Software Analysis & Design Project

Team Members:
- John Majdi Fahim Zawaiedeh 2336871
- Abdulwahab Anwar Mustafa Shdaifat 2338071
- Murad Shehab Mohammed AlZyoud 2241528

System Overview

This document describes the Mobile Library Assistant application, a mobile system that allows Hashemite University users to search for books, reserve existing books, propose new titles for purchase, and manage hall reservations for events inside the library. The system targets smartphones and focuses on a simple, intuitive interface suitable for non-technical users.

Stakeholders and Users

- Library members (students, academic staff, employees) who use the app to search, reserve, suggest books, and book halls.
- Librarians who manage books, suggestions, and hall schedules using the back-end or existing library tools.
- University administration, which is interested in better utilization of library resources and halls.

Functional Requirements

1. Search for a book
   - The system shall allow the user to search for books by title, author, category, or keyword.
   - The system shall display a list of matching books with key details (title, author, availability status, location).
   - The system shall allow the user to open a book details page showing description and current reservation status.
2. Book an existing book

- The system shall allow the user to select an available book and submit a reservation request.
- The system shall verify that the book is currently available for reservation before confirming.
- The system shall create a reservation record linked to the user and update the book status to "reserved" for the selected period.

3. Suggest a book to be bought by the library
   - The system shall allow the user to submit a suggestion form for a new book, including at least title, author, publisher (if known), and justification.
   - The system shall store the suggestion and mark it as "pending review" for librarians.
   - The system shall notify the user (inside the app or via simple status) that the suggestion has been recorded.

4. Search for a suitable hall to make an event
   - The system shall allow the user to define basic event information: date, start time, end time, expected number of attendees, and event type (e.g., children, disability-friendly, general).
   - The system shall search available halls, including: American Corner, Prince Hussain Corner, Disabilities Corner, Children Corner, Khadija Corner, and Wikipedia Corner.
   - The system shall return only halls that match capacity, type, and availability in the requested time period.

5. Book a time and a hall
   - The system shall allow the user to select one of the suggested available halls and a specific time slot.
   - The system shall check for conflicting reservations before confirming the booking.
   - The system shall store the reservation linked to the user and selected hall and show a confirmation summary (hall, date, time).

Nonfunctional Requirements

Usability:
- The application shall provide a clear, mobile-friendly interface with simple navigation between main functions (books, suggestions, halls).

- Forms for search, reservation, and suggestion shall be short and validated with clear error messages.

Performance:
- Typical book and hall searches shall complete within a few seconds under normal network conditions.
- The system shall support multiple concurrent users without noticeable slowdown for basic operations.

Reliability:
- The system shall prevent double booking of the same book or hall for the same time interval.
- The system shall maintain data consistency between reservations and availability status in the underlying databases.

Portability:
- The system shall be deployable on Android and iOS devices

Constraints:
- The application must implement all five main functions defined in the project description: book search, book reservation, book suggestion, hall search, and hall booking, using the specified library halls.
- The solution is a design-level project; actual deployment can rely on mock or simplified data sources as long as interfaces and behaviors are clearly defined.

Member
+memberid: int
+name: String
+email: String
+password: String

BookReservation
+reservationId: int
+reservationDate: Date
+startDate: Date
+endDate: Date
+status: ReservationStatus

Book
+bookId: int
+title: String
+author: String
+category: String
+status: BookStatus

+isAvailable(): Boolean

«enumeration» BookStatus
AVAILABLE
RESERVED
LOST

«enumeration» ReservationStatus
PENDING
CONFIRMED
CANCELLED

BookSuggestion
+suggestionId: int
+title: String
+author: String
+publisher: String
+reason: String
+suggestionDate: Date

Hall
+hallId: int
+name: String
+capacity: int
+type: String

HallReservation
+reservationId: int
+date: Date
+startTime: Time
+endTime: Time
+status: ReservationStatus

BookSearchCriteria
+title: String
+author: String
+category: String
+keyword: String

LibrarySystem
+searchBooks(criteria: BookSearchCriteria): List<Book>
+getBookDetails(bookId: int): Book
+reserveBook(member: Member, book: Book, startDate: Date, endDate: Date): BookReservation
+cancelBookReservation(reservationId: int): boolean
+createBookSuggestion(member: Member, title: String, author: String, publisher: String, reason: String): BookSuggestion
+getSuggestionsForMember(member: Member): List<BookSuggestion>
+searchHalls(date: Date, startTime: Time, endTime: Time, expectedAttendees: int, eventType: String): List<Hall>
+getHallDetails(hallId: int): Hall
+reserveHall(member: Member, hall: Hall, date: Date, startTime: Time, endTime: Time): HallReservation
+cancelHallReservation(reservationId: int): boolean

Relationships:
Member 1 — 0..* BookReservation
BookReservation 0..* — 1 Book
Member 1 — 0..* BookSuggestion
Member 1 — 0..* HallReservation
Hall 1 — 0..* HallReservation

Use case diagram:

Member actor connected to:

- **Suggest Book** — note: fill form (title, author, justification), submit
- **View Book Details** — note: display name, author, etc.
- **Search Book** «include» View Book Details — note: enter criteria, display search results
- **Reserve Book** «extend» Search Book — note: select book, confirm reservation
- **View Hall Details** — note: display name, location, capacity, etc.
- **Search Hall** «include» View Hall Details — note: enter hall name, display search results
- **Reserve Hall** «extend» Search Hall — note: select hall, confirm reservation

Librarian actor connected to:

- **Review Suggestions** — note: list suggestions, mark as approved/rejected

Member

Book Reservation Request (selected book, dates)

Book Search Request

Book Search Results

Book Reservation Confirmation / Failure Message

Hall Search Request

Hall Reservation Request

Hall Search Results

Hall Reservation Confirmation / Failure Message

Suggestion Details (title, author, publisher, reason)

Suggestion Stored Message

Manage Book Search & Reservation

Manage Hall Search & Reservations

Manage Book Suggestions

New Book Reservation (reservation record)

Matching Books

Search Criteria

Availability Status

Availability Check (bookId, dates)

Check Conflicts / Store Reservation Status

List of Suitable Halls

Hall Search Query

Store Suggestion

Request Suggestion List

List of Suggestions

Book Database

Reservation Database

Hall Database

Suggestion Database

Reservation Status

View Suggestions

List of Suggestions

Librarian