جامعة الإمام عبدالرحمن بن فيصل
IMAM ABDULRAHMAN BIN FAISAL UNIVERSITY

Imam Abdulrahman Bin Faisal University
College of Computer Science & Information Technology
Department of Computer Science

CS 411 – Software Engineering
Term 1 – 2024/2025

Software Design Specifications

For

Automated Attendance System

Version 1.0

CS 2024, G1

Dr. Nehad Mohammed Ibrahim

02/11/2024

This Software Design Specification was prepared and provided as a deliverable for Software Engineering, CS 411, 1st term, and it will be used by Imam Abdulrahman University.

This document is based in part on the IEEE Recommended Practice for Software Design Descriptions.

Student	ID
Ali Albaqqal	2220000245
Ahmad Alsowayan	2220000086
Hassan Alibrahim	2220004350
Hassan Alzourei	2220004853
Hussain Alghubari	2220004326
Feras Alameer	2220004198

Table of Content

Revision History

Name	Date	Reason For Changes	Version
Ali Albaqqal	Nov. 14, 2024	Prepared initial version	0.1
Ali Albaqqal	Nov. 14, 2024	Updated section 1	0.2
Feras Alameer	Nov. 14, 2024	Updated section 2	0.3
Ali Albaqqal	Nov. 15, 2024	Updated section 3	0.4
Hassan Alzourei	Nov. 15, 2024	Updated section 4	0.5
Hassan Alzourei	Nov. 15, 2024	Updated section 5	0.6
Ahmad Alsowayan	Nov. 15, 2024	Updated section 4	0.7
Feras Alameer	Nov. 15, 2024	Updated section 6	0.8
Hassan Alibrahim	Nov.16, 2024	Updated section 7-10	0.9
All members	Nov. 16, 2024	Complete review - Final version	1.0

Table of Tables

Table of Figures

1. Introduction

This document, Software Design Specification (SDS) proposes to convert the SRS into data, architecture, interfaces, and components to implement and complete Automated Attendance System application. It shows how the whole system is assembled to meet Automated Attendance System requirements to make an efficient application. SDS will be divided into two phases:

• **The initial design phase:** it includes the first five sections. In this phase, we design components and interfaces for data.

• **The detailed design phase:** in this phase, we demonstrate the first phase in detail which contains the components design, and the detailed system design.

In this section of SDS, we will introduce the topics that are related to ASS's SDS document. It covers the document's purpose, scope, definitions, acronyms, abbreviations, and list of references used.

1.1 Purpose

The purpose of this Software Design Specification (SDS) document is to outline the design architecture, components, and detailed technical approach for implementing the Automated Attendance System. This system leverages facial recognition and geolocation technologies to provide an accurate and efficient solution for attendance tracking. This document is intended for:

- **University Administration**:
  To understand the system's design and confirm that it adheres to institutional policies on data security, privacy, and efficiency, as established in the Software Requirements Specification (SRS).
- **Development Team**:
  To serve as a blueprint for the detailed design, development, and integration of the system with the university's infrastructure, based on the requirements defined in the SRS.
- **Faculty and Students**:
  To offer a high-level understanding of the system's design and user interfaces, facilitating end-user familiarity with the technical aspects of the system's functionality for improved attendance processes.

1.2 Scope

The Automated Attendance System will be designed for seamless integration with the university's existing mobile application. The system architecture and user interfaces will support distinct functionalities tailored to each user role, as detailed below:

Role	Functionality
Student	- Design interface for Face ID check-in and geolocation-based verification. - Enable secure access to personal attendance records. - Implement notifications for attendance status updates.
Instructor	- Develop access controls for attendance reports specific to assigned classes. - Integrate real-time monitoring features for student check-ins. - Provide interface for managing and addressing attendance discrepancies.
Administrator	- Implement administrative controls for managing user accounts and role-based permissions. - Design real-time data synchronization with the university's central database. - Create automated reporting functions for generating attendance summaries.

Table 1: Design specifications for each end-user role

1.3 Definitions, Acronyms, and Abbreviations

Acronym	Meaning
AAS	Automated Attendance System
API	Application Programming Interface
Face ID	Facial Recognition Technology
GPS	Global Positioning System
IEEE	Institute of Electrical and Electronics Engineers
SPMP	Software Project Management Plan
SRS	Software Requirements Specification

Table 2: Acronyms used in the SDS

Term	Definition
Automated Attendance System (AAS)	A biometric attendance solution that utilizes Face ID and geolocation to streamline attendance tracking within the university.
Attendance Report	A report showing attendance records for a specific period, accessible to instructors and administrators.
Biometric Authentication	Security mechanisms use unique physical traits, such as facial recognition, to verify identity.
Face ID	Biometric technology that authenticates users' identities through facial recognition.
Geolocation	Technology that determines a user's location to confirm attendance within specified areas.
Real-Time Synchronization	The system’s capability to immediately update attendance data in the university database as students check in.
User Access Levels	Role-based permissions within the system, with levels for students, instructors, and administrators.

Table 3: Terminologies used in the Software Design Specification

1.4 References

[1] Internal Document, "Software Project Management Plan (SPMP) for Automated Attendance System," Version 1.0, Imam Abdulrahman University, October 2024.

[2] Internal Document, "Software Requirements Specification (SRS) for Automated Attendance System," Version 1.0, Imam Abdulrahman University, October 2024.

[3] IEEE Standard for Information Technology – Software Design Descriptions, IEEE Std 1016-2009, 2009.

2. System overview

This section provides a general description of the Automated Attendance System, including its key functionalities, design considerations, and its role within the university's broader administrative ecosystem.

2.1 General Description

The Automated Attendance System is designed to streamline attendance tracking using geolocation and biometric technology (Face ID). It integrates seamlessly with the university's existing infrastructure to improve accuracy, efficiency, and data security. The system communicates with scheduling APIs and student databases while operating independently as a standalone solution.

The system's design ensures interoperability with various hardware and software components, making it a scalable and adaptable addition to the university's ecosystem. A detailed block diagram illustrating system relationships with existing components will be included in the final design document.

2.2 Key Design Considerations

The system's architecture and functionality are influenced by the following constraints and requirements:

- **System Interfaces**: Integration with the university's scheduling API and central database.
- **User Interfaces**: Simple and intuitive interfaces tailored for students, instructors, and administrators.
- **Hardware Interfaces**: Compatibility with mobile devices and biometric scanners.
- **Software Interfaces**: Conformance with university protocols for secure and reliable data exchange.
- **Communication Interfaces**: Secure channels for transmitting attendance data.
- **Performance**: Efficient operation within hardware constraints.
- **Site Adaptation**: Flexibility to adapt to different classroom setups and configurations.

