V Software Requirements Specification

for

Trackademia

Version 2

Prepared by

GROUP: Trackademia

Tajaun Tomlinson	**620163325**	**tajaun.tomlinson@mymona.uwi.edu**
Davaun Glave	**620163313**	**davaun.glave@mymona.uwi.edu**
Seantay Johnson	**620155318**	**seantay.johnson@mymona.uwi.edu**
Rory Williams	**620165266**	**rory.williams@mymona.uwi.edu**
Ricardo Foster	**620168530**	**ricardo.foster@mymona.uwi.edu**
Solay Wellington	**620170530**	**solay.wellington@mymona.uwi.edu**

Course Instructor: **Dr. Ricardo Anderson**

Course: **COMP2140 – Introduction to Software Engineering**

Studio Facilitator: **Mr. Eyton Ferguson**

Date: **08/11/2025**

TABLE OF CONTENTS

Overall Description..... 1
1.1 Product Context and Need..... 1
1.2 Product Functionality..... 1
1.3 Stakeholders and Users Characteristics..... 2
1.4 Operating Environment..... 2
1.5 Design and Implementation Constraints..... 2
1.6 Assumptions and Dependencies..... 3

Specific Requirements..... 5
2.1 External Interface Requirements..... 5
2.2 System Features..... 7
2.3 Use Case View..... 15

Other Non-functional Requirements..... 16
3.1 Safety and Security Requirements..... 16
3.2 Software Quality Attributes..... 17
3.2 Software Quality Attributes..... 18

Other Requirements..... 14

Appendix..... 15

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Full Group members	Initial Draft of the SRS based on project proposal and team analysis	31/10/25
1.1	All group members	Added three new functional requirements (FR-4, FR-5, FR-6) for student roster management and reporting. Updated relevant sections.	02/11/2025
1.2	All group members	Added performance, safety & security and updated the software quality attributes of the non-functional requirements	07/11/2025
1.3	Tajaun Tomlinson	Altered requirement #4 from allowing lecturer being to add students to lecturer's course, allowing Admin being to add students to lecturer's course	

● Overall Description

1.1 Product Context and Need

Trackacademia is a software system that was created with intentions to automate and enhance the academic institutions student attendance procedures. The existing methods used to take attendance, roll calls and paper sign-in sheets, are prone to error, in efficiency and the possibility of proxy attendance. This solution improves academic integrity and administrative efficiency by meeting the demand for a dependable, safe, and transparent system that gives instructors control over the attendance process and enables students access to real-time attendance data.

1.2 Product Functionality

The Trackademia system's primary features include:

- For Students: To see their own attendance histories, along with percentages and whole class history, for courses they have enrolled in.
- For instructors: To set up and oversee class sessions, specify time slots for sign-in, and get attendance data for their classes.
- In order to ensure that attendance is only recorded from the authorized lecture location, the system will validate a student's eligibility to sign in by verifying their physical location and network connections.

1.3 Stakeholders and Users Characteristics

- Students:
    - Primary users. They examine their own attendance records and log into classes using the technology.
    - Features: They have an easy-to-use interface; they have various levels of technical expertise; they can only access their own data.
- Teachers and lecturers:
    - Vital Users. They control the list of enrolled students, establish class sessions, provide attendance settings(time, place), produce and export attendance reports, and keep track of attendance.
    - Features: Moderate to high technical skill; needs access to data management tools and summary data; needs control and configuration choices.
- Academic Administrative:
    - Secondary users. They control system-wide parameters and produce institutional reports.
    - Feature: High-level system access with an emphasis on reporting and data quality.

1.4 Operating Environment

Trackademia will be a web-based application that can be accessed on desktop and mobile devices using contemporary web browsers (Chrome, Firefox, Safari, and Edge). The server will run in a Node.js runtime environment on a cloud platform (like AWS or Azure). The system will utilize a relational database (PostgreSQL). The device's location service (GPS) and network connection must be accessible to client-side applications.

1.5 Design and Implementation Constraints

1. Hardware & Device Constraints

- The attendance marking feature depends on students' mobile devices supporting GPS functionality.

- Accuracy is limited by the quality of the device's location hardware, environmental factors , and available internet connectivity.
- Some low-end devices may take longer to obtain accurate coordinates, affecting response times.

2. Security and Privacy Constraints

- Location data must be securely stored and transmitted using HTTPS/TLS.
- Student personal data (ID numbers, attendance records, location logs) must comply with institutional privacy policies and local data protection regulations.
- Authentication must use secure session management (e.g., token-based authentication).

3. Network & Communication Constraints

- Continuous internet access is required for attendance submission; offline mode attendance is not supported.

- GPS verification requires real-time communication with the backend for validation.

1.6 Assumptions and Dependencies

Assumptions

- In order to facilitate real-time attendance submission and data retrieval, instructors and students have dependable internet connectivity on their personal devices or institutional networks.
- Students will provide the system location rights to verify their attendance if they have GPS-enabled devices (such as laptops or cellphones with location services).

- GPS signals will be precise enough to confirm a user's location within the necessary range of a lecture hall or classroom.
- When registering, users will give accurate information (such as their ID number), and the school will guarantee that student and course data is accurate.
- Location data collection and storage for academic attendance reasons will be permitted per institutional rules.

Dependencies

- For precise location retrieval, the system depends on the dependability and accessibility of third-party geolocation services or browser APIs.
- To connect accounts with official student ID numbers, the system requires access to the school's student database or identity management system.
- The system's ability to process and store data depends on web hosting and server infrastructure, including database servers.
- It may be necessary to integrate with other services (such as SMS/email notification APIs or authentication providers), and its unavailability may have an impact on operations.
- To accurately calculate attendance requirements for each course, the system could rely on class schedule data supplied by the university.

● Specific Requirements

2.1 External Interface Requirements

2.1.1 Hardware Interfaces

Standard web-enabled technology, including laptops, cellphones, and institutional servers, is used by the system. There is no need for specific bespoke hardware because GPS interaction is handled using browser APIs.

2.1.2 Software Interfaces

Through a compatible web browser, the web-based attendance monitoring system communicates indirectly with the host operating system. Networking and location services are among the device-level resources that the operating system controls and that the system can access through browser APIs.

Operating Systems Supported

- The system is made to function on the main operating systems, such as:

  Mobile operating systems: iOS and Android

  Desktop operating systems: Linux, macOS, and Windows

This guarantees accessibility and flexibility for instructors and students alike.

2.1.3 Communications Interfaces

Using a secure internet connection, the system will interact with client devices (lecturers and students) using common web protocols. Every communication between the client and server will take place via HTTPS, guaranteeing that all data, including login passwords, location data, and attendance records, is encrypted while being transmitted. The system will transmit and receive data

using common HTTP request/response forms, such as JSON. The backend server will reply with confirmation messages, attendance records, or instructor updates after receiving GPS coordinates and authentication data from the browser.

