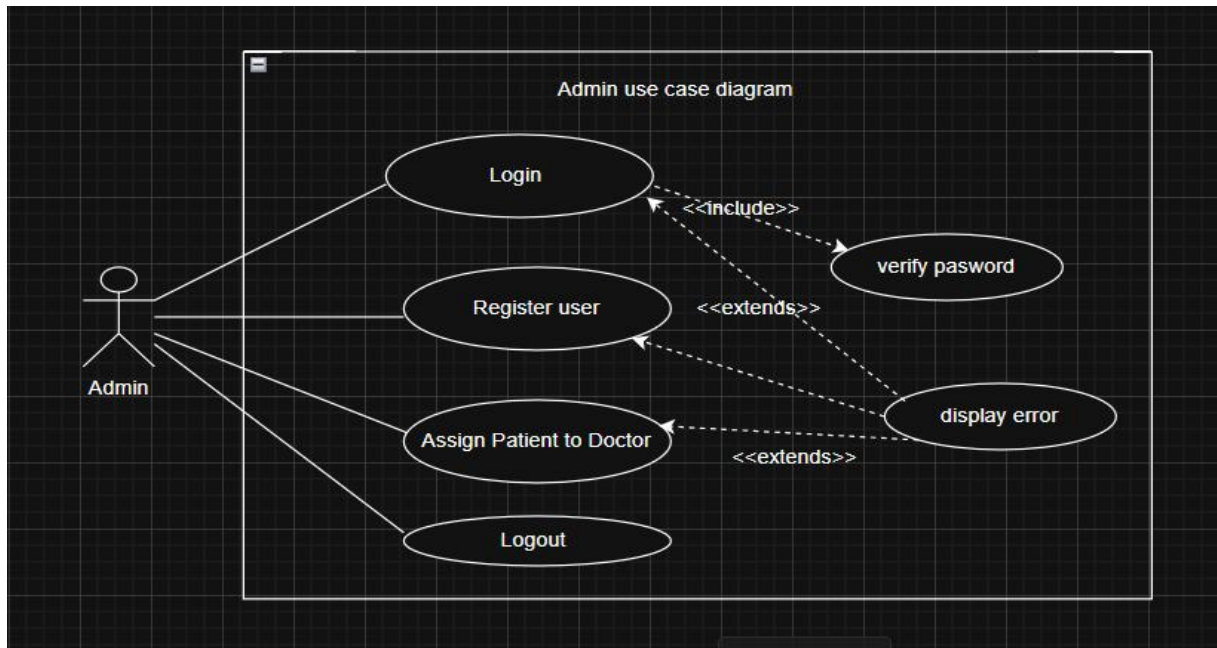


Figure 6: Admin Usecase

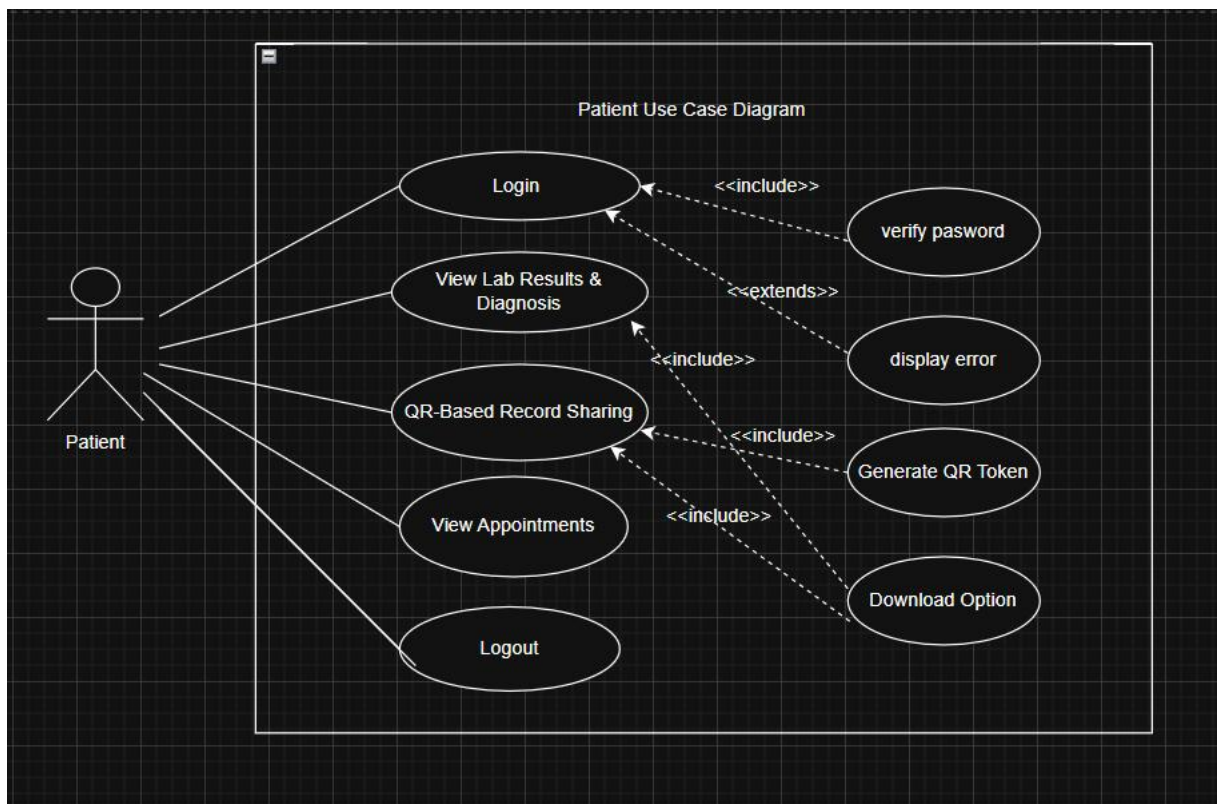


Figure 7: Patient Usecase

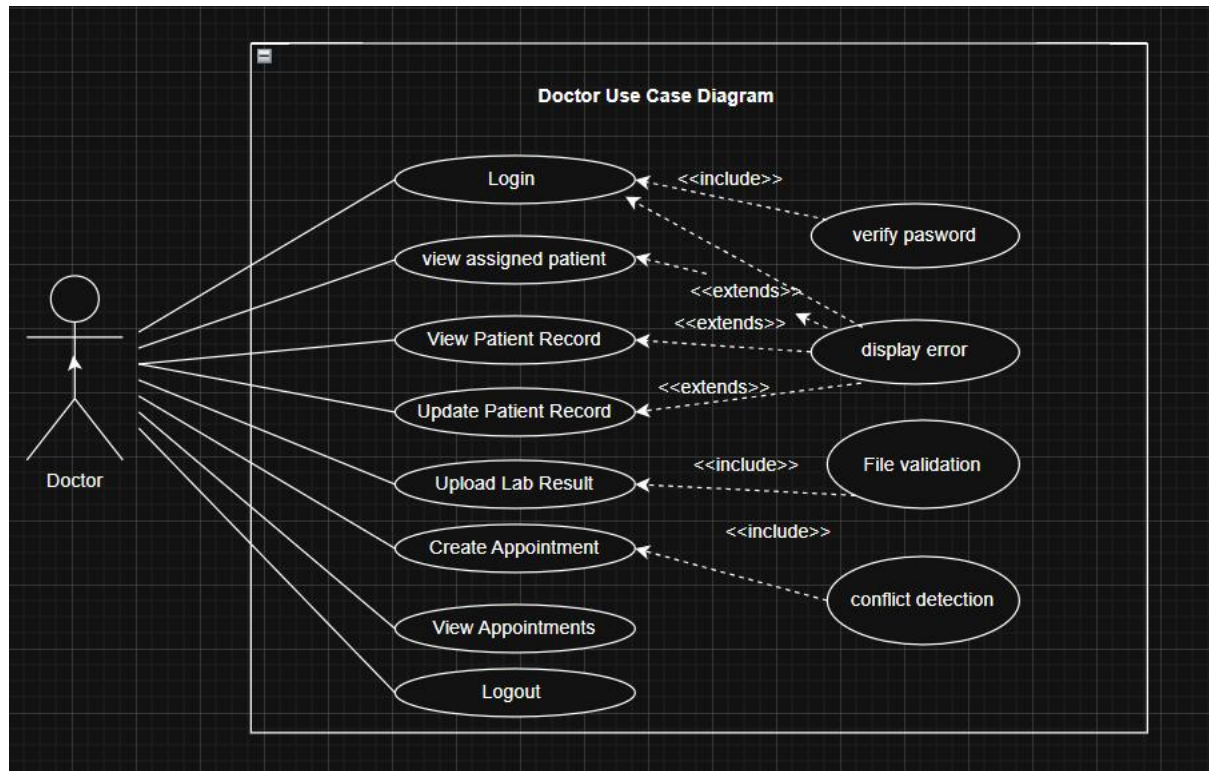


Figure 8: Doctor Usecase

3.4 Non-Functional Requirements

3.4.1 Performance

- ✓ The system will respond to 95% of the user requests (login, access to patient record, or emergency access by QR code scan) within a few seconds on a minimum 3G connection.
- ✓ The HealthNet database will support high number of user concurrency at full utilization without performance degradation maintaining stability.
- ✓ Emergency access through QR code will access critical patient information without noticeable delay.

3.4.2 Reliability

- ✓ The system will be accessible about 99.5% of the time except during planned maintenance.
- ✓ Failed transactions (e.g., form submission of incomplete forms) will be logged and recoverable.

3.4.3 Availability

- ✓ The system will run 24 hours a day, 7 days a week from authorized sites.
- ✓ Planned maintenance will occur only during designated low activity periods, with prior notice to users.

3.4.4 Security

- ✓ All communication from client to server will be HTTPS .
- ✓ Role-based authentication will be used, with access to data restricted by user type (e.g.physician, nurse and admins).
- ✓ Passwords will be securely hashed using current modern hashing mechanisms.