2.3 Functionality Overview

The Automated Attendance System includes the following core functions:

1. **Attendance Tracking**: Uses Face ID and geolocation to automatically record attendance.
2. **Real-Time Data Syncing**: Updates attendance records directly to the university database.
3. **User Management**: Enables administrators to manage user accounts and permissions.
4. **Reporting**: Generates detailed attendance reports for instructors and administrators.
5. **User Notifications**: Sends real-time alerts about attendance status and discrepancies.

2.4 User Characteristics

The system is designed to cater to the following primary user groups:

- **Students**: Require a quick, simple method for marking attendance.
- **Instructors**: Need efficient tools for monitoring attendance and generating reports.
- **Administrators**: Manage user accounts, oversee system operations, and resolve exceptions.

2.5 Design Constraints

Several constraints impact the system's design and development:

- **Time Constraints**: Development and deployment must align with the academic term schedule.
- **Budget Constraints**: Resources are limited, influencing technology choices.
- **Compliance**: Adherence to the university’s privacy and security policies is mandatory.

2.6 Assumptions and Dependencies

The system’s success relies on the following assumptions and dependencies:

- **Assumptions**:
    - Users will have access to compatible devices for attendance verification.
    - University systems will be available for seamless integration.
    - The project team will deliver tasks effectively within allocated hours.
- **Dependencies**:
    - Access to the university’s APIs for data synchronization.
    - Availability of biometric hardware for development and testing.

2.7 Development Prioritization

The system will be developed incrementally, focusing on critical functions first:

1. **Phase 1**: Implementation of attendance tracking and real-time data syncing.
2. **Phase 2**: Development of reporting and user management features.
3. **Phase 3**: Refinement of user interfaces and notification systems.

This approach ensures critical functionalities are delivered early, with subsequent phases enhancing overall system capabilities.

3. Design Considerations

3.1 Assumptions and Dependencies

3.1.1 Related Software or Hardware

The Automated Attendance System relies on several related software and hardware components. These include the university's existing mobile application, biometric scanners for Face ID, and geolocation services. The system must integrate seamlessly with these components to function correctly.

3.1.2 Operating Systems

The system is designed to support multiple operating systems, including iOS and Android for mobile devices. This ensures that students, instructors, and administrators can access the system regardless of their device's operating system.

3.1.3 End-User Characteristics

End-users of the system include students, instructors, and administrators. Students require a quick and straightforward method to mark their attendance, while instructors need efficient tools for monitoring attendance and generating reports. Administrators manage user accounts, oversee system operations, and resolve exceptions.

3.1.4 Possible and/or Probable Changes in Functionality

The system may need to accommodate future changes in functionality, such as additional biometric authentication methods or enhanced reporting features. These changes should be anticipated and planned for during the design phase to ensure the system remains adaptable and scalable.

Assumption/Dependency	Description
Related software or hardware	Software or hardware that the system depends on.
Operating systems	Operating systems that the system will support.
End-user characteristics	Characteristics of the end-users that may affect the design.
Possible and/or probable changes in functionality	Anticipated changes in functionality that could impact the design.

Table 4: Assumptions and Dependencies

3.2 General Constraints

3.2.1 Hardware or Software Environment

The system will operate within the university's existing hardware and software environment. This includes mobile devices used by students and faculty, as well as the university's central database and scheduling APIs.

3.2.2 End-User Environment

End-users will interact with the system primarily through their mobile devices. The user interface must be intuitive and responsive to ensure a positive user experience across different devices and screen sizes.

3.2.3 Availability or Volatility of Resources

The system's design must account for the availability and stability of resources, such as network connectivity and server uptime. Any volatility in these resources could impact the system's performance and reliability.

3.2.4 Standards Compliance

The system must comply with relevant standards, including data privacy and security regulations. This ensures that user data is protected and that the system operates within legal and institutional guidelines.

Standard	Description
IEEE 1016	Standard for Software Design Descriptions.
GDPR	General Data Protection Regulation for data privacy.
ISO/IEC 27001	Information security management standards.
University IT Policies	Internal policies for data security and IT management.

Table 5: Standards Compliance

3.2.5 Interoperability Requirements

The system must be able to interact with other university systems, such as the central database and scheduling APIs. This requires adherence to established protocols and standards for data exchange.

Requirement	Description
API Integration	Integration with university scheduling and database APIs.
Data Format	Standard data formats for interoperability (e.g., JSON, XML).
Protocols	Communication protocols (e.g., HTTPS, REST).
Compatibility	Compatibility with existing university systems and applications.

Table 6: Interoperability Requirements

3.2.6 Interface/Protocol Requirements

The system will use standard interfaces and protocols to communicate with other systems. This includes RESTful APIs for data exchange and secure communication protocols to protect data in transit.

Interface/Protocol	Description
RESTful API	For data exchange between the system and university databases.
HTTPS	Secure communication protocol for data transmission.
OAuth	Authentication protocol for secure access.
WebSockets	Real-time communication protocol for instant updates.

Table 7: Interface/Protocol Requirements

3.2.7 Data Repository and Distribution Requirements

The system will store attendance data in a central database. This data must be accessible to authorized users and updated in real-time to ensure accuracy and reliability.

Requirement	Description
Central Database	Storage of attendance data in a central database.
Data Backup	Regular backups to prevent data loss.
Data Access	Controlled access to data based on user roles.
Data Distribution	Efficient distribution of data to authorized users.

Table 8: Data Repository and Distribution Requirements

3.2.8 Security Requirements

Security is a critical consideration for the system. This includes protecting user data through encryption, ensuring secure authentication methods, and implementing access controls to prevent unauthorized access.

Requirement	Description
Data Encryption	Encryption of data at rest and in transit.
Access Control	Role-based access control to restrict data access.
Authentication	Biometric and password-based authentication.
Audit Logs	Logging of all access and changes to data.

Table 9: Security Requirements

3.2.9 Memory and Other Capacity Limitations

The system must operate efficiently within the memory and capacity limitations of mobile devices. This includes optimizing the application to minimize resource usage and ensure smooth performance.

Limitation	Description
Mobile Device Memory	Optimization to run efficiently on devices with limited memory.
Server Capacity	Ensuring server capacity can handle peak loads.
Data Storage	Efficient use of storage to manage large volumes of attendance data.

Table 10: Memory and Capacity Limitations

3.2.10 Performance Requirements

The system must perform reliably under various conditions, including high user loads. This requires efficient coding practices and robust testing to identify and address potential performance bottlenecks.

Requirement	Description
Response Time	The system should respond within 2 seconds for most operations.
Throughput	Ability to handle a high number of concurrent users.
Reliability	System uptime should be 99.9% during operational hours.
Scalability	Ability to scale resources based on demand.

Table 11: Performance Requirements

3.2.11 Network Communications

The system relies on network communications to sync attendance data with the central database. This requires stable and secure network connections to ensure data integrity and availability.

Requirement	Description
Bandwidth	Sufficient bandwidth to support real-time data syncing.
Latency	Low latency to ensure timely updates.
Network Security	Secure network connections to prevent unauthorized access.
Redundancy	Network redundancy to ensure continuous operation.