The client and server will communicate using REST-style web service protocols, with distinct endpoints for record retrieval, attendance submission, and authentication. In order to sustain responsive communication, the system relies on the reliability of third-party or institution-provided hosting. Sensitive data will be safeguarded using standard communication security techniques (such TLS), and only authorized users will be able to access protected data thanks to session management or authentication tokens. Beyond basic network access, no specialist communication gear is needed.

2.2 System Features

Feature ID: FR-1

Use Case: Check Location/Verify Network/IP

Rationale: This feature is necessary so that the system can ensure that the students are actually at the lecture when they sign attendance

Relates to/Dependencies:

Priority: High

Team Owner: Rory Williams

User Requirement: The system should only allow attendance to be taken when the device's location matches the lecture room's location and a valid network connection is established.

System Requirements:

1. The system shall be able to check the device's location and compare it to the given area of the lecture room.

2. The system shall be able to verify the IP address of the connected network on the device and ensure that the device is connected to a valid IP address.

3. The system shall be able to set a range from the exact location where attendance can be taken.

Acceptance Criteria:

1. The system is able to find the location of the device and compare it to a set location 100% of the time.

2. The system is able to distinguish whether the user is connected to a valid network 100% of the time.

Feature ID: FR-2

Title: View Attendance Record

Use Case: Viewing Attendance History

Rationale: To allow students to track their attendance performance, identify gaps and ensure compliance with course requirements.

Relates To / Dependencies: Student Portal / Attendance Module

Priority: High

Team Owner: Seantay Johnson

User Requirement:

The system shall allow the student to view their attendance record for each enrolled class in a clear and accessible format.

System Requirements:

1. The system shall display the student's attendance record for all enrolled courses.

2. The system shall show the attendance percentage, total classes attended and total classes held for each course.

3. The system shall accurately retrieve attendance data from the database for the logged-in student.

Acceptance Criteria:

1. The system shall display the attendance record for every course in which the student is enrolled.

2. The system shall show the attendance percentage, total classes attended and total classes held for each course.

3. The system shall display a message stating "No attendance data available at this time" if the student has no recorded attendance.

4. The system shall allow only the logged-in student to view their own attendance data.

Feature ID: FR-3

Title: Session Notifications and Alerts System

Use Case: Receive Session Alerts and Notifications

Rationale:

To proactively engage users by providing timely, relevant information about session status changes, attendance events, and academic performance metrics. This feature enhances user experience, reduces missed attendance opportunities, and improves academic accountability through automated communication channels.

Relates to/Dependencies:

- FR-1 (Location/Network Verification): Notifies when verification fails

- FR-2 (View Attendance History): Sends alerts for low attendance percentages

- FR-4 (Student Information Upload): Notifies lecturers of new student enrollments

- FR-5 (Generate Attendance Reports): Alerts when reports are ready for download

- FR-6 (Generate Timestamp): Confirms successful attendance with timestamp

Priority: Medium-High

Team Owner: Tajaun Tomlinson

User Requirements:

For Students:

1. The system shall notify students 15 minutes before each scheduled session start time.

2. The system shall provide immediate confirmation when attendance is successfully marked.

3. The system shall alert students when their attendance percentage falls below 75% for any enrolled course.

4. The system shall notify students of any changes to session details (time, location, cancellation).

5. The system shall send weekly attendance summaries every Monday morning.

For Lecturers:

1. The system shall alert lecturers when a session they created becomes active.

2. The system shall notify lecturers when attendance for a session falls below 50%.

3. The system shall alert lecturers about students with consecutive absences (3+ sessions).

4. The system shall notify lecturers when attendance reports are successfully generated.

5. The system shall allow lecturers to send custom notifications to all students in a course.

System Requirements:

1. **Notification Delivery System:**

- The system shall support multiple notification channels: in-app alerts, email, and optional push notifications.

- The system shall deliver notifications within 60 seconds of the triggering event.

- The system shall maintain a delivery log with timestamps and status for all notifications.

2. **Notification Configuration:**

- The system shall allow users to customize which notifications they receive.

3. **Automated Triggers:**

- The system shall automatically trigger notifications based on:

- Session timing (start, end, upcoming)

- Attendance events (success, failure, threshold breaches)

- System events (maintenance, updates)

4. **Manual Notification Capability**:

- The system shall provide lecturers with a notification composer interface.

- The system shall allow lecturers to send notifications to:

- Individual students

- Course-specific groups

- All enrolled students

- The system shall provide delivery confirmation for manual notifications.

5. **Notification Management:**

- The system shall maintain a notification history accessible to users.

- The system shall allow users to mark notifications as read/unread.

- The system shall provide notification filtering and search capabilities.

Acceptance Criteria:

1. **Timeliness:**

- Session reminders are delivered exactly 15 minutes before session start time.

- Attendance confirmations are delivered within 5 seconds of successful marking.

- System-generated alerts are delivered within 60 seconds of triggering conditions.

2. **Accuracy:**

- Notifications contain correct session details (course, time, location).

- Attendance alerts reflect accurate percentages and statistics.

- Recipient lists for course notifications include all currently enrolled students.

3. **User Control:**

- Users can successfully enable/disable notifications.

4. **Reliability:**

- System delivers 99% of notifications successfully.

- Notification logs accurately record all delivery attempts and outcomes.

5. **Performance:**

- Notification delivery does not impact core attendance marking functionality.

6. **Content Quality:**

- Notifications include clear, actionable information.

- Error notifications provide specific guidance for resolution.

- Email notifications include appropriate subject lines and formatting.

- All notifications include timestamps and clear sender identification.

Feature ID: FR-4

Title: Student Information Upload

Use Case: Upload Students Information to the System

Rationale: To allow a Admin to update the system of their course with students' information such as their name and ID numbers.

Relates to/Dependencies: None

Priority: High

Team Owner: Davaun Glave

User Requirements:

- The Admin shall be able to input or upload a student's name and ID number so that the student is added to the Lecturers course's student list.

System Requirements:

- The system shall allow the Lecturer to provide a student's name and ID number.
- The system shall validate that both the name and ID number conform to the required format and are not blank.

- The system shall store the student's name and ID number in the associated course's student list upon successful validation.
- The system shall check for existing records and prevent duplicate student ID entries in the same course.

Acceptance Criteria:

- The system shall store a student's name and ID number in the course's student list when uploaded by the Lecturer.
- The system shall only store a student's information if both the name and ID number are provided and meet the expected input format. If not, an error message shall be displayed.
- If the student ID already exists in the course's student list, the system shall prevent duplicate entry and notify the Lecturer that the student is already enrolled.

Feature ID: FR5

Title: Allow the lecturer to export the attendance statistics.

Use Case: Generating Attendance Summary

Rationale: To generate attendance reports for each course, grade submissions and analyze the relationship between attendance and the students' overall performance.

