DATABASE SYSTEMS

DETAILED REQUIREMENT DOCUMENT

Session: 2024-2028

Submitted By:

2024-CS-144

2024-CS-154

2024-CS-160

Supervised By:

SIR SAMYAN QAYYUM WAHLA

Department of Computer Science

University Of Engineering and Technology, Lahore

Attendance Management System

Description

The Attendance Management System facilitates and organizes all schools, colleges, and training centers to keep track of student attendance. This system offers two types of users here namely Admin and Student. Admin including teachers and professors, who is able to administer the whole system, has powers such as adding students, adding courses with allots, and credits allocated for every course and holidays marked. Each course has a name, code, and credit hours attached to it associated with the students and the attendance records. Student is now able to access the system via login and see his/her attendance, courses in which the student is enrolled, and attendance percentage of each course. A student may also be able to check future holidays and be updated on the schedule of classes. The teacher or admin signs daily attendance by choosing a date, course, and marking each student present, absent, or late. Absence in a holiday would mean automatically skipping the attendance for a particular day. The entries for attendance are exactly secure and filtered out by student, course, or date for easy access. Reports are also generated to indicate the count of classes attended by a student in every course for both the teachers and the students to monitor progress. Thus, it ensures that attendance will not be taken on holidays, and the attendance for each course will be associated with the respective credit hours. It also gives alerts to the students whose attendance is short. In fact, this attendance management system is user friendly, easy to create, and deployable into schools, colleges, or coaching institutions. Time saving, manual error minimalization, and easier processes in dealing with managing attendance and course-related information. In all, this system helps maintain proper academic records as well as help the students realize their responsibility while relieving the workload of teachers and admins.

Use cases

The Use Case Names for the Attendance Management System are as follows:

1. Mark Attendance
2. Generate Attendance Reports
3. Login Authentication
4. Password Reset
5. Student Login
6. Admin Privileges
7. Professor Privileges
8. CRUD for Students
9. CRUD for Professors
10. CRUD for Departments
11. CRUD for Courses
12. Assign Professors to Courses
13. Assign Students to Courses
14. Set Attendance Date and Time

15. Edit Attendance
16. View Attendance Records
17. Filter Attendance by (i.e. by Course, Date, or Student)
18. View Student Attendance Summary
19. Export Attendance Data (PDF/Excel)
20. Notification or Alert (i.e. Low Attendance)
21. Search (i.e. by Course, or Student)

Use Case: Mark Attendance

Level: User Goal

Primary Actor: Professor

Preconditions

- Professor is logged in with valid credentials.
- The professor has at least one course assigned.
- Students are enrolled in the selected course and section.
- System has access to an active internet connection and database.

Success Guarantees

- Attendance for each student is successfully saved to the database
- Contact hours for the course and students are updated.
- Credit hour tracking is validated or recalculated.
- No duplicate, missing, or unauthorized attendance entries are allowed.

Basic Flow

1. Professor logs in and opens the AMS desktop application.

2. System loads the dashboard based on the professor's role.

3. Professor navigates to the "Mark Attendance" module.

- **Problem:** Module fails to load.
  Solution: Display: "Unable to load the attendance module. Please check your connection or try again." System retries or prompts user to contact admin.

4. System shows all assigned courses and sections.

- **Problem:** Course list is empty.
  Solution: Display: "No courses assigned. Please contact the administrator."

5. Ensure course-professor assignments exist.

6. Professor selects a course and section for the current day.

7. System checks course schedule and confirms it's valid for today.

- **Problem:** Selected date is outside schedule.
  Solution: Display "Attendance cannot be marked today for this course."

8. System loads list of students enrolled in the selected course/section.

- **Problem:** List fails to load due to internet or database error.
  Solution: Display "Student data could not be loaded. Please check your internet or contact the admin." Provide Retry button.

9. Professor marks each student as Present, Absent, or Leave. Professor adds remarks (leave, medical reason). Remark field is linked to each student and saved with the record.

- **Problem:** Some students are left unmarked.
  Solution: Display "Some students are not marked. Please complete attendance for all before submitting." Disable Submit until resolved.

10. Professor clicks “Submit Attendance.”

- **Problem:** Attendance already submitted for this course and date.
  Solution: Display "Attendance for this course on this date is already recorded." and block duplicate entries.

