College of Computer and Information Sciences (CCIS)

Course: SE201 – Introduction to Software Engineering

Sillah (صلة): Family Health Management System

Instructor: Ms. Ishrat Khan

Section #: 1320

Project Contributors:

SHOUG FAWAZ ABDULLAH ALOMRAN 223410392

ALJOHARA WALEED A ALBAWARDI 223410346

ALJAWHARA MOHMMAD NASSER ALRUZUQ 223410216

ALJAWHARAH TURKI ABDULLAH ALSALEH 223410217

Phase I: Project Proposal

1.1 Introduction

The **Sillah (صلة)** project is a preventive health management system designed to help Saudi families identify hereditary cardiac risks through an easy-to-use digital platform.

It enables users to record their family health history, receive automatic risk alerts, and book preventive clinic appointments.

This report documents all project phases — from requirements to design, prototype, and testing — demonstrating a full software development process aligned with SE201 objectives.

1.2 Project Overview

1.2.1 Goals:

1. Promote Preventive Health Awareness: Promote public awareness regarding hereditary diseases and encourage early detection.
2. Simplify Family Health Tracking: Provide a simple-to-use interface for families to follow and chart medical histories.
3. Enable Smart Risk Detection: Use rule-based reasoning to automatically detect hereditary cardiac risk patterns.
4. Support Action Through Integration: Link users to accredited clinics for consultations and checkups.
5. Empower Public Health Systems: Demonstrate a scalable model that can be embedded in national healthcare systems.

1.2.2 Objectives:

To provide a user-friendly system that allows citizens to record family health data, analyze hereditary patterns, and take proactive steps toward cardiac screening.

1.2.3 Main Features:

- Add and manage family members with health events
- Detect hereditary risks automatically

- Send personalized alerts and recommendations
- Access an Awareness Hub with educational content and checklists
- Book appointments with local clinics

1.2.4 Users:

1. General citizens (families)
2. Healthcare Providers (view bookings)
3. Administrators (manage awareness content)

Phase II: Requirements Analysis & Documentation

2.1 Functional Requirements

ID	Requirement	Description	Priority
FR-01	Add Family Member	User can record family members with name, relation, and health conditions	High
FR-02	Add Health Event	User can specify a condition and age of onset	High
FR-03	Generate Risk Alert	System detects hereditary risk and sends an alert	High
FR-04	Book Appointment	User can select a clinic and confirm booking	Medium
FR-05	View Awareness Content	User can access health tips and checklists	Medium
FR-06	View Alert History	User can view past alerts	Low

2.2 Non-Functional Requirements

2.2.1 Introduction & Overview

This document specifies the Non-Functional Requirements (NFRs) for the Sillah Family Health Management System across **Usability**, **Reliability**, and **Security**. The goal is to ensure the platform is simple to use, robust, and safe for all users in Saudi Arabia. Items marked [Conceptual] indicate production-level expectations. In Phase IV (Java prototype), those items will be simulated or documented, while core functionality will be implemented.

2.2.2 Usability Requirements

UR-01: Mobile-First Responsive Design

Priority: Critical

Phase IV: Implement

Rationale: Most users in Saudi Arabia access services via mobile; optimizing for phones maximizes reach.

User Stories: SIL-7 to SIL-16

Requirement: Interface must work across screen sizes, prioritizing mobile.

Acceptance Criteria:

- Passes Google Mobile-Friendly Test
- Verified on iPhone (Safari) and Android (Chrome)
- Viewport meta tag is set correctly
- Touch targets ≥ 44×44 px (Web Content Accessibility Guidelines)

- No horizontal scrolling

UR-02: Web Content Accessibility Guidelines 2.1 Level AA Accessibility

Priority: High

Phase IV: Implement core; document full compliance

Rationale: Public services must be accessible and legally compliant.

User Stories: SIL-7 to SIL-16

Requirement: Meet Web Content Accessibility Guidelines 2.1 AA guidelines.

Acceptance Criteria:

- Web Accessibility Evaluation Tool/Accessibility Engine (WAVE/axe) automated scan: zero critical errors
- Manual test with NVDA or JAWS
- Keyboard-only navigation succeeds
- Color contrast verified with WebAIM (≥ 4.5:1 normal text)

UR-03: Task Completion Efficiency

Priority: High

Phase IV: Implement

Rationale: Faster tasks increase adoption and adherence to preventive actions.

User Stories: SIL-7, SIL-13, SIL-14

Requirement: Key tasks should be quick and low-effort.

Acceptance Criteria:

- Build a basic family tree in < 5 minutes (≥ 15 participants)
- Find & download a checklist in < 3 minutes
- ≥ 90% success on critical tasks without help

UR-04: Bilingual Interface (Arabic & English)

Priority: High

Phase IV: Implement

Rationale: Serves both Arabic speakers and expatriates.

User Stories: SIL-7 to SIL-16

Requirement: Full bilingual support.

Acceptance Criteria:

- 100% UI translated
- RTL layout correct for Arabic
- Language preference persists
- Medical terminology reviewed by an Arabic-speaking clinician

UR-05: Clear Error Messages & Guidance

Priority: High

Phase IV: Implement

Rationale: Friendly, actionable errors reduce frustration and support load.

User Stories: SIL-7, SIL-8, SIL-15

Requirement: Errors must be specific, polite, and near the relevant field.

Acceptance Criteria:

- Messages free of technical jargon
- Inline placement beside fields
- Visible success states with confirmation

UR-06: Intuitive Information Architecture

Priority: High

Phase IV: Implement

Rationale: Logical structure lowers cognitive load.

User Stories: SIL-7 to SIL-16

Requirement: Content and features must be easy to find.

Acceptance Criteria:

- ≤ 3 clicks to any feature
- “Ease of finding” rating ≥ 4/5
- Breadcrumbs available site-wide

UR-07: Consistent Design System

Priority: Medium

Phase IV: Implement

Rationale: Consistency speeds learning and builds trust.

User Stories: SIL-7 to SIL-16

Requirement: Use a shared design system and components.

Acceptance Criteria:

- Style guide documented
- Component library in use
- Visual QA confirms consistency

UR-08: Onboarding & First-Time UX

Priority: Medium

Phase IV: Implement

Rationale: A strong first session boosts retention.

User Stories: SIL-15

Requirement: New users understand the product and complete setup easily.

Acceptance Criteria:

- ≥ 80% onboarding completion
- Average time < 3 minutes
- Consent captured with timestamp

UR-09: Plain-Language Readability

Priority: High

Phase IV: Implement

Rationale: Health literacy varies; content must be clear.

User Stories: SIL-11, SIL-12

Requirement: Public-friendly language with minimal jargon.

Acceptance Criteria:

- Reviewed by non-medical staff
- Flesch-Kincaid ≈ 8th grade

- User tests confirm understanding

UR-10: Visual Feedback & System Status

Priority: Medium

Phase IV: Implement

Rationale: Clear feedback reduces anxiety and confusion.