Relates to/Dependencies: FR4- Allow the lecturer to upload the list of students (names and IDs) who are enrolled in a given course section.

Priority: Medium

Team Owner: Solay Wellington

User Requirement: The system shall allow the lecturer to use the data collected to export attendance statistics in various ways for a respective course in a format that can be downloaded.

***System Requirements*:**

1. The system shall generate a summary of attendance data for a specific course, including the amount of presents and absents for each student.
2. The system shall support exporting attendance data in Excel.

3. The system shall allow the lecturer to filter out data according to a specific data range or choose “full semester” before exporting the statistics.

Acceptance Criteria:

1. The lecturer should be able to export attendance statistics all of the time when a valid course and data range is entered.
2. The exported file downloads successfully and correctly containing complete attendance data for every student in a specific course.
3. If no attendance data exists for a specific time period then the system will display an error message instead of generating an empty file.

Feature ID: FR-6

Title: Generator Timestamp

Use Case: Sign-in for Students

Rationale: To reliably create an audit trail and provide precise data for attendance reporting and dispute resolution by automatically and precisely recording the moment a student signs in.

Relates to/Dependencies: FR-1, FR-2, FR-3, and FR-5

Priority: High

Team Owner: Ricardo Foster

User Requirement: When a student successfully logs in for a class, the system will automatically record the time and date.

System Requirements:

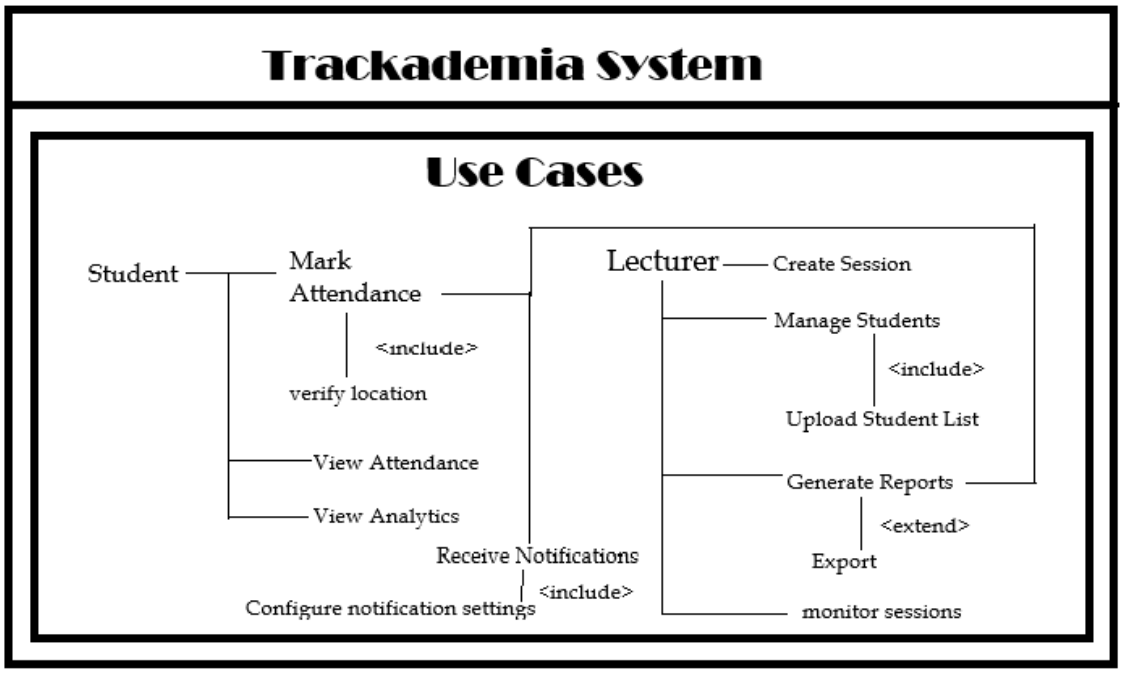
1. The system will record the current server date and time after location and network verification are successfully completed.

2. This timestamp will be permanently linked to the student's attendance record for that particular class session by the system, which will save it in the database.
3. Both the teacher (in the course attendance report) and the student (in their personal record) will see this timestamp displayed by the system.

Acceptance Criteria:

100% of the time, the system logs a timestamp for each successful sign-in. The timestamp matches the server's time within one second.

2.3 Use Case View



2.3 Use Case Narratives

UC-01: Mark Attendance

- **ID**: UC-01
- **Primary Actor**: Student
- **Secondary Actors**: System (location verification), Lecturer (session creator)
- **Preconditions**:
    - Student is authenticated and logged into the system
    - Student is enrolled in at least one course with active sessions
    - Student's device has location services enabled
    - At least one active session exists for the student's enrolled courses
- **Main Success Scenario**:
    - Student navigates to the "Mark Attendance" section from their dashboard
    - System displays a list of active sessions for the student's enrolled courses
    - Student selects a specific session to mark attendance
    - System automatically captures the device's GPS coordinates
    - System verifies the location is within the allowed radius of the session location
    - System verifies the device is connected to a valid institutional network
    - System records attendance with the current server timestamp
    - System displays confirmation: "Attendance successfully marked at [time]"
    - System triggers an "Attendance Confirmed" notification to the student
- **Alternative Flows**:
    - **A1: Location Outside Allowed Range** 5a. System detects the student's location is outside the allowed radius 5b. System displays error: "You must be within [X] meters

of [location] to mark attendance" 5c. System triggers an "Attendance Failed - Location" notification 5d. Use case ends with attendance not recorded

    - **A2: Network Verification Fails** 6a. System cannot verify a valid institutional network connection 6b. System displays error: "Connect to campus Wi-Fi to mark attendance" 6c. System triggers an "Attendance Failed - Network" notification 6d. Use case ends with attendance not recorded
    - **A3: Attendance Already Marked** 7a. System detects an existing attendance record for this session 7b. System displays information: "Attendance already marked at [previous time]" 7c. Use case ends without creating a duplicate record
- **Postconditions**:
    - If successful: Attendance record created in the database with timestamp
    - If failed: No attendance record created; error logged in the system
    - Notification sent to student regarding the outcome
    - Lecturer can view updated attendance status in real time
- **Special Requirements**:
    - Requires GPS hardware on the student's device
    - Requires internet connectivity
    - Location accuracy must be within 10 meters
- **Frequency of Use**: Multiple times daily during class periods