11. System validates attendance records, then attempts to save to the database.

- **Problem:** Save operation fails (e.g., timeout, database error).
  Solution: Attempt automatic retries up to three times. If it fails, it displays "Failed to save attendance. Please try again or contact the admin."

Upon successful save:

1. Contact hours are updated for each present student.

2. Credit hour thresholds are recalculated if applicable.

3. Confirmation message is shown "Attendance saved successfully."

4. Professor may export or view attendance report before exiting.

Admin Resolution Scenarios:

- Student List Fails to Load:
  Admin checks system internet/database. Admin may restore from backups or use internal tools to reload student data. After resolution, professor refreshes or reopens the module.
- Attendance Fails to Save Even After Retries:
  Admin uses the “Attendance Recovery Panel” or a database interface to insert missed records. Admin grants permission to resubmit if needed.

Use Case: Reports for the Attendance Management System

Level: User Goal

Primary Actor: Professor

Preconditions

- The professor is authorized and recognized as someone who can view attendance data.
- Attendance records exist for the desired date range.

Success Guarantees

- The attendance report is generated successfully.
- The report properly reflects the data on student presence.
- Any errors occurring during the generation of reports are logged.

Basic Flow

1. The professor requests the creation of an attendance report for a specific date range.

   - **Problem**: Start date is after end date.
     Solution: Validate that the start date is before the end date; otherwise, an error message should be issued stating "Start date cannot be after end date."

2. The system validates the given date range.

   - **Problem**: No attendance records exist for the date range requested.
     Solution: Query the database for any records; if none are found, inform the professor "Attendance records do not exist for that date range."

3. The system queries attendance records based on the given parameters.

   - **Problem:** The database is unable to connect.
     Solution: The system enables multiple attempts for connection, if not successful, it logs the error and informs the professor about connectivity issues.

4. The system generates resulting attendance data into reports (PDF, Excel).

   - **Problem:** Fields are incomplete or do not match.
     Solution: Treats all data with a same approach in case of an issue, i.e., a message will be sent to the professor for their action to verify, "Report Data is incomplete; please check attendance records."

5. The instructor is viewing the report preview shown on screen.

   - **Problem:** Alignment issues or incorrect display of the report layout.
     Solution: Check for responsive design and correct markup display. If the problem persists, the instructor will be prompted to regenerate the report with a different layout choice.

6. The instructor clicks ‘Yes’ to download or print the report.

- **Problem:** Download fails as this permission is denied to the instructor.
  Solution: Check permissions before proceeding with the download. If denied, informs the instructor, "You are not authorized to download this report."

7. The system will generate the requested report and initiate the download process.

- **Problem:** Total failure during file generation (i.e., due to insufficient disk space).
  Solution: Verify system resources prior to actual file generation. If file generation fails, the instructor is informed, "File Generation Failed due to Insufficient Storage."

8. The instructor reports confirmation of a successful download of the report.

- **Problem:** No confirmation was received for the instructor.
  Solution: Wrap alerts or fold errors within the notification mechanism. If none show an alert confirming, “Confirm that the report downloaded/printed successfully.”
- **Problem:** The system crashed during report generation.
  Solution: Log the error, show an error message to the user, and give randomness.

9. What actions have been taken by the professors to review the attendance reports in light of any actions or inconsistencies?

- **Problem:** The professor perceives data inconsistencies.
  Solution: Provide a mechanism that will allow us to backtrack any discrepancies to the attendance-tracking system and initiate follow-up actions.

Use Case: Admin Privileges

Level: User Goal

Primary Actor: Admin

Preconditions

- The admin has permission to manage users.
- Database is ready and functional.
- User data (student/professor) is available or said data is ready to be added.

Success Guarantees

- Users have been added, updated or deleted successfully.
- Any errors get logged and appropriately messaged to Admin.

Basic Flow

1. Admin opens the user management function.

- **Problem:** Required fill-in fields (name, email, role) is empty.
  Solution: Display “Please fills in all required fields.”

2. Admin adds a user.

- **Problem:** Duplicate email for uniqueness**.**
  Solution: Show “There is already user with this email saves user data.
- **Problem:** Database connection fails**.**
  Solution: Retrying to save and if this is not success we will show “User could not be saved. Please try again later**.”**

3. Admin assigns a user role.

- **Problem:** No role or invalid selection of role.
  Solution: Using dropdown from lookup show “Please select a valid user role.”

4. Admin filters or searches user list.

- **Problem:** No users matched the search term.
  Solution: Display “No users were found for the selected criteria.”

