ADMIN PRD

✅ v1.0 – Login & School Profile (unique to the School)

🔹 1. Feature: User Login (Authentication)

✅ **Acceptance Criteria:**

- User must be able to log in using a valid email and password.
- Invalid login attempts should return clear, user-friendly error messages (e.g., “Incorrect email or password”).
- Only one role exists initially: **Admin**.
- Upon successful login, user is redirected to the dashboard or school profile page.
- User session persists until explicit logout.
- Passwords are securely hashed and stored.
- Basic validation on inputs (e.g., no empty fields, proper email format).

🧑‍💻 **User Story:**
As an admin,
I want to log into the system securely
So that I can access and manage my school settings.

🔹 2. Feature: View & Update School Profile

✅ **Acceptance Criteria:**

- Admin can view current school profile details, including:
    - School Name
    - Address
    - Email and Phone contact(s)
    - School Logo (optional)
- Admin can update any of these fields via a form.
- If no data exists initially, the form should allow fresh entry of school profile info.
- Validation ensures required fields (school name, at least one contact method) are filled before saving.
- On successful update, a confirmation message is displayed (e.g., “Profile updated successfully”).
- Data is stored in a table, allowing only one record (enforce single school per system at this stage).
- Changes are reflected immediately on the dashboard or relevant UI.

👩‍💼 **User Story:**
As an admin,
I want to view and update my school’s profile information
So that my dashboard reflects the correct and current school data.

✅ v1.1 – Academic Session & Term Management

🔹 1. Feature: Academic Session Setup

✅ Acceptance Criteria:

- Admin can create new academic sessions (e.g., “2024/2025”).
- Sessions have a start date and end date.
- Sessions must not overlap with existing sessions (validation required).
- Admin can view a list of all academic sessions.
- Admin can update or delete existing sessions (with constraints: e.g., cannot delete sessions with linked terms).
- Session data is saved in a `sessions` table with unique session names.
- Clear success/failure messages on create, update, and delete operations.

👩‍💼 User Story:
As an admin,
I want to create and manage academic sessions
So that the school’s academic years are correctly defined and organized.

🔹 2. Feature: Academic Term Setup (Linked to Sessions)

✅ Acceptance Criteria:

- Admin can create academic terms (e.g., 1st Term, 2nd Term, 3rd Term) linked to a specific session.
- Term has a name, start date, and end date.
- Terms must not overlap within the same session (validation required).
- Admin can view a list of all terms per session.
- Admin can update or delete terms (with constraints: cannot delete terms linked to students or results).
- Term data is saved in a `terms` table, linked via foreign key to the `sessions` table.

- Display clear success/failure messages on all operations.

👩‍💼 **User Story:**
As an admin,
I want to create and manage academic terms within sessions
So that the academic calendar is correctly structured for each academic year.

✅ v1.2 – Class & Arm Setup

🔹 1. Feature: Class Creation

✅ **Acceptance Criteria:**

- Admin can create and name classes (e.g., JSS1, SS2).
- Admin can view a list of all created classes.
- Admin can update or delete classes (deletion restricted if linked to arms or students).
- Validation to prevent duplicate class names in a school(unique to the school).
- Data stored in a `classes` table.
- Success/failure messages displayed on operations.

👩‍💼 **User Story:**
As an admin,
I want to create and manage classes
So that the school's academic levels are clearly defined.

🔹 2. Feature: Class Arm Creation

✅ **Acceptance Criteria:**

- Admin can create arms/sections for each class (e.g., A, B, C).
- Arms must be linked to a valid class.
- Admin can view a list of arms under each class.
- Admin can update or delete arms (deletion restricted if linked to students).
- Validation to prevent duplicate arms within the same class in a school(unique to the school).
- Clear success/failure feedback on CRUD actions.

🧑‍💼 **User Story:**
As an admin,
I want to create arms for each class
So that students can be grouped into manageable class sections.