Table 12: Network Communication Requirements

3.2.12 Verification and Validation Requirements

The system must undergo thorough verification and validation to ensure it meets all specified requirements. This includes functional testing, performance testing, and security testing.

Requirement	Description
Functional Testing	Testing to ensure all functions work as intended.
Performance Testing	Testing to ensure the system meets performance requirements.
Security Testing	Testing to identify and fix security vulnerabilities.
User Acceptance Testing	Testing with end-users to ensure the system meets their needs.

Table 13: Verification and Validation Requirements

3.2.13 Other Means of Addressing Quality Goals

Quality goals will be addressed through continuous integration and deployment practices, regular code reviews, and user feedback mechanisms to identify and resolve issues promptly.

3.2.14 Other Requirements Described in the Requirements Specification

All additional requirements outlined in the SRS must be considered during the design phase. This ensures that the system meets all user needs and operates within the defined constraints.

Goal	Description
Usability	Ensuring the system is easy to use for all user roles.
Maintainability	Ensuring the system is easy to maintain and update.
Portability	Ensuring the system can run on various devices and platforms.

Table 14: Other Quality Goals

4. User Interface Design

4.1 Overview of User Interface

To improve the process of taking attendance, the Automated Attendance System provides an easy-to-use interface. Students, teachers, and administrators are the main user roles; each has specific access requirements and features.

- **Students** will log in to the system via Face ID upon entering the classroom. The interface will confirm successful attendance recording with visual feedback (e.g., a green check mark or message saying "Attendance Recorded") and display real-time attendance status. At the end of the class, students will also use Face ID to log out, receiving a similar confirmation.
- **Instructors** will see a dashboard listing all students enrolled in their classes, with live updates on who has logged in or out of the classroom. The system will notify instructors of absentees or late arrivals.
- **Administrators** will access a management interface with reporting features to view and analyze attendance records across different courses, dates, and student demographics.

Feedback Information: The system provides instant feedback through status icons, success or error messages, and a summary of recorded attendance at the end of each session.

4.2 Interface Design Rules

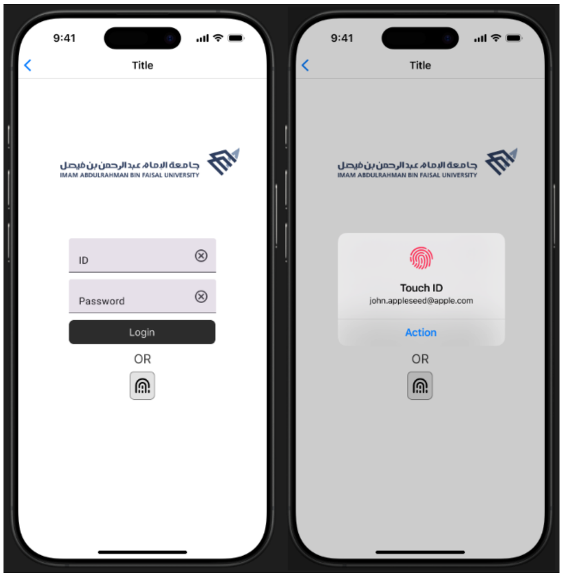
The interface will follow these design rules to ensure consistency, accessibility, and ease of use:

- **Consistency:** Use uniform colors, fonts, and layout structures across all screens for a cohesive look and feel.
- **Accessibility:** Interface elements should be accessible for users with disabilities. All buttons will be labeled, and the interface will support screen readers.
- **Simplicity:** Information and actions on each screen will be clear and straightforward. Only relevant data will be displayed to avoid information overload.
- **Error Handling:** User errors, such as failed Face ID recognition, will prompt error messages with suggestions for corrective action, and errors will be logged for administrator review.

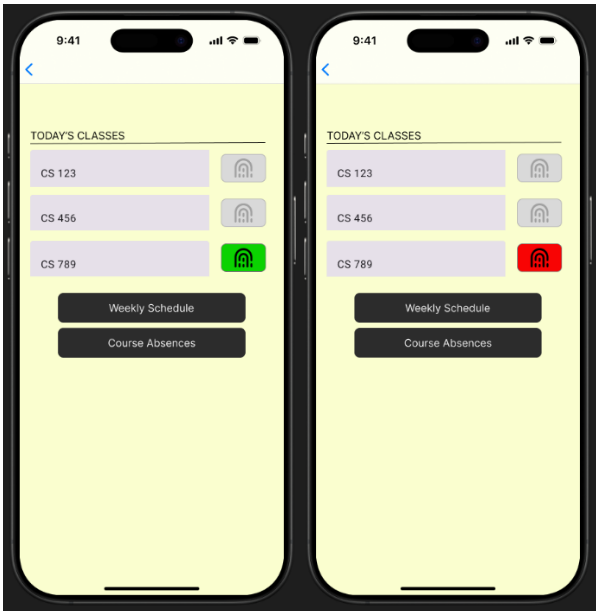
4.3 Screen Images

Display mockups of the primary interface screens:

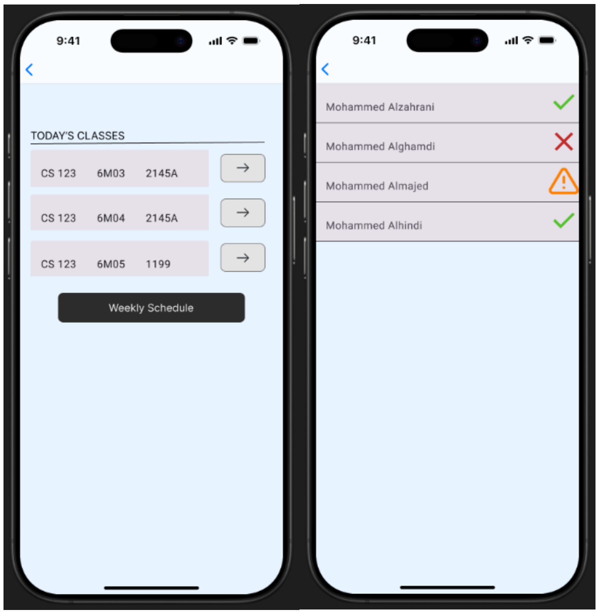
- **Login Screen:** Displays fields for Face ID verification and optional manual login with a student ID number. Confirmation messages will appear post-authentication.



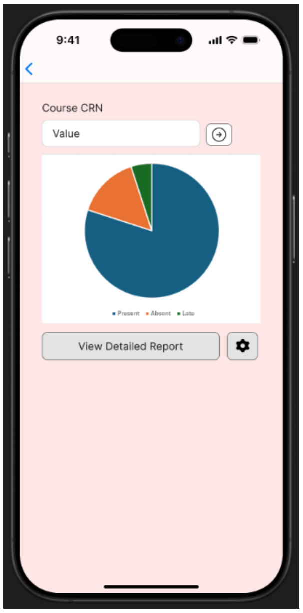
- **Student Dashboard:** Shows the current attendance status and class schedule. It includes buttons for logging in and out attendance.