User Stories: SIL-7 to SIL-16

Requirement: Always show what's happening and what comes next.

Feedback Examples:

- Spinner for quick actions (< 2s)
- Progress bar for longer actions (> 2s)
- Success/error toasts (top-right, auto-dismiss)
- Skeleton screens during content loads

Acceptance Criteria:

- All loading states implemented
- Success/error feedback within 1 second
- No "frozen" actions

UR-11: Help & Support Access

Priority: Medium

Phase IV: Document

Rationale: Accessible help reduces drop-off and tickets.

User Stories: SIL-7 to SIL-16

Requirement: Help is easy to find and use.

Acceptance Criteria:

- Help in Arabic & English
- FAQ covers ≥ 80% common questions
- Email support live; WhatsApp noted as future option
- Responses within 48 hours

UR-12: System Usability Scale (SUS)

Priority: High

Phase IV: Document

Rationale: SUS is a validated, standard usability metric.

User Stories: SIL-7 to SIL-16

Requirement: Run SUS and meet score targets.

Acceptance Criteria:

- ≥ 15 participants
- Standard 10 SUS items used
- Target ≥ 80 (A); minimum ≥ 70 (C)

2.2.3 Reliability Requirements

RR-01: Availability & Uptime [Conceptual]

Priority: Critical

Phase IV: Document

Rationale: Users need access when risk alerts matter most.

User Stories: SIL-7 to SIL-16

Requirement: High availability with minimal unplanned downtime.

Acceptance Criteria:

- Monthly uptime target 99.5% (≤ 3.65 hours downtime)
- Automated monitoring & alerts
- Public status page

RR-02: Backup & Disaster Recovery [Conceptual]

Priority: Critical

Phase IV: Document

Rationale: Family health data must not be lost.

User Stories: SIL-7 to SIL-10

Requirement: Robust backups and tested recovery.

Acceptance Criteria:

- Automated backups verified
- RPO ≤ 6 hours; RTO ≤ 4 hours
- Quarterly DR drills

RR-03: Graceful Degradation & Fault Tolerance

Priority: High

Phase IV: Implement

Rationale: Partial service is better than outage.

User Stories: SIL-7 to SIL-16

Requirement: Keep core features usable during failures.

Acceptance Criteria:

- Map fallback to list view when the map API is down
- No raw stack traces shown to users
- Fallbacks enabled for core features
- Friendly error messages

RR-04: Data Accuracy & Integrity

Priority: Critical

Phase IV: Implement

Rationale: Incorrect risk results undermine the system's purpose.

User Stories: SIL-7 to SIL-10

Requirement: Protect data quality and detect risks accurately.

Acceptance Criteria:

- False negatives ≤ 1% on a defined test set
- False positives < 5% from entry errors
- DB constraints prevent impossible data
- 100% detection on known-risk test trees

RR-05: Browser & Device Compatibility

Priority: High

Phase IV: Implement

Rationale: Equal access across common environments.

User Stories: SIL-7 to SIL-16

Requirement: Consistent behavior on supported browsers/devices.

Acceptance Criteria:

- Identical behavior on latest two versions
- Visual consistency verified
- Unsupported browsers get an upgrade notice

RR-06: Performance Under Load [Conceptual]

Priority: High

Phase IV: Document

Rationale: Campaigns can create traffic spikes.

User Stories: SIL-7 to SIL-16

Requirement: Remain reliable under heavy load.

Acceptance Criteria:

- Capacity for 10,000 concurrent users validated
- Error rate < 1% at peak
- Auto-scaling configured and tested

RR-07: Content Consistency & Currency

Priority: High

Phase IV: Document

Rationale: Medical information must be accurate and current.

User Stories: SIL-11, SIL-12

Requirement: Editorial process ensures quality.

Acceptance Criteria:

- Medical approval before publishing
- Annual content review
- Visible review dates

RR-08: Session Reliability & State

Priority: Medium

Phase IV: Implement

Rationale: Users shouldn't lose in-progress work.

User Stories: SIL-7, SIL-8

Requirement: Stable sessions and preserved drafts.

Acceptance Criteria:

- Auto-save every 30 seconds (timeout aligned with SR-03)
- Warning 2 minutes before expiry with extend option
- Form data survives refresh
- No data loss during timeout tests

RR-09: API Reliability & Error Handling

Priority: High

Phase IV: Implement

Rationale: Networks fail; the app should not.

User Stories: SIL-7 to SIL-16

Requirement: Handle errors gracefully and retry wisely.

Acceptance Criteria:

- ≥ 95% successful responses under normal load
- Exponential backoff retries (3 attempts)
- Circuit breaker avoids cascades
- Clear user-facing errors

RR-10: Notification Delivery Reliability

Priority: Medium

Phase IV: Simulate

Rationale: Missing a risk alert defeats the purpose.

User Stories: SIL-9, SIL-10

Requirement: Deliver critical alerts reliably and track outcomes.

Acceptance Criteria:

- Queued notifications
- Delivery tracking in DB
- 99% delivery target for critical alerts
- Failed deliveries visible to admins

2.2.4 Security Requirements

SR-01: Encryption at Rest [Conceptual]

Priority: Critical

Phase IV: Document

Rationale: Protect PHI in line with Saudi Personal Data Protection Law (PDPL).

User Stories: SIL-7 to SIL-16

Requirement: Encrypt stored sensitive health data.

Acceptance Criteria:

- AES-256 for PHI fields
- Keys stored separately
- Prototype documents the policy; production uses a KMS

SR-02: Encryption in Transit [Conceptual]

Priority: Critical

Phase IV: Document

Rationale: Prevent eavesdropping/MITM.

User Stories: SIL-7 to SIL-16

Requirement: Enforce HTTPS with modern TLS.

Acceptance Criteria:

- HTTP → HTTPS redirect
- TLS 1.3 enforced
- Valid CA-issued certificate
- HSTS documented for production

SR-03: Authentication

Priority: Critical

Phase IV: Implement

Rationale: Only authorized users should access health data.

User Stories: SIL-15

Requirement: Follow secure authentication practices.

Acceptance Criteria:

- Weak passwords rejected
- Session tokens rotated at login
- Auto-logout after 15 minutes (mirrors RR-08)
- Prototype documents hashing policy; production uses bcrypt ≥ 12 rounds + breached-password checks

SR-04: Role-Based Access Control (RBAC)

Priority: Critical

Phase IV: Implement

Rationale: Enforce least privilege by role.

User Stories: SIL-7 to SIL-16

Requirement: Three roles with appropriate permissions.

Acceptance Criteria:

- Role set at account creation
- Unauthorized access attempts blocked and logged
- UI hides unavailable options
- Server-side role checks on every protected API

SR-05: Consent Management

Priority: High

Phase IV: Implement

Rationale: Ethical/legal basis for processing health data.

User Stories: SIL-7, SIL-15

Requirement: Collect explicit consent before data entry.