---

UC-02: Create Session

- **ID**: UC-02

- **Primary Actor**: Lecturer
- **Secondary Actors**: System (scheduling), Students (notification recipients)
- **Preconditions**:
    - Lecturer is authenticated and logged into the system
    - Lecturer has at least one course created in the system
    - Lecturer has appropriate permissions for session creation
- **Main Success Scenario**:
    - Lecturer navigates to "Create Session" from the dashboard
    - System displays the session creation form with required fields
    - Lecturer selects a course from the dropdown list of their courses
    - Lecturer enters session details:
        - Session name/title
        - Date and start time
        - Duration in minutes
        - Physical location (building/room)
        - Allowed attendance radius (10-500 meters)
    - System validates all entered information
    - Lecturer submits the session creation form
    - System creates the session with "upcoming" status
    - System schedules notifications for:
        - Lecturer: Session creation confirmation
        - Enrolled students: Session reminder (15 minutes before start)
    - System displays confirmation: "Session created successfully"

- **Alternative Flows**:
    - **A1: Invalid Date/Time** 5a. System detects the session time is in the past 5b. System displays error: "Session time must be in the future" 5c. Lecturer corrects the date/time and resubmits
    - **A2: Overlapping Sessions** 5d. System detects a time conflict with an existing session 5e. System displays warning: "Session overlaps with [existing session]" 5f. Lecturer can choose to proceed or adjust the times
- **Postconditions**:
    - New session record created in the database
    - Session appears in the lecturer's session management view
    - Enrolled students can see the upcoming session
    - Notifications scheduled for session reminders
- **Special Requirements**:
    - Session times must respect institutional scheduling policies
    - Maximum session duration: 4 hours
- **Frequency of Use**: Multiple times per week per course

UC-03: Manage Students

- **ID**: UC-03
- **Primary Actor**: Lecturer
- **Secondary Actors**: System (validation), Students (affected users)
- **Preconditions**:

    - Lecturer is authenticated and logged in
    - Lecturer has at least one active course
    - Lecturer has permission to manage student rosters
- **Main Success Scenario**:
    - Lecturer selects a course from their dashboard
    - System displays the current student roster for the selected course
    - Lecturer chooses the "Add Student" option
    - System displays the student addition interface
    - Lecturer can either:
        - Manually enter student ID and name
        - Upload a CSV file with student data
    - For manual entry: a. Lecturer enters student ID and full name b. System validates the ID format and checks for duplicates c. System adds the student to the course roster
    - For CSV upload: a. Lecturer selects and uploads a CSV file b. System validates the file format and data c. System processes valid records and skips invalid ones d. System provides an import summary report
    - System sends a "Course Enrollment" notification to the added students
    - System updates attendance tracking for the newly added students
- **Alternative Flows**:
    - **A1: Duplicate Student Entry** 6b. System detects the student is already enrolled in the course 6c. System displays warning: "Student [ID] already enrolled" 6d. Lecturer can skip or update student information

    - **A2: Invalid CSV Format** 7b. System detects a malformed CSV file 7c. System displays specific format errors 7d. Lecturer corrects the file and re-uploads
    - **A3: Remove Student** 3a. Lecturer selects "Remove Student" instead 3b. System displays a confirmation dialog 3c. Lecturer confirms removal 3d. System removes the student from the course roster 3e. System retains attendance history but stops future tracking
- **Postconditions**:
    - Course roster updated with added/removed students
    - Students receive enrollment notifications
    - Attendance tracking adjusted accordingly
    - Lecturer can view the updated roster immediately
- **Special Requirements**:
    - Student ID must follow institutional format
    - Maximum class size: 300 students
- **Frequency of Use**: Beginning of each semester, with weekly adjustments

---

UC-04: Generate Reports

- **ID**: UC-04
- **Primary Actor**: Lecturer
- **Secondary Actors**: System (data processing), Students (data subjects)
- **Preconditions**:
    - Lecturer is authenticated and logged in

    - Lecturer has at least one course with attendance data
    - Course has completed at least one session
- **Main Success Scenario**:
    - Lecturer selects "Generate Reports" from the course dashboard
    - System displays report generation options:
        - Attendance summary (current session)
        - Course attendance history (date range)
        - Student performance analytics
        - Export format selection (PDF, Excel, CSV)
    - Lecturer selects report type and date range
    - Lecturer selects export format (default: PDF)
    - System processes attendance data and generates the report
    - System displays a preview of the generated report
    - Lecturer can: a. Download the report immediately b. Save the report to their document library c. Schedule regular report generation
    - System sends a "Report Generated" notification to the lecturer
    - System logs the report generation for audit purposes
- **Alternative Flows**:
    - **A1: No Attendance Data** 5a. System detects no attendance data for the selected period 5b. System displays message: "No attendance data available for selected period" 5c. Lecturer adjusts the date range or cancels

- **A2: Large Data Set** 5d. System detects a large data set requiring extended processing 5e. System displays: "Processing large data set, estimated time: X minutes" 5f. System sends a notification when processing completes
- **A3: Export to Excel** (<<extend>> relationship) 4a. Lecturer selects "Export to Excel" format 4b. System generates an Excel file with formatted data 4c. System includes multiple tabs: Summary, Details, Charts 4d. System applies Excel formatting and formulas

- **Postconditions**:
  - Report generated and available for download
  - Report saved to the lecturer's document library (if selected)
  - Notification sent to the lecturer
  - Audit log entry created
- **Special Requirements**:
  - Reports must exclude personally identifiable information when shared
  - Excel exports must work with Excel 2010 and later versions
- **Frequency of Use**: Weekly for course management, end of semester for grading

---

UC-05: Receive Notifications

- **ID**: UC-05
- **Primary Actor**: Student/Lecturer
- **Secondary Actors**: System (notification engine)
- **Preconditions**:

    - User is authenticated and logged into the system
    - User has notifications enabled in their settings
    - User's device/browser supports push notifications (if enabled)
- **Main Success Scenario**:
    - System event triggers a notification requirement
    - System checks whether the user has notifications enabled
    - If enabled, system formats the notification message
    - System delivers a push notification to the user's device
    - User receives the notification alert
    - User can click the notification to view details in the system
    - System logs notification delivery
- **Alternative Flows**:
    - **A1: Notifications Disabled** 2a. System detects the user has disabled notifications 2b. System skips notification delivery 2c. Event is still processed, but no notification is sent
    - **A2: Push Notification Blocked** 4a. User's browser/device blocks push notifications 4b. System falls back to in-app notification only 4c. Notification appears the next time the user accesses the system
    - **A3: Critical Notification** 2d. System detects the notification is critical (e.g., system outage) 2e. System overrides the disabled setting and delivers the notification 2f. Includes a "Critical" label in the notification
- **Postconditions**:
    - User receives timely information about system events

    - Notification logged for record keeping
    - User can view notification history in the app
- **Special Requirements**:
    - Push notifications must work on Chrome, Firefox, Safari, and Edge
    - Notifications delivered within 30 seconds of the event
    - Critical notifications cannot be disabled
- **Frequency of Use**: Multiple times daily during active periods