- **Instructor Dashboard:** Displays a list of students with icons indicating present, absent, or late statuses.



- **Administrator Interface:** Provides access to detailed attendance reports, analytics, and settings for managing the attendance system.



4.4 Screen Objects and Actions

- **Login Button (Student and Instructor):** Initiates Face ID or manual verification for login.
- **Attendance Status Icon (Student):** Shows if the user is currently marked as present or absent. Tapping on the icon provides additional attendance details.

- **Class Roster (Instructor):** Displays each student's name, status, and timestamps for login and logout. Clicking on a student's name shows their full attendance record.
- **Reports Tab (Administrator):** Generates and displays attendance reports based on various parameters, such as date, course, or student ID.

4.5 Other Interfaces

4.5.1 University API Integration Interface

The attendance system integrates with the university's existing website API to upload attendance data securely and accurately.

- **Technology & Protocol:** The interface will use REST API with JSON for data format. HTTPS will be implemented to ensure secure data transfer.
- **Error Conditions & Messages:** If the API call fails due to network or authentication issues, an error message will be logged and displayed in the interface.
- **Handshake & Initiation:** Authentication tokens will be requested at the start of each session, and session closures will be confirmed via a logout API call.

4.5.2 Mobile App Widget

An integrated widget for the university's mobile app will enable students to check their attendance status in real-time and verify previous attendance records.

- **Technology:** The widget will be developed using the mobile app's SDK.
- **Data Syncing:** Attendance records will update automatically every 10 seconds during class hours.
- **Error Handling:** If the widget fails to connect to the server, a message will prompt students to reconnect or verify their network connection.

5. System Architecture

This section explains the high-level architecture and structure of the Automated Attendance System. Separating the system into discrete subsystems has made it easier to scale, modularize, and maintain. The system's primary functions include user authentication, attendance monitoring, data storage, and reporting. Streamlining these duties and ensuring that each component plays a distinct part in the overall functionality are the goals of the decomposition into subsystems.

5.1 Architectural Design Approach

The architectural approach follows a **modular design** pattern, emphasizing loose coupling and high cohesion among components to allow each subsystem to function independently. This strategy was used to provide simple upgrades, maintenance, and system scalability as it expands. The architecture makes use of a client-server paradigm in which user devices communicate with a centralized server that manages essential functions including data processing, storage, and authentication. The **RESTful API** design enables communication between components and ensures secure data transfer. This modular breakdown minimizes dependencies between subsystems, thus enhancing system resilience and reliability.

5.2 Architectural Design

The system architecture is illustrated in the diagram below (you can include a high-level diagram showing the major subsystems such as User Interface, Authentication, Attendance Processing, Data Storage, and Reporting, along with the relationships between them).

The main components and their roles are:

1. **User Interface Subsystem**
   - This subsystem handles user interactions. It includes interfaces for students, instructors, and administrators, enabling each user role to access specific features such as login, attendance tracking, and reporting.
2. **Authentication Subsystem**
   - Responsible for Face ID and location-based authentication. It verifies the identity of students using facial recognition, integrating securely with the university's API to ensure that only authorized users access the system.
3. **Attendance Processing Subsystem**
   - Manages the attendance records, logging students in or out based on Face ID and location data. This subsystem updates real-time attendance status for instructors and processes attendance data for reporting.
4. **Data Storage Subsystem**
   - A centralized database stores attendance records, user data, and course schedules. This database is designed for fast querying and retrieval, ensuring efficient access for attendance validation and reporting purposes.
5. **Reporting Subsystem**
   - Provides analytical reports on attendance data, accessible by administrators. This subsystem enables analysis of attendance patterns and generation of reports for various stakeholders.

Subsystem Collaboration:

- **User Interface** interacts with the **Authentication Subsystem** to validate user access, and then communicates with the **Attendance Processing Subsystem** to log attendance.
- **Attendance Processing** interacts with **Data Storage** to retrieve and store attendance records. The **Reporting Subsystem** pulls data from **Data Storage** for analysis and displays it through the **User Interface** for administrator use.

5.3 Subsystem Architecture

Each subsystem has been further divided as follows:

1. **User Interface Subsystem**
    - The UI is structured around **three main components** for each user role: Student UI, Instructor UI, and Admin UI. Each component has unique screens and functionalities relevant to the user role.
    - *Diagram:* Sequence diagram showing user login, attendance tracking, and reporting interactions.
2. **Authentication Subsystem**
    - **Components:** Face ID verification and location verification services.
    - *Object Diagram:* Showing interactions between the user interface and authentication modules, illustrating successful and failed login attempts based on Face ID and location.
3. **Attendance Processing Subsystem**
    - **Data Flow:** Captures attendance data and timestamps from students, logs them, and updates instructors.
    - *Data Flow Diagram (DFD):* A DFD illustrating how attendance data flows from student login to data storage.
4. **Data Storage Subsystem**
    - **Database Design:** Relational database that includes tables for student information, attendance logs, course schedules, and reports.
    - *ER Diagram:* Entity Relationship diagram showing major tables and relationships such as “Students,” “Classes,” “Attendance Logs,” and “Reports.”
5. **Reporting Subsystem**
    - **Components:** Data retrieval and report generation module.
    - *Use Case Diagram:* Depicting how administrators and instructors generate reports and access attendance analytics.

6. Data Design

To manage vital data including student records, attendance logs, and session specifics, the Automated Attendance System needs a well-structured data design. The structure of the data entities, the database schema that underpins the system's operation, and the data organization are all covered in this part.

6.1 Data Description

Students, classes, attendance logs, and administrators are the core data entities that make up the system's information domain. Relational databases are used to efficiently store, retrieve, and alter data. The data entities are arranged and structured as follows:

- **Students:** Holds personal details, Face ID information, and student ID for unique identification.
- **Classes:** Contains information about courses, schedules, and instructors.
- **Attendance Logs:** Records attendance status, timestamps, and login/logout details for each student per class.
- **Administrators:** Stores information for users with administrative privileges, allowing them to access and analyze attendance data.

These entities interact to support user authentication, attendance logging, and report generation. The relationships between entities, such as the many-to-one relationship between students and classes, enable efficient querying and data integrity.