Acceptance Criteria:

- Consent screen cannot be bypassed
- Timestamped consent recorded
- Users can withdraw and request deletion
- Disclaimer: “alerts are screening recommendations, not diagnoses”

SR-06: Audit Logging

Priority: Medium

Phase IV: Document

Rationale: Monitoring, forensics, and accountability.

User Stories: SIL-7 to SIL-16

Requirement: Log important security events.

Acceptance Criteria:

- Automatic logging of defined events
- Tamper-resistant storage
- Retention ≥ 90 days

SR-07: Input Validation & Sanitization

Priority: High

Phase IV: Implement

Rationale:Prevent injection and XSS (Cross-Site Scripting).

User Stories: SIL-7, SIL-8, SIL-11, SIL-12

Requirement: Validate and sanitize all inputs.

Acceptance Criteria:

- Server-side validation
- Parameterized DB queries
- Output encoding for user content

SR-08: Privacy by Design – Data Minimization

Priority: High

Phase IV: Implement

Rationale: Reduce risk and comply with Personal Data Protection Law (PDPL).

User Stories: SIL-7 to SIL-16

Requirement: Collect only what's necessary; minimize retention.

Acceptance Criteria:

- Documented justification for each data point
- No unnecessary optional fields
- Anonymous analytics (no individual tracking)

SR-09: Secure Password Reset

Priority: Medium

Phase IV: Implement

Rationale: Password reset must not be an attack vector.

User Stories: SIL-15

Requirement: Harden the reset flow.

Acceptance Criteria:

- Token valid for 1 hour
- Email delivery only
- Account lockout after 5 failed attempts

SR-10: Awareness Hub Content Security

Priority: Medium

Phase IV: Document

Rationale: Keep downloads safe and content trustworthy.

User Stories: SIL-11, SIL-12

Requirement: Secure uploads and content management.

Acceptance Criteria:

- AV scan on uploads
- File-type whitelist
- Admin 2FA for content management
- Approval workflow before publishing

2.3 Security Testing Requirements

Validate security by:

1. Penetration Testing: Independent assessment before production.
2. Automated Scanning: OWASP ZAP (or similar) in CI/CD.
3. Code Review: Focus on auth and data access paths.
4. Compliance Check: Verify against Saudi Personal Data Protection Law (PDPL).

2.4 Traceability Matrix

Requirement ID	Priority	Phase IV Status	Related User Stories	Verification (How We'll Check It Works)
UR-01	Critical	Implement	SIL-7 to SIL-16	Must work properly on phones and tablets (tested on mobile).
UR-02	High	Implement	SIL-7 to SIL-16	The system should be easy to read and navigate for all users.
UR-03	High	Implement	SIL-7,13,14	Users can finish main tasks (like adding data) in under 5 minutes.
UR-04	High	Implement	SIL-7 to SIL-16	The system should fully support Arabic and English without errors.
UR-05	High	Implement	SIL-7,8,15	Error messages must be clear and easy to understand.
UR-06	High	Implement	SIL-7 to SIL-16	Users should reach any feature within 3 clicks.

UR-07	Medium	Implement	SIL-7 to SIL-16	Pages and buttons look consistent across the system.
UR-08	Medium	Implement	SIL-15	New users can understand and finish setup easily.
UR-09	High	Implement	SIL-11,12	Text should be simple and easy to understand (everyday language).
UR-10	Medium	Implement	SIL-7 to SIL-16	The system shows progress or feedback quickly after each action.
UR-11	Medium	Document	SIL-7 to SIL-16	A Help section is available in both Arabic and English.
UR-12	High	Document	SIL-7 to SIL-16	Users should find the system easy to use (high satisfaction score).
RR-01	Critical	Document	SIL-7 to SIL-16	The system should be available most of the time with little downtime.
RR-02	Critical	Document	SIL-7,8,9,10	Data is backed up regularly and can be restored if needed.
RR-03	High	Implement	SIL-7 to SIL-16	If something fails, the rest of the system still works.
RR-04	Critical	Implement	SIL-7,8,9,10	Health data stays correct and does not get lost or changed.
RR-05	High	Implement	SIL-7 to SIL-16	The system runs smoothly on major browsers and devices.

RR-06	High	Document	SIL-7 to SIL-16	The system handles many users at once without crashing.
RR-07	High	Document	SIL-11,12	Health content is reviewed and updated by reliable sources.
RR-08	Medium	Implement	SIL-7,8	User input is saved automatically and not lost.
RR-09	High	Implement	SIL-7 to SIL-16	The system handles connection errors smoothly and shows proper messages.
RR-10	Medium	Simulate	SIL-9,10	Alerts and notifications are delivered correctly to the user.
SR-01	Critical	Document	SIL-7 to SIL-16	Health data stored in the system is protected and private.
SR-02	Critical	Document	SIL-7 to SIL-16	All communication between system parts is secure.
SR-03	Critical	Implement	SIL-15	Login uses strong passwords and logs users out after inactivity.
SR-04	Critical	Implement	SIL-7 to SIL-16	Each user type (citizen, admin, provider) only accesses their data.
SR-05	High	Implement	SIL-7,15	Users give consent before their data is collected or used.

SR-06	Medium	Document	SIL-7 to SIL-16	System keeps activity records for safety and accountability.
SR-07	High	Implement	SIL-7,8,11,12	User input is checked and cleaned before being saved.
SR-08	High	Implement	SIL-7 to SIL-16	System collects only necessary information, nothing extra.
SR-09	Medium	Implement	SIL-15	Password reset is safe and prevents unauthorized access.
SR-10	Medium	Document	SIL-11,12	Uploaded files and content are scanned to make sure they're safe.

2.5 Implementation Summary

Total NFRs: 32

- Critical: 9
- High: 15
- Medium: 8

Phase IV plan:

- Implement: 18 (prototype core)
- Simulate: 1 (notification delivery)
- Document: 13 (production-scale, conceptual)

Prototype focus:

- Responsive family health tree
- Basic risk rules with validation
- Arabic/English content
- Secure authentication

- Reliable session + auto-save
- Clear feedback and messaging

Architectural drivers for Phase III:

- RR-08 Session reliability → session store strategy
- UR-02 Accessibility → UI framework & components
- SR-03 Authentication → security module design
- UR-01 Mobile-first → responsive patterns
- RR-03 Degradation → fault-tolerance approach
- UR-04 Bilingual → i18n/RTL framework

Phase III: Software Initial Design

3.1 Initial Software Design Approach

The Sillah system follows an Object-Oriented Design (OOD) approach, emphasizing modularity, encapsulation, and separation of concerns. The design centers around core domain entities such as User, FamilyMember, HealthEvent, and Clinic, each encapsulating their respective data and behaviors. This approach enables maintainability, extensibility, and clear responsibility assignment across system components.

Key design principles applied include:

- Encapsulation: All class attributes are private with public accessor methods
- Single Responsibility: Each class has a clearly defined, focused purpose
- Loose Coupling: Components interact through well-defined interfaces
- High Cohesion: Related functionality is grouped within the same modules

3.2 Software Architecture

3.2.1 Architectural Overview

Sillah employs a Three-Layer Architecture consisting of Presentation, Business Logic, and Data layers. This layered approach provides clear separation of concerns, enabling independent development, testing, and maintenance of each layer.