---

UC-06: Toggle Notifications

- **ID**: UC-06
- **Primary Actor**: Student/Lecturer
- **Secondary Actors**: System (settings storage)
- **Preconditions**:
    - User is authenticated and logged into the system
    - User has access to the settings page
- **Main Success Scenario**:
    - User navigates to "Settings" from the profile menu
    - System displays the settings page with a "Notifications" section
    - User sees a simple toggle switch: "Enable Notifications"
    - Current status is shown (On/Off)
    - User clicks the toggle to change the setting
    - System shows a brief confirmation: "Notifications [Enabled/Disabled]"

    - System immediately applies the new setting
    - Changes take effect for the next notification event
- **Alternative Flows**:
    - **A1: Browser Permission Required** 4a. User enables notifications for the first time 4b. Browser shows a permission request for push notifications 4c. User grants permission 4d. System confirms successful setup
    - **A2: Permission Denied** 4e. User denies browser permission for push notifications 4f. System shows message: "Notifications will only appear in-app" 4g. System still enables in-app notifications
- **Postconditions**:
    - Notification setting saved to the user profile
    - Setting applied immediately to future notifications
    - User receives confirmation of the change
- **Special Requirements**:
    - Setting must be stored persistently
    - Change must take effect within 5 seconds
    - Must work across all modern browsers
- **Frequency of Use**: Initial setup, with occasional adjustments

● Other Non-functional Requirements

3.1 Performance Requirements

System Response Time

- The system shall load a student's attendance record within 3 seconds under normal network conditions.
- Rationale*:* Students must quickly access real-time attendance data to maintain academic accountability.

Attendance Submission Processing

- When a student attempts to sign in for a class, the system shall validate GPS location and network eligibility within 5 seconds.
- Rationale: Attendance sessions are time-restricted, and delays may prevent students from being marked present.

Instructor Dashboard Performance

- When instructors generate attendance reports, the system shall process and display results within 7 seconds for up to 200 students per course.
- Rationale*:* Ensures smooth class management and reduces administrative bottlenecks.

Concurrent Users & Scalability

- The system shall support at least 500 simultaneous student logins and attendance submissions without performance degradation.

- Rationale*:* Peak usage occurs at the start of class periods across the institution.

3.2 Safety and Security Requirements

Accurate Location Validation

- The system shall prevent attendance submission if the student's location cannot be reliably verified (e.g., inaccurate GPS, disabled location services).

- Rationale: Prevents proxy attendance and maintains academic integrity.

Session Timeout for Safety

- All login sessions shall automatically expire after 30 minutes of inactivity to prevent unauthorized access on shared or misplaced devices.

- Rationale*:* Reduces the risk of unintended data exposure.

Security Requirements (Client Expectations)

The client expects a high level of data protection due to the sensitive nature of student information, location data, and academic records. The system must ensure the confidentiality, integrity, and availability of all stored and transmitted data.

Major Security Requirements:

Secure Authentication

- The system shall require token-based authentication with secure session management.

- Multi-factor authentication (MFA) shall be required for instructors and administrators.

Encrypted Data Transmission

- All data (including GPS coordinates, student IDs and attendance logs) shall be transmitted using TLS/HTTPS encryption.

Data Privacy Compliance

- The system shall comply with institutional privacy policies and any applicable data protection regulations.
- Location logs shall be stored only for the minimum time necessary.

Role-Based Access Control (RBAC)

- Students shall access only their own attendance data.
- Instructors shall access only their course-related records.
- Administrators shall have controlled, audited access to system-wide reports.

Audit Logging

- All attendance submissions, logins, and administrative actions shall be logged for audit purposes.

Database Security

- Sensitive fields such as student identifiers shall be encrypted at rest.

3.3 Software Quality Attributes

- The interface shall be designed with clear navigation, simple labeling, and mobile-friendly layouts to accommodate students with varying technical skill levels. This is achieved with consistent page layouts, and mobile-responsive UI design.
- The system shall be designed to allow for the expansion of users (e.g., from one department to multiple campuses) without requiring major architectural changes by using scalable cloud services and database indexing.

- The system's code will feature a modular design and will be documented, allowing for future updates (e.g., adding new attendance features or analytics reports) can be implemented with minimal rework. The use of frameworks such as Node.js and PostgreSQL, version control and adherence to naming conventions.
- The system shall be maintained by frequent checks by the Academic Administrative Team.

Appendix

1. The first brainstorming sessions for the project

Dates: September 15-29, 2025

Participants: Solay Wellington, Ricardo Foster, Seantay Johnson, Davaun Glave, Rory Williams and Tajaun Tomlinson

Method: Problem identification and cooperative brainstorming Results:

- identified issues with the manual attendance methods in use today.
- identified the main parties involved (students, instructors, and administrators)
- defined the goals and scope of the core system
- Initial feature concepts and limitations were documented.
- Minutes of meetings and concept diagrams kept on a shared team drive under "Project Initiation Documentation" serve as proof.

2. Analysis of Stakeholders Date of Workshop:September 27- October 7th, 2025

Participants: The whole development team; methodology: organized mapping and analysis of stakeholders

Results:

- Stakeholders were categorized based on their interest and impact.
- Specific user attributes for every stakeholder group
- found criteria that conflicted across several user groups.
- Priority user groups were established for preliminary development.

- Evidence: User persona templates and a stakeholder analysis matrix produced during the workshop

3. Consultation sessions with clients

Date: October 10, 2025

Participants were members of the development team (Tajaun Tomlinson, Seantay Johnson), with Dr. Ricardo Anderson, the course teacher, serving as a client proxy.

Semi-structured interview technique

Important Topics for Discussion:

Concerns about academic integrity given the present techniques of attendance

3. Institutional guidelines for the privacy of student data

Requirements for integration with current university systems

Departmental administration's reporting requirements

Documented Client Responses:

"The system must prevent proxy attendance - this is our primary concern"

"Lecturers need flexibility in setting attendance parameters for different course types"

"Any location data collection must comply with university data protection policies"

"Export functionality is essential for grade submission and administrative reporting"

Proof: October 10, 2025, requirement validation notes and email receipts with client

4. Analysis of Technical Feasibility

Date: October 22, 2025

Participants: Rory Williams and Ricardo Foster from the technical sub-team

Method: Identification of constraints and assessment of the technology stack

Results:

- ❖ GPS accuracy constraints were assessed for indoor location confirmation.
- ❖ evaluated the limitations of network reliance
- ❖ determined the requirements for browser compatibility
- ❖ Implementation issues with security and privacy that have been documented
- ❖ Evidence: discussions of architectural decisions and technical feasibility reports

5. Non-functional prerequisites Examination

Date: November 5, 2025

Participants: Davaun Glave and Solay Wellington's quality assurance team

Method: Scenario analysis of quality attributes

Results:

- Clearly defined, quantifiable performance standards
- Constraints on privacy and security
- established objectives for maintainability and usefulness
- Requirements for documented compliance

