Software Requirement Specifications

Sehat Pal

Version: 2.00

Project Code	*N/A*
Supervisor	*Fizza Mansoor*
Co Supervisor	*N/A*
Project Team	*Syed Ukkashah* *Ibrahim Johar*
Submission Date	*7th May, 2025*

Document History

Version	Name of Person	Date	Description of change
1.0	*Syed Ukkashah*	*2nd May*	*Document Created*
1.1	*Syed Ukkashah*	*3rd May, 2025*	*Added Non-functional requirements*
2.0	*Ibrahim Johar*	*5th May, 2025*	*Added Use Cases*

Distribution List

Name	Role
Fizza Mansoor	*Supervisor*

Document Sign-Off

[Following table will contain sign-off details of document. Once the document is prepared and revised, this should be signed-off by the sign-off authority.

Any subsequent changes in the document after the first sign-off should again get a formal sign-off by the authorities.]

Version	Sign-off Authority	Sign-off Date
2.00	*Fizza Mansoor*	

Table of Contents

1. Introduction

1.1. Purpose of Document

This Software Requirements Specification (SRS) document outlines the requirements and specifications for Sehat Pal, a web-based symptom checker developed as a university project by Syed Ukkashah (backend developer) and Ibrahim Johar (frontend developer). The primary purpose of this document is to define the system's functionality, constraints, and design considerations to guide its development, implementation, and evaluation within an academic context. Specifically, it serves the following objectives:

- ***Define System Scope**: The SRS provides a clear scope for Sehat Pal, a prototype designed to assist a small user base (5-10 users) in Pakistan with preliminary health assessments, medical record management, and emergency access. It highlights the integration of Firebase Hosting, Firebase Authentication (with email functionality for signup, verification, and password reset), Infermedica API (trial basis: 2,000 API calls or 60 days), Bootstrap 5 for UI design, Axios for API calls, and a simple URL redirect for an emergency helpline.*
- ***Guide Development**: It serves as a blueprint for the developers (Syed Ukkashah and Ibrahim Johar) to ensure the system meets specified functional and non-functional requirements, such as secure data handling, user authentication, and performance under the constraints of Firebase free-tier limits and the Infermedica API trial.*
- ***Facilitate Academic Evaluation**: The document is a formal submission for assessment by the course instructor, Miss Fizza Mansoor, and other academic evaluators, demonstrating adherence to software engineering principles, including requirements analysis, system design, and documentation standards.*
- ***Communicate with Stakeholders**: It provides a comprehensive reference for target users (individuals in Pakistan with basic digital literacy), future maintainers, and collaborators, explaining the system's intended use, limitations (e.g., preliminary assessments only), and safety considerations (e.g., disclaimers to consult doctors).*
- ***Mitigate Risks**: By detailing assumptions, constraints, and safety requirements, the SRS helps identify potential issues (e.g., API limits, network variability in Pakistan) and proposes mitigations to ensure a reliable prototype.*

This document is structured to balance technical detail for developers and evaluators with accessibility for users, ensuring all stakeholders understand Sehat Pal's purpose, capabilities, and limitations within the university project's scope.

1.2. Intended Audience

This document, the Software Requirements Specification (SRS) for Sehat Pal, is intended for the following individuals and groups who are concerned with or expected to use it:

- ***Course Instructor (Miss Fizza Mansoor)**: The primary academic evaluator responsible for assessing the university project's compliance with software engineering principles, functionality, and documentation quality. This SRS serves as a formal submission to meet course requirements and provide a clear blueprint for the system's design and implementation.*
- ***Project Developers (Syed Ukkashah and Ibrahim Johar)**: The backend and frontend developers who designed and implemented Sehat Pal. They use this document as a reference to ensure the system aligns with specified requirements, including integration with Firebase Hosting, Firebase Authentication, Infermedica API (trial basis: 2,000 API calls or 60 days), Bootstrap 5, and Axios, as well as the emergency helpline redirect.*

- ***Academic Evaluators****: Faculty members or peers involved in reviewing the project for academic purposes, such as grading or feedback, who need to understand the system's scope, constraints, and functionality for a small user base (5-10 users) in Pakistan.*
- ***Target Users****: The end-users, primarily individuals in Pakistan with basic digital literacy, who will interact with Sehat Pal for preliminary health assessments, medical record management, and emergency access. This document helps them understand the system's intended use and limitations (e.g., reliance on Firebase services and the Infermedica API trial).*
- ***Future Maintainers or Collaborators****: Potential students or developers who may extend or maintain the system post-project, using this SRS as a foundation for understanding its architecture and requirements.*

The document is tailored to be accessible to this diverse audience, balancing technical detail for developers and evaluators with clear explanations for users and instructors.

1.3 Abbreviations

- ***API****: Application Programming Interface - A set of rules and tools allowing software components (e.g., Infermedica API) to communicate with Sehat Pal.*
- ***Axios****: A JavaScript library used for making HTTP requests to the Infermedica API from the client-side frontend.*
- ***Bootstrap 5****: The latest version of the Bootstrap CSS framework, utilized for responsive and mobile-first UI design in Sehat Pal.*
- ***Firebase****: A platform by Google providing cloud-based services, including Authentication, Firestore, Realtime Database, and Hosting, used for Sehat Pal's backend and hosting.*
- ***Firestore****: A scalable NoSQL cloud database within Firebase, used to store medical records in Sehat Pal.*
- ***HTTPS****: Hypertext Transfer Protocol Secure - A secure communication protocol (using TLS 1.2 or higher) for data transmission between Sehat Pal and external services.*
- ***Infermedica API****: The AI-driven API by Infermedica, integrated on a trial basis (2,000 API calls or 60 days), for symptom analysis and diagnosis in Sehat Pal.*
- ***Realtime Database****: A NoSQL cloud database within Firebase, used to store and sync chat history in Sehat Pal in real time.*
- ***SRS****: Software Requirements Specification - This document, outlining the requirements and specifications for Sehat Pal.*
- ***TLS****: Transport Layer Security - A cryptographic protocol ensuring secure data transmission, employed in HTTPS for Sehat Pal.*
- ***UI****: User Interface - The visual and interactive elements of Sehat Pal, designed with Bootstrap 5 by Ibrahim Johar*

1.4 Document Convention

Arial 16 for Headings, Arial 12 for Sub-Headings and Arial 10 for Paragraphs

2. Overall System Description

2.1. Project Background

Sehat Pal is a healthcare technology initiative developed to enhance access to preliminary health assessments within the growing digital health sector. The project was undertaken by a two-person team: Syed Ukkashah (Backend developer) who focused on user authentication and data storage, and Ibrahim Johar (Frontend developer), who designed the UI/UX for the website.

The project was triggered by the significant problem of limited access to timely healthcare consultations, particularly in underserved or remote areas where medical facilities and professionals are scarce. This lack of access often results in delayed diagnoses and untreated health conditions, exacerbating patient outcomes. Sehat Pal capitalizes on the opportunity to provide an AI-driven platform that supports early health assessments, empowering users to take proactive steps toward seeking professional care and reducing delays in the healthcare process.

2.2. Project Scope

Sehat Pal is a web-based healthcare platform developed as a university project to provide an efficient, digital, and portable symptom checker for users in third-world countries like Pakistan, particularly those unable to access healthcare facilities due to travel constraints, and anyone seeking accessible health assessments. The platform was built by Syed Ukkashah (backend developer), who implemented Firebase for user management and data storage and integrated the Infermedica API for client-side symptom diagnosis, and Ibrahim Johar (frontend developer), who designed the UI/UX, custom theme, and logo using HTML, CSS, and JavaScript. The system operates on standard web browsers, supporting desktop and mobile devices, and is designed for a small user base of 5-10 users due to the trial limitations of the Infermedica API.

The main functionalities addressed in the system include:

- *User authentication, enabling secure signup and login via Firebase Authentication.*
- *A symptom checker chatbot, powered solely by the Infermedica API (trial basis, limited to 2,000 API calls or 60 days), that analyzes user-reported symptoms to provide disease diagnoses with percentage-based probabilities.*
- *Secure storage of medical records, implemented using Firebase Firestore.*
- *Retention of chat history across user sessions, managed through Firebase Realtime Database.*
- *Static pages, including an About Us page with branding elements, a Contact Us page for user inquiries, and a Call an Ambulance page that redirects to an external emergency helpline via a simple URL link, all developed by Ibrahim Johar.*

***Project Boundaries**:*

- *The system is a web-based application and does not include a dedicated mobile app.*
- *It is limited to a small-scale deployment (5-10 users) due to the university project context and the Infermedica API's trial constraints (2,000 API calls or 60 days, whichever expires first).*
- *The platform focuses on preliminary health assessments and is not a substitute for professional medical services.*

2.3. Not In Scope

The following functionalities are explicitly excluded from the current project:

- *Development of a dedicated mobile application.*
- *Real-time consultations with healthcare professionals.*
- *Advanced diagnostic capabilities beyond the Infermedica API's trial functionality.*
- *Integration with wearable devices or other external hardware.*
- *Support for large-scale user bases or high traffic volumes, as the system is constrained by the university project scope and the Infermedica API trial limits (2,000 API calls or 60 days).*
- *Additional static pages beyond About Us, Contact Us, and Call an Ambulance.*

2.4. Project Objectives

Sehat Pal aims to address the problem of limited access to timely healthcare consultations in underserved areas of Pakistan, where travel constraints prevent many individuals from visiting doctors. By leveraging the opportunity to provide an AI-driven, portable symptom checker, the project seeks to empower users with preliminary health assessments. The specific objectives include:

- *Deliver accurate symptom-based disease diagnoses using the Infermedica API (trial basis) to support early health awareness for 5-10 users.*
- *Provide secure storage and management of medical data through Firebase Firestore, ensuring user privacy within the university project's scope.*
- *Offer an accessible and intuitive web interface, to enable users with basic digital literacy to navigate the platform effectively.*
- *Facilitate emergency access through a simple URL redirect to an external helpline, addressing critical health needs.*

.