- ✓ The system automatically locks out accounts following five successive failed login attempts.
- ✓ QR-code emergency access displays only restricted, pre-approved health information (e.g. medication, allergies).

3.4.5 Maintainability

- ✓ The system will be modular in design with separate modules for user administration, patient details, and emergency access to support independent updating.
- ✓ Any update or patch to the system will install and test in under few hours.
- ✓ Source code will follow standard documentation practices (inline comments, API documentation).
- ✓ System logs will be maintained and verified periodically for debugging convenience and performance optimization.

3.4.6 Portability

- ✓ The web application shall be compatible with common and latest browsers (Chrome , Firefox , Safari and Edge).
- ✓ The system shall be executed on Windows and Linux servers.
- ✓ The front-end interface shall be responsive and usable on mobile, tablet, and desktop systems.

3.4.7 Usability

- ✓ It is expected that new users are able to access basic functionality like login and patient search with no formal training.
- ✓ Standard iconography, tool-tips, and form validation will direct user input and reduce data entry errors.

3.4.8 Scalability

- ✓ Vertical and horizontal scaling for supporting more healthcare facilities across the nation will be facilitated by the system.
- ✓ Designing the database to support future partitioning by facility or region to scale millions of patient records will be supported.

3.4.9 Legal and Ethical Compliance

- ✓ Admin permission would be required for divulging information outside of the treatment center.

3.5 Inverse Requirements

The **HealthNet** system is explicitly designed not to perform the following functions:

1. The system shall not allow Patients to edit, alter, or delete any medical data (diagnoses, lab results) entered by a Doctor or Lab Technician. They may only view this information.
2. The system shall not allow Doctors to view or access medical records of patients who are not explicitly assigned to them.
3. The system shall not process payments, manage billing, or handle financial transactions.
4. The system shall not facilitate direct messaging or chat between users (e.g., Patient to Doctor).
5. The system shall not automatically share or transmit patient data to external health information exchanges or other third-party systems without explicit administrative action and patient consent.

3.6 Design Constraints

The design and implementation of the *HealthNet* system shall adhere to the following constraints:

Database Constraints:

- ✓ The system shall use a relational database management system (implied by the use of terms like "PostgreSQL," "users table," and "foreign key constraint" in the use cases).