3.2.2 Component Responsibilities

Presentation Layer

- Java Console Interface: Handles all user interaction in the prototype through menu-driven console input/output
- Family Health Tree Interface: Manages the display and input for adding/viewing family members and health history
- Risk Dashboard: Presents risk assessment results and alerts to the user
- Awareness Hub Interface: Displays educational content and preventive checklists
- Clinic Booking Interface: Facilitates clinic selection and appointment scheduling

Responsibilities: User input validation, data presentation, user experience flow, and interaction management.

Business Logic Layer

- Risk Calculation Engine: Implements rule-based algorithms to analyze family health patterns and detect hereditary cardiac risks based on criteria such as multiple SCD cases under age 50

- Alert Notification Service: Generates and manages risk alerts, determining appropriate messaging and urgency levels
- Family History Analyzer: Processes and interprets family relationship data to identify hereditary patterns
- Clinic Matching Service: Matches users with appropriate healthcare providers based on location and specialty
- Booking System: Manages appointment scheduling, availability checks, and confirmation processes

Responsibilities: Core business rules, data processing, decision-making, and application logic.

Data Layer

- User Profiles Manager: Handles user account data including authentication information and personal details
- Family History Repository: Manages persistent storage of family member records and health events
- Clinic Database: Maintains information about certified healthcare providers and their availability
- Educational Content Store: Manages awareness materials, articles, and preventive checklists

Responsibilities: Data persistence, retrieval, integrity maintenance, and data access operations.

3.3 Architectural Pattern

3.3.1 Pattern Selection: Layered Architecture

The system implements the Layered Architecture pattern (also known as N-Tier Architecture). This pattern organizes the system into horizontal layers where each layer has a specific role and interacts only with adjacent layers.

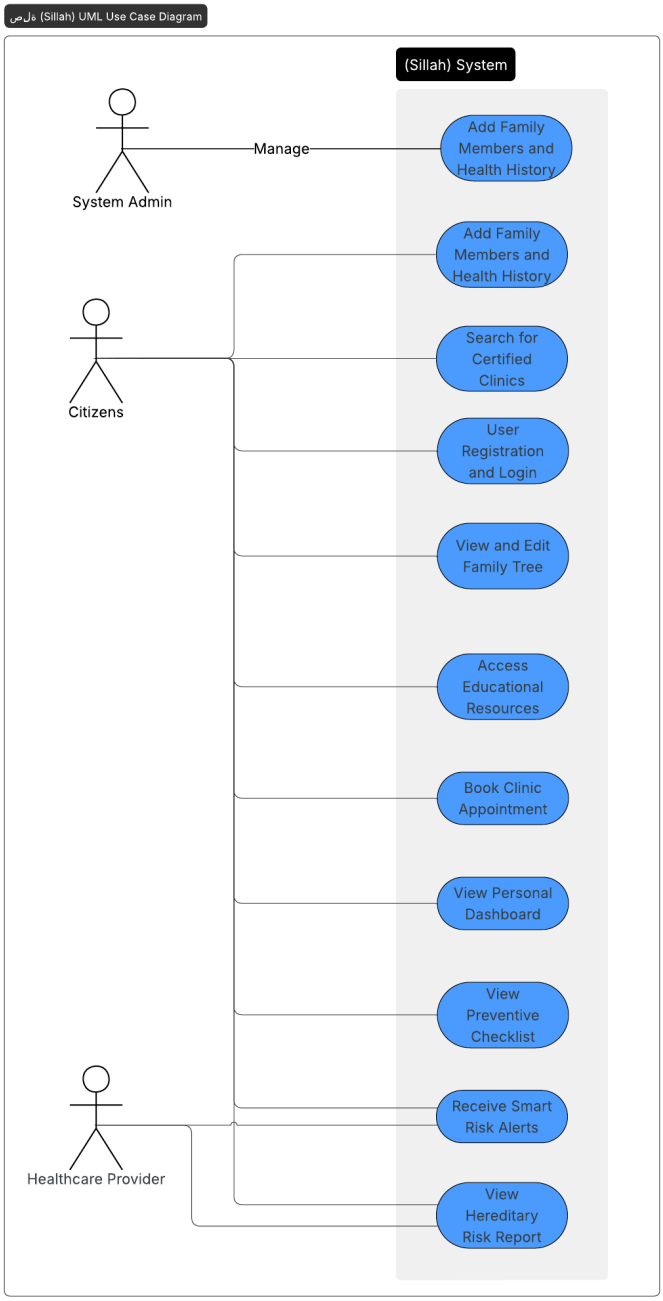
3.3.2 Justification for Pattern Selection

The Layered Architecture pattern was selected for several key reasons:

1. Separation of Concerns: Clear distinction between user interface, business logic, and data management enables specialized development and testing of each layer.
2. Maintainability: Changes in one layer (e.g., switching from console to web interface) minimally impact other layers, reducing maintenance complexity.
3. Testability: Each layer can be tested independently:
    - Presentation Layer: User interaction flows
    - Business Logic: Risk algorithms and business rules
    - Data Layer: Data storage and retrieval operations
4. Team Development: The layered structure allows parallel development by team members with different specializations (UI, business logic, database).
5. Scalability Path: While the prototype uses in-memory data structures, the layered design facilitates seamless transition to database persistence and web-based interfaces in future iterations.

6. Academic Alignment: This pattern effectively demonstrates fundamental software engineering principles of modularity and separation of concerns, aligning with SE201 course objectives.

The architecture supports the system's functional requirements while providing a foundation for future enhancements such as web interfaces, mobile applications, and integration with healthcare provider systems.



3.4 UML Design

3.4.1 Use Case Diagram

The **Sillah (صلة) Use Case Diagram** illustrates major interactions of the system with its three actors: **Citizens**, **Healthcare Providers**, and **System Administrators** — each playing distinct roles within the platform.

Actors:

- **Citizens:** Log family medical histories, access checklists, receive risk alerts, and schedule clinic appointments.
- **Healthcare Providers:** Access hereditary risk reports and respond to system alerts.
- **Administrators:** Manage users, clinics, and awareness content.

System Scope:

The diagram represents the **Sillah Family Health Management System**, encompassing preventive care, hereditary risk detection, and awareness education.

Primary Use Cases:

- Register and log in
- Add and edit family health history
- View family tree
- Search and book certified clinics
- Access awareness resources and preventive checklist
- Get and view smart risk alerts
- Manage users and system data (admin)

Summary:

The diagram defines the functional boundaries of the system and shows how each actor interacts with Sillah (صلة) to achieve preventive health goals through education, data entry, and clinic integration.

3.4.2 Tabular Use Case Descriptions

Use Case – Add Family Member

Field	Description
Primary Actor	Citizen (User)
Goal	To allow the user to record a family member's basic information and health condition.
Preconditions	Users must be logged into the system.