2.5. Stakeholders

The Sehat Pal platform, involves various stakeholders who interact with or contribute to the system's development, deployment, and usage. The system will be hosted on Firebase or Netlify, and once deployed, it will be reviewed by the software engineering course instructor. Below are the identified business user classes and technical personnel involved:

- ***Business User Classes**:*
    - ***End-Users**: Individuals in Pakistan, particularly those with limited access to healthcare due to travel constraints, and anyone seeking an efficient, digital, and portable symptom checker. These users will interact with the system to perform preliminary health assessments, manage medical records, and access emergency resources. The system is designed for a small user base (5-10 users) due to the trial limitations of the Infermedica API, but any user fitting the target audience can use it post-deployment.*
    - ***Emergency Service Providers**: External organizations linked via the Call an Ambulance page's URL redirect, providing emergency helpline services to users in critical need.*

- ***Technical Personnel****:*
    - ***Syed Ukkashah (Backend Developer)****: Responsible for developing and testing the backend, including Firebase Authentication for user management, Firebase Firestore for medical record storage, Firebase Realtime Database for chat history retention, and integration of the Infermedica API for client-side symptom diagnosis.*
    - ***Ibrahim Johar (Frontend Developer)****: Responsible for developing and testing the frontend, including the UI/UX, custom theme, and logo using HTML, CSS, and JavaScript, as well as static pages (About Us, Contact Us, and Call an Ambulance).*
    - ***Miss Fizza Mansoor (Course Instructor)****: The software engineering course instructor who will review the deployed system to evaluate its functionality, design, and adherence to project requirements.*
    - ***Healthcare Advisors (Potential)****: Medical professionals who may be consulted to validate the accuracy of the Infermedica API's symptom-disease mappings, ensuring the system's reliability for preliminary assessments.*

2.6. Operating Environment

The system is hosted on Firebase Hosting and designed for a small user base of 5-10 users due to the trial constraints of the Infermedica API (2,000 API calls or 60 days). Below is a description of the environment in which the software will operate:

- ***Hardware Platform****:*
    - ***User Devices****: The platform is accessible on standard user devices, including PCs, smartphones, and tablets with internet access, suitable for the target audience in Pakistan. No specific hardware requirements are imposed beyond devices capable of running modern web browsers.*
    - ***Server Infrastructure****: The system is hosted on cloud-based servers provided by Firebase Hosting, which manages the backend and static content delivery. No local server hardware is required.*
- ***Operating System****:*
    - *As a web-based application, Sehat Pal is compatible with any operating system supporting modern web browsers, including Windows, macOS, Linux, Android, and iOS. Supported browsers include Google Chrome, Mozilla Firefox, and Safari (latest or near-latest versions recommended).*
- ***Network Environment****:*
    - *The system requires an active internet connection for users to access the platform and interact with its services. It is designed to be lightweight to accommodate variable network conditions in Pakistan, ensuring accessibility even in areas with moderate connectivity.*
    - *All communications between the client, Firebase, and the Infermedica API use HTTPS to ensure secure data transfer. The system employs efficient data handling to minimize latency.*
- ***Software Components and Applications****:*
    - ***Firebase Services*** *(managed by Syed Ukkashah):*
        - ***Firebase Authentication****: Facilitates secure user signup and login.*
        - ***Firebase Firestore****: Enables secure storage and retrieval of medical records.*

        - ***Firebase Realtime Database****: Supports retention and retrieval of chat history.*
        - ***Firebase Hosting****: Hosts the web application, serving static content and integrating with backend services.*
    - ***Infermedica API*** *(integrated by Syed Ukkashah): A trial-based API (limited to 2,000 API calls or 60 days) used for client-side symptom analysis and disease diagnosis, accessed via RESTful HTTPS requests.*
    - ***Frontend*** *(developed by Ibrahim Johar): Built with HTML, CSS, JavaScript, and Bootstrap CSS, delivering a responsive UI/UX with a custom theme and logo, compatible with standard web browsers.*
    - ***External Emergency Helpline****: An external website linked via a simple URL redirect on the Call an Ambulance page, requiring no direct integration beyond browser-based navigation.*
    - ***Coexisting Applications****: Sehat Pal operates independently within standard browser environments and does not require coexistence with specific user-installed applications. It relies solely on browser support for HTML5, CSS3, JavaScript, and Bootstrap CSS.*

The operating environment is tailored to the university project's scope, ensuring accessibility for a small user base in Pakistan while leveraging Firebase Hosting and AI-driven diagnostics within the constraints of the Infermedica API trial.

2.7. System Constraints

Sehat Pal, a web-based symptom checker developed as a university project by Syed Ukkashah (backend developer) and Ibrahim Johar (frontend developer), operates under several constraints imposed by the external environment, including stakeholders, business conditions, technical issues, and academic requirements. The system targets a small user base (5-10 users), particularly those with limited healthcare access, and is hosted on Firebase Hosting with a trial-based Infermedica API. Below are the constraints affecting the system:

- ***Software Constraints****:*
    - *The system relies on Firebase services (Authentication, Firestore, Realtime Database, Hosting), which are subject to free-tier limitations, such as restricted database reads/writes, storage capacity, and bandwidth, limiting scalability.*
    - *The Infermedica API, used for symptom diagnosis, is on a trial basis (2,000 API calls or 60 days, whichever expires first), constraining the number of users and duration of active use.*
    - *The frontend uses Bootstrap CSS, which may limit design flexibility compared to fully custom CSS, as it imposes predefined styling conventions.*
- ***Hardware Constraints****:*
    - *The system must support low-to-mid-range devices (PCs, smartphones, tablets) commonly used in Pakistan, with no specialized hardware requirements. This ensures accessibility but restricts performance expectations to standard consumer hardware.*
- ***Cultural Constraints****:*

- *The interface must support English as the primary language and potentially Urdu, the national language of Pakistan, to ensure accessibility for the target audience with varying linguistic backgrounds.*
- *The design must be culturally sensitive, avoiding imagery, terminology, or content that may be inappropriate or misunderstood in the Pakistani cultural context.*

- ***Legal Constraints**:*
  - *The system must comply with applicable data protection regulations, such as Pakistan's Data Protection Bill (if enacted) or general privacy standards, to safeguard medical records and user data stored in Firebase Firestore.*
  - *Adherence to the Infermedica API's terms of service is required, particularly for trial usage, limiting modifications or extensions of its functionality.*
- ***Environmental Constraints**:*
  - *The system must function in variable network conditions, such as intermittent or low-bandwidth internet common in rural Pakistan, requiring efficient data handling to maintain accessibility.*
  - *As a text-based platform with no audio features, the system is unaffected by noisy environments and includes no sound events, aligning with diverse usage settings.*
- ***User Constraints**:*
  - *The system is designed for users with basic digital literacy, necessitating a simple and intuitive interface with clear textual and graphical controls, facilitated by Bootstrap CSS and the Infermedica API's user-friendly question flow.*
  - *The interface must accommodate users unfamiliar with medical terminology, relying on the Infermedica API to present symptoms and diagnoses in accessible language.*
- ***Off-the-Shelf Component Constraints**:*
  - ***Firebase Free-Tier**: Limits on database operations, storage, and bandwidth restrict the system's ability to scale beyond the university project's small user base (5-10 users).*
  - ***Infermedica API Trial**: The 2,000 API call or 60-day limit restricts the number of symptom checker interactions and the project's operational lifespan without additional licensing.*
  - ***Bootstrap CSS**: Predefined styles may constrain unique design elements, requiring additional effort for customizations beyond the framework's capabilities.*
- ***Academic Constraints**:*
  - *The system must meet the evaluation criteria set by the software engineering course instructor, Miss Fizza Mansoor, focusing on functionality, usability, and adherence to academic software engineering principles.*
  - *The project is constrained by the university timeline and resources, limiting its scope to a prototype for 5-10 users rather than a production-ready application.*

These constraints shape the development and deployment of Sehat Pal, ensuring it meets the needs of its target audience within the boundaries of the university project and available technologies.

2.8. Assumptions & Dependencies

- ***Assumptions:***
    - ***User Capabilities****: Users in Pakistan, particularly those with limited healthcare access, have basic digital literacy and access to low-to-mid-range devices (PCs, smartphones, tablets) with modern web browsers (e.g., Chrome, Firefox, Safari) to interact with the platform.*
    - ***Infermedica API Reliability****: The trial-based Infermedica API provides accurate and reliable symptom-disease mappings, validated by its internal medical database, sufficient for preliminary health assessments during the project's duration.*
    - ***Network Availability****: Users have sufficient internet access, despite variable network conditions in Pakistan (e.g., intermittent or low-bandwidth connections), to access the web-based platform without requiring extensive offline functionality.*
    - ***Academic Evaluation****: The system meets the evaluation criteria of the software engineering course instructor, Miss Fizza Mansoor, without requiring additional features or integrations beyond the current scope.*
    - ***Deployment Stability****: Firebase Hosting provides reliable uptime and performance for the project's duration, ensuring the platform remains accessible for the small user base.*
- ***Dependencies:***
    - ***Firebase Services****: The system relies on Firebase for core functionality, including Firebase Authentication for secure user signup and login, Firebase Firestore for medical record storage, Firebase Realtime Database for chat history retention, and Firebase Hosting for deploying the web application.*
    - ***Infermedica API****: The symptom checker chatbot depends entirely on the trial-based Infermedica API (limited to 2,000 API calls or 60 days), which handles client-side symptom analysis and disease diagnosis via RESTful HTTPS requests.*
    - ***Bootstrap CSS****: The frontend, developed by Ibrahim Johar, depends on Bootstrap CSS for responsive styling, impacting the UI/UX design and requiring compatibility with its framework.*
    - ***External Emergency Helpline****: The Call an Ambulance page relies on the availability of an external emergency helpline website, accessible via a simple URL redirect, which must remain operational for the redirect to function.*
    - ***Internet Connectivity****: The system requires a stable internet connection for users to access Firebase services, interact with the Infermedica API, and load the web-based interface.*
    - ***Browser Compatibility****: The system depends on modern web browsers supporting HTML5, CSS3, JavaScript, and Bootstrap CSS to render the frontend correctly and ensure a consistent user experience.*

These assumptions and dependencies define the operational boundaries of Sehat Pal, ensuring it meets the needs of its target audience within the constraints of the university project and available technologies.