5. Admin updates user record.

- **Problem:** Record not found or deleted.
  Solution: Can refresh data and show “The user selected does not exist.”

Use Case: Entry of Professors in the Attendance Management System

Level: User Goal

Primary Actor: Admin

Preconditions

- Admin logs in with valid credentials.
- AMS establishes a connection to the database.
- There are pre-populated cases for the departments and the courses.

Success Guarantees

- The Professor is added successfully.
- The validation gets completed.
- Any errors get logged and appropriately messaged to Admin.

Basic Flow

1. Admin accesses add/edit/delete professors from the Admin Dashboard.

- **Problem**: The page did not load.
  Solution: Check whether the server or internet connection has issues. The system gives a message that says, "Unable to load Add Professor module. Please try again later."

2. Admin enters the details of the professor (name, ID, email, contact, etc.).

- **Problem:** There are empty fields.
  Solution: Ensure that all fields are filled before submission. It displays the message "All fields are mandatory."
- **Problem:** Invalid email format.
  Solution: Validate the email format. It displays: "Invalid email format."
- **Problem:** Duplicate professor ID.
  Solution: Ensure the professor ID is unique. Displays: "Professor ID already exists. Please use a unique ID."

3: Admin selects a department from the dropdown.

- **Problem:** The List of Departments is empty or fails to load.
  Solution: Make sure departments have been pre-defined in the system. It displays "No departments were found. Please add departments before assigning professors."

4: Admin assigns one or more courses to the profile.

- **Problem:** Courses are not loading
  Solution: Check the course database connection or reload the list.
- **Problem:** The selected course has been assigned to another professor.
  Solution: Restrict assignment or allow multiple professors per course. It displays "This course is already assigned to another professor."

5: Admin creates a UserID and a password for the professor.

- **Problem:** Weak password.
  Solution: Strong password rules to be enforced.
- **Problem:** Duplicate Professor UserID.
  Solution: Ensure a unique UserID. It displays "Professor UserID already exists. Please choose a unique ID."

6: Admin clicks Save and submit to save professors data.

- **Problem:** Database error, or timeout.
  Solution: Retry the operation and check if there are server connection issues. It displays a message saying "Failed to save data. Please try again", this also auto logs the failure and alerts the admin for follow-up.

7: The System acknowledged that the professor has been added and can optionally mail the login credentials.

- **Problem:** Email not sent.
  Solution: Check email settings are properly configured, or else send the credentials manually. It displays the message "Professor added, but could not send email."
- **Problem:** No confirmation message appears in the display.

Solution: Check if there is good feedback for the system shown. It displays the message "Professor added successfully."

Use Case: Department Entry in Attendance Management System

Level: User Goal

Primary Actor: Admin

Precondition

- The admin has to log in with valid credentials.
- The Attendance Management System is connected to the database.
- No duplicate nurture codes or names should exist in the system.

Success Guarantees

- The department has been added successfully.
- Some validation is done for the input.
- Some errors will be logged, and appropriate messages will appear to the admin.

Basic Flow

1. Admin opens the Add/Edit/Delete Departments section of the Admin Dashboard.

- **Problem:** The page is not loading.
  Solution: Check the internet or server connection. Shows the message "Unable to load the Department module. Please try again later."

2. The admin fills up the required details like Department Name, and Department Code.

- **Problem:** Some of the required fields have not been filled.
  Solution: To submit the form, it is mandatory to fill in all fields which are compulsory. The message reads "All fields are mandatory."
- **Problem:** The department name or code is invalid.
  Solution: Accept valid inputs only. It would display a message "Invalid characters in Department Name or Code."
- **Problem:** The name of the department or its code already exists.
  Solution: Unique Department Name and Code. It displays the message "The department already exists. Please use a unique name and code."

3. Admin saves the department when he submits the form.

- **Problem:** While saving, a database error or timeout occurred.
  Solution: Retry the database connectivity and check for the connection. It displays the message "Failed to save department. Please try again." The system logs an entry for the failure and sends a notification to the admin after several failed attempts.

4. The system will acknowledge the addition of the department.

- **Problem:** No confirmation was displayed.
  Solution: Give feedback that the entry has been made successfully. The message is "Department added successfully."