Main Flow	1. The user selects "Add Family Member." 2. System prompts for relation, age, and health condition. 3. The user enters details. 4. The system validates input data. 5. The system saves the new family member.
Alternative Flow	If data is invalid (e.g., empty fields or impossible age), the system displays an error message and prompts for correction.
Postcondition	The new family member record is successfully saved and displayed in the user's family tree.

Use Case – Generate Risk Alert

Field	Description
Primary Actor	System
Goal	To automatically analyze family data and determine whether hereditary cardiac risk exists.
Preconditions	Family members and their health conditions are recorded in the system.
Main Flow	1. The system scans all recorded family members. 2. It checks for patterns (e.g., two or more relatives with SCD under age 50). 3. If a pattern is found, a risk alert is generated.

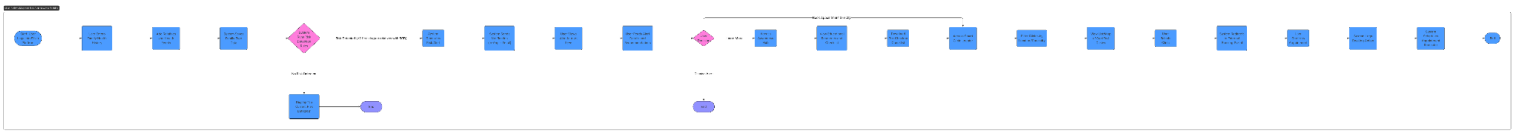
	4. An alert is displayed to the user.
Alternative Flow	If no risk is detected, the system displays a “No immediate hereditary risk detected” message.
Postcondition	Risk level (Low, Moderate, or High) is shown to the user and can trigger a clinic recommendation.

Use Case – Book Clinic Appointment

Field	Description
Primary Actor	Citizen (User)
Goal	To book a preventive health screening appointment at an available clinic.
Preconditions	The user has received a risk alert and clinic options are available.
Main Flow	1. The user selects a recommended clinic. 2. The system checks clinic availability. 3. User confirms preferred time slot. 4. The system books the appointment and confirms details.
Alternative Flow	If no appointment slots are available, the system suggests alternative clinics or future dates.

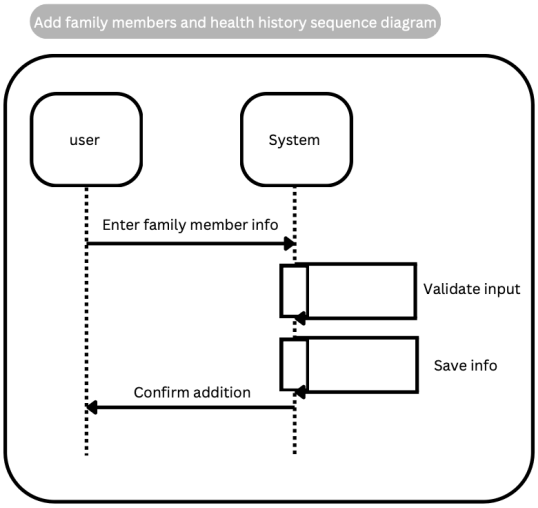
Postcondition	Appointment confirmation message is shown and stored in the user's record.

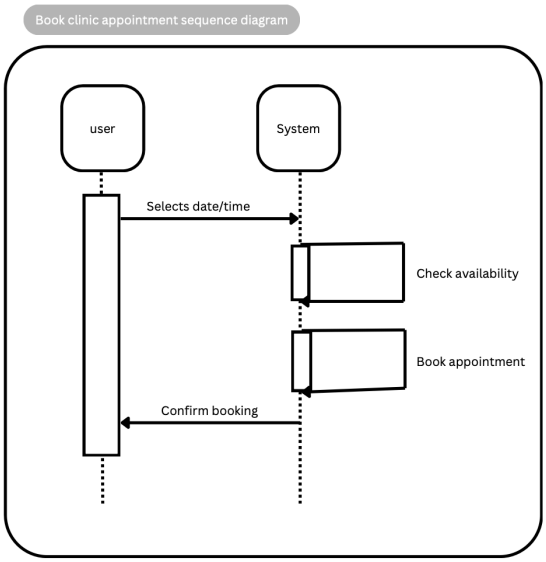
3.4.3 Activity Diagram

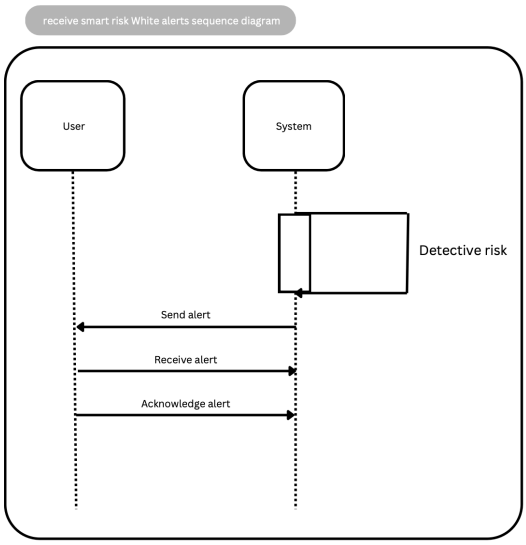


- The diagram will be provided in the zip file for better quality.