Evidence: Validation standards and quality attribute scenarios

Software Design Specification

for

Tracademia

Version 3.0

Prepared by

Tajaun Tomlinson	**620163325**	**tajaun.tomlinson@mymona.uwi.edu**
Davaun Glave	**620163313**	**davaun.glave@mymona.uwi.edu**
Seantay Johnson	**620155318**	**seantay.johnson@mymona.uwi.edu**
Rory Williams	**620165266**	**rory.williams@mymona.uwi.edu**
Ricardo Foster	**620168530**	**ricardo.foster@mymona.uwi.edu**
Solay Wellington	**620170530**	**solay.wellington@mymona.uwi.edu**

Course Instructor: **Dr. Ricardo Anderson**

Course:	**COMP2140 – Introduction to Software Engineering**
Studio Facilitator:	**Mr. Eyton Ferguson**
Date:	**03/12/2025**

TABLE of CONTENTS

Project Overview……2-3
1.1 Purpose
1.2 Client Background
1.3 Problem Description
1.4 Intended Users
1.5 Context of Use
1.6 Key User Requirements
1.7 Relevant Non-Functional Requirements

Architectural Design……4-11
2.1 General Constraints
2.2 Alternatives Considered
2.3 System Architecture Diagram
2.3.1 Architectural Description
2.3.2 Architecture Justification

Architecture Decomposition……12-15
3.1 Component Decomposition – *Classes OR Modules*
- Component Table (Requirements → Component → Class/Module)
3.2 Structural Design – *Class Diagram OR Structure Chart*
3.2.1 Design Notes

1.0 Project Overview

Trackacademia is being developed for academic institutions whose aim is to improve and secure their student attendance procedures. Many institutions continue to utilize antiquated procedures like roll calls, paper sign-in sheets, or manual records, which are ineffective, prone to human error, and susceptible to proxy attendance. These restrictions lower the accuracy of attendance statistics, which is frequently linked to participation grades, academic achievement, and institutional reporting.

Trackacademia tackles this issue by offering a centralized, automated, and secure attendance management system. Instructors are able to create and oversee class sessions, specify time intervals for attendance, and obtain comprehensive attendance reports with the software. Conversely, students are able to evaluate their own academic involvement and make well-informed decisions throughout the semester by viewing their attendance history, entire course summaries and percentages. Additionally, the system uses network and location verification to guarantee that attendance is only recorded from approved locations, improving academic integrity.

This project is designed to meet major user criteria: students must be able to check their attendance records in real time, instructors must be able to easily manage attendance sessions, and the system must accurately validate attendance eligibility. The overall architecture is further

influenced by a number of non-functional requirements, such as usability: an easy-to-use interface suitable for both students and instructors, performance: quick data loading and processing, security: to prevent unauthorized access or fraudulent attendance, and reliability: consistent and accurate attendance retrieval. These specifications together contribute to the Trackacademia system's technological design choices.

2.0 Architectural Design

2.1 General Constraints

Tracademia has to operate within the limitations of academic IT environments and common web technologies. Users will access the system through contemporary web browsers like Chrome or Firefox, and it will operate on standard university hardware using a client-server architecture. In order to support a range of internet speeds, the architecture must offer lightweight data transfers and responsive web pages.

The system must be compatible with SQL-based data representations to connect the Virtual Learning Environment (VLE), an existing student database. Furthermore, attendance data must load within realistic time limitations (ideally less than two seconds) due to performance constraints, even during periods of high usage. Network restrictions arise as a result of the system's need to manage concurrent access from a large number of students.

Additional restrictions are imposed by security requirements: all client-server communication must be encrypted using HTTPS; user credentials must be securely verified; and access control must be implemented to guarantee that students are limited to seeing their personal attendance data.

2.2 Alternatives Considered

Though we narrowed down to the Layered Architecture Pattern, there were other options that could have been used, however, they were not well suited as the final choice. The other two architectural patterns that were considered during the design of Trackademia includes the Model View Controller (MVC) and the Client Server Architecture.

The Model View Architecture was evaluated because it effectively separates the presentation and interaction from the system data. It has three distinct components: the View component that handles the user interface, the Model component for data handling and the Controller component for managing user interaction. This is perfect for systems with highly interactive interfaces that are constantly updated. Furthermore, the Model View Controller's main focus is on the user interface separation and does not provide strong structure for areas such as location verification, security validation or attendance processing rules. Trackademia requires multiple responsibilities beyond user interface handling, so MVC alone would not be enough to manage all the other parts of a full attendance system. This project needs something more scalable and more modular.

Client Server Architecture was also analyzed due to its simplicity and suitability for centralized data access. In this model, clients are able to send requests to a single server that processes all attendance., authentication and reporting. Even though this pattern may be easy to implement, this architecture introduces multiple limitations. These include high traffic since multiple students are marking attendance simultaneously, which can overload the server which slows down performance. The server will then become a single point of failure which reduces reliability for a system that just b e at its peak during attendance hours. These issues made the client–server pattern less appropriate for Trackacademia’s real-time and scalable requirements.

2.3 System Architecture Diagram

2.3.1 Architectural Description

1. Presentation Layer - This layer is the user interface of the system, providing distinct experiences for students and instructors.
    - Student Web/Mobile Interface: Students can view their individual attendance records with this component. It will produce dashboards that clearly and intuitively display their attendance history, course summaries, and computed percentages.
    - Instructor Web/Mobile Interface: This component provides instructors with the tools to manage class sessions. It includes forms to create new sessions, set time intervals for attendance, and generate complete attendance reports.
    - Attendance Component: This is a crucial sub-component within both interfaces. It provides the specific form or button a student uses to mark their attendance during an active session.

Interaction & Requirement Fulfillment:

- These interfaces will be created with an emphasis on clarity, simplicity, and responsive design to function flawlessly on both web and mobile devices in order to satisfy the usability criterion.
- By offering the required panels and controls, they enable instructors to "easily manage attendance sessions" and students to "check their attendance records in real-time"—two essential functional requirements.
- To ensure that no business rules are handled on the client side and to support security, the interfaces will submit requests for all data and operations to the Business Logic Layer.

2. Business Logic Layer - This is where the logic occurs, business rules, validations and processes are executed.
    - Attendance Session Management: The lifespan of an attendance session is managed by this component. It handles teachers' requests to start, stop, and modify sessions while enforcing guidelines like the time limits.
    - Location & Network Verification: This is the main element that guarantees academic integrity. This engine compares the device's GPS position and/or IP address (to confirm access to a particular school network) with the permitted parameters for that session when a student tries to record attendance.
    - User Authentication & Authorization: This security feature makes sure that a user can only access authorized functions and data after logging in (Authorization) and checks user login credentials (Authentication). It addresses important security issues by, for instance, preventing an instructor from accessing a course they do not teach or a student from examining the data of another student.
    - Reporting & Analysis Engine: Requests for reports from the instructor interface are handled by this component. It compiles the database's raw attendance data, computes statistics (such as the proportion of students present), and arranges the results into thorough reports for the instructor.

