جامعة الأمير سلطان
PRINCE SULTAN UNIVERSITY

Sillah Phase 1

CS340: Introduction to Databases Systems

Section: 1629

Instructor: Maram Alajlan

Prepared by:

Yara Albugami - 223410190

Rose Alrakan - 223410382

Shoug Alomran - 223410392

Raghad Abdulaziz - 222511358

Table of contents

1. Application idea **2**
1.1 Application Purpose and Scope *2*
1. Users **2**
2.1 Target users and usage scenarios *2*
2. System Architecture **3**
3.1 Data requirements *3*
3.2 Functional and Non-functional system requirements *4*
Functional Requirement 4
Non-Functional Requirement 4
1. Usability: 4
2. Reliability: 5
3.3 High-level system Architecture *5*
Software Architecture 5
Architectural Pattern 6
3.4 Paper prototype and UI wireframes *6*
3.4.1 Design Principles 6
3.4.2 Main Screens Overview 7
1. Login / Registration Screen 7
2. User Dashboard (Citizen) 7
3. Add / Edit Family Member Screen 8
4. Risk Alerts Screen 8
5. Clinic Booking Screen 9
6. Awareness Hub Screen 9
3.4.3 Admin Interface (High-Level) 10
3.4.4 Prototype Scope and Limitations 10
3.4.5 Summary 10
3.5 Teamwork strategy and task distribution *11*
3.5.1 Collaboration Tools and Workflow 11
3.5.2 Coordination, Review, and Accountability 11
3.6 Individual contribution description *12*

1. Application idea

1.1 Application Purpose and Scope

Sillah is a preventive family health management system designed to support Saudi families with hereditary health issues, including cardiac and other hereditary diseases. The system enables users to record family health histories, generate automated hereditary risk warnings, and schedule preventive clinic visits. The purpose of the application is to raise preventive health awareness, simplify family health management, and support early risk detection through data analysis.

The application manages user data, family members, health events, risk warnings, clinic information, appointments, and health awareness content using a relational database management system. It focuses on storing and analyzing health-related data to support preventive decision-making across multiple hereditary conditions. Integration with real hospital systems, real-time medical diagnoses, and live data exchange with healthcare providers are outside the scope of this project. The system is intended solely for educational and preventive purposes and does not provide medical diagnoses or treatments.

1. Users

2.1 Target users and usage scenarios

Sillah's target audience includes general citizens, healthcare providers, and administrators. To elaborate further, families are the main users of the application. To list a few, the system can send an alert once any detection of hereditary risks comes up. Another use scenario includes the users ability to input a specific condition and age of onset.

2. System Architecture

3.1 Data requirements

Data Category	Description
User Data	Saves users data such as: user ID, email address, full name, role (family member, administrator, healthcare provider) and login credentials.
Family Member Data	Includes details about each family member, such as their: relationship, name, gender, linked family group, and date of birth.
Medical History Data	Documents hereditary and non-hereditary conditions, condition type, age of onset, severity level, and associated family member.
Health Event Data	Store users – entered health- related events, including past modifications, overtime, and condition updates.
Risk Alert Data	Keeps track of created alerts, including alert types, detection, date, description, and alert status (active or resolved).
Appointment Data	Keeps track of appointment details including the time and date of the appointment, the clinic of choice, booking status, and confirmation details.
Awareness Content Data	Contains health awareness, materials, including educational articles, health advice, and preventive checklists.
Audit and Log Data	Records system activities, including user actions, login attempts, and modifications for security monitoring and troubleshooting.

3.2 Functional and Non-functional system requirements

Functional Requirement

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

Non-Functional Requirement

The main goal for the Sillah Family Health Management System is to ensure that the application is easy to use, robust, and most importantly safe for all users. For a more clear understanding, we further delved into the three main categories of the non-functional requirements as follows.