Use Case: Course Entry in Attendance Management System

Level: User Purpose

Primary Actor: Administrator

Preconditions

- Admin is logged in with legitimate access.
- Admin can manage courses.
- Database is connected and working.
- UI is up with access to a course form for inserting the course.
- All required reference data (e.g., semester) is available.

Success Guarantees

- The course has been inserted into the database.
- The course code is validated to not exist.
- Admin is provided with a message of success.
- Logs do update with added course data (course code, admin id, and timestamp).

Basic Flow

1 Admin logs in and lands to the Attendance Management interface. The admin proceeds to add/update/delete Courses.

- **Problem:** The Add/Update/Delete module fails to run.
  Solution: System displays "Cannot load course management module. Please try again."

2 System loads course management interface. Admin clicks "Add course" and fills in the course form by filling in the Course code, Course Name, Description, Credit Hours, and Semester.

- **Problem:** One or more fields is empty or invalid.
  Solution: Prompt "please fill all mandatory fields with valid values." Indicate the problematic fields.

3 The admin submits the form and the system checks the data.

- **Problem**: Duplicate Course Code.
  Solution: Display "Course code already exists. Please enter a unique code." Do not submit the form.

4 The system attempts to insert the new course record into the database.

- **Problem:** Database issue (e.g. deadlock) prevents record from being inserted into database.
  Solution: Attempt to insert the course in the database up to three times. If it still fails, display "Unable to add course. Please try again or contact support." Log the error.

5 When the course is added, a new course record is now in the database. The UI course list is refreshed. Display a confirmation message to the admin that reads "Course added successfully."

- **Problem:** The Course UI is unable to load.
  Solution: Check Internet, check app logs, try to restart this part of the module you are in, contact system administrator for further instructions on how to resolve issue.
- **Problem:** Required field has been left blank.
  Solution: UI highlights any fields you are missing and prompts the admin to fill in that information. Duplicate Course code is presented. It suggests unique course code of what is already accessed or auto-generates based on rules for naming.
- **Problem:** Unable to insert record into the database.
  Solution: Reattempt the action, or if you still have issues, reaching out to someone in technical support.
- **Problem:** Invalid data types presented (text in the field for credit hours).
  Solution: Validate client side and server side before hitting the database.
- **Problem:** Error message appears as semester list does not load.
  Solution: Check to see if you can load the form again or go out to the lookup table for a list.

Use Case: Add Student

Level: User Goal

Primary Actor: Administrator

Preconditions

- Admin is logged in with valid credentials.
- Admin has access rights to manage student data.
- The database is connected and functional.
- At least one department and course exist in the system.

Success Guarantees

- Student record is saved correctly in the database.
- All required fields are validated and unique.
- The student is successfully assigned to a department and optionally to courses.
- No duplicate entries or incomplete records are saved.

Basic Flow

1. Admin logs in and accesses the AMS dashboard.

2. Admin selects the Add/Edit/Delete Students module.

3. System loads the student management interface.

- **Problem:** Interface fails to load.
  Solution: Display "Unable to load student management module. Please try again." Retry or alert the admin.

4. Admin clicks “Add New Student”.

5. Admin fills in student details:

- **Problem:** Required fields are empty.
  Solution: Highlight empty fields and show: "All fields marked with * are mandatory."
- **Problem:** Invalid formats (e.g., email or phone).
  Solution: Validate input; display: "Invalid email or phone number format."
- **Problem:** Duplicate Student ID or Email.
  Solution: System checks for uniqueness; display: "A student with this ID or email already exists."

6. Admin selects the student’s department from dropdown.

- **Problem:** Department list is empty.
  Solution: Display "No departments found. Please add departments before assigning students."

7. Admin clicks “Save Student”.

8. System validates all fields and checks backend connection.

- **Problem:** Save fails due to server/database issue.
  Solution: Retry up to 3 times. If still fails, it displays "Student could not be saved. Please try again or contact technical support."

9. Allow admin to re-submit without re-entering all data.

On successful save

1. Student record is added to the database.

2. Student is linked to department and (if chosen) courses.

3. Confirmation message shown "Student added successfully."

Admin Resolution Scenarios

- UI or form fails to load.

Admin checks internet or application logs. Restart app or notify system administrator.

- Validation fails or duplicate detected.
  Admin corrects the data and retries submission.

- Missing department or course data:
  Admin first adds department or course using appropriate modules, then returns to add student.

Use Case: Edit Student

Level: User Goal

Primary Actor: Administrator

Preconditions

- Admin is logged in with valid credentials.
- Admin has access rights to manage student data.
- The student to be edited already exists in the system.
- Database is connected and functional.