6.2 Data Dictionary

The **Data Dictionary** provides details of key data entities in the system, describing their types and purposes. Below is an example structure for the data dictionary (you can expand this based on your exact database schema):

Entity	Type	Description
Student_ID	Integer	Unique identifier for each student
Face_ID	Binary/Blob	Biometric data for Face ID authentication
Name	String	Full name of the student
Email	String	Email address of the student
Class_ID	Integer	Unique identifier for instructors associated with each class
Attendance_Status	Boolean	Indicates whether the student is present (True) or absent (False)
Timestamp	DateTime	Date and time of attendance record
Admin_ID	Integer	Unique identifier for administrators accessing the system
Report_ID	Integer	Unique identifier for each generated attendance report
Location	Coordinates	GPS coordinates for validating location-based attendance

Table 15: Data Dictionary

6.3 Database Description

The database supporting the **Automated Attendance System** is a **relational database** designed for scalability and efficient data handling. The database includes the following tables:

1. **Students Table**
   - Contains personal information and Face ID data for each student.
2. **Classes Table**
   - Includes details of courses, schedules, and instructor IDs.
3. **Attendance Logs Table**
   - Stores attendance records, including student IDs, timestamps, and status for each class session.
4. **Administrators Table**
   - Holds information for administrators who have access to reporting and system management functions.
5. **Reports Table**
   - Maintains records of generated attendance reports, which can be filtered by course, date, and instructor.

7. Component Design

This section details the major software components of the Automated Attendance System, providing an overview of the core functions, responsibilities, and interactions. The components are categorized based on user roles (Admin, Instructor, Student) and common functionalities.

7.1 Common Components

7.1.1 Login

The Login component handles user authentication through Face ID or password input. It verifies user credentials and provides secure access to the system. If Face ID verification fails, the user can log in using their registered password.

Pseudocode:

```
Function Login(user_id, password):
    If VerifyFaceID(user_id):
        Return "Login Successful"
    Else If VerifyPassword(user_id, password):
        Return "Login Successful"
    Else:
        Display "Invalid Credentials"
        Return "Login Failed"
```

7.1.2 Profile Management

This component allows users (Admin, Instructor, Student) to view and update their personal information. Updates are reflected in real-time in the system database.

Pseudocode:

```
Function ViewProfile(user_id):
    profile_data = FetchFromDatabase(user_id)
    Display profile_data


Function UpdateProfile(user_id, new_data):
    If ValidateData(new_data):
        UpdateDatabase(user_id, new_data)
        Display "Profile Updated Successfully"
    Else:
        Display "Invalid Data"
```

7.1.3 Notification System

The Notification component is responsible for sending alerts and updates to users regarding their attendance status, schedule changes, and system updates.

Pseudocode:

```
Function SendNotification(user_id, message):
    contact_info = GetUserContact(user_id)
    SendPushNotification(contact_info, message)
    LogNotification(user_id, message)
```

7.2 Admin Components

7.2.1 User Management

This component enables the Admin to manage user accounts, including adding, editing, and removing users. It also handles setting user permissions based on roles.

Pseudocode:

```
Function ManageUser(action, user_id, user_data):
    If action == "Add":
        CreateUser(user_data)
        Display "User Added Successfully"
    Else If action == "Edit":
        UpdateUser(user_id, user_data)
        Display "User Updated Successfully"
    Else If action == "Delete":
        DeleteUser(user_id)
        Display "User Deleted Successfully"
    Else:
        Display "Invalid Action"
```

7.2.2 Attendance Reporting

Generates comprehensive attendance reports for instructors and university administrators. The reports include details on student attendance, course participation, and discrepancies.

Pseudocode:

```
Function GenerateReport(course_id, date_range):
    attendance_data = FetchAttendanceData(course_id, date_range)
    report = FormatReport(attendance_data)
    Return report
```

7.3 Instructor Components

7.3.1 Attendance Tracker

The Attendance Tracker allows instructors to view and update student attendance records. It integrates with Face ID and geolocation to ensure accurate tracking.

Pseudocode:

```
Function MarkAttendance(student_id, location):
    If VerifyLocation(location) And VerifyFaceID(student_id):
        UpdateAttendanceRecord(student_id, "Present")
        Display "Attendance Marked"
    Else:
        Display "Verification Failed"
```

7.3.2 Schedule Viewer

This component provides instructors with an interface to view their teaching schedules and student attendance records.

Pseudocode:

```
Function ViewSchedule(instructor_id):
    schedule = FetchInstructorSchedule(instructor_id)
    Display schedule
```

7.4 Student Components

7.4.1 Attendance Check-in

The Check-in component allows students to mark their attendance using Face ID and geolocation technology.

Pseudocode:

```
Function CheckIn(student_id, current_location):
    If VerifyFaceID(student_id) And VerifyLocation(current_location):
        MarkAttendance(student_id, "Present")
        Display "Check-in Successful"
    Else:
        Display "Check-in Failed"
```

7.4.2 Attendance History

This component allows students to view their past attendance records, including classes attended and any absences.

Pseudocode:

```
Function ViewAttendanceHistory(student_id):
    history = FetchAttendanceHistory(student_id)
    Display history
```

8. Detailed System Design

This section provides a comprehensive breakdown of system components, including classifications, constraints, uses/interactions, and required resources.

8.1 Classification, Definition, and Responsibilities

Table 16: Component Classification and Responsibilities

Component	Type	Description
Login	**Function**	**Authenticates users through Face ID or password input, ensuring secure access to the system.**
Profile Management	**Class**	**Manages user profile data, allowing users to view and update personal information.**
Notification System	**Class**	**Handles sending real-time alerts and updates to users about attendance status and schedule changes.**
User Management	**Class**	Allows Admin to manage user accounts, including adding, editing, and removing users.
Attendance Tracker	**Function**	**Enables instructors to mark, track, and review student attendance using Face ID and location.**
Check-in	**Function**	**Allows students to mark attendance using Face ID and geolocation verification.**
Schedule Viewer	**Function**	**Displays instructors' teaching schedules and student attendance records.**
Attendance History	**Function**	**Allows students to view their attendance history, including absences and attended classes.**

8.2 Definition

The following section highlights each component’s definition.

Table 17: System Component Definition

Component	Definition and Responsibilities
Common Interfaces	
Login	This function allows the user to log in by entering their username and password, or by using Face ID.
Forgot Password	Checks the user’s identification and authentication answers to allow the reset of a password if needed.
Sign Out	Enables users to securely log out of the system.
ViewProfile	Allows users to view their profile details, including personal and role-specific information.
UpdateProfile	Enables users to update their profile information, with changes saved in real-time.
Admin Interfaces	
Admin	
AdminHomepage	Provides a central menu for the admin to access system management functions.
User Management	Allows the admin to view, add, update, and delete user accounts and set user permissions.
Report Generation	Generates detailed attendance and activity reports accessible to admin users.
System Monitoring	Provides the admin with tools to monitor and manage system performance.
Instructor Interfaces	
Instructor	This class defines the behavior and properties of the instructor interface.
InstructorHomepage	Displays the main menu for instructors to access attendance and schedule management features.
Attendance Tracker	Allows instructors to mark student attendance, using Face ID and geolocation for verification.
Schedule Viewer	Enables instructors to view and manage their teaching schedules and student attendance data.
ViewAttendance	Provides instructors with access to attendance records for each class session.
Student Interfaces	
Student	This class sets up the behavior and properties of the student interface.