1. **Usability:**

ID	Title	Requirement
UR-01	Mobile-First Responsive Design	Interface must work across screen sizes, prioritizing mobile
UR-04	Bilingual Interface (Arabic & English)	Full bilingual support

UR-05	Clear Error Messages & Guidance	Errors must be specific, polite, and near the relevant field

2. Reliability:

ID	Title	Requirement
RR-01	Availability & Uptime	High availability with minimal unplanned downtime
RR-02	Data Backup & Restore	Define backup schedule and recovery targets
RR-03	Data Accuracy & Integrity	Protect data quality and detect risks accurately

3. Security:

ID	Title	Requirement
SR-03	Authentication	Follow secure, authentication practices
SR-04	Role-Based Access Control (RBAC)	Three roles with appropriate permissions
SR-05	Secure Input Validation & Output Encoding	Validate on server side; encode all untrusted output

3.3 High-level system Architecture

Software Architecture

Aiming to find an approach that provides a more clear separation between layers for overall development, testing and maintenance, Sillah employs a three-layer architecture. The architecture layers consist of presentation layer, business logic layer, and lastly the data layer.

Layer	Function
Presentation layer	Handles all user interactions
Business logic/domain layer	Implements business logic and functionality of system
Data/integration layer	Represents future connectivity with external systems

Architectural Pattern

Sillah's system implements the Layered Architecture pattern, which organizes the system into horizontal layers that only interact with adjacent layers. Main reasons for this choice of pattern was separation of concerns, maintainability, and testability.

3.4 Paper prototype and UI wireframes

In this section, the paper prototype and the UI wireframes for the Sillah Family Health Management System are presented. The purpose of the wireframes is to show the interaction between the users and the system, as well as how the users will interact with the underlying database using the application interface.

The wireframes emphasize simplicity and direct mapping to the database operations CRUD. The wireframes were created to accommodate the major functional requirements as discussed in Phase 1.

3.4.1 Design Principles

The paper prototype will follow the following guidelines:

- **Database-first interaction:** All screens will be associated with one or more database operations (INSERT, SELECT, UPDATE).
- **Minimal navigation depth:** All functionalities will be accessible within 2-3 levels of navigation.
- **Role-based access:** All screens will differ based on the roles (Citizen, Healthcare Provider, Admin).
- **Mobile-friendly design:** Vertical design for mobile and web versions.

3.4.2 Main Screens Overview

The following wireframes represent the **minimum viable interface** required to support the system's functionality.

1. Login / Registration Screen

Purpose:

Authenticate users and assign roles.

Main Elements:

- Email / Username field
- Password field
- Login button
- Registration link

Database Interaction:

- SELECT user credentials from `USER`
- Role retrieval to determine access permissions

2. User Dashboard (Citizen)

Purpose:

Provide a central overview of the user's family health data and alerts.

Main Elements:

- Summary of family members
- Latest risk status (Low / Moderate / High)
- Quick action buttons:
    - Add Family Member
    - View Alerts

- Book Appointment
- Awareness Hub

Database Interaction:

- SELECT from `FAMILY_MEMBER`
- SELECT from `APPOINTMENT`
- SELECT from `ALERT`

3. Add / Edit Family Member Screen

Purpose:

Allow users to record family members and their health information.

Main Elements:

- Relation (e.g., Father, Mother, Sibling)
- Health condition (dropdown / text)
- Age
- Save button

Database Interaction:

- INSERT into `FAMILY_MEMBER`
- UPDATE existing records if edited
- INSERT into `HEALTH_EVENT`

4. Risk Alerts Screen

Purpose:

Display generated hereditary risk alerts and explanations.

Main Elements:

- List of alerts with timestamps
- Short explanation of why the alert was generated
- Risk level (Low / Moderate / High)

Database Interaction:

- SELECT from `ALERT`
- Aggregation queries on `HEALTH_EVENT` (used to determine risk level)

5. Clinic Booking Screen

Purpose:

Enable users to schedule preventive screening appointments.

Main Elements:

- List of available clinics
- Location and specialty
- Available appointment slots
- Confirm booking button

Database Interaction:

- SELECT from `CLINIC`
- INSERT into `APPOINTMENT`
- Foreign key validation (User ↔ Clinic)

6. Awareness Hub Screen

Purpose:

Provide educational content and preventive checklists.