3.4.4 Sequence Diagrams

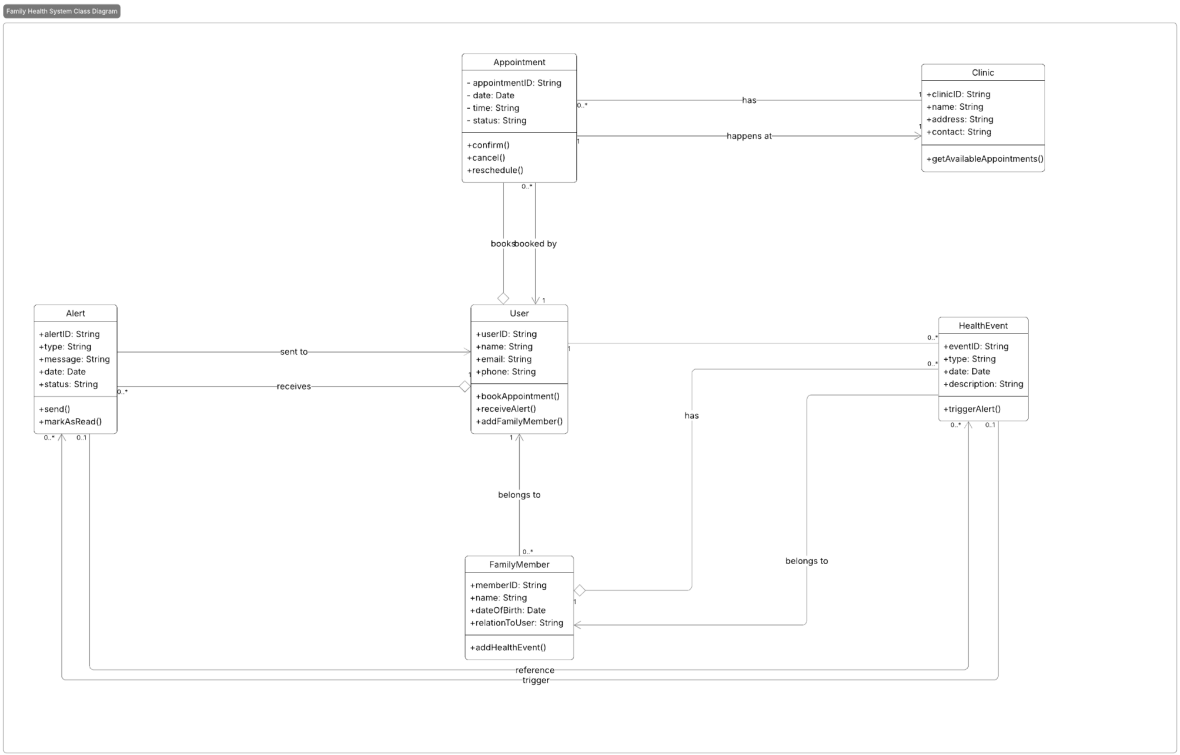




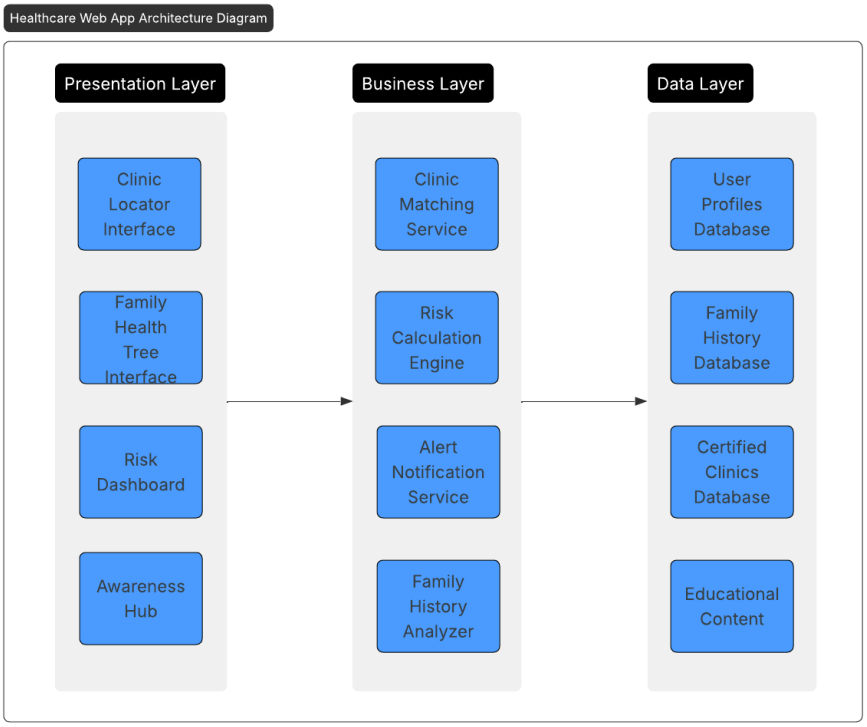


Each SSD demonstrates clear messages and lifelines without redundant user/system boxes.

3.4.5 Class Diagram



3.4.6 System Architecture Diagram



Phase IV - Prototype and Testing

4.1 Prototype Implementation (Java)

The Phase IV prototype was implemented in **Java**, demonstrating the interaction between these layers:

- The **Presentation Layer** (console interface) interacts with users.
- The **Business Layer** handles logic such as risk assessment, alerts, and appointment booking.
- The **Data Layer** uses in-memory structures (lists and classes) for simulation instead of a database.

4.2 Awareness Hub Simulation

The Awareness Hub prototype simulates educational and preventive features through:

- Menu-based navigation for viewing health topics.
- A simple checklist that tracks user awareness progress.
- Educational summaries related to hereditary disease prevention.

4.3 Testing & Results

Test Case	Input	Expected Output	Result
TC-01	Enter 1 SCD relative	"No immediate hereditary risk detected."	passed
TC-02	Enter 2 SCD relatives under 50	"High Risk: Please schedule preventive screening."	passed
TC-03	Awareness Hub menu	Displays educational content and checklist	passed

4.4 Conclusion

The Sillah system successfully integrates software engineering concepts across all SE201 phases — from requirements and design to implementation and testing.

The team achieved a fully functional Java prototype reflecting preventive health awareness, risk detection, and basic system reliability.

Future improvements could include a web-based GUI, secure authentication, and database persistence.

4.5 References

Alomran, S. F. (2025). *Sillah (صلة): Preventive Family Health Risk Detection System* [Java project]. Prince Sultan University, SE201 – Introduction to Software Engineering.

https://github.com/Shoug-Alomran/SE201--Sillah-Project

https://se-201-sillah-project-eb6q97l6e-shoug-alomrans-projects.vercel.app

Atlassian Jira (2025). *Sillah Project Management Board.* Prince Sultan University – SE201 Course.