3. External Interface Requirements

[This section is intended to specify any requirements that ensure that the new system will connect properly to external components. Place a context diagram showing the external interfaces at a high level of abstraction.]

3.1. Hardware Interfaces

Below are the characteristics of each interface between the software and hardware components, including supported device types and the nature of data and control interactions.

- ***User Devices**:*
  - ***Supported Device Types**: The system is accessible on low-to-mid-range PCs, smartphones, and tablets commonly used in Pakistan, ensuring affordability and accessibility for the target audience. Devices must support modern web browsers (e.g., Google Chrome, Mozilla Firefox, Safari, latest or near-latest versions) capable of rendering HTML5, CSS3, JavaScript, and Bootstrap CSS (used by Ibrahim Johar for responsive UI/UX).*
  - ***Data Interactions**:*
    - ***Input**: Users interact with the software through standard input mechanisms, including keyboards (physical or on-screen) for entering symptom data and login credentials, touchscreens for navigating the interface (e.g., selecting options in the chatbot), and mice for clicking buttons or links (e.g., About Us, Contact Us, Call an Ambulance pages). Input data is processed by the browser and sent via HTTPS to Firebase or the Infermedica API.*
    - ***Output**: The software delivers visual output to device displays, rendering the web interface (developed by Ibrahim Johar), chatbot responses (powered by the Infermedica API), medical records (retrieved from Firebase Firestore), and static pages. Output is formatted using Bootstrap CSS for responsive design, ensuring compatibility across varying screen sizes. No audio, haptic, or other sensory outputs are involved, as the system is text-based.*
  - ***Control Interactions**: The software controls the browser environment to process user inputs, render dynamic content, and initiate HTTPS requests to Firebase services (Authentication, Firestore, Realtime Database) and the Infermedica API (trial basis: 2,000 API calls or 60 days). Control is limited to standard browser-based events, such as form submissions, button clicks, and page navigation (e.g., URL redirect to the emergency helpline website). No low-level hardware control (e.g., sensors, cameras, or peripherals) is required, ensuring compatibility with standard consumer devices.*
  - ***Characteristics**: The interface is lightweight, requiring minimal processing power and memory to support low-to-mid-range devices. The system does not impose specific hardware performance thresholds (e.g., CPU, RAM) beyond those needed to run modern browsers, making it accessible to the target audience in Pakistan.*
- ***Server Infrastructure**:*
  - ***Supported Device Types**: Sehat Pal is hosted on Firebase Hosting's cloud-based servers, which are managed externally and require no direct interaction with local server hardware. The system does not interface with physical server components, as Firebase Hosting abstracts server management.*

- ***Data Interactions****: No direct data exchange occurs between the software and server hardware, as Firebase Hosting handles static content delivery (HTML, CSS, JavaScript, Bootstrap CSS) and connects to Firebase services via cloud APIs. Data interactions are managed at the software level (see Software Interfaces).*
- ***Control Interactions****: The software has no control over server hardware, relying on Firebase Hosting's infrastructure for deployment and scalability. Control is limited to configuring Firebase Hosting settings (e.g., domain, content delivery), managed by Syed Ukkashah during deployment.*
- ***Characteristics****: The cloud-based hosting ensures high availability and scalability within Firebase's free-tier limits, suitable for the university project's small user base (5-10 users). No hardware-specific configurations are required.*

These hardware interfaces ensure Sehat Pal operates seamlessly on standard user devices in Pakistan, leveraging browser-based interactions to deliver its functionality within the constraints of the university project and the Infermedica API trial.

3.2. Software Interfaces

Sehat Pal, a web-based symptom checker developed as a university project by Syed Ukkashah (backend developer) and Ibrahim Johar (frontend developer), connects to external software components to deliver its functionality to a small user base (5-10 users) in Pakistan. The system, hosted on Firebase Hosting, integrates with Firebase services, the Infermedica API (trial basis: 2,000 API calls or 60 days), Bootstrap 5, Axios, and an external emergency helpline website. This subsection describes the connections, including component names and versions, the purpose of data items or messages exchanged, services needed, the nature of inter-component communications, and shared data.

- ***Firebase Authentication (Latest Stable Version)****:*
  - ***Connection****: Managed by Syed Ukkashah, connects the Sehat Pal backend to Firebase's cloud-based authentication service via HTTPS.*
  - ***Purpose of Data Items/Messages****: Exchanges user credentials (email, password) and session tokens to authenticate users for signup and login. Returns authentication status (success/failure) and user IDs.*
  - ***Services Needed****: Secure user authentication and session management.*
  - ***Inter-Component Communications****: RESTful HTTPS requests initiated by the client (browser) to Firebase's authentication endpoints, using Firebase SDK. Responses are JSON-formatted, containing user IDs and tokens.*
  - ***Shared Data****: User IDs are shared with Firebase Firestore and Realtime Database to link user accounts with medical records and chat history.*
- ***Firebase Firestore (Latest Stable Version)****:*
  - ***Connection****: Managed by Syed Ukkashah, connects the backend to Firebase's NoSQL cloud database via HTTPS for storing and retrieving medical records.*
  - ***Purpose of Data Items/Messages****: Stores encrypted medical records, including user IDs, record metadata (e.g., file name, upload date), and file contents (e.g., PDFs, images). Retrieves records based on user queries, returning record data to the client.*
  - ***Services Needed****: Secure, scalable storage and retrieval of medical data.*
  - ***Inter-Component Communications****: RESTful HTTPS requests via Firebase SDK, with queries to create, read, update, or delete records. Data is exchanged in JSON format, secured by Firebase security rules.*

- ***Shared Data**: User IDs link records to authenticated users, shared with Firebase Authentication and Realtime Database.*

- ***Firebase Realtime Database (Latest Stable Version)**:*
  - ***Connection**: Managed by Syed Ukkashah, connects the backend to Firebase's real-time NoSQL database via HTTPS for managing chat history.*
  - ***Purpose of Data Items/Messages**: Stores chat logs, including user IDs, timestamps, symptom inputs, and Infermedica API responses (diagnoses). Retrieves chat history for display upon user login.*
  - ***Services Needed**: Real-time storage and retrieval of chat session data.*
  - ***Inter-Component Communications**: WebSocket and HTTPS requests via Firebase SDK, enabling real-time updates for chat logs. Data is exchanged in JSON format, secured by Firebase security rules.*
  - ***Shared Data**: User IDs link chat history to authenticated users, shared with Firebase Authentication and Firestore.*

- ***Firebase Hosting (Latest Stable Version)**:*
  - ***Connection**: Managed by Syed Ukkashah, connects the system to Firebase's cloud hosting service to serve the web application.*
  - ***Purpose of Data Items/Messages**: Serves static content (HTML, CSS, JavaScript, Bootstrap 5, Axios) developed by Ibrahim Johar to the client's browser. No dynamic data exchange occurs, only content delivery.*
  - ***Services Needed**: High-availability hosting for static assets and integration with Firebase backend services.*
  - ***Inter-Component Communications**: HTTPS requests from the client to retrieve static files from Firebase Hosting's content delivery network (CDN). Responses are standard HTTP content (e.g., HTML, CSS).*
  - ***Shared Data**: None, as Firebase Hosting serves static content linked to backend services via user IDs.*

- ***Infermedica API (Latest Trial Version, 2,000 API Calls or 60 Days)**:*
  - ***Connection**: Integrated by Syed Ukkashah, connects the client-side application to the Infermedica API's cloud service via HTTPS for symptom diagnosis, using Axios for API calls.*
  - ***Purpose of Data Items/Messages**: Sends symptom inputs (e.g., user-reported symptoms, responses to follow-up questions) and receives disease diagnoses with percentage-based probabilities. Messages are JSON-formatted, containing symptom IDs, user demographics (if provided), and diagnosis results.*
  - ***Services Needed**: AI-driven symptom analysis and disease prediction, limited to 2,000 API calls or 60 days under the trial.*
  - ***Inter-Component Communications**: RESTful HTTPS requests initiated by the client (browser) using Axios, including an API key for authentication. Responses are JSON-formatted diagnosis data, processed client-side and stored in Firebase Realtime Database.*
  - ***Shared Data**: Symptom inputs and diagnoses are stored in Firebase Realtime Database as chat history, linked by user IDs.*

- ***Bootstrap 5 (Version 5)**:*

- ***Connection****: Used by Ibrahim Johar in the frontend, integrated as a client-side library to style the web interface.*
- ***Purpose of Data Items/Messages****: No data exchange; Bootstrap 5 provides predefined styles and responsive layouts for HTML elements, rendering the UI/UX in the browser.*
- ***Services Needed****: Responsive and consistent styling across devices (PCs, smartphones, tablets).*
- ***Inter-Component Communications****: None, as Bootstrap 5 is a static library loaded by the browser via Firebase Hosting. It interacts with HTML/JavaScript for rendering, with no external communication.*
- ***Shared Data****: None, as Bootstrap 5 affects presentation only, with no data shared with other components.*