StudentHomepage	Displays the main menu for students, allowing access to attendance check-in, history, and notifications.
Check-in	Enables students to mark attendance by verifying their Face ID and geolocation within campus.
Attendance History	Allows students to review their attendance records, including details of classes attended or missed.
Notifications	Displays system notifications for students, including attendance status and schedule changes.

8.3 Responsibilities

The following table outlines the primary responsibilities and behaviors of each component within the *Automated Attendance System*. Each component is designed to accomplish specific roles and provide essential services to its users.

Table 18: System Component Responsibilities

Component	Responsibilities
Login	Authenticates users by verifying their credentials (Face ID or password), ensuring only authorized access to the system. It plays a critical role in security by controlling access to system resources. This component also logs login attempts for auditing purposes.
Forgot Password	Provides password recovery options to users by verifying their identity through security questions or email. Ensures a secure method for users to regain access without compromising security.
Sign Out	Allows users to securely exit the system, terminating their active session and freeing system resources. It ensures that no unauthorized person can continue using the system after a user has logged out.
ViewProfile	Allows users to access their personal information and role-specific data. Ensures data privacy by limiting access to only the user's information. This component supports the user experience by providing easily accessible personal information.

UpdateProfile	Enables users to modify their personal details, which are stored in real-time in the database. Ensures that the latest information is available across the system, providing an updated and accurate profile.
AdminHomepage	Provides a dashboard for Admins to access management tools, reports, and system monitoring options. This component serves as the control center for administrative tasks, allowing Admins to oversee the system's overall operation.
User Management	Allows Admins to create, edit, or delete user accounts and set permissions based on user roles. This component is responsible for managing user access and ensuring role-based security within the system. It plays a crucial role in maintaining system integrity by controlling who can access specific resources.
Report Generation	Provides Admins with detailed reports on system usage, attendance statistics, and user activity. This component helps monitor and evaluate system performance, supporting decision-making and ensuring compliance with reporting requirements.
System Monitoring	Enables Admins to monitor real-time system performance and detect potential issues. It provides insights into user activity, data flow, and system health, ensuring stability and quick response to any irregularities.
InstructorHomepage	Acts as the main interface for instructors, allowing them access to attendance tracking, scheduling, and reporting tools. This component organizes instructor-related functions in a user-friendly layout, enhancing productivity.
Attendance Tracker	Allows instructors to mark attendance accurately by verifying students’ Face ID and geolocation. It provides a secure method for tracking presence, ensuring data reliability and compliance with attendance policies.
Schedule Viewer	Provides instructors with access to their schedules and student attendance records. This component ensures instructors have up-to-date class and student data, supporting effective class management.
ViewAttendance	Allows instructors to view detailed attendance records for each class session, ensuring they can track and manage attendance over time.

	This component supports instructors in identifying attendance trends and discrepancies.
StudentHomepage	Serves as the primary interface for students, giving them access to attendance check-in, history, and notifications. This component organizes student-specific functionalities, ensuring ease of use and accessibility.
Check-in	Enables students to mark their attendance using Face ID and geolocation, confirming their presence on campus. This component ensures accurate tracking of student attendance and supports compliance with attendance requirements.
Attendance History	Allows students to view their past attendance records, providing transparency and helping students monitor their attendance status. This component encourages accountability and self-management among students.
Notifications	Delivers important updates to students, such as attendance status, schedule changes, and other alerts. This component helps keep students informed and supports timely responses to system updates.

8.4 Constraints

The following table outlines the assumptions, limitations, and constraints for each component within the *Automated Attendance System*. Each component is subject to specific rules and conditions related to timing, storage, data formats, synchronization, and exception handling.

Table 19: System Component Constraints

Component	Constraints
Login	Timing: Login must occur within a set timeout period to prevent unauthorized access attempts. Preconditions: Valid credentials (Face ID or password) must be provided. Exceptions: Lock account after multiple failed attempts.
Forgot Password	Assumptions: Users have set up recovery options (security questions or email) in advance. Constraints: Secure verification to prevent unauthorized password resets. Postconditions: A new password is generated securely.
Sign Out	Preconditions: User must be logged in to use this function. Postconditions: All session data must be cleared upon sign-out. State Constraints: The user session state is set to inactive after logging out.
ViewProfile	Data Access: Profile data must be accessed securely; only authorized users should view profiles. Data Format: Must conform to predefined format. Constraints: Profile data must not be modified in this view.
UpdateProfile	Data Validation: Must validate all inputs to prevent invalid or malicious data entry. Preconditions: User is logged in and authorized to update profile. Postconditions: Updated profile information saved in the database.
AdminHomepage	Access Constraints: Restricted to Admin role; unauthorized access must be blocked. Timing: Loading time must be optimized for quick access to admin features. State Constraints: Must display active system information.

User Management	Audit Logging: All actions (add, edit, delete) must be recorded for accountability. Data Format: User data must follow a standard format. Synchronization: Changes to user roles should be immediately reflected across the system.
Report Generation	Timing: Large reports must be generated within an acceptable time frame. Storage: Sufficient storage required for storing historical report data. Data Accuracy: Data used for reports must be consistent and up-to-date.
System Monitoring	Real-time Requirement: Monitoring must refresh at regular intervals to provide current data. Storage Constraints: Logs and monitoring data may require substantial storage. Exception Handling: Any system error should trigger alerts.
InstructorHomepage	Access Constraints: Only accessible by users with Instructor role. Data Consistency: Must display real-time attendance and scheduling data. Exception Handling: Display error message if data fails to load.
Attendance Tracker	Preconditions: Student must be physically within campus boundaries for attendance marking. Timing: Attendance records must be updated immediately. Data Access: Only authorized instructors can view and update attendance records.
Schedule Viewer	Data Consistency: Schedule changes must propagate in real time. Constraints: Only instructors assigned to a course should view its schedule. Postconditions: The schedule view must refresh automatically when data changes.
ViewAttendance	Data Privacy: Attendance records are confidential and must be securely accessed. Synchronization: Attendance data should reflect any changes immediately. Constraints: Only instructors of the course can view the attendance data
StudentHomepage	Access Constraints: Only accessible by users with Student role. Exception Handling: Must provide error message if any student-specific data fails to

	load. Data Format: Information must conform to a standardized display format.
Check-in	Preconditions: Student must pass Face ID and be within geolocation boundaries. Timing: Attendance marking should occur within a limited time window for each class session. Postconditions: Attendance status is recorded in the system.
Attendance History	Data Integrity: Records should be immutable and reflect accurate attendance history. Access Constraints: Only accessible by the student or authorized personnel. Storage: Requires storage capacity for long-term data retention.
Notifications	Timing: Notifications must be delivered in real time to be effective. Data Access: Must securely access user contact information. Exception Handling: Retry sending notifications if delivery fails initially.

8.5 Composition

The following table outlines the primary subcomponents within each main component of the *Automated Attendance System*, describing their use and meaning within the overall structure.