https://sillah.atlassian.net

Appendix A

Main Class

```
import java.util.Scanner;

public class Main {

   private static final String title = "=== Sillah (صلة) Preventive Health System
===";

   public static void main(String[] args) {

       Scanner input = new Scanner(System.in);

       while (true) {

           printMenu();

           String choice = input.nextLine().trim();

           if ("1".equals(choice)) {

               runLowRiskDemo();

           } else if ("2".equals(choice)) {

               runHighRiskDemo(); // high-risk alert scenario

           } else if ("3".equals(choice)) {

               AwarenessHub hub = new AwarenessHub();

               hub.start(input);
```

```
            } else if ("0".equals(choice)) {
                System.out.println("Goodbye!");
                break;
            } else {
                System.out.println("Invalid choice. Please select 1, 2, 3, or 0.");
            }
            System.out.println();
            System.out.println("Press ENTER to return to the menu...");
            input.nextLine();
        }
       input.close();
    }
    private static void printMenu() {
        System.out.println();
        System.out.println(title);
        System.out.println("Select a demo scenario:");
        System.out.println("  1) Low/No Risk family (baseline)");
        System.out.println("  2) High Risk family (>=2 SCD relatives under 50)");
        System.out.println("  3) Awareness Hub (education + checklist)");
        System.out.println("  0) Exit");
        System.out.print("Enter choice: ");
    }
    // === Scenario 1: Your existing baseline (no immediate hereditary risk) ===
    private static void runLowRiskDemo() {
        System.out.println();
        System.out.println("--- Demo: Low/No Risk Family ---");
        User user = new User("Shoug Alomran");
        FamilyMember f1 = new FamilyMember("Father", 55, "Healthy");
```

```
        FamilyMember f2 = new FamilyMember("Brother", 30, "Healthy");
        user.addFamilyMember(f1);
        user.addFamilyMember(f2);

        runFlow(user);
    }
    // === Scenario 2: High-risk example (2 relatives with SCD under 50) ===
    private static void runHighRiskDemo() {
        System.out.println();
        System.out.println("--- Demo: HIGH Risk Family ---");
        User user = new User("Shoug Alomran");
        // Two relatives with SCD and age < 50 → should trigger High Risk alert
        FamilyMember f1 = new FamilyMember("Father", 45, "SCD");
        FamilyMember f2 = new FamilyMember("Brother", 30, "SCD");
        FamilyMember f3 = new FamilyMember("Mother", 48, "Healthy"); // optional
        user.addFamilyMember(f1);
        user.addFamilyMember(f2);
        user.addFamilyMember(f3);
        runFlow(user);    }
    // Shared method: risk check → recommend clinic → book appointment
    private static void runFlow(User user) {
        System.out.println(title);
        AlertSystem alertSystem = new AlertSystem();
        String alert = alertSystem.checkRisk(user);
        System.out.println(alert);
        BookingSystem bookingSystem = new BookingSystem();
        Clinic clinic = bookingSystem.findClinic("Riyadh Heart Center");
        if (clinic != null) {
```

```
            System.out.println("Recommended Clinic: " + clinic.getName());

        } else {

            System.out.println("No clinic found. Using default clinic.");

            clinic = new Clinic("General Health Clinic", "Riyadh");

        }

        System.out.println("Booking appointment...");

        Appointment appointment = bookingSystem.bookAppointment(user, clinic);

       System.out.println("Appointment booked for " + appointment.getUser().getName()
+ " at " + appointment.getClinic().getName());

        System.out.println("=== End of Simulation ===");

    }

}
```

Message Class

```
enum Lang {

    EN, AR

}

public class Message {

    private final Lang lang;

    public Message(Lang lang) {

        this.lang = lang;

    }

    public String title() {

        return lang == Lang.AR ? "=== صلة: نظام الصحة الوقائية ===" : "=== Sillah (صلة)
Preventive Health System ===";

    }
```

```
    public String highRisk() {
        return lang == Lang.AR ? "مخاطر عالية: يرجى حجز فحص وقائي."
                : "High Risk: Please schedule a preventive screening.";
    }
    public String moderateRisk() {
        return lang == Lang.AR ? "مخاطر متوسطة: يُنصح باستشارة طبية."
                : "Moderate Risk: Consider a medical consultation.";
    }
    public String lowRisk() {
        return lang == Lang.AR ? "لا توجد مخاطر وراثية فورية." : "No immediate hereditary
risk detected.";
    }
    public String recommendedClinic() {
        return lang == Lang.AR ? "العيادة الموصى بها: " : "Recommended Clinic: ";
    }
    public String end() {
        return lang == Lang.AR ? "=== نهاية المحاكاة ===" : "=== End of Simulation ===";
    }
}
```

User Class

```
import java.util.*;

public class User {
    private String name;
    private List<FamilyMember> familyMembers = new ArrayList<>();

    // Validation
    public User(String name) {
        if (name == null || name.trim().isEmpty()) {
```

```
            throw new IllegalArgumentException("Name required");
        }
        this.name = name.trim();
    }

    // Validation
    public void addFamilyMember(FamilyMember fm) {
        if (fm != null)
            familyMembers.add(fm);
    }

    public List<FamilyMember> getFamilyMembers() {
        return Collections.unmodifiableList(familyMembers);
    }

    public String getName() {
        return name;
    }
}
```

Health Event Class

```
public class HealthEvent {

    private String condition;

    private int ageAtDiagnosis;

    private String description;

    public HealthEvent(String condition, int ageAtDiagnosis, String description) {

        if (condition == null || condition.trim().isEmpty()) {

            throw new IllegalArgumentException("Condition required");

        }

        if (ageAtDiagnosis < 0 || ageAtDiagnosis > 120) {

            throw new IllegalArgumentException("Age must be between 0 and 120");

        }
```

```
        this.condition = condition.trim();
        this.ageAtDiagnosis = ageAtDiagnosis;
        this.description = (description == null) ? "" : description.trim();
    }
    public String getCondition() {
        return condition;
    }
    public int getAgeAtDiagnosis() {
        return ageAtDiagnosis;
    }
    public String getDescription() {
        return description;
    }
    @Override
    public String toString() {
        return condition + " (diagnosed at age " + ageAtDiagnosis + ")";
    }
}
```

Family Members Class

```
import java.util.*;
public class FamilyMember {
    private String relation;
    private int age;
    private List<HealthEvent> healthEvents = new ArrayList<>();
    // Constructor with validation
```

```
    public FamilyMember(String relation, int age) {
        if (relation == null || relation.trim().isEmpty()) {
            throw new IllegalArgumentException("Relation required");
        }
        if (age < 0 || age > 120) {
            throw new IllegalArgumentException("Age must be between 0 and 120");
        }
        this.relation = relation.trim();
        this.age = age;
    }
    // Adds an initial HealthEvent automatically.
    public FamilyMember(String relation, int age, String healthCondition) {
        this(relation, age);
        if (healthCondition != null && !healthCondition.trim().isEmpty()) {
            // Use a simple guessed ageAtDiagnosis (you can change it when you know
the real
            // age)
            addHealthEvent(
                    new HealthEvent(healthCondition.trim(), Math.max(0, Math.min(age,
50)), "Initial condition"));
        }
    }
    public String getRelation() {
        return relation;
    }
    public int getAge() {
        return age;
    }
    public void addHealthEvent(HealthEvent event) {
```

```
        if (event != null) {
            healthEvents.add(event);
        }
    }
    public List<HealthEvent> getHealthEvents() {
        return Collections.unmodifiableList(healthEvents);
    }
    public String getHealthCondition() {
        return healthEvents.isEmpty() ? "Healthy" :
healthEvents.get(0).getCondition();
    }
    @Override
    public String toString() {
        return relation + " (Age: " + age + ", Events: " + healthEvents.size() + ")";
    }
}
```

Clinic Class

```
public class Clinic {
    private String name;
    private String location;
    public Clinic(String name, String location) {
        this.name = name;
        this.location = location;
    }
    public String getName() {
```

```
        return name;
    }
    public String getLocation() {
        return location;
    }
}
```

Booking System Class

```
public class BookingSystem {
    public Clinic findClinic(String name) {
        if (name != null && name.equalsIgnoreCase("Riyadh Heart Center"))
            return new Clinic("Riyadh Heart Center", "Riyadh");
        return new Clinic("General Health Clinic", "Riyadh");
    }
    // Book an appointment for the user at the specified clinic while validating
    // inputs
    public Appointment bookAppointment(User user, Clinic clinic) {
        if (user == null || clinic == null)
            throw new IllegalArgumentException("User and clinic are required");
        System.out.println("Booking appointment...");
        return new Appointment(user, clinic);
    }
}
```