Success Guarantees

- Student details are updated correctly in the database.
- All fields are validated to ensure no invalid or incomplete data is saved.
- The system prevents duplicate entries or invalid data updates.
- The student's updated information is reflected across the system.

Basic Flow

1. Admin logs in and accesses the AMS dashboard.

2. Admin selects the Add/Edit/Delete Students module.

3. System loads the student management interface.

4. Admin clicks the Edit option next to the student's name.

- **Problem:** Interface fails to load.
  Solution: Display "Unable to load student management module. Please try again." Retry or alert the admin.

5. Admin searches for the student to edit by entering student ID, name, or email.

- **Problem:** No matching student found.
  Solution: Display "No student found matching the provided criteria."

6. System displays the student's existing information (name, contact info, department, courses, etc.).

7. Admin can modify the student's data

- **Problem:** Required fields left empty.
  Solution: Highlight empty fields with a message: "This field is required."
- **Problem:** Invalid format for email or phone number.
  Solution: System validates input and displays: "Invalid email or phone number format."
- **Problem:** Duplicate email or phone number detected.
  Solution: System checks for existing records with the same email or phone. It displays "This email or phone number is already in use."

8. Admin updates the student's department.

- **Problem:** Department list not loading.
  Solution: Display "No departments found. Please add departments before assigning students or retry later"

9. Admin updates student's course(s) (if applicable).

- **Problem:** Course list not loading.
  Solution: Display "Courses not found. Please assign department courses before editing student information or retry later"

10. Admin clicks "Save Changes".

System validates all fields and checks the backend connection.

- **Problem:** Save fails due to server/database issue.
  Solution: Retry again. If still fails. It displays "Changes could not be saved. Please try again later."

On successful save

- Student record is updated in the database.
- System confirms the action with: "Student details updated successfully."
- Admin exits the student management module or returns to the student list.
- System refreshes the student data to reflect the changes.

Admin Resolution Scenarios

- User Interface or form fails to load.
  Admin checks internet or application logs or restart the app
- Validation fails or duplicate detected.
  Admin corrects the data and retries submission.

- Save operation fails consistently.
  Admin contacts the database administrator. Admin may use internal tools to manually update the student record.

- Missing department or course data.
  Admin adds the missing department or course using the appropriate modules, then returns to edit the student.

Use Case: Delete Student

Level: User Goal

Primary Actor: Administrator

Preconditions

- Admin is logged into the system with valid credentials.
- Admin has rights to manage student records.
- The system is connected to the database and it is operational.
- There is at least one student record in the system.

Success Guarantees

- The student record has been removed from (either permanently deleted or soft-deleted) the database successfully.
- All data related to the student (attendance, enrollment, etc.) is deleted, archived, or blocked safely according to dependency rules.
- The user interface is updated immediately to inform the admin that the student record was deleted successfully.
- Admin receives confirmation that the student record is successfully removed.

Basic Flow

1. Admin logs into the AMS and reaches the AMS dashboard.

2. Admin selects the Add/Edit/Delete Students module.

3.The system loads the student management user interface.

- **Problem:** The user interface is not populated with the student management user interface.
  Solution: Provide display: "Unable to load student management module. Please try again."

4. Admin selects a student from the list of students to delete.

- **Problem:** Admin does not select a student record to delete.

Solution: Generate prompt: "Please select a student record to delete."The system generates a confirmation dialogue: "Are you sure you want to delete this student?" Admin confirms that the student record will be deleted.

5.System attempts to remove the student record from the database.

- **Problem:** A server/database error is preventing the deletion.
  Solution: Attempt to delete up to. If that fails, display: "There was an issue deleting the student. Please try again later or reach out to technical support. "

6. After the system has successfully deleted the record:

7. The student's record will be completely removed from the database.

8. The existing UI student list will be refreshed. "A student was deleted successfully" confirmed.

Administrator Resolution Scenarios

- UI or form won't load.
  Check the network/application logs, retry delete, or contact the system administrator.

- Student wasn't selected.
  Print a message to the admin, directing them to select a record before deletion.

- Data dependency prevents deletion.
  Admin can check the linked student data (i.e. search for attendance, grades, etc.) and resolve, or use a "soft-delete" change database can't delete the student record. Retry, or if that is still the issue, contact the database administrator.