Architectural Constraints:

- ✓ The system shall enforce a role-based access control (RBAC) model with three distinct roles: Patient, Doctor/Lab Technician, and Administrator.

3.7 Logical Database Requirements

A relational database management system (RDBMS) will be used as the primary data storage solution for the HealthNet system. This is essential for maintaining the structured, secure, and persistent storage of all Electronic Health Records (EHR), user credentials, appointment schedules, and system relationships.

3.7.1 Logical Requirements

1. Data Formats

- ✓ Email addresses, phone numbers, and dates must follow standardized formats
- ✓ Unique Patient Identifier (UPI) and usernames must be system-generated and unique
- ✓ Medical codes and file types must be stored in structured fields.

2. Storage Capabilities

- ✓ Must support 100,000+ users and millions of associated records based on the technical equipments used.
- ✓ Architecture must allow for vertical and horizontal scaling
- ✓ Complex queries should execute within 2 seconds under typical load

3. Data Retention

- ✓ Patient medical records: Retained indefinitely
- ✓ Audit logs and activity records: Minimum 5-7 years
- ✓ Inactive accounts: Archived/anonymized after 3-5 years of inactivity

4. Data Integrity

- ✓ Enforce referential integrity through foreign key constraints
- ✓ Ensure primary keys are unique and non-null
- ✓ Validate data domains (e.g., blood type values, future appointment dates)
- ✓ Maintain business logic constraints (e.g., doctors can only access assigned patients)

5. Data Security & Access

- ✓ Schema must support Role-Based Access Control (RBAC) implementation
- ✓ Include audit trail fields (timestamps, user IDs) in critical tables
- ✓ Design should allow for future data partitioning by region/facility

3.7.2 Schema Design

To ensure the safety and security of the system's data, all dynamic information will be stored in a well-organized database. Good logical database is also important for future maintenance. The database used by the system will have the following structure. *This is just to show case our database design, any changes might apply while implementing the design.*

Table 23: Database Schema

Database Table	Attributes	Description
Users	<ul style="list-style-type: none"> • user_id (PK) • name • email • password_hash • phone • role_id (FK) • status • created_at 	Stores login credentials and basic profile data for Admins, Doctors, and Patients. Each user is linked to a role.
Roles	<ul style="list-style-type: none"> • role_id (PK) • role_name 	Defines user roles in the system (Admin, Doctor, Patient).
Patients	<ul style="list-style-type: none"> • patient_id (PK) • user_id (FK) • dob • gender • address • created_at 	Contains personal and demographic data of patients; linked to their user account.
Doctors	<ul style="list-style-type: none"> • doctor_id (PK) • user_id (FK) • specialization • license_no • contact_info • created_at 	Contains doctor details such as specialization, license number, and linked user ID.
Assignments	<ul style="list-style-type: none"> • assignment_id(PK) • doctor_id (FK) • patient_id (FK) • assigned_date 	Links doctors and patients; used to enforce access control (only assigned doctors can update patient data).
Diagnoses	<ul style="list-style-type: none"> • diagnosis_id(PK) • patient_id (FK) • doctor_id (FK) • diagnosis_text • treatment_plan • created_at 	Stores all diagnostic information and treatment notes made by doctors for assigned patients.

Lab_results	<ul style="list-style-type: none"> • lab_result_id (PK) • patient_id (FK) • doctor_id (FK) • file_path, result_type • created_at 	Contains uploaded or structured lab test results linked to patients and doctors.
Appointments	<ul style="list-style-type: none"> • appointment_id (PK) • doctor_id (FK) • patient_id (FK) • date, time • reason • status 	Records all scheduled appointments between doctors and patients, ensuring no time conflicts.
Insurance	<ul style="list-style-type: none"> • insurance_id (PK) • patient_id (FK) • provider_name • policy_number • valid_until 	Stores insurance provider information, policy numbers, and coverage validity for each patient.
Emergency_Info	<ul style="list-style-type: none"> • emergency_id (PK) • patient_id (FK) • blood_type 	Contains life-saving emergency details (visible

3.8 Other Requirements

Training-related Requirements:

- ✓ The system shall be intuitive enough for Patients and Doctors to use with minimal formal training.
- ✓ Administrator training may be required for bulk user registration, complex user assignment, and understanding system audit logs.