🔹 3. Feature: Class Arm Section (Optional)

✅ **Acceptance Criteria:**

- Admin can assign a color or label (e.g., Pink, Blue, White) to each class arm for easier identification.
- Colors/labels appear in listings and dropdowns.
- Can update or remove color/label.
- Color data stored as an optional attribute in `class_arms`.
- Admin can view a list of section under each class arm.
- Admin can update or delete section (deletion restricted if linked to students).
- Validation to prevent duplicate section within the same class).
- Clear success/failure feedback on CRUD actions.
- Changes trigger confirmation messages.

🧑‍💼 **User Story:**
As an admin,
I want to label class arms with colors or sections
So that visual distinction among arms is easy for staff and students.

✅ v1.3 – Parent Management (Basic CRUD)

🔹 1. Feature: Parent Registration (Add New Parent)

✅ **Acceptance Criteria:**

- Admin can add new parent records with fields such as:
    - Full Name
    - Email (optional)
    - Phone Number (required)
    - Address (optional)
    - Occupation (optional)
- Validation ensures required fields are filled (e.g., phone number).
- Email and phone number must be unique across parents.
- On successful creation, parent is added to the system and visible in the list.
- User-friendly error messages for validation or duplication issues.

🧑‍💼 **User Story:**
As an admin,
I want to add new parents into the system
So that students can be linked to their respective parents.

🔹 2. Feature: View Parent List

✅ **Acceptance Criteria:**

- Admin can view a list of all registered parents.
- List displays key info: Name, Phone, Email, Number of Linked Students.
- Search and filter options available (e.g., by name or phone).
- Clicking a parent's name opens detailed parent profile.

🧑‍💼 **User Story:**
As an admin,
I want to view and search parents
So that I can quickly find a parent's details when needed.

🔹 3. Feature: Edit Parent Information

✅ **Acceptance Criteria:**

- Admin can update parent details
- Validation runs on update (e.g., unique email/phone).
- Confirmation message shown on successful update.

🧑‍💼 **User Story:**
As an admin,
I want to edit parent contact and personal information
So that parent records remain accurate and up to date.

🔹 4. Feature: Delete Parent

✅ **Acceptance Criteria:**

- Admin can delete a parent record if no students are linked to that parent.

- If linked students exist, deletion is blocked with an explanatory error message.
- Successful deletion removes the parent from listings.

🧑‍💼 **User Story:**
As an admin,
I want to delete parent records
So that outdated or incorrect records can be removed safely.

✅ v1.4 – Student Management (Link to Parent, Class,class arm,class Session)

🔹 1. Feature: Student Registration

✅ **Acceptance Criteria:**

- Admin can register a new student with the following mandatory fields:
  - Admission Number(auto generate using the section/No EG 2024/2025/001
  - Full Name
  - Date of Birth
  - Gender
  - Assigned Parent (select from existing parents)
  - Assigned Class
  - Assigned Class Arm
  - Assigned Class section (optional)
  - Academic Session (e.g., 2024/2025)
- Optional fields:, Address, Medical Info, Photo.

- Validation to ensure all required fields are completed before submission.
- Dropdown and searchable select inputs for Parent, Class, Class Arm, and Session.
- On successful registration, student appears in student listings and can be linked to results, attendance, etc.
- Duplicate student checks based on name and DOB with warning prompts (optional).

🧑‍💼 **User Story:**
As an admin,
I want to register students into the system with links to their parent and academic class info
So that student records are accurately captured for academic and administrative tracking.

🔹 2. Feature: View Student List & Search

✅ **Acceptance Criteria:**

- Admin can view a paginated, sortable list of students, filtered by session,class,class-arm and class-arm-section(optional).
- Display key info: Name, Class, Arm, Parent Name, Session.
- Search functionality by student name, admission number, class, or parent.
- Clicking on the view button, the student detailed profile page is opened.