    Interaction & Requirement Fulfillment:

    - After receiving requests from the Presentation Layer, this layer carries out intricate business logic and provides the outcomes. It is the main enforcer of the company and system security regulations.

- By directly addressing a fundamental issue statement, the Location & Network Verification and Validation Rules components are essential for preventing proxy attendance.
- In terms of performance, this layer is made to effectively handle data queries and validation checks, with the goal of meeting the sub-two-second response time even when concurrent access is being used.

3. Data Access Layer - This layer acts as an abstraction between the business logic and the database, managing all data persistence and retrieval operations.
    - Database Connection Management: This part is in charge of effectively establishing, maintaining, and pooling database connections. Because it eliminates the overhead of constantly opening and shutting connections, particularly when there is a high concurrent load, this is essential for performance and reliability.
    - Data Repositories: All CRUD (Create, Read, Update, Delete) activities for particular data entities are handled by these structured classes or modules. When a student successfully marks their attendance, for instance, the AttendanceRecordsRepository would include a method to InsertNewAttendanceRecord.
    - Query Handling & Data Mapping: This component converts database results into application objects that the Business Logic Layer can comprehend, for as turning a database row into a Student object, and vice versa.

Interaction & Requirement Fulfillment:

- The Data Access Layer is called by the Business Logic Layer to retrieve or save data without requiring knowledge of the underlying database structure. By preventing direct database exposure, this division improves both security and maintainability.
- This layer supports the performance need for fast data loading by centralizing all data access, ensuring consistent and effective database interaction.

4. Database Layer - This is the persistent storage layer where all system data is stored.
    - Student Table: Stores student profiles(ID, name, etc), linked to the existing university database.
    - Course & Session Tables: keeps track of course information as well as the specifics of each instructor-created attendance session, such as the course ID, time, location, and network restrictions.
    - Attendance Records Table: The core audit table records all attendance events, including timestamps, student IDs, and session IDs. This gives a consistent, unchanging history.
    - Systems Logs / Audit Trails Table: This table is critical for security, reliability, and debugging. It logs crucial system events such as login attempts (successful and unsuccessful), failures, and changes to critical data, allowing administrators to track down any problems or suspect behavior.

Interaction & Requirement Fulfillments:

- The Data Access Layer interacts directly with this layer by executing SQL queries to retrieve and save data.

- The structured tables, including Attendance Records and System Logs, assure data reliability and correctness, which is essential for the system's purpose.
- The design facilitates reporting and real-time data viewing by offering a structured and efficient source of information.

2.3.2 Architecture Justification

The layered architecture was selected for the Trackacademia system because it provides a structured, scalable, and maintainable approach that aligns directly with the system's functional goals (attendance tracking, location validation, analytics, user management) and the non-functional requirements (performance, security, reliability, maintainability, availability). It separates each responsibility into their own sections which allow for a simple and effective way to allow the system to be evolved over time. Each layer handles a specific function which allows for cleaner code, easier debugging and greater productivity.

This architecture also allows for scaling and upgrading. New features may be added and this layered approach allows for individual areas and sections to be changed without having to change the entire system. It also supports stronger security by ensuring that sensitive data is hard to retrieve by only allowing specific layers to have direct access to them. Unit testing is easier to be completed using this architecture and testing can be done on different layers simultaneously. It enables optimized database queries instead of processing in the UI and allows the business logic layer to be processed before it interacts with the database. Having each layer separated allows for connection with the cloud to be easier.

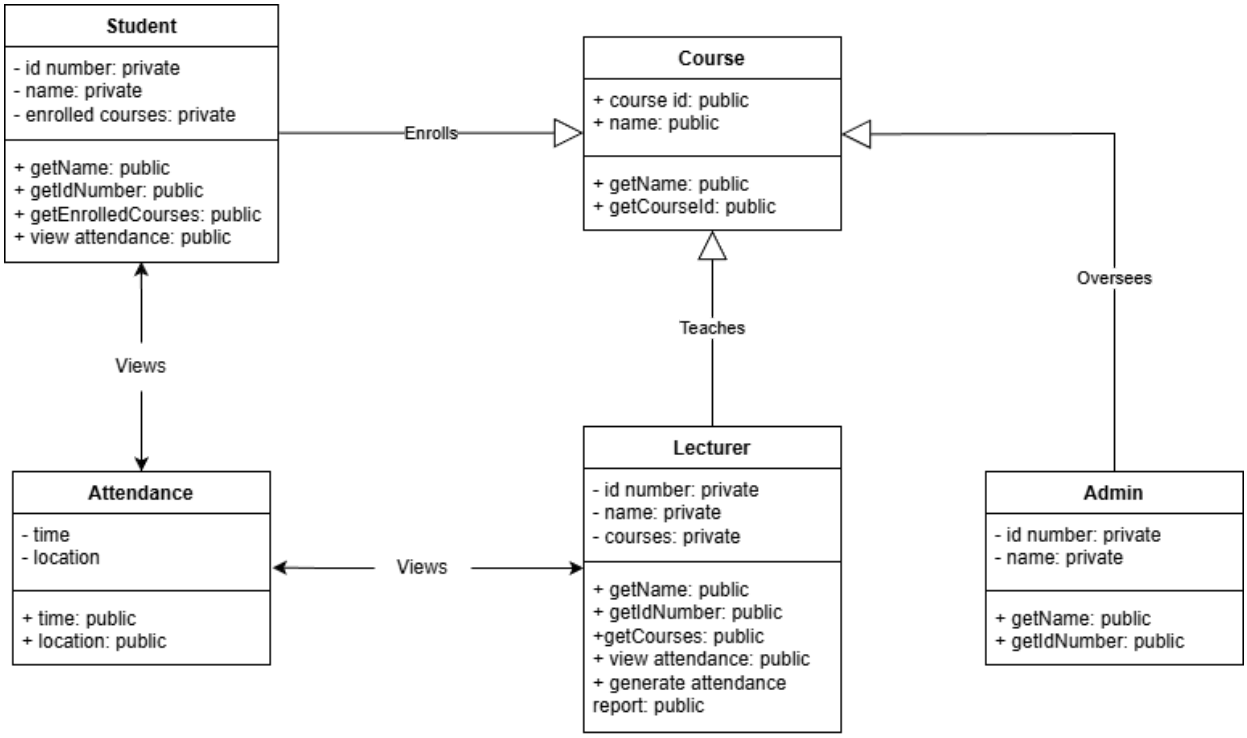
3.0 Architecture Decomposition

3.1 Component Decomposition – Classes

Class Name	**Architecture Component**	**Responsibilities/Methods**	**Related Requirement IDs**
UserUI	Presentation Layer	-View courses and attendance(students) -Mark attendance and view class lists(lecturers)	REQ-UI-01, REQ-UI-03, REQ-UI-05
InputValidator	Presentation Layer	-Validate login inputs, course codes and session dates	REQ-UI-02
AuthService	Application/Service Layer	-Handle login/logout -Track sessions -Manage authorization	REQ-AUTH-O1, REQ-AUTH-02