Awareness Hub Class

```
import java.util.Scanner;

public class AwarenessHub {

   public void start(Scanner input) {

       while (true) {

           System.out.println("\n=== Awareness Hub ===");

           System.out.println("1) View Educational Topics");

           System.out.println("2) Preventive Checklist");

           System.out.println("0) Back to main menu");

           System.out.print("Enter choice: ");

           String choice = input.nextLine().trim();

           if ("1".equals(choice)) {

               showTopics(input);

           } else if ("2".equals(choice)) {

               showChecklist();

           } else if ("0".equals(choice)) {

               System.out.println("Returning to main menu...");

               break;

           } else {

               System.out.println("Invalid choice.");

           }

       }

   }

   private void showTopics(Scanner sc) {

       System.out.println("\nAvailable Topics:");

       System.out.println("1) Understanding Sickle Cell Disease");

       System.out.println("2) Importance of Genetic Screening");

       System.out.println("3) Healthy Heart Tips");
```

```
        System.out.print("Select topic: ");

        String choice = input.nextLine().trim();

        if ("1".equals(choice)) {

          System.out.println("SCD affects red blood cells and can be hereditary...");

        } else if ("2".equals(choice)) {

        System.out.println("Genetic screening helps detect hereditary risks early.");

        } else if ("3".equals(choice)) {

     System.out.println("Maintain a balanced diet, exercise, and regular
checkups.");

        } else {

            System.out.println("Invalid topic.");

        }

    }

    private void showChecklist() {

        System.out.println("\n=== Preventive Checklist ===");

        System.out.println("[✓] Add family members to your health tree");

        System.out.println("[ ] Schedule your first screening");

        System.out.println("[ ] Read one Awareness article");

        System.out.println("Progress: 1/3 completed");

    }

}
```

Appointment Class

```
public class Appointment {

    private User user;

    private Clinic clinic;
```

```
    public Appointment(User user, Clinic clinic) {
        this.user = user;
        this.clinic = clinic;
    }

    public User getUser() {
        return user;
    }

    public Clinic getClinic() {
        return clinic;
    }
}
```

Alert System Class

```
public class AlertSystem {
    public String checkRisk(User user) {
        int earlySCDCount = 0;
        for (FamilyMember fm : user.getFamilyMembers()) {
            for (HealthEvent event : fm.getHealthEvents()) {
                if ("SCD".equalsIgnoreCase(event.getCondition()) &&
event.getAgeAtDiagnosis() < 50) {
                    earlySCDCount++;
                }
            }
        }
```

```
        if (earlySCDCount >= 2) {

            return "High Risk: Multiple early SCD cases detected. Please schedule a
preventive screening.";

        } else if (earlySCDCount == 1) {

            return "Moderate Risk: One hereditary case detected. Consult your doctor
for screening.";

        } else {

           return "No immediate hereditary risk detected.";

        }

    }

}
```

Code Simulation

=== Sillah (صلة) Preventive Health System ===

Select a demo scenario:

1) Low/No Risk family (baseline)

2) High Risk family (>=2 SCD relatives under 50)

3) Awareness Hub (education + checklist)

0) Exit

Enter choice: 1

--- Demo: Low/No Risk Family ---

=== Sillah (صلة) Preventive Health System ===

No immediate hereditary risk detected.

Recommended Clinic: Riyadh Heart Center

Booking appointment...

Booking appointment...

Appointment booked for Shoug Alomran at Riyadh Heart Center

=== End of Simulation ===

Press ENTER to return to the menu...

=== Sillah (صلة) Preventive Health System ===

Select a demo scenario:

1) Low/No Risk family (baseline)

2) High Risk family (>=2 SCD relatives under 50)3) Awareness Hub (education + checklist)

0) Exit

Enter choice: 2

--- Demo: HIGH Risk Family ---

=== Sillah (صلة) Preventive Health System ===

High Risk: Multiple early SCD cases detected. Please schedule a preventive screening.

Recommended Clinic: Riyadh Heart Center

Booking appointment...

Booking appointment...

Appointment booked for Shoug Alomran at Riyadh Heart Center

=== End of Simulation ===

Press ENTER to return to the menu...

=== Sillah (صلة) Preventive Health System ===

Select a demo scenario:

1) Low/No Risk family (baseline)

2) High Risk family (>=2 SCD relatives under 50)

3) Awareness Hub (education + checklist)

0) Exit

Enter choice: 3

=== Awareness Hub ===

1) View Educational Topics2) Preventive Checklist

0) Back to main menu

Enter choice: 1

Available Topics:

1) Understanding Sickle Cell Disease

2) Importance of Genetic Screening

3) Healthy Heart Tips

Select topic: 1

SCD affects red blood cells and can be hereditary...

=== Awareness Hub ===

1) View Educational Topics

2) Preventive Checklist

0) Back to main menu

Enter choice: 1

Available Topics:

1) Understanding Sickle Cell Disease

2) Importance of Genetic Screening

3) Healthy Heart Tips

Select topic: 2

Genetic screening helps detect hereditary risks early.

=== Awareness Hub ===

1) View Educational Topics2) Preventive Checklist

0) Back to main menu

Enter choice: 1

Available Topics:

1) Understanding Sickle Cell Disease

2) Importance of Genetic Screening

3) Healthy Heart Tips

Select topic: 3

Maintain a balanced diet, exercise, and regular checkups.

=== Awareness Hub ===

1) View Educational Topics

2) Preventive Checklist

0) Back to main menu

Enter choice: 2

=== Preventive Checklist ===

[✓] Add family members to your health tree

[ ] Schedule your first screening

[ ] Read one Awareness article

Progress: 1/3 completed

=== Awareness Hub ===

1) View Educational Topics

2) Preventive Checklist0) Back to main menu

Enter choice: 0

Returning to main menu...

Press ENTER to return to the menu...

=== Sillah (صلة) Preventive Health System ===

Select a demo scenario:

1) Low/No Risk family (baseline)

2) High Risk family (>=2 SCD relatives under 50)

3) Awareness Hub (education + checklist)

0) Exit

Enter choice: 0

Goodbye!

Appendix B

To support project planning and agile task tracking, the *Sillah (صلة)* team utilized **Atlassian Jira** for sprint management and backlog organization.

The board below represents **SIL Sprint 1**, containing 10 user stories aligned with system functionalities defined in the requirements and design phases.

Epic	User Stories	Description
Family Health Tree	SIL-7, SIL-8	Add and view family members, record health history
Risk Alert System	SIL-9, SIL-10	Generate hereditary risk alerts and display risk reports
Awareness Hub	SIL-11, SIL-12	Display educational content and preventive health checklist
Clinic Locator & Booking	SIL-13, SIL-14	Search for certified clinics and book appointments
User Management & Dashboard	SIL-15, SIL-16	Registration, login, and personal dashboard view

This setup ensured every requirement (functional and non-functional) was represented as a Jira issue, enabling traceability from the *requirements phase* to *prototype implementation*.