Table 20: System Component Composition

Component	Subcomponent	Description
Login	Credential Verification	Verifies the provided Face ID or password to authenticate users.
Login	Access Control	Determines user role (Admin, Instructor, Student) upon login to assign appropriate permissions.
Forgot Password	Identity Verification	Confirms user identity through security questions or email verification before allowing password reset.
Forgot Password	Password Reset	Generates a new password upon successful verification, updating it in the database.

Sign Out	Session Termination	Ends the current session and clears any session-specific data from the system.
ViewProfile	Profile Retrieval	Accesses user profile information from the database to display personal details.
UpdateProfile	Data Validation	Ensures new profile data meets required standards before saving.
UpdateProfile	Database Update	Updates the database with new profile information, reflecting changes in real-time.
AdminHomepage	Dashboard Navigation	Provides links to core admin functions, such as User Management and Report Generation.
AdminHomepage	System Overview	Displays a summary of system status, including recent activities and alerts for admin users.
User Management	Account Creation	Allows admins to add new user accounts with role-based permissions.
User Management	Account Modification	Enables editing of user information, including updating roles or access levels.
User Management	Account Deletion	Permits removal of user accounts, ensuring only active users remain in the system.
Report Generation	Data Aggregation	Collects and organizes attendance and usage data for report creation.
	Report Formatting	Formats collected data into structured reports for review and analysis by admins.
System Monitoring	Activity Logging	Tracks user actions and system events for real-time monitoring and troubleshooting.
System Monitoring	Performance Alerts	Generates alerts for system performance issues or unusual activity.

Attendance Tracker	Attendance Recording	Marks student attendance based on Face ID and geolocation, ensuring accuracy and compliance.
Attendance Tracker	Attendance Verification	Confirm that the recorded attendance is accurate by checking Face ID and location boundaries.
Schedule Viewer	Schedule Retrieval	Fetches instructor schedules from the database for real-time viewing.
Schedule Viewer	Update Notification	Alerts instructors to any updates in their schedule, ensuring they have the latest information.
Check-in	Face ID Verification	Confirms student identity using Face ID before marking attendance.
Check-in	Geolocation Check	Verifies that the student is within the allowed campus boundaries for check-in.
Notifications	Notification Queue	Manages notifications awaiting delivery to users, ensuring they are sent in order of priority.
Notifications	Delivery Tracking	Confirms successful notification delivery or retries if the initial attempt fails.

8.6 Uses/Interactions

The following table describes how each component interacts with other components in the *Automated Attendance System*, including dependencies and any potential side effects.

Table 21: Component Uses and Interactions

Component	Collaborations and Interactions
Login	Collaborates with **User Management** to verify user credentials and access levels. Interacts with **Session Management** to establish user sessions after logging in. Successful login initiates interactions with role-specific homepages (Admin, Instructor, or Student).
Forgot Password	It depends on **User Management** for identity verification. Interacts with **Notification System** to send password reset instructions to the user's registered email. Any changes made during the reset are updated in the **User Profile** component.
Sign Out	Interacts with **Session Management** to terminate user sessions and log out the user. Triggers cleanup of temporary session data and ensure no lingering access rights remain active, affecting components relying on active user sessions, such as **Notifications** and **Attendance Tracker**.
ViewProfile	It depends on **User Management** and **Database Access** to retrieve user data. Updates from **User Profile** are displayed through this component, ensuring any changes made to a profile are accurately reflected.
UpdateProfile	Interacts with **User Management** and **Database Access** to validate and store updated user details. Changes to profile information propagate to all components accessing user data, such as **Login** (for updated credentials) and **Notifications**.
AdminHomepage	Acts as a central interface, interacting with **User Management**, **Report Generation**, and **System Monitoring**. Allows admins to manage users, generate reports, and monitor system activity, initiating interactions with various subsystems based on admin actions.
User Management	Collaborates with **AdminHomepage** for user creation, modification, and deletion. Interacts with **Login** for credential verification and

	Profile Management for updating user details. User changes impact all components that access user data, such as **Attendance Tracker** and **Notifications**.
Report Generation	Interacts with **Database Access** to retrieve attendance and usage data. Collaborates with **AdminHomepage** to display generated reports. Report changes do not directly affect other components but are used for analysis and auditing by admins.
System Monitoring	Collaborates with **Activity Logging** to track user actions and system events. Interacts with **Notification System** to alert admins of unusual activity. Monitoring results impact components like **Login** (e.g., if suspicious login behavior is detected).
InstructorHomepage	Provides access to **Attendance Tracker**, **Schedule Viewer**, and **ViewAttendance** components. Acts as a hub for instructors to manage attendance and view schedules, initiating interactions based on the instructor's selected task.
Attendance Tracker	Collaborates with **Check-in** to verify attendance marked by students. Interacts with **Database Access** to store attendance records and with **Notifications** to alert students of their attendance status. Instructor updates to attendance data affect the **Attendance History** for students.
Schedule Viewer	It depends on **Database Access** to retrieve and display class schedules. Interacts with **ViewAttendance** to link schedules with attendance records for each session. Schedule updates directly impact **InstructorHomepage** by displaying the latest information.
ViewAttendance	Depends on **Attendance Tracker** for data consistency and accuracy. Interacts with **InstructorHomepage** to display session-specific attendance. Attendance changes affect **Attendance History** and **Report Generation** for accuracy in tracking and reporting.
StudentHomepage	Central access point for students to interact with **Check-in**, **Attendance History**, and **Notifications**. Collaborates with these components based on the student's actions and choices, impacting only the specific component chosen by the student.

Check-in	Interacts with **Attendance Tracker** to confirm attendance based on Face ID and location data. Collaborates with **Notifications** to inform students of check-in success or failure. A successful check-in updates **Attendance History** and **InstructorHomepage** for real-time records.
Attendance History	Depends on **Attendance Tracker** and **Check-in** for up-to-date attendance records. Interacts with **StudentHomepage** for display and with **Report Generation** for generating historical attendance reports. Changes in history records reflect attendance tracking trends.
Notifications	Collaborates with all role-specific homepages (**AdminHomepage**, **InstructorHomepage**, **StudentHomepage**) to display system alerts. Interacts with **User Management** and **Attendance Tracker** to provide updates about attendance status, schedule changes, and other critical alerts.

8.7 Resources

The following table outlines the resources required, affected, or managed by each component within the *Automated Attendance System*. Additionally, potential race conditions and deadlock scenarios are identified, along with strategies for resolution.