Packaging Requirements:

- ✓ The final delivered system shall be a packaged web application deployable on a standard web server with a connected backend application server and database.
- ✓ All installation and configuration procedures shall be documented in a README.md file included with the source code.

Legal Requirements:

- ✓ All source code developed for the project shall be the intellectual property of the developing team/organization, unless otherwise specified.
- ✓ The application shall display a Privacy Policy and Terms of Use, detailing how patient data is collected, stored, and used.

4. Change Management Process

The following process shall be used to manage changes to this Software Requirements Specification (SRS). Any team member can propose a change by submitting it to the team's Telegram group or during a Zoom or physical meeting; the proposal must include the specific SRS section, the rationale for the change, and the proposed new text. These requests will be discussed and reviewed collectively during weekly team meetings, where the advisor may be consulted for input. The group members retain final authority to approve or reject changes based on their impact on scope, timeline, and technical feasibility. Once a change is approved, the designated member will update the SRS document on GitHub, ensuring the revision history table is amended with the date, description, and author. Communication of the updated SRS will be confirmed by announcing the changes in the Telegram group and providing a direct link to the latest version on GitHub, ensuring all team members are synchron

References

This subsection provides a complete list of all sources, interviews, and references that informed the requirements in this Software Requirements Specification.

❖ Ethiopian Federal Ministry of Health

Ethiopian eHealth Strategy 2025

- ✓ Publishing Organization: Federal Democratic Republic of Ethiopia, Ministry of Health
- ✓ Date: 2021
- ✓ Source: Ministry of Health official publications and policy documents
- ✓ This document provided the strategic framework and national compliance requirements for the **HealthNet** system.

❖ Yekatit 12 Hospital Medical College

Formal Interviews with Hospital Staff

- ✓ Participants: Doctors, Nurses, Administrative Staff, and IT Personnel
- ✓ Date: Conducted October-November 2025
- ✓ Location: Yekatit 12 Hospital Medical College, Addis Ababa
- ✓ Source: Interview transcripts and notes maintained in project documentation
- ✓ These interviews provided critical insights into current workflow challenges, data management practices, and user needs within a functioning Ethiopian healthcare facility.

❖ Yekatit 12 Hospital Medical College

Extended Clinical Internship and Domain Expertise

- ✓ Team Member: Martha Tegegne (HealthNet Team Member)
- ✓ Duration: 2-month clinical internship (September-October 2025)
- ✓ Role: Direct hospital experience and continuous requirements validation
- ✓ Source: Daily clinical exposure, system workflow observation, and regular team consultations
- ✓ Provided deep, practical understanding of patient record flows, doctor-patient interactions, medication processes, and real-world hospital operational challenges that directly shaped system functionality.

❖ Fayda Digital ID System

Technical Specifications and Implementation Framework

- ✓ Publishing Organization: National ID Program, Ethiopia
- ✓ Date: Ongoing implementation (2023-present)
- ✓ Source: Publicly available technical documentation and implementation guidelines
- ✓ Informed the design principles for the Unique Patient Identifier (UPI) system, including interoperability considerations, biometric data handling protocols, and national identification standards for integration with Ethiopia's official digital identity system.

❖ Personal Healthcare Experiences

Team Member Patient Experiences

- ✓ Context: Various Ethiopian healthcare facilities over multiple years
- ✓ Date: Cumulative experiences 2010-2025
- ✓ Source: Team discussions and requirements brainstorming sessions

- ✓ Provided grassroots understanding of patient journey pain points, record-keeping inefficiencies, and opportunities for digital improvement in the Ethiopian healthcare context.

❖ **Informal Healthcare Sector Consultations**

Discussions with Hospital Working Personnel

- ✓ Participants: Various healthcare professionals across multiple facilities
- ✓ Date: Ongoing throughout 2025
- ✓ Source: Team member field notes and consolidated insights
- ✓ Supplemented formal interviews with additional perspectives on daily operational challenges, technology adoption barriers, and practical feature priorities.