Main Elements:

- List of health topics
- Article summaries
- Preventive checklist with completion status

Database Interaction:

- SELECT from `AWARENESS_CONTENT`
- SELECT / UPDATE checklist progress

3.4.3 Admin Interface (High-Level)

Purpose:

Allow administrators to manage system data.

Main Elements:

- User management
- Clinic management
- Awareness content management

Database Interaction:

- INSERT / UPDATE / DELETE operations across multiple tables
- Role-based access enforcement

3.4.4 Prototype Scope and Limitations

The paper prototype represents **logical screen flow only**.

It does not include:

- Final visual styling
- Animations
- Real hospital system integration

These wireframes serve as a **conceptual bridge** between system requirements and the database-backed implementation that will be developed in later phases.

3.4.5 Summary

The paper prototype and UI wireframes offer a clear visualization of how users will interact with the Sillah system and how each interaction will directly relate to database operations.

They ensure that the interface design will support the underlying relational database structure and prepare the system for implementation in the next phases.

3.5 Teamwork strategy and task distribution

This project is developed collaboratively by a four-member team through the implementation of a structured teamwork strategy that promotes clear responsibility allocation, constant progress through the phases of the project, and individual accountability.

This team works collaboratively through a phase-based task distribution strategy in which the allocated roles are given on the basis of the individual team members' strengths.

3.5.1 Collaboration Tools and Workflow

To support effective teamwork, the following tools and practices are used:

- **GitHub & Project Documentation Website:** GitHub is utilized for version control of all documentation and code artifacts. Moreover, a project documentation website is maintained to provide a structured and accessible format for all project phases, diagrams, and technical information.
- **WhatsApp Group**: Primary communication channel for coordination, quick clarifications, and scheduling.
- **Google Docs**: Collaborative drafting and review of written deliverables.

All team members review major deliverables before submission to ensure consistency and correctness.

3.5.2 Coordination, Review, and Accountability

The group has a formal coordination mechanism that is used to ensure that there is constant progress and individual accountability during the entire process. In addition, we have a plan for the tasks at the start and finish of each phase. There are internal checks to ensure that the tasks are in line with the needs of the course requirements.

There is also a review of the work done by another individual before the work is submitted to ensure that the work is correct, consistent, and complete.

3.6 Individual contribution description

The following table summarizes the individual contributions of each team member across all project phases. Responsibilities reflect both completed work and planned contributions for later phases.

Member	Contribution Description
Yara Albugami	**Phase 1:** Users and system requirements analysis **Phase 2:** Core entities, attributes, strong/weak entities, and key identification **Phase 3:** EER design, table structure, and primary/foreign key definition **Phase 4:** SQL scripts for core tables and integrity constraints **Phase 5:** Implementation of core backend features, development of several SQL queries, and main data entry screens
Shoug Alomran	**Phase 1**: System architecture design and UI wireframes **Phase 2:** Relationships, cardinality, and participation constraints **Phase 3:** Data dictionary, attribute domains, and integrity rules **Phase 4:** SQL scripts for relationship and transaction tables, foreign keys **Phase 5:** Backend logic for alerts and appointments, development of related SQL queries, and frontend integration
Rose Al Rakan	**Phase 1:** Data requirements analysis and team planning **Phase 2:** Specialization/generalization and EER refinements **Phase 3:** Relational schema mapping, normalization, and relational diagrams **Phase 4:** SQL scripts for support and lookup tables, validation constraints **Phase 5:** Implementation of awareness content module, development of related SQL queries, and UI integration

Raghad Abdulaziz	**Phase 1:** Documentation refinement and scope validation **Phase 2:** Review and validation of EER model and constraints **Phase 3:** Schema consistency checks and design quality review **Phase 4:** Data insertion scripts, test data preparation, and query testing **Phase 5:** Contribution to SQL query development, system testing, debugging, and full frontend-backend integration