Table 22: Component Resources

Component	Resources Managed / Needed	Potential Issues and Solutions
Login	**Memory:** Temporary storage for session data during login. **Processor:** For verifying Face ID or password.	**Race Condition:** Multiple simultaneous login attempts could lead to conflicting session data. **Solution:** Use session locks to ensure only one login request per user at a time.
Forgot Password	**Database Access:** For identity verification and password updates. **Notification Service:** For sending reset emails.	**Deadlock:** Simultaneous access requests for the password reset feature might block other processes. **Solution:** Implement a timeout mechanism to release locks on email service or database access.
Sign Out	**Memory:** Frees up session data.	**Race Condition:** Simultaneous logouts could attempt to release the same session data. **Solution:** Ensure each session has a unique identifier to avoid conflicts.
ViewProfile	**Database Access:** Retrieves user profile data for display.	**Race Condition:** Multiple access requests could cause data retrieval conflicts. **Solution:** Implement read locks to ensure data consistency during concurrent access.
UpdateProfile	**Database Access:** For updating user details in real-time.	**Race Condition:** Concurrent profile updates may lead to inconsistent data. **Solution:** Use write locks

		to ensure only one update is processed at a time.
AdminHomepage	**Database Access:** Reads system and user activity data for display.	**Deadlock:** Concurrent requests for system status updates could lead to resource locking. **Solution:** Use a priority queue to manage admin access to system data.
User Management	**Database Access:** For creating, updating, and deleting user accounts.	**Race Condition:** Multiple admins modifying user data simultaneously could result in data conflicts. **Solution:** Implement transaction-based operations with rollbacks in case of conflicts
Report Generation	**Processor:** For aggregating and formatting report data. **Database Access:** Reads historical data for report accuracy.	**Deadlock:** Large report requests may block database access for other components. **Solution:** Allocate processing priority based on request type and use batch processing for large reports.
System Monitoring	**Memory:** For storing real-time system status updates. **Database Access:** Logs system events.	**Race Condition:** Multiple monitoring requests may cause data inconsistencies. **Solution:** Use timestamped event logs to ensure sequence consistency.
InstructorHomepage	**Database Access:** Fetches attendance and schedule data.	**Deadlock:** Multiple instructors accessing shared data could lead to resource blocking. **Solution:** Use concurrent read operations with controlled write access to avoid deadlocks.
Attendance Tracker	**Database Access:** For recording student attendance data. **Processor:** For Face ID and geolocation verification.	**Race Condition:** Simultaneous attendance updates may result in inconsistent data. **Solution:** Implement locking on individual

		student records during attendance updates.
Schedule Viewer	**Database Access:** For retrieving and displaying up-to-date class schedules.	**Race Condition:** Concurrent schedule access could cause data display errors. **Solution:** Apply read locks to ensure consistency during multiple accesses.
ViewAttendance	**Database Access:** For retrieving student attendance records.	**Race Condition:** Multiple concurrent data retrievals may cause timing conflicts. **Solution:** Use query caching to manage frequent access requests.
StudentHomepage	**Memory:** For loading user-specific data.	**Race Condition:** Simultaneous access to student data could cause loading conflicts. **Solution:** Use session locks and data caching for efficient loading.
Check-in	**Processor:** For Face ID and geolocation processing. **Database Access:** For updating attendance records.	**Deadlock:** If Face ID and location verification are accessed simultaneously, it may cause deadlock. **Solution:** Sequentially process Face ID and geolocation data.
Attendance History	**Database Access:** For accessing and displaying historical attendance records.	**Race Condition:** Multiple access requests could lead to delayed response times. **Solution:** Implement read-only access for historical data to ensure non-conflicting retrieval.
Notifications	**Notification Service:** For delivering real-time alerts. **Memory:** For queueing pending notifications.	**Deadlock:** Multiple notifications queued for the same user could block the service. **Solution:** Use a message queue with prioritization to avoid delays.

8.8 Processing

The detailed processing logic for each component, including input and output handling, state changes, concurrency, and exception management, is outlined in Section 3.2 of the *CS 411 - SRS* G1 document. This section provides specific descriptions of system components, algorithms, data processing flows, and handling of abnormal conditions.

8.9 Interface/Exports

The following table provides an overview of each component's interfaces, services, and exports, specifying the set of data types, subroutines, and constants each provides to other parts of the system.

Table23: components services.

Component	Services and Exports
Login	AuthenticateUser(username, password): Validates credentials and returns session token. LockAccount(userID): Locks account after failed attempts.
Forgot Password	SendResetEmail(userID): Sends password reset link. VerifyResetToken(token): Confirms reset token validity.
Attendance Tracker	MarkAttendance(studentID, timestamp): Marks attendance and returns status. VerifyLocation(coordinates): Confirms student location within campus bounds.
Report Generation	GenerateAttendanceReport(courseID, dateRange): Aggregates and returns report data. ExportReport(format): Exports report in specified format (PDF, CSV).
Notifications	SendNotification(userID, message): Delivers real-time notifications. QueueNotification(userID, message): Adds notifications to a pending queue for batch processing.

8.10 Detailed Subsystem Design

The detailed subsystem design, including component structure, behavior, and information flow, is outlined in Section 3.2 of the *CS 411 - SRS* G1 document. This section provides in-depth descriptions of each software component, supported by flowcharts and diagrams that illustrate interactions and data flow

9. Other Design Features

Additional design features include:

- **Security Protocols**: Multi-layer authentication (Face ID and password) for sensitive functions like attendance tracking and report generation.
- **Optimization Techniques**: Caching frequently accessed data (e.g., student schedules) to improve load times and reduce server calls.
- **Scalability Considerations**: Modular design supports system scaling, allowing additional user roles or features without significant refactoring.
- **Exception Logging**: All exceptions are logged with stack traces and error codes, enabling quick debugging and providing data for future improvements.

10. Requirements Traceability Matrix

The Requirements Traceability Matrix below links each functional requirement from the *System Requirements Specification (SRS)* to the corresponding design elements in this *System Design Specification (SDS)*. This traceability ensures that each requirement is addressed within the system's design.

Table24: Requirements Traceability Matrix

Association Code in SRS	Functional Requirement	User Role(s)	Technical Assumptions or User Needs	Association Code in SDS	System Component
3.2.1.1	Login functionality allowing all roles to access the system	Admin, Instructor, Student	Secure credential verification (username/password)	7.1.1	Login Component
3.2.1.2	Password reset functionalities for all roles	Admin, Instructor, Student	Secure identity verification	7.1.2	Forgot Password Component
3.2.1.3	Profile editing functionalities for all users	Admin, Instructor, Student	Real-time profile update and data consistency	7.1.3	Profile Management Component
3.2.2.1	Admin can approve or reject requests	Admin	Admin permissions for request management	7.2.1	Admin Interface
3.2.3.1	Instructor can view and manage schedules	Instructor	Real-time access to class schedules	7.3.1	Schedule Management

3.2.3.2	Instructor can update attendance status	Instructor	Accurate tracking and updating of attendance	7.3.2	Schedule Viewer
3.2.4.1	Students can view their attendance status	Student	Accurate attendance reflection	7.4.1	Attendance Tracker
3.2.4.2	Students can view attendance history	Student	Accessible history for attendance tracking	7.4.2	Schedule Viewer
3.2.4.4	Student attendance verification with Face ID or OTP	Student	Biometric and OTP functionality	7.4.1	Check-in Component