AttendanceService	Application/Service Layer	-Retrieve and calculate attendance -Validate sign-ins -Send notification	REQ-ATT-01, REQ-ATT-04, REQ-SRV-01
Repositories	Data Access Layer	-AttendanceRepository: Create, read, update and delete attendance records -UserRepository: Manage users (lecturer, student, admin)	REQ-DATA-01, REQ-DATA-02 ,REQ-DATA-03
Database Tables	Database Layer	-AttendanceTable: Store attendance logs -UserTable: Store account details and roles	REQ-DB-01, REQ-DB02

3.2 Structural Design – <Class Diagram / Structure Chart not both>



3.2.1 Design Notes

The relationships primarily follow a **"HAS-A"** structure, meaning one item holds a reference to another. For example, a Student "**HAS A**" link to many Courses they are enrolled in, and vice-versa (Many-to-Many enrollment). Similarly, one Lecturer "**HAS A**" relationship to multiple Courses, but each Course is tied to only one Lecturer for accountability. The Admin follows the same rule: one admin oversees many Courses. The Attendance class is just a record that exists only to connect a specific Student to a specific Course. All personal data are private, while the actions that users need to perform are public.

Assumptions

1. Student and Lecturer roles are distinct; their behaviors and access are role-specific.
2. Admin is a separate class responsible for system management rather than day-to-day attendance or course activities.

3. Course is central; both students and lecturers are linked through it.

4. Attendance is simplified to only track student-course-session relationships.

Constraints

1. Many-to-many relationship between Students and Courses requires an underlying attendance or enrollment linking mechanism.

2. Admin manages everything but does not participate in courses or attendance marking directly.

3. The diagram focuses on core system functionality; additional features like notifications or input validation are omitted.

Why Relationships Were Specified

1. Student ↔ Course ↔ Lecturer

    ➔ Represents real-world LMS structure: lecturers teach courses, students enroll and attendance is recorded per course.

2. Attendance linked to Student and Course

    ➔ Ensures all attendance data is traceable to both the student and the course.

3. Admin linked to all classes

    ➔ Shows the management/control role in the system, aligning with the administrative layer of Trackademia.

Test Code

GROUP 5: Trackademia

Tajaun Tomlinson	**620163325**	**tajaun.tomlinson@mymona.uwi.edu**
Davaun Glave	**620163313**	**davaun.glave@mymona.uwi.edu**
Seantay Johnson	**620155318**	**seantay.johnson@mymona.uwi.edu**
Rory Williams	**620165266**	**rory.williams@mymona.uwi.edu**
Ricardo Foster	**620168530**	**ricardo.foster@mymona.uwi.edu**
Solay Wellington	**620170530**	**solay.wellington@mymona.uwi.edu**

Course Instructor: **Dr. Ricardo Anderson**

Course: **COMP2140 – Introduction to Software Engineering**

Studio Facilitator: **Mr. Eyton Ferguson**

Date: **03/12/2025**

Test Case	Test Data	Associated Requirement	Expected Result	Actual Result	Pass/Fail	Comment
TC1: Validate device is inside lecture room range and authorized network IP	Device GPS: *18.0050, -76.7500* (inside radius)	FR-1.1	Attendance allowed	Attendance is allowed with message “Attendance marked successfully!”	Pass	
TC2: Reject device outside lecture room range	Device GPS: *18.0090, -76.7600* (outside radius)	FR-1.1	Attendance blocked with message	Attendance blocked with message “Cannot mark attendance: Too far from lecture room”	Pass	
TC3: Validate network IP is authorized	Network IP: *172.16.10.5* (valid campus IP)	FR-1.2	Attendance allowed	Attendance allowed	Pass	

Test Case	Test Data	Associated Requirement	Expected Result	Actual Result	Pass/Fail	Comment
TC4: Reject unauthorized IP	Network IP: *192.168.1.15* (home WiFi)	FR-1.2	Attendance blocked with message	Attendance blocked with message “Cannot mark attendance: Not on campus network”	Pass	
TC5: Display attendance for enrolled courses	Student ID: *620163325*	FR-2.1	System displays list of all enrolled courses + attendance percentage	0% - 100%, status poor - good	Pass	
TC6: Show message when no attendance exists	Student ID with empty record	FR-2.1	“No attendance data available at this time”	Total sessions = 0, sessions attended = 0, attendance percentage = 0%, status poor	Pass	Data is well integrated into a table
TC7: Prevent viewing another student’s data	Student A tries accessing Student B data	FR-2 Security	Access Denied	Student cannot access students data without login credentials	Pass	Provide medium level security
TC8: Send session reminder 15 minutes before class	Session start 10:00 AM	FR-3.1	Notification delivered at 9:45 AM	Reminder not sent	Failed	
TC9: Alert student of low attendance (<75%)	Attendance = 68%	FR-3.3	“Your attendance is below 75%” alert sent	Alert not sent	Failed	
TC10: Notify lecturer when session becomes active	Student marks attendance	FR-3.2	“Attendance Marked” notification within 5 sec	Alert not sent	Failed	
TC12: Upload valid student record	Name: *John Brown* ID: *620111111 , Major:* Information Technology	FR-4.1	Student enrolled to course list	Student enrolled to specific lecturer course	Pass	It does this by utilizing a table and add button
TC13: Reject upload with missing fields	Name: “ ”, ID: *620111111*	FR-4.2	Error: “Name and ID required”	Uploads any data enter	Failed	
TC14: Prevent duplicate student ID entries	Existing student ID already in list	FR-4.3	Error: “Student already enrolled”	Adds student Either way	Failed	
TC15: Export Excel summary for valid course	Course: COMP2190 Date Range: Full Semester	FR-5.1	Excel file downloaded successfully with full data	Exports as cvs file	Failed	
TC16: Filter export by date range	Course: COMP2190 Week 1–4	FR-5.3	File shows only attendance for selected range	Does not show export data range	failed	
TC17: Display error when no attendance exists	Course with no recorded sessions	FR-5.3	Message: “No attendance data available for selected period”	Displays that no attendances have been for a particular session	Pass	
TC18: Record timestamp after successful sign-in	Student signs in at 10:03:12 AM	FR-6.1	Timestamp saved and displayed (10:03:12 AM)	Stores time and date of all activity	Pass	It displays this in the bash

Test Case	Test Data	Associated Requirement	Expected Result	Actual Result	Pass/Fail	Comment
TC19: Timestamp must match server time within 1 second	Server time: 10:05:00	FR-6.1	Logged timestamp: 10:05:00–10:05:01	Matches server time	Pass	Can be observed in bash

https://github.com/Tajilinson/trackedamia