- ***Axios (Latest Stable Version)****:*
  - ***Connection****: Used by Ibrahim Johar in the frontend, integrated as a client-side JavaScript library to facilitate HTTP requests to the Infermedica API.*
  - ***Purpose of Data Items/Messages****: Sends JSON-formatted symptom inputs to the Infermedica API and receives JSON-formatted diagnosis responses. Handles HTTP request/response lifecycle for client-side API calls.*
  - ***Services Needed****: Simplified and reliable HTTP request handling for API interactions.*
  - ***Inter-Component Communications****: Initiates RESTful HTTPS requests to the Infermedica API, configured with API keys and headers. Responses are processed client-side and passed to the UI or Firebase Realtime Database.*
  - ***Shared Data****: Symptom inputs and diagnoses are shared with Firebase Realtime Database (chat history), linked by user IDs.*
- ***Emergency Helpline Website (External, Version Unknown)****:*
  - ***Connection****: Linked via a URL redirect from the Call an Ambulance page, developed by Ibrahim Johar, with no direct integration.*
  - ***Purpose of Data Items/Messages****: No data or messages are exchanged; the browser navigates to the external website's URL when the user clicks the redirect link.*
  - ***Services Needed****: Availability of the external helpline website to provide emergency services.*
  - ***Inter-Component Communications****: Standard HTTP redirect initiated by the browser, with no data transfer or API interaction. The redirect is stateless and relies on the external site's availability.*
  - ***Shared Data****: None, as the redirect involves no data exchange with Sehat Pal.*
- ***Operating Systems****:*
  - ***Connection****: Sehat Pal runs within modern web browsers (e.g., Chrome, Firefox, Safari) on any operating system (Windows, macOS, Linux, Android, iOS).*
  - ***Purpose of Data Items/Messages****: Limited to standard browser-OS interactions, such as file uploads for medical records (e.g., PDFs, images) via browser file dialogs, which may prompt OS-level permissions.*
  - ***Services Needed****: Browser compatibility and basic file system access for uploads.*

    - ***Inter-Component Communications****: Handled by the browser, with no direct OS integration. File uploads use standard HTML input elements, processed client-side and sent to Firebase Firestore via Axios or Firebase SDK.*
    - ***Shared Data****: File data (medical records) is shared with Firebase Firestore after upload, linked by user IDs.*
- ***Development Tools****:*
    - ***Git (Latest Stable Version)****: Used for version control, managing source code by Syed Ukkashah and Ibrahim Johar.*
    - ***Visual Studio Code (Latest Stable Version)****: Primary code editor for development, supporting frontend and backend tasks.*
    - ***Browser Dev Tools****: Built-in browser tools (e.g., Chrome DevTools) for debugging frontend UI/UX and API interactions.*
    - ***GitHub (Latest Stable Version)****: Hosts the project repository, facilitating collaboration and code management.*
    - ***Firebase CLI (Latest Stable Version)****: Used by Syed Ukkashah for deploying the application to Firebase Hosting and configuring Firebase services.*
    - ***Purpose****: Support development, debugging, version control, and deployment.*
    - ***Communications****: Non-runtime, involving command-line interactions (Firebase CLI, Git) or web-based interfaces (GitHub). No direct impact on system operation.*
    - ***Shared Data****: None, as tools are development-only.*

Services Needed Summary

- ***Authentication****: Secure user management (Firebase Authentication).*
- ***Database****: Secure storage for medical records (Firestore) and chat history (Realtime Database).*
- ***Hosting****: Delivery of static content (Firebase Hosting).*
- ***Symptom Diagnosis****: AI-driven analysis (Infermedica API).*
- ***Styling****: Responsive UI/UX (Bootstrap 5).*
- ***HTTP Requests****: Client-side API calls (Axios).*
- ***Emergency Access****: Navigation to external helpline (URL redirect).*
- ***Development****: Code management, editing, debugging, and deployment (Git, VSCode, Browser Dev Tools, GitHub, Firebase CLI).*
- ***Shared Data Summary***
- ***User IDs****: Link Firebase Authentication, Firestore, and Realtime Database to associate users with their records and chat history.*
- ***Medical Records****: Stored in Firestore, accessible to authenticated users via the frontend.*
- ***Chat History****: Stored in Realtime Database, including symptom inputs and Infermedica API diagnoses, linked by user IDs.*
- ***Static Content****: Served by Firebase Hosting, rendered with Bootstrap 5 in the browser.*
- ***API Data****: Symptom inputs and diagnoses exchanged via Axios with Infermedica API, stored in Realtime Database.*

These software interfaces ensure Sehat Pal integrates seamlessly with external components, supporting its functionality within the university project's scope and the constraints of the Infermedica API trial.

3.3. Communications Interfaces

Sehat Pal relies on specific communication functions to enable its functionality for a small user base (5-10 users) in Pakistan. The system, hosted on Firebase Hosting, uses web browsers for user interaction, HTTPS for secure network communications with Firebase services and the Infermedica API, HTTP for an emergency helpline redirect, electronic forms for user input, and email for Firebase Authentication during signup and password reset. This subsection details the requirements for these communication functions, including message formatting, security and encryption, data transfer rates, and synchronization mechanisms, within the constraints of the Infermedica API trial (2,000 API calls or 60 days).

- ***Web Browser**:*
  - ***Requirement**: Users access Sehat Pal through modern web browsers (e.g., Google Chrome, Mozilla Firefox, Safari, latest or near-latest versions) on devices (PCs, smartphones, tablets) running any operating system (Windows, macOS, Linux, Android, iOS).*
  - ***Function**: Renders the frontend interface (HTML, CSS, JavaScript, Bootstrap 5) developed by Ibrahim Johar, processes user inputs (e.g., form submissions, clicks), and initiates communications with Firebase services and the Infermedica API via Axios.*
  - ***Message Formatting**: Browser handles HTML for UI rendering, JavaScript for dynamic interactions, and JSON for API requests/responses (via Axios for Infermedica API calls). No specific browser-level formatting is required beyond standard web protocols.*
  - ***Security/Encryption**: HTTPS ensures secure communication between the browser and Firebase/Infermedica API, with TLS 1.2 or higher enforced for all requests.*
  - ***Data Transfer Rates**: Supports low-latency page loads (<3 seconds) and API responses (<2 seconds for Infermedica API, <1 second for Firebase queries), suitable for a small user base. Lightweight design accommodates Pakistan's variable network conditions.*
  - ***Synchronization**: No browser-level synchronization; handled by Firebase Realtime Database for chat history updates.*
- ***Network Communications (HTTPS)**:*
  - ***Requirement**: The system uses HTTPS for all communications between the client (browser) and external components, including Firebase services (Authentication, Firestore, Realtime Database, Hosting) and the Infermedica API.*
  - ***Function**:*
    - ***Firebase Communications**: Managed by Syed Ukkashah, handles user authentication (credentials, tokens, verification emails), medical record storage/retrieval (record data), chat history updates (chat logs), and static content delivery (HTML, CSS, JavaScript).*
    - ***Infermedica API Communications**: Managed by Syed Ukkashah, sends symptom inputs and receives diagnoses via client-side Axios HTTPS requests, integrated by Ibrahim Johar in the frontend.*

    - ***Message Formatting****: JSON-formatted messages for all HTTPS requests/responses:*
        - ***Firebase Authentication****: Credentials (email, password), tokens, user IDs, verification email triggers.*
        - ***Firebase Firestore****: Medical record data (user IDs, metadata, file contents, e.g., PDFs, images).*
        - ***Firebase Realtime Database****: Chat logs (user IDs, timestamps, symptom inputs, diagnoses).*
        - ***Infermedica API****: Symptom inputs (symptom IDs, user responses), diagnosis outputs (disease names, probabilities).*
    - ***Security/Encryption****: HTTPS with TLS 1.2 or higher ensures secure data transfer. Firebase security rules restrict data access to authenticated users, with password security (secure hashing, e.g., bcrypt) for Authentication. Infermedica API requests include an API key for authentication.*
    - ***Data Transfer Rates****: Low-latency interactions (<2 seconds for Infermedica API responses, <1 second for Firebase queries), optimized for 5-10 users. Data exchange is lightweight to support variable network conditions in Pakistan.*
    - ***Synchronization****: Firebase Realtime Database uses WebSocket for real-time chat history updates. Other Firebase services and Infermedica API operate on a request-response model, requiring no additional synchronization.*
- ***Network Communications (HTTP)****:*
    - ***Requirement****: The system uses a standard HTTP redirect for the Call an Ambulance page to navigate to an external emergency helpline website.*
    - ***Function****: Developed by Ibrahim Johar, the redirect allows users to access emergency services by clicking a link, with no data exchange.*
    - ***Message Formatting****: No message formatting; the redirect uses a simple URL (e.g., http://emergency-helpline.example.com) loaded by the browser.*
    - ***Security/Encryption****: HTTP redirect is unencrypted, as no sensitive data is exchanged. The external website's security is outside Sehat Pal's control.*
    - ***Data Transfer Rates****: Instantaneous redirect (<1 second), dependent on browser and network speed, with no significant data transfer.*
    - ***Synchronization****: None, as the redirect is stateless and does not involve data persistence.*
- ***Electronic Forms****:*
    - ***Requirement****: The system uses HTML forms for user input, including signup/login (with email verification), symptom input, and medical record uploads, processed client-side and sent to Firebase or Infermedica API.*
    - ***Function****:*
        - ***Signup/Login Forms****: Collect email and password, sent to Firebase Authentication via HTTPS, triggering verification emails for signup.*
        - ***Symptom Input Forms****: Collect user responses to chatbot questions, sent to Infermedica API via Axios HTTPS requests.*
        - ***Medical Record Upload Forms****: Allow file uploads (e.g., PDFs, images, <10 MB assumed), sent to Firebase Firestore via HTTPS.*