🧑‍💼 **User Story:**
As an admin,
I want to view and search through the student list
So that I can quickly find student records and verify details.

🔹 3. Feature: Update Student Information

✅ **Acceptance Criteria:**

- Admin can edit any student’s details via a form.

- Validation on all editable fields, especially class and parent associations.
- Changes saved immediately and reflected across all linked modules (results, attendance).
- Confirmation message after successful update.

👩‍💼 **User Story:**
As an admin,
I want to update student records as needed
So that all information remains current and accurate.

🔹 4. Feature: Link Student to Parent, Class, Session

✅ **Acceptance Criteria:**

- Students must be linked to a single Parent record at registration.
- Students must belong to exactly one Class and Class Arm and class section (optional) per academic session.
- Students must be enrolled in a specific Academic Session.

👩‍💼 **User Story:**
As an admin,
I want to ensure each student is correctly linked to their parent, class arm, and session
So that reporting and academic tracking are consistent and reliable.

🔹 5. Feature: Delete Student Record

✅ **Acceptance Criteria:**

- Admin can delete student records if there are no dependent records (results, attendance, fee payments).
- Deletion blocked with explanatory error if dependencies exist.
- Confirmation prompt before deleting to avoid accidental removals.

🧑‍💼 **User Story:**
As an admin,
I want to delete student records when necessary
So that the database only contains active and valid student data.

Note: And these must be done in one controller @ api/v1 and must be unique to one school(logged in school)

✅ v1.5 – Staff & Teacher Management (Basic CRUD)

🔹 1. Feature: Staff/Teacher Profile Creation

✅ **Acceptance Criteria:**

- Admin can add new staff or teacher profiles with the following mandatory fields:
    - Full Name
    - Email (unique)
    - Phone Number
    - Role e.g accountant , teacher for frontend (dropdown)
    - Gender
- Optional fields: Address, Qualifications, Employment Start Date, Photo.
- Password setup or invite email for account creation
- Form validation to ensure required fields are completed and email uniqueness.
- On success, the staff profile is saved and visible in staff listings.
- Details should also be saved at user table

🧑‍💼 **User Story:**
As an admin,
I want to create staff and teacher profiles
So that I can manage school personnel and assign them roles.

🔹 2. Feature: View Staff/Teacher List & Search

✅ **Acceptance Criteria:**

- Admin can view a paginated list of staff/teachers.
- Display key info: Name, Role/Position, Email, Phone.
- Search and filter by name, role, or status (active/inactive).
- Ability to click on a profile to view detailed staff information.

🧑‍💼 **User Story:**
As an admin,
I want to browse and search the list of staff and teachers
So that I can easily find and review personnel details.

🔹 3. Feature: Update Staff/Teacher Profiles

✅ **Acceptance Criteria:**

- Admin can edit staff/teacher information.
- Validation on updated fields, including email uniqueness and valid phone format.
- Changes immediately reflected in all related modules.
- Confirmation message after successful update.

🧑‍💼 **User Story:**
As an admin,
I want to update staff or teacher details when necessary
So that all personnel records stay current and accurate.

🔹 4. Feature: Assign Roles to Staff/Teachers

✅ **Acceptance Criteria:**

- Staff must have a defined role or position upon creation or edit.
- Staff number should be auto generated session/No eg (2022/2023/001)
- Backend enforcement to prevent invalid role assignments.
- UI should display role clearly in listings and profiles.

🧑‍💼 **User Story:**
As an admin,
I want to assign clear roles to each staff member
So that their responsibilities and access can be managed effectively later.

🔹 5. Feature: Delete Staff/Teacher Profiles

✅ **Acceptance Criteria:**

- Admin can delete a staff or teacher profile only if there are no active dependencies (e.g., assigned classes, subjects).
- Confirmation dialog to prevent accidental deletion.
- Deletion blocked with clear error if dependencies exist.

🧑‍💼 **User Story:**
As an admin,
I want to safely delete staff profiles when appropriate
So that outdated or incorrect personnel data can be removed.

Note: And these must be done in one controller @ api/v1 and must be unique to one school(logged in school)

✅ v1.6 – Subjects

🔹 1. Feature: Subject Management (CRUD)

✅ **Acceptance Criteria:**

- Admin can create new subjects with fields such as:
    - Subject Name (e.g., Mathematics, Biology)
    - Subject Code (optional, unique eg. BIO)
    - Description (optional)
- Admin can view a list of all subjects.
- Admin can update subject details.
- Admin can delete subjects only if no assignments or results depend on them.
- Validation for required fields (Subject Name) and unique subject code if provided.
- Confirmation messages on create, update, and delete actions.

👩‍💼 **User Story:**
As an admin,
I want to manage the list of subjects offered at the school
So that the academic curriculum is well defined and maintained.

🔹 v1.7. Feature: Assign Subjects to Classes

✅ **Acceptance Criteria:**

- Admin can assign one or more subjects to a specific class arm and class arm section (e.g., JSS1 A SCIENCE,).
- Admin can view and modify subject assignments per class.

- Validation to ensure only valid subjects and classes are assigned.
- Changes should be reflected immediately in the class academic setup.

🧑‍💼 **User Story:**
As an admin,
I want to assign subjects to classes
So that each class has the correct curriculum subjects set.

🔹 3. Feature: Assign Teachers to Subjects per Class

✅ **Acceptance Criteria:**

- Admin can assign teachers to teach specific subjects for a class arm or class section (e.g., Teacher A teaches Mathematics in JSS1 Arm A section science).
- List existing assignments and allow updates or removals.
- Validation to ensure only valid staff with teacher roles can be assigned.
- UI clearly shows assigned teachers per subject/class arm or class arm section

🧑‍💼 **User Story:**
As an admin,
I want to assign teachers to subjects for specific classes and arms
So that teaching responsibilities are clearly defined and tracked.

🔹 4. Feature: Subject & Assignment Search and Filters

✅ **Acceptance Criteria:**

- Ability to search subjects by name or code.
- Ability to filter subject assignments by class or teacher.
- Pagination for large data sets.
- Search and filter UI should be responsive and intuitive.

🧑‍💼 **User Story:**
As an admin,
I want to search and filter subjects and their assignments
So that I can easily find and manage subject-teacher-class relationships.

Note: 1.And these must be done in one controller @ api/v1 and must be unique to one school(logged in school)

---

v1.8.1 – Assessment Components Management

---

1. Feature: Create Assessment Components

- **User Story**: As an admin, I want to create and manage assessment components for each subject and term so that I can accurately define how students are evaluated.
- **Acceptance Criteria**:
    - Input:
        - **Name** (e.g., “CA1”, “Mid-Term Exam”, “CA2”, “Exam”)
        - **Weight** (numeric, percentage of final grade)
        - **Order** (integer, order of the component)
        - **Label** (optional description)
        - **school_id**, **session_id**, **term_id** (required associations)
    - Validation: Ensure the weight is positive, and name is not empty.
    - Response:
        - Success: Returns a message confirming the assessment component creation.

        - Failure: Returns an error message with the validation issues.

---

2. Feature: View Assessment Components

- **User Story**: As an admin, I want to view a list of all assessment components for each term and subject so that I can track how students are evaluated.
- **Acceptance Criteria**:
    - Response:
        - Returns a list of assessment components with details like `name`, `weight`, `order`, and `term/subject association`.
    - **Database**: Fetch from the `assessment_components` table.

---

3. Feature: Update Assessment Components

- **User Story**: As an admin, I want to update existing assessment components so that I can adjust the evaluation methods as needed (e.g., change weight or labels).
- **Acceptance Criteria**:
    - Input:
        - **Name** (optional, for updating)
        - **Weight** (optional, for updating)
        - **Order** (optional, for updating)
        - **Label** (optional)
    - Validation: Ensure the weight is positive and name is unique per session and term.

- Response:
    - Success: Returns a message confirming the assessment component update.
    - Failure: Returns an error message with the validation issues.

---

4. Feature: Delete Assessment Components

- **User Story**: As an admin, I want to delete assessment components that are no longer needed so that I can maintain accurate and up-to-date evaluation criteria.
- **Acceptance Criteria**:
    - Validation:
        - Prevent deletion if there are linked student results.
        - If linked, return an error message.
    - Response:
        - Success: Returns a message confirming the deletion.
        - Failure: Returns an error message if dependencies exist.

---

5. Feature: Manage Assessment Component Dependencies

- **User Story**: As an admin, I want to ensure that assessment components cannot be deleted or modified if they have existing dependencies (e.g., linked student results) to avoid data inconsistencies.
- **Acceptance Criteria**:
    - Before deletion or modification, the backend must check the `results` table for any linked records.
    - If dependencies exist, return a failure message explaining the issue.

Versioning Considerations

- **Assessment Components** should be implemented under **v1.8.1** to track the various components of assessment (e.g., CA1, CA2, Exam) and manage them for each subject, term, and session.

✅ v1.9 – Results & Assessment

🔹 1. Feature: Input Subject Scores per Student

✅ **Acceptance Criteria:**

- Teachers/Admins can input scores for each student in each subject for a specific term and session.
- Scores are broken down by assessment components (e.g., CA1, CA2, Exam).
- Input validation ensures scores are within allowed ranges (e.g., 0 to 100).
- Ability to save partial entries and update scores later before final submission.
- Confirmation message upon successful save or update.
- Prevent duplicate score entries for the same student-subject-term-session combination.

👩‍🏫 **User Story:**
As a teacher,
I want to input assessment scores for my students per subject
So that I can accurately record and track their academic performance.

🔹 2. Feature: Input Teacher Comments/Remarks on Results

✅ **Acceptance Criteria:**

- Teachers can add comments or remarks per student per subject per term.
- Comments should be optional but editable until results are finalized.
- Comments are visible to parents and staff on the results page.
- Input length and content validation (e.g., max characters).
- Confirmation of saved comments.

🧑‍💼 **User Story:**
As a teacher,
I want to provide comments or feedback on my students' results
So that parents and staff understand each student's performance better.

🔹 3. Feature: View Results Summary

✅ **Acceptance Criteria:**

- Users (teachers, parents, and admins) can view summarized results for a student per term/session.
- Summary includes total scores, average scores, grades, position in class, and teacher remarks.
- Display Lowest In Class (LIC), Highest In Class (HIC), and Class Average for each subject.
- Responsive UI that shows results clearly and neatly.
- Ability to print or export the results summary.

🧑‍💼 **User Story:**
As a parent or teacher,
I want to view a summary of a student's results
So that I can easily understand the student's academic standing for the term.

🔹 4. Feature: Results Validation & Integrity

✅ **Acceptance Criteria:**

- Backend checks to prevent invalid or duplicate entries.
- Validation rules for each assessment component's weight and total score calculation.
- System calculates total score per subject based on weighted assessment components.
- Automatic assignment of grades based on the grading scale.
- Error messages if validation fails, with guidance for correction.

🧑‍💻 **User Story:**
As a system,
I want to validate all result inputs and calculations
So that the academic results are accurate and reliable.

🔹 5. Feature: Result Locking & Finalization (Optional for v1.7.x)

✅ **Acceptance Criteria:**

- Ability for admin/teacher to lock results for a term/session to prevent further edits.
- Locked results cannot be changed unless unlocked by authorized personnel.
- Notification when trying to edit locked results.

🧑‍💻 **User Story:**
As an admin,
I want to lock finalized results
So that no accidental changes can be made after official publication.

✅ v2.0 – Skills/behavior

🔹 1. Feature: Record Skills/Behavior per Student

✅ **Acceptance Criteria:**

- Admins and teachers can record non-academic skills and behavior traits (e.g., neatness, punctuality, respect) for each student.
- Skills are associated with a specific term and session.
- Input fields allow rating or qualitative remarks (e.g., numeric scale 1–5 or descriptive text).
- Ability to update skill records until the end of the term.
- Display skill ratings on the student's profile and results summary.

🧑‍💼 **User Story:**
As a teacher,
I want to track and record student skills and behaviors
So that the student's holistic development is monitored beyond academics.

🔹 2. Feature: Generate Result PINs for Students

✅ **Acceptance Criteria:**

- System can generate unique, secure PIN codes for each student's results.
- PINs are linked to a specific term/session and student.
- PINs can be regenerated or invalidated by admin.
- PINs are displayed to parents and authorized users for secure result access.
- Validation ensures no duplicate PINs exist.
- PIN generation can be bulk or individual.

🧑‍💼 **User Story:**
As an admin,
I want to generate unique PINs for student results
So that parents can securely access their child's academic results online.

🔹 3. Feature: Export/print Results of a class

✅ **Acceptance Criteria:**

- Users can export class results
- PDFs are properly formatted and printable.
- Option to export individual student results or bulk export for classes.

🧑‍💻 **User Story:**
As a parent or teacher,
I want to download or print class results as PDF
So that I have an official and portable copy of academic performance.

🔹 4. Feature: User Interface for Skills, PIN, and PDF Export

✅ **Acceptance Criteria:**

- Dedicated pages/forms for recording skills and generating PINs.
- Clear buttons/links to export results as PDFs.
- Responsive and user-friendly UI that guides users through the process.
- Status indicators for PIN generation and PDF export success or failure.

🧑‍💻 **User Story:**
As a user,
I want an easy-to-use interface to manage skills, PINs, and export PDFs
So that I can perform these tasks efficiently and accurately.

.

✅ v2.0 – Student Promotion & Academic Year Rollover

🔹 1. Feature: Student Promotion Logic

✅ Acceptance Criteria:

- Admins can promote students from one class/arm/section to the next for a new academic session.
- Promotion supports bulk processing of multiple students.
- Student's academic session and class/arm details update accordingly after promotion.
- Option to retain or update other student data during promotion (e.g., assigned subjects).
- Validation prevents promotion if target class/session does not exist or is invalid.
- Ability to view promotion history for auditing.

🧑‍💼 User Story:
As a school admin,
I want to promote students to their next class and session at the end of an academic year
So that student academic records reflect their current level accurately.

🔹 2. Feature: Academic Year Rollover (Session & Term Setup)

✅ Acceptance Criteria:

- Admins can roll over academic sessions and terms to create a new academic year.
- Rollover clones or resets relevant academic data to prepare for a new session (e.g., new session and terms).
- Previous academic data remains intact and accessible for historical reporting.

- Ability to set start and end dates for new sessions and terms.
- Validation to prevent overlapping or duplicate sessions during rollover.
- Notification or confirmation prompt before rollover operation.

👩‍💻 **User Story:**
As a school admin,
I want to create a new academic year with new sessions and terms based on the previous year
So that the system is ready for new academic activities without losing past data.

🔹 3. Feature: Bulk Student Promotion UI & API

✅ **Acceptance Criteria:**

- UI allows selecting multiple students for promotion with filters (e.g., by class, session).
- API endpoint supports bulk update of students' class/arm/section and session.
- Clear success or error messages after promotion process.
- Undo or rollback option for accidental promotions (optional but preferred).

👩‍💻 **User Story:**
As an admin,
I want a simple interface to promote many students at once
So that I can efficiently manage academic progression.

🔹 4. Feature: Promotion Reports & Logs

✅ **Acceptance Criteria:**

- System logs each promotion action with user, timestamp, and details.
- Admins can generate reports on student promotion status.
- Reports filterable by session, class.

- Export options for reports (CSV, PDF).

🧑‍💼 **User Story:**
As a school admin,
I want to view and export reports on student promotions
So that I can audit academic progress and share with stakeholders.

✅ v21 – Attendance Management (Student & Staff)

🔹 1. Feature: Student Attendance Tracking

✅ **Acceptance Criteria:**

- Admins and teachers can record daily attendance for students per class and session.
- Attendance status options: Present, Absent, Late, Excused.
- Ability to mark attendance for individual students or bulk mark for a class.
- Attendance records linked to date, class, session, and student.
- Attendance data can be viewed, edited, and deleted by authorized users.
- Attendance reports available for specific date ranges, classes, or individual students.
- Export reports in CSV and PDF formats.
- Validation to prevent duplicate attendance entries for the same student/date.

🧑‍💼 **User Story:**
As a teacher,
I want to record daily attendance for my students
So that I can monitor their presence and participation accurately.

🔹 2. Feature: Staff Attendance Tracking

✅ **Acceptance Criteria:**

- Admins can record and track daily attendance for staff members.
- Attendance status options: Present, Absent, On Leave, Late.
- Attendance data linked to staff member, date, and branch (if multi-branch).
- Ability for staff to view their own attendance records.
- Generate attendance summary reports for staff filtered by date range and department.
- Export reports in CSV and PDF.
- Alerts or notifications for excessive absences or late arrivals (optional feature).

🧑‍💼 **User Story:**
As an admin,
I want to track staff attendance daily
So that I can manage payroll and ensure adequate staffing.

🔹 3. Feature: Attendance User Interface

✅ **Acceptance Criteria:**

- UI to select date, class, and session for student attendance marking.
- Simple and intuitive attendance marking interface (checkboxes, toggles).
- Staff attendance UI accessible only to authorized users with filtering options.
- Attendance summary dashboards for quick overview of student and staff attendance.
- Search and filter options for attendance records.

🧑‍💼 **User Story:**
As a teacher or admin,

I want an easy-to-use interface to mark and review attendance
So that the process is efficient and error-free.

🔹 4. Feature: Attendance Reports & Analytics

✅ **Acceptance Criteria:**

- Generate detailed attendance reports for students and staff over selectable periods.
- Summary views showing attendance trends and absenteeism rates.
- Identify students or staff with poor attendance for follow-up.
- Exportable reports (CSV, PDF) for sharing with school management or parents.

👩‍💼 **User Story:**
As a school admin,
I want to analyze attendance data
So that I can identify attendance issues and improve school operations.

✅ *V22 – Bulk Student Uplodad*

🔹 *1. Feature: Downloadable CSV Template*

✅ ***Acceptance Criteria:***

- *System provides a clearly labeled **"Download Template"** button for bulk student upload.*
- *CSV file includes **well-named, descriptive column headers** to guide the user (e.g. `First Name`, `Last Name`, `Gender`, `Date of Birth`, `Class`, `Session`, `Parent Email`, etc.).*
- *Each column includes inline comments or example rows where supported (e.g. `M/F` for Gender, `YYYY-MM-DD` for Date of Birth).*
- *Template reflects **current system data structures** — e.g., valid class names, sessions, and section IDs.*

- *Any change in database schema automatically updates the CSV template structure to prevent mismatch.*

- *File is downloadable in `.csv` format and readable by spreadsheet tools (Excel, Google Sheets, etc.).*

🧑‍💻 ***User Story:***

As an admin,
I want to download a structured CSV template with predefined columns
So that users can correctly fill in student data for upload without errors.

🔹 ***2. Feature: CSV Upload & Data Validation***

✅ ***Acceptance Criteria:***

- *Users can upload a filled CSV file through a clean, guided interface.*

- *System validates:*

    - ***File type*** *(only `.csv` allowed)*

    - ***Required fields*** *are not empty*

    - ***Field format validation*** *(e.g., date, email, numeric)*

    - ***Duplicate student entries*** *(based on name + admission number or email)*

    - ***Class and session mapping*** *— uploaded values must match existing records.*

- *If any validation errors occur, the system displays:*

    - *The **specific row and column** causing the error.*

    - ***Clear error messages*** *(e.g. “Invalid date format in row 7, column: Date of Birth”).*

- *Uploads that pass validation are staged for confirmation before saving.*

- *Option to **preview parsed data** before final submission.*

🧑‍💻 ***User Story:***

As an admin,
I want to upload student data in bulk using a CSV file
So that I can quickly register many students without manual entry errors.

🔹 ***3. Feature: Bulk Save & Error Handling***

✅ ***Acceptance Criteria:***

- *System performs **atomic upload** — partial uploads do not occur (either all records save successfully or none).*
- *Duplicate or invalid records are automatically skipped and reported.*
- *Successful uploads generate a summary:*
    - *Total records processed*
    - *Successfully inserted count*
    - *Failed record count (with downloadable error log)*
- *Error log downloadable as a `.csv` with original data and error description per row.*
- *Upload progress indicator for large files (optional asynchronous handling).*

🧑‍💻 ***User Story:***

As an admin,
I want bulk uploads to either fully succeed or fail gracefully
So that the database remains consistent and error-free.

🔹 ***5. Feature: User Interface & Experience***

✅ ***Acceptance Criteria:***

- *Intuitive UI with clear steps:*
    1. *Download Template*
    2. *Fill Template*

3. *Upload File*
4. *Preview & Confirm*

- *Upload component supports **drag-and-drop** and **progress feedback**.*
- *Tooltip or on-screen guide explaining each column's purpose.*
- *Clear success/error notifications with color-coded feedback.*
- *Mobile-friendly and responsive layout.*

🧑‍💻 ***User Story:***

As a user,
I want a simple step-by-step interface to upload student data in bulk
So that the process is fast, transparent, and user-friendly.

🔹 ***6. Feature: Integration & Extensibility***

✅ ***Acceptance Criteria:***

- *Bulk upload integrates with **student registration***
- *Uploaded records trigger creation of linked records (e.g., `StudentProfile`, `ParentProfile`, `Enrollment`).*
- *Support for future bulk uploads (staff, subjects, fees) using a shared reusable module.*
- *API endpoint available for automated imports via external systems.*

🧑‍💻 ***User Story:***

As a system developer,
I want a consistent and reusable bulk upload module
So that I can extend the feature to other entities with minimal code duplication.

✅ v23 – Analytics & Academic Dashboards

🔹 v3.0.1 – Analytics & Academic Dashboards

✅ Acceptance Criteria:

- Display school-wide academic stats (average scores per class, subject, term).
- Visual charts (bar, pie, line) showing:
    - Class performance comparison
    - Subject difficulty (lowest/highest average)
    - Number of students with failing/passing grades
- Filter by session, term, class, subject.
- Show total number of students, teachers, and subjects per class.
- Dashboard accessible to Admin only (RBAC controlled).
- Must load data efficiently, even for large datasets.

👩‍💻 User Story:

As an admin,
I want to view academic dashboards with charts and statistics
So that I can analyze student and class performance over time.

✅ v23 – Fee Management unique to each school

🔹 v2.4.1 – Fee Structure Setup

✅ **Acceptance Criteria:**

- Admin can define individual billing items (e.g., Tuition, Uniform, PTA dues).
- Billing items can be grouped by category.
- Each fee structure must be linked to a specific class, session, and term.
- Admin can reuse/copy an existing fee structure for a new term/session.
- Prevent duplicate structures for the same class/session/term combination.
- Validation: amount fields must