    - ***Message Formatting****: Forms use standard HTML form data (key-value pairs) or JSON (for API submissions via Axios). Client-side validation (e.g., email format, file size limits) is implemented using JavaScript and Bootstrap 5 classes for styling.*
    - ***Security/Encryption****: HTTPS ensures secure transmission of form data. Firebase security rules validate data writes, with password security for Authentication. File uploads are encrypted in Firestore.*
    - ***Data Transfer Rates****: Form submissions are processed quickly (<1 second for Firebase, <2 seconds for Infermedica API), with file uploads dependent on file size and network speed (<10 MB assumed).*
    - ***Synchronization****: No synchronization required; form submissions are processed as individual requests, with results stored in Firebase Firestore or Realtime Database.*
- ***Email****:*
    - ***Requirement****: Firebase Authentication, integrated by Syed Ukkashah, sends emails for signup verification and password reset, required for user account management.*
    - ***Function****: Sends automated emails to users during signup (to verify email addresses) and upon password reset requests (to provide recovery links), triggered via the signup/login interface.*
    - ***Message Formatting****: Standard email format (HTML or plain text) generated by Firebase, containing verification or password reset links (e.g., https://sehatpal.firebaseapp.com/verify?token=xyz). No custom formatting required.*
    - ***Security/Encryption****: Emails are sent over secure SMTP via Firebase's infrastructure, with links using HTTPS for verification/reset actions. Email content avoids sensitive data beyond tokens.*
    - ***Data Transfer Rates****: Email delivery is near-instantaneous (<5 seconds), dependent on Firebase's email service and the user's email provider.*
    - ***Synchronization****: None, as emails are asynchronous and stateless.*
- ***Security and Encryption Summary***
- ***HTTPS****: All network communications (Firebase, Infermedica API) use TLS 1.2 or higher for encryption, ensuring secure data transfer.*
- ***Firebase Security****: Security rules restrict data access in Firestore and Realtime Database to authenticated users. Password security (e.g., bcrypt hashing) protects user credentials in Authentication.*
- ***Infermedica API Key****: Authenticates API requests, ensuring secure access to trial-based services.*
- ***Form Security****: Client-side validation and HTTPS prevent injection attacks or data leaks.*
- ***Email Security****: Firebase's secure SMTP and HTTPS links protect verification and password reset functionality.*
- ***HTTP Redirect****: Unencrypted but involves no sensitive data, as it's a simple URL navigation.*
- ***Data Transfer Rates Summary***

- ***Browser Rendering****: Page loads in <3 seconds, optimized for variable networks in Pakistan.*
- ***Firebase Queries****: Authentication, Firestore, and Realtime Database responses in <1 second.*
- ***Infermedica API****: API responses in <2 seconds, dependent on trial service performance.*
- ***Form Submissions****: Processed in <1-2 seconds, with file uploads varying by size (<10 MB assumed).*
- ***HTTP Redirect****: Instantaneous (<1 second), browser-dependent.*
- ***Email****: Delivery in <5 seconds.*
- ***Synchronization Mechanisms Summary***
- ***Firebase Realtime Database****: Uses WebSocket for real-time updates of chat history, ensuring users see the latest logs upon login.*
- ***Other Components****: Operate on a request-response model (Firebase Authentication, Firestore, Infermedica API, email), requiring no additional synchronization.*
- ***Forms and Redirects****: Stateless, with no synchronization needs.*

These communications interfaces ensure Sehat Pal operates reliably within the university project's scope, supporting secure and efficient interactions for its target audience in Pakistan while adhering to the Infermedica API trial constraints and Firebase's security requirements.

4. Functional Requirements

4.1. Functional Hierarchy

- ***User Management***
    - ***Description****: Manages user accounts and secure access to Sehat Pal, ensuring only authenticated users can interact with protected features (e.g., symptom checker, medical records).*
    - ***Sub-Functions****:*
        - ***Signup****: Allows users to create an account by providing an email and password, with email verification via a Firebase Authentication-generated link to confirm ownership.*
        - ***Login****: Authenticates users with email and password, granting access to personalized features using Firebase Authentication tokens.*
        - ***Password Reset****: Provides a secure password recovery option, sending a reset link via email through Firebase Authentication.*
        - ***Session Management****: Maintains user sessions with Firebase Authentication tokens, automatically expiring after inactivity (e.g., 1 hour) to ensure security.*
- ***Symptom Checker***
    - ***Description****: Enables users to input symptoms and receive AI-driven preliminary health assessments, powered by the Infermedica API, with results stored for future reference.*
    - ***Sub-Functions****:*
        - ***Symptom Input****: Collects user-reported symptoms and responses to follow-up questions via interactive HTML forms, styled with Bootstrap 5 and sent to the Infermedica API using Axios.*
        - ***AI-Driven Diagnosis****: Processes symptom inputs through the Infermedica API (trial version) to generate disease diagnoses with percentage-based probabilities, displayed in the UI.*
        - ***Chat History Storage and Retrieval****: Stores symptom inputs and diagnosis results as chat logs in Firebase Realtime Database, linked to user IDs, and retrieves them for display upon login.*
- ***Medical Records***
    - ***Description****: Allows users to upload, store, and retrieve medical records securely, supporting personal health management within the prototype's scope.*
    - ***Sub-Functions****:*
        - ***File Upload****: Enables users to upload medical files (e.g., PDFs, images, <10 MB) via a form, validated client-side (JavaScript, Bootstrap 5) and stored in Firebase Firestore.*
        - ***Secure Storage****: Stores uploaded files in Firebase Firestore, encrypted and linked to user IDs, with access restricted by Firebase security rules.*
        - ***Record Retrieval and Display****: Retrieves user-specific records from Firestore and displays them in the UI, with options to view or download files.*

- ***Emergency Access***
    - ***Description****: Provides quick access to emergency services through a redirect to an external helpline, ensuring users can seek urgent help when needed.*
    - ***Sub-Functions****:*
        - ***URL Redirect****: Redirects users to an external emergency helpline website via a simple HTTP link on the Call an Ambulance page, implemented by Ibrahim Johar.*
        - ***Emergency Instructions****: Displays clear instructions and disclaimers in the UI, emphasizing the redirect's use for urgent situations only.*
- ***Information Pages***
    - ***Description****: Offers static content to inform users about the system, provide support, and ensure transparency, enhancing usability and trust.*
    - ***Sub-Functions****:*
        - ***About Us Page****: Presents details about Sehat Pal, its purpose (preliminary health assessments), and disclaimers clarifying its non-professional status, styled with Bootstrap 5.*
        - ***Contact Us Page****: Provides contact information (e.g., developer email) for user inquiries, supporting academic evaluation and user feedback.*
        - ***Help Page****: Includes FAQs, usage instructions, and safety guidance, hosted on Firebase Hosting, to assist users with basic digital literacy.*

4.2. Use Cases

4.2.1. [Title of use case]

[Use Case Diagram]
[Use Case Description]

Use Case 1: Sign Up and Verify Email

Field	*Details*
Use case ID	*UC-01*
Actors	*New User, Firebase Authentication*
Preconditions	*- The user has a valid email address and internet access.* *- Firebase Authentication service is available.*
Main Flow	***Step 1**: The user navigates to the signup page via the Sehat Pal homepage.* ***Step 2**: The user enters their email and password in the signup form, styled with Bootstrap 5.* ***Step 3**: The system validates the input (e.g., email format, password length ≥ 6 characters).* ***Step 4**: The user submits the form, triggering a Firebase Authentication request.* ***Step 5**: Firebase Authentication creates the user account and sends a verification email.* ***Step 6**: The user receives the email and clicks the verification link, confirming their email address.* ***Step 7**: The system updates the user's status to verified, allowing login.*
Postconditions	*- The user account is created and email-verified, enabling access to protected features.* *- The user can log in using their credentials.*
Exceptions	*- If the email is already in use, the system displays an error ("Email already registered").* *- If the verification email fails to send, the user is prompted to retry or contact support.*

Use Case 2: Log In

Field	*Details*
Use case ID	*UC-02*
Actors	*Registered User, Firebase Authentication*
Preconditions	*- The user has a verified email and password from signup.* *- Firebase Authentication service is available.*
Main Flow	***Step 1**: The user navigates to the login page via the Sehat Pal homepage.* ***Step 2**: The user enters their email and password in the login form.* ***Step 3**: The system validates the input and submits it to Firebase Authentication.* ***Step 4**: Firebase Authentication verifies the credentials and returns a session token.* ***Step 5**: The system grants access, redirecting the user to the homepage with personalized features (e.g., symptom checker, medical records).*

Postconditions	*- The user is logged in and can access protected features.* *- A session token is active, expiring after inactivity (e.g., 1 hour).*
Exceptions	*- If credentials are incorrect, the system displays an error ("Invalid email or password").* *- If the email is unverified, the system prompts the user to verify their email.*

Use Case 3: Perform Symptom Check

Field	*Details*
Use case ID	*UC-03*
Actors	*Logged-In User, Infermedica API, Firebase Realtime Database*
Preconditions	*- The user is logged in with a verified account.* *- The Infermedica API trial has remaining API calls (<2,000) and is within 60 days.*
Main Flow	***Step 1**: The user navigates to the Symptom Checker page.* ***Step 2**: The user enters initial symptoms (e.g., "fever, cough") via an interactive form, styled with Bootstrap 5.* ***Step 3**: The system sends the symptoms to the Infermedica API using Axios.* ***Step 4**: The Infermedica API returns follow-up questions, displayed dynamically in the form.* ***Step 5**: The user answers the questions, and the system submits responses to the API.* ***Step 6**: The Infermedica API returns a diagnosis (e.g., "Possible flu, 85% probability").* ***Step 7**: The system displays the diagnosis with a disclaimer ("Consult a doctor for professional advice").* ***Step 8**: The chat log (symptoms, questions, diagnosis) is stored in Firebase Realtime Database, linked to the user's ID.*
Postconditions	*- The user receives a preliminary diagnosis and can view the chat history.* *- The chat log is stored for future reference.*
Exceptions	*- If the Infermedica API limit is reached, the system displays an error ("API limit exceeded").* *- If the network fails, the system prompts the user to retry.*

Use Case 4: Upload and View Medical Records

Field	*Details*
Use case ID	*UC-04*
Actors	*Logged-In User, Firebase Firestore*
Preconditions	*- The user is logged in with a verified account.* *- Firebase Firestore has available storage (<1 GB) and write quotas (<20,000 daily).*
Main Flow	***Step 1**: The user navigates to the Medical Records page.*

	***Step 2**: The user selects a file (e.g., PDF, image, <10 MB) using the upload form, styled with Bootstrap 5.* ***Step 3**: The system validates the file (e.g., format, size) and prompts the user to confirm.* ***Step 4**: The user submits the file, triggering an upload to Firebase Firestore via a secure HTTPS request.* ***Step 5**: Firebase Firestore stores the file, linked to the user's ID, and returns a success confirmation.* ***Step 6**: The system updates the UI with the uploaded file's metadata (e.g., name, date).* ***Step 7**: The user clicks to view a record, and the system retrieves it from Firestore, displaying the file or a download link.*
Postconditions	*- The medical record is securely stored and linked to the user's account.* *- The user can view or download their records.*
Exceptions	*- If the file exceeds 10 MB or is an unsupported format, the system displays an error ("Invalid file").* *- If Firestore storage is full, the system notifies the user ("Storage limit reached").*

Use Case 5: Access Emergency Helpline

Field	*Details*
Use case ID	*UC-05*
Actors	*Logged-In User or Guest User*
Preconditions	*- The user has internet access.* *- The external emergency helpline website is operational.*
Main Flow	***Step 1**: The user navigates to the Call an Ambulance page via the Sehat Pal homepage or menu.* ***Step 2**: The user reads the emergency instructions and disclaimer on the page.* ***Step 3**: The user clicks the redirect button, triggering an HTTP redirect to the external helpline URL.* ***Step 4**: The system opens the helpline website in the user's browser.*
Postconditions	*- The user is redirected to the emergency helpline website for assistance.* *- No user data is exchanged during the redirect.*
Exceptions	*- If the helpline URL is unavailable, the system displays an error ("Helpline unavailable, try again later").*

Use Case 6: View Information Pages

Field	*Details*
Use case ID	*UC-06*
Actors	*Logged-In User or Guest User*

Preconditions	*- The user has internet access.* *- Firebase Hosting is operational.*
Main Flow	***Step 1**: The user navigates to an information page (About Us, Contact Us, or Help) via the homepage or menu.* ***Step 2**: The system loads the static content (e.g., project details, FAQs) from Firebase Hosting, styled with Bootstrap 5.* ***Step 3**: The user reads the content, such as disclaimers on the About Us page or guidance on the Help page.*
Postconditions	*- The user gains access to the requested information.* *- No user interaction beyond viewing is required.*
Exceptions	*- If the page fails to load, the system displays an error ("Page unavailable, check network").*

5. Non-functional Requirements

5.1. Performance Requirements

Sehat Pal, a web-based symptom checker developed as a university project by Syed Ukkashah (backend developer) and Ibrahim Johar (frontend developer), is designed to meet the performance requirements of a small-scale system serving 5-10 users in Pakistan, particularly those with limited healthcare access. Hosted on Firebase Hosting, the system integrates Firebase services, the Infermedica API (trial basis: 2,000 API calls or 60 days), Bootstrap 5, Axios, and an emergency helpline redirect to provide efficient, secure, and reliable preliminary health assessments. This section outlines the performance characteristics required by the business (academic context), including speed, precision, concurrency, capacity, safety, and reliability, tailored to the project's prototype scope.

- ***Speed**:*
  - ***Page Load Time**: The frontend, built with HTML, CSS, JavaScript, and Bootstrap 5, loads in less than 3 seconds on standard browsers (e.g., Chrome, Firefox, Safari) under typical network conditions in Pakistan, ensuring quick access to the interface.*
  - ***Firebase Operations**: Authentication (signup/login, email verification), Firestore (medical record storage/retrieval), and Realtime Database (chat history updates) queries complete in less than 1 second, leveraging Firebase's cloud infrastructure for low-latency performance.*
  - ***Infermedica API Responses**: Symptom analysis and diagnosis requests, sent via Axios, return results in less than 2 seconds, dependent on the Infermedica API's trial service performance and network conditions.*
  - ***Email Delivery**: Firebase Authentication emails for signup verification and password reset are delivered in less than 5 seconds, dependent on Firebase's email service and the user's email provider.*
  - ***HTTP Redirect**: The Call an Ambulance page redirect to the external emergency helpline website completes in less than 1 second, browser-dependent.*
  - ***Rationale**: These speed requirements ensure a responsive user experience for a small user base, accommodating variable network conditions in Pakistan without requiring advanced optimization.*

- ***Precision***:
    - ***Symptom Diagnosis***: *The Infermedica API (latest trial version) provides accurate symptom-disease mappings based on user inputs, with percentage-based probabilities, sufficient for preliminary health assessments. Precision depends on the API's internal medical database, assumed to be reliable for the trial period.*
    - ***Data Storage/Retrieval***: *Firebase Firestore and Realtime Database ensure precise storage and retrieval of medical records (e.g., user IDs, file metadata, contents) and chat history (e.g., timestamps, symptom inputs, diagnoses), with no data loss or corruption under normal operation.*
    - ***User Authentication***: *Firebase Authentication accurately validates user credentials and email verification, ensuring correct user access control.*
    - ***Rationale***: *Precise diagnosis and data handling are critical for user trust and functionality, meeting the academic requirement for a functional prototype.*
- ***Concurrency***:
    - ***Concurrent Users***: *The system supports up to 5-10 concurrent users, constrained by the Infermedica API trial (2,000 API calls or 60 days) and Firebase free-tier limits (e.g., database reads/writes, bandwidth).*
    - ***Behavior***: *Firebase services handle concurrent authentication, data storage, and chat history updates efficiently for a small user base, with no significant performance degradation. The Infermedica API processes sequential client-side requests without concurrency issues for limited users.*
    - ***Rationale***: *The small-scale university project scope does not require support for large-scale concurrency, aligning with the target user base and API constraints.*
- ***Capacity***:
    - ***Data Storage***: *Firebase Firestore stores medical records (e.g., PDFs, images, <10 MB assumed per file) for 5-10 users, within the free-tier quota (e.g., 1 GB storage, 20,000 daily writes). Firebase Realtime Database stores chat history (e.g., text-based logs, <1 MB per user) within similar limits.*
    - ***API Usage***: *The Infermedica API supports up to 2,000 API calls (e.g., symptom queries, diagnoses) or 60 days, sufficient for 5-10 users performing occasional health assessments.*
    - ***Hosting***: *Firebase Hosting delivers static content (HTML, CSS, JavaScript, Bootstrap 5, Axios) with a free-tier bandwidth limit (e.g., 10 GB/month), adequate for small-scale traffic.*
    - ***Rationale***: *The system's capacity meets the needs of a prototype, handling limited data volumes without requiring commercial-scale resources.*
- ***Safety***:
    - ***Data Security***: *HTTPS with TLS 1.2 or higher secures all communications (Firebase, Infermedica API). Firebase security rules restrict Firestore and Realtime Database access to authenticated users. Firebase Authentication uses secure password hashing (e.g., bcrypt) for user credentials, supporting signup verification and password reset.*
    - ***API Security***: *Infermedica API requests include an API key for authentication, preventing unauthorized access.*

- ***Form Security**: Client-side validation (using JavaScript and Bootstrap 5) and HTTPS prevent injection attacks or data leaks in electronic forms (signup, symptom input, file uploads).*
- ***Email Security**: Firebase Authentication emails (signup verification, password reset) use secure SMTP and HTTPS links, avoiding sensitive data exposure.*
- ***Rationale**: Safety is critical for protecting medical data and user privacy, meeting academic and ethical requirements for a healthcare-related prototype.*

- ***Reliability**:*
  - ***Uptime**: The system targets 99.9% availability, leveraging Firebase Hosting's cloud infrastructure, which provides high uptime for static content and backend services. Reliability depends on Firebase and Infermedica API service availability, assumed stable for the project's duration.*
  - ***Error Handling**: The frontend (via JavaScript and Axios) handles API errors (e.g., Infermedica API rate limits, network failures) with user-friendly messages. Firebase services provide robust error handling for authentication and database operations.*
  - ***Data Integrity**: Firebase Firestore and Realtime Database ensure no data loss under normal operation, with backups not required for the prototype scope. The Infermedica API is assumed to deliver consistent diagnoses within trial limits.*
  - ***Rationale**: High reliability ensures the system remains functional for evaluation by the course instructor (Miss Fizza Mansoor) and user testing, meeting academic requirements for a working prototype.*

These performance characteristics ensure Sehat Pal delivers a responsive, accurate, secure, and reliable experience for its target audience in Pakistan, within the constraints of the university project, Firebase free-tier, and Infermedica API trial.

5.2. Safety Requirements

This section specifies requirements to prevent data breaches, misdiagnosis, system misuse, and service unavailability, defines safeguards and actions to be taken, identifies dangerous actions to prevent, and notes applicable safety certifications, policies, or regulations, tailored to the prototype's academic context.

- ***Possible Loss, Damage, or Harm***
- ***Data Breaches**: Unauthorized access to medical records (stored in Firebase Firestore) or user credentials (managed by Firebase Authentication) could compromise user privacy, leading to identity theft or exposure of sensitive health information.*
- ***Misdiagnosis**: Inaccurate diagnoses from the Infermedica API (trial version) could mislead users, potentially delaying professional medical care and exacerbating health conditions.*
- ***System Misuse**: Users treating Sehat Pal as a substitute for professional medical advice could make uninformed health decisions, risking harm due to lack of qualified diagnosis or treatment.*
- ***Service Unavailability**: Downtime of Firebase services or the Infermedica API could prevent users from accessing health assessments or the emergency helpline redirect, critical for urgent needs.*
- ***Safeguards and Actions***
- ***Data Security**:*

- *All communications (Firebase, Infermedica API) use HTTPS with TLS 1.2 or higher for encryption, ensuring secure data transfer.*
- *Firebase security rules restrict access to Firestore (medical records) and Realtime Database (chat history) to authenticated users, preventing unauthorized access.*
- *Firebase Authentication employs secure password hashing (e.g., bcrypt) for user credentials, with email verification during signup and secure password reset functionality.*
- *Infermedica API requests include an API key for authentication, ensuring secure access to trial-based services.*

- ***Disclaimers**:*
  - *The frontend, developed by Ibrahim Johar, includes clear UI messages (e.g., on the symptom checker and About Us pages) stating that Sehat Pal provides preliminary health assessments only and is not a substitute for professional medical advice. Users are encouraged to consult doctors for accurate diagnoses and treatment.*
- ***Emergency Access**:*
  - *The Call an Ambulance page, developed by Ibrahim Johar, provides a reliable HTTP redirect to an external emergency helpline website, ensuring users can access critical services in urgent situations.*
- ***Input Validation**:*
  - *Client-side validation (using JavaScript and Bootstrap 5) on electronic forms (signup, symptom input, medical record uploads) prevents malicious inputs (e.g., SQL injection, cross-site scripting).*
  - *File uploads to Firebase Firestore are restricted to safe formats (e.g., PDFs, images) and size limits (<10 MB assumed), validated client-side to prevent exploits.*
- ***Error Handling**:*
  - *The frontend (via JavaScript and Axios) displays user-friendly error messages for API failures (e.g., Infermedica API rate limits, network issues), guiding users to retry or seek alternatives.*
  - *Firebase services handle authentication and database errors gracefully, ensuring system stability.*
- ***Dangerous Actions to Prevent***
- ***Unauthorized Data Access**:*
  - *Prevented by Firebase security rules requiring authentication for Firestore and Realtime Database access, and by HTTPS encryption for all data transfers.*
  - *Firebase Authentication's password security (e.g., bcrypt hashing) and email verification during signup reduce the risk of credential theft.*
- ***Over-Reliance on Diagnoses**:*
  - *Prevented by explicit disclaimers in the UI (developed by Ibrahim Johar) clarifying that the Infermedica API's diagnoses are preliminary and not a replacement for professional medical advice.*
  - *The system encourages users to consult healthcare professionals via UI prompts and the emergency helpline redirect.*

- ***Malicious Inputs****:*
    - *Prevented by client-side validation on all forms (signup, symptom input, file uploads) to block injection attacks or unsafe files.*
    - *Firebase Firestore enforces data schema validation to reject malformed uploads.*
- ***Misuse of Emergency Redirect****:*
    - *Prevented by ensuring the Call an Ambulance redirect links to a reputable external helpline, with clear UI instructions to use it only for emergencies.*
- ***Safety Certifications, Policies, or Regulations***
- ***Firebase Terms of Service****: The system must comply with Firebase's terms for free-tier usage, including limits on database operations, storage, and bandwidth, ensuring proper handling of user data.*
- ***Infermedica API Terms of Service****: The trial version (2,000 API calls or 60 days) requires adherence to usage restrictions, prohibiting modifications to API functionality and ensuring secure API key management.*
- ***Pakistan's Data Protection Bill****: If applicable (assumed as a best practice for a university project), the system must align with data protection principles for medical data, such as user consent, data minimization, and secure storage, enforced by Firebase security rules and HTTPS.*
- ***General Data Privacy Standards****: The system follows GDPR-like principles for medical data (e.g., confidentiality, integrity), ensuring user privacy through encryption, authentication, and restricted access, even in a prototype context.*
- ***Academic Ethical Guidelines****: As a university project evaluated by Miss Fizza Mansoor, the system must adhere to ethical guidelines for healthcare-related software, including transparency about limitations (via disclaimers), user safety (via secure data handling), and clear communication of non-professional status.*
- ***No Formal Safety Certifications****: As a small-scale academic prototype, no formal safety certifications (e.g., ISO 13485 for medical devices) are required, but the system adopts best practices for data security and user safety to meet academic standards.*

These safety requirements ensure Sehat Pal mitigates risks of data breaches, misdiagnosis, misuse, and service unavailability, protecting users and their data while meeting the academic and ethical expectations of the university project within the constraints of the Infermedica API trial and Firebase free-tier.

5.3. Security Requirements

- ***Data Transmission Security****:*
    - *All communications between the client (browser) and external components (Firebase Authentication, Firestore, Realtime Database, Hosting; Infermedica API) use HTTPS with TLS 1.2 or higher, ensuring encrypted data transfer to prevent interception or tampering.*
- ***Authentication Security****:*
    - *Firebase Authentication, integrated by Syed Ukkashah, secures user signup and login with email and password, using secure password hashing (e.g., bcrypt). Email verification during signup ensures account ownership, and password reset functionality provides secure recovery via emailed HTTPS links.*
- ***Authorization Security****:*

- *Firebase security rules, configured by Syed Ukkashah, restrict access to Firestore (medical records) and Realtime Database (chat history) to authenticated users only, preventing unauthorized data access. Rules validate user IDs against session tokens.*

- ***Input Validation**:*
  - *Client-side validation, implemented by Ibrahim Johar using JavaScript and Bootstrap 5, on electronic forms (signup, symptom input, medical record uploads) prevents injection attacks (e.g., SQL injection, cross-site scripting). File uploads are restricted to safe formats (e.g., PDFs, images) and size limits (<10 MB assumed).*

- ***API Security**:*
  - *Infermedica API requests, sent via Axios, include an API key for authentication, ensuring secure access to trial-based services and preventing unauthorized API calls.*

- ***Error Handling**:*
  - *The frontend (via JavaScript and Axios) handles errors (e.g., API rate limits, network failures) with user-friendly messages, preventing system crashes or exposure of sensitive information.*

Integrity Requirements

- ***Data Storage Integrity**:*
  - *Firebase Firestore and Realtime Database, managed by Syed Ukkashah, ensure accurate storage and retrieval of medical records (user IDs, metadata, file contents) and chat history (user IDs, timestamps, symptom inputs, diagnoses), with no data corruption under normal operation.*

- ***Diagnosis Integrity**:*
  - *The Infermedica API (latest trial version) is assumed to provide reliable symptom-disease mappings, maintaining diagnosis integrity for preliminary health assessments. Client-side validation ensures accurate symptom inputs.*

- ***Data Consistency**:*
  - *Firebase security rules enforce data schema validation, preventing malformed or inconsistent data writes. Client-side error handling (via Axios) ensures consistent API interactions with the Infermedica API.*

- ***Auditability**:*
  - *Chat history in Firebase Realtime Database includes timestamps and user IDs, allowing traceability of user interactions for integrity verification, though not required for auditing in a prototype.*

Privacy Requirements

- ***Data Minimization**:*
  - *The system collects only necessary data: email and password for authentication, symptom inputs for diagnosis, and medical records (e.g., PDFs, images) for storage. No unnecessary personal data (e.g., full names, addresses) is collected unless provided voluntarily.*

- ***Data Encryption**:*
  - *Data in transit is encrypted via HTTPS (TLS 1.2+). Data at rest in Firebase Firestore and Realtime Database is encrypted using Firebase's built-in encryption, protecting medical records and chat history.*

- ***User Consent**:*
    - *Users implicitly consent to data collection and processing by signing up, with clear UI disclaimers (developed by Ibrahim Johar) explaining data usage for health assessments and storage. Disclaimers are displayed on signup and symptom checker pages.*
- ***Data Access Control**:*
    - *Only authenticated users can access their own medical records and chat history, enforced by Firebase security rules. No third-party data sharing occurs beyond Firebase (for storage) and Infermedica API (for diagnosis).*
- ***Email Privacy**:*
    - *Firebase Authentication emails (signup verification, password reset) contain minimal data (e.g., verification/reset links) and are sent over secure SMTP, avoiding exposure of sensitive information.*

User Authentication and Authorization Requirements

- ***Authentication**:*
    - *Users must sign up with a valid email address and password via Firebase Authentication, integrated by Syed Ukkashah. Email verification is required post-signup to confirm account ownership, sent via a secure HTTPS link.*
    - *Login requires email and password, with secure password reset functionality available via emailed HTTPS links, ensuring account recovery without compromising security.*
    - *Authentication sessions are managed with Firebase tokens, refreshed automatically for active users.*
- ***Authorization**:*
    - *Authenticated users are authorized to access their own medical records (Firebase Firestore) and chat history (Firebase Realtime Database), linked by user IDs. Firebase security rules ensure that users cannot access data belonging to others.*
    - *No administrative roles are defined, as the system is designed for individual end-users only, with no need for elevated privileges in the prototype scope.*
- ***Session Management**:*
    - *Firebase Authentication maintains user sessions via secure tokens, with automatic expiration after inactivity (configurable, assumed 1 hour for prototype). Users must re-authenticate after session expiry.*

Security and Privacy Policies or Certifications

- ***Firebase Terms of Service**:*
    - *The system must comply with Firebase's free-tier terms, including limits on database operations (e.g., 20,000 daily writes), storage (e.g., 1 GB), and bandwidth (e.g., 10 GB/month), and adhere to security best practices (e.g., security rules, encryption).*
- ***Infermedica API Terms of Service**:*
    - *The trial version (2,000 API calls or 60 days) requires adherence to usage restrictions, secure API key management, and no modifications to API functionality, ensuring compliance with Infermedica's privacy and security policies.*

- ***Pakistan's Data Protection Bill**:*
    - *If applicable (assumed as a best practice for a university project), the system aligns with principles such as user consent, data minimization, secure storage, and right to access/delete data, implemented via Firebase security rules, HTTPS, and UI disclaimers.*
- ***General Data Privacy Standards**:*
    - *The system follows GDPR-like principles for medical data, including confidentiality (encryption, access control), integrity (secure storage), and transparency (disclaimers), even in a prototype context, to protect user privacy.*
- ***Academic Ethical Guidelines**:*
    - *As a university project evaluated by Miss Fizza Mansoor, the system adheres to ethical guidelines for healthcare-related software, ensuring:*
        - *Transparency about the system's limitations (preliminary assessments only, via UI disclaimers).*
        - *User safety through secure data handling and clear communication of non-professional status.*
        - *Privacy protection through encryption and access control, meeting academic standards.*
- ***No Formal Certifications**:*
    - *As a small-scale academic prototype, no formal security or privacy certifications (e.g., ISO 27001, HIPAA) are required. The system adopts best practices for data security and privacy to meet academic and ethical expectations.*

These security and privacy requirements ensure Sehat Pal protects user data, maintains system integrity, and safeguards user privacy, meeting the needs of its target audience in Pakistan while adhering to the university project's scope, Firebase free-tier constraints, and Infermedica API trial limitations.

5.4. User Documentation

- ***User Manual**:*
    - ***Description**: A concise document providing comprehensive instructions for using Sehat Pal, tailored to users in Pakistan with basic digital literacy. It covers system setup, core functionalities, and safety information.*
    - ***Content**:*
        - ***Introduction**: Overview of Sehat Pal as a preliminary symptom checker, with disclaimers emphasizing it is not a substitute for professional medical advice.*
        - ***Setup**: Steps for signing up (entering email/password, verifying email via Firebase Authentication link) and logging in (including password reset instructions).*
        - ***Usage**: Instructions for using the symptom checker (entering symptoms via forms, responding to Infermedica API questions), uploading medical records (e.g., PDFs, images, <10 MB) to Firebase Firestore, viewing chat history (Firebase Realtime Database), and accessing the Call an Ambulance redirect.*
        - ***Safety and Privacy**: Explanation of data protection (HTTPS, Firebase security rules), privacy practices (data minimization, no third-party sharing), and guidance to consult doctors for accurate diagnoses.*

        - ***Troubleshooting****: Common issues (e.g., login failures, API errors) and solutions (e.g., check email for verification, retry after network issues).*
    - ***Format****: Delivered as a PDF or static web page, hosted on Firebase Hosting, downloadable or viewable via a dedicated link in the UI (e.g., "Help" or "Documentation" menu).*
    - ***Purpose****: Provides a standalone reference for users to understand and navigate the system, meeting academic requirements for clear documentation.*
- ***Online Help****:*
    - ***Description****: Integrated help content embedded in the frontend (developed by Ibrahim Johar), offering brief instructions and FAQs to assist users during interaction with the system.*
    - ***Content****:*
        - ***Help Page****: A static page (accessible via a UI link, e.g., "Help" button) summarizing key functionalities (signup, symptom checker, record uploads, emergency redirect) and FAQs (e.g., "How do I verify my email?", "What file types can I upload?").*
        - ***Disclaimers****: Reinforces that Sehat Pal provides preliminary assessments and users should consult doctors, displayed prominently on the help page.*
        - ***Contact Information****: Optional contact details (e.g., developer email) for user queries, if required for academic evaluation.*
    - ***Format****: HTML page styled with Bootstrap 5, hosted on Firebase Hosting, accessible via the application's navigation menu.*
    - ***Purpose****: Offers immediate, in-app guidance for users, enhancing usability for those unfamiliar with the system.*
- ***Context-Sensitive Help****:*
    - ***Description****: Limited inline help provided through tooltips or hints on specific UI elements, particularly forms, to guide users during data entry.*
    - ***Content****:*
        - ***Form Tooltips****: Hints on signup/login forms (e.g., "Enter a valid email address"), symptom input forms (e.g., "Select all applicable symptoms"), and medical record upload forms (e.g., "Supported formats: PDF, JPG, PNG; max 10 MB").*
        - ***Error Messages****: Inline feedback for invalid inputs (e.g., "Password must be at least 6 characters") or failed actions (e.g., "File size exceeds limit").*
    - ***Format****: Implemented using JavaScript and Bootstrap 5 tooltips/popovers, rendered client-side within the browser.*
    - ***Purpose****: Provides real-time assistance during user interactions, reducing errors and improving the experience for users with basic digital literacy.*
- ***Tutorials****:*
    - ***Description****: No extensive tutorials are included due to the prototype's small scope, but a brief "Getting Started" section is incorporated into the user manual to guide first-time users.*
    - ***Content****:*
        - ***Getting Started Guide****: A short section in the user manual outlining the initial steps to use Sehat Pal, including signing up, verifying email, performing a*

symptom check, and uploading a medical record. Includes screenshots or text descriptions of key UI elements (e.g., symptom checker form, upload button).

- ***Format**: Embedded in the user manual (PDF or web page), hosted on Firebase Hosting.*
- ***Purpose**: Offers a quick introduction to the system's core features, sufficient for academic evaluation and user onboarding in a prototype context.*

- ***Delivery and Accessibility***
- ***Delivery**: All documentation components are delivered digitally, hosted on Firebase Hosting alongside the Sehat Pal application. The user manual is accessible via a downloadable PDF or static web page link in the UI. Online help and context-sensitive help are integrated into the frontend, accessible during system use.*
- ***Accessibility**: Documentation is designed for users with basic digital literacy, using clear, concise language and visual aids (e.g., screenshots, tooltips styled with Bootstrap 5). Content is available in English, with potential for Urdu translations if required for the Pakistani audience (not implemented in prototype scope).*
- ***Maintenance**: Documentation is maintained by Syed Ukkashah and Ibrahim Johar during the project's development and evaluation phase, updated as needed for academic submission to Miss Fizza Mansoor.*

These user documentation components ensure Sehat Pal is accessible and usable for its target audience in Pakistan, providing clear guidance on system functionality and safety while meeting the academic requirements of the university project within the constraints of the Infermedica API trial and Firebase free-tier.

6. References

Sehat Pal, a web-based symptom checker developed as a university project by Syed Ukkashah (backend developer) and Ibrahim Johar (frontend developer), relies on various documents for its development, implementation, and documentation. This section provides a complete list of all documents referenced in the Software Requirements Specification (SRS) at the time of writing, identified by title, report number (if applicable), date, publishing organization, and source. These references support the system's technical implementation and academic requirements for a small user base (5-10 users) in Pakistan, using Firebase Hosting, the Infermedica API (trial basis: 2,000 API calls or 60 days), Bootstrap 5, Axios, and Firebase Authentication.

1. ***Firebase Documentation***
    - ***Title****: Firebase Documentation*
    - ***Report Number****: Not applicable*
    - ***Date****: 2025 (latest version as of development)*
    - ***Publishing Organization****: Google LLC*
    - ***Source****: https://firebase.google.com/docs*
    - ***Description****: Comprehensive guide for Firebase services, including Authentication (user signup, login, email verification), Firestore (medical record storage), Realtime Database (chat history), and Hosting (web application deployment), used by Syed Ukkashah for backend implementation.*
2. ***Infermedica API Documentation***
    - ***Title****: Infermedica API Documentation*
    - ***Report Number****: Not applicable*
    - ***Date****: 2025 (latest trial version as of development)*
    - ***Publishing Organization****: Infermedica*
    - ***Source****: https://developer.infermedica.com/docs*
    - ***Description****: Technical documentation for the Infermedica API (trial version, 2,000 API calls or 60 days), detailing RESTful HTTPS requests, symptom analysis, and diagnosis outputs, used by Syed Ukkashah for symptom checker integration.*
3. ***Bootstrap 5 Documentation***
    - ***Title****: Bootstrap 5 Documentation*
    - ***Report Number****: Not applicable*
    - ***Date****: 2025 (version 5, latest as of development)*
    - ***Publishing Organization****: Bootstrap Team*
    - ***Source****: https://getbootstrap.com/docs/5.0/*
    - ***Description****: Official documentation for Bootstrap 5, covering responsive styling, UI components, and JavaScript plugins, used by Ibrahim Johar for frontend development and documentation components (e.g., tooltips, forms).*
4. ***Axios Documentation***
    - ***Title****: Axios Documentation*
    - ***Report Number****: Not applicable*
    - ***Date****: 2025 (latest stable version as of development)*

- ***Publishing Organization**: Axios Contributors*
- ***Source**: https://axios-http.com/docs/intro*
- ***Description**: Guide for Axios, a JavaScript library for client-side HTTP requests, used by Ibrahim Johar to implement Infermedica API calls in the frontend.*

5. ***Software Requirements Specification Template***
   - ***Title**: Software Requirements Specification Template*
   - ***Report Number**: Not applicable*
   - ***Date**: 2025 (course-specific, assumed)*
   - ***Publishing Organization**: FAST-NUCES Software Engineering Department*
   - ***Source**: Provided by course instructor, Miss Fizza Mansoor, via Google Classroom*
   - ***Description**: Academic template guiding the structure and content of the Sehat Pal SRS, including sections for operating environment, constraints, interfaces, performance, safety, security, and documentation.*
6. ***Ethical Guidelines for Healthcare-Related Software Projects***
   - ***Title**: Ethical Guidelines for Healthcare-Related Software Projects*
   - ***Report Number**: Not applicable*
   - ***Date**: 2025 (course-specific, assumed)*
   - ***Publishing Organization**: [University Name] Software Engineering Department*
   - ***Source**: Provided by course instructor, Miss Fizza Mansoor, via university learning management system or course materials*
   - ***Description**: University or course-specific guidelines outlining ethical considerations for healthcare-related software, including transparency, user safety, and data privacy, used to ensure Sehat Pal aligns with academic standards.*

Notes

- ***Online Sources**: Technical documentation (Firebase, Infermedica, Bootstrap 5, Axios) is freely accessible via the provided URLs, subject to updates by their respective organizations. Users should refer to the latest versions as of 2025.*

This bibliography ensures all referenced materials are properly documented, supporting the development and evaluation of Sehat Pal within the university project's scope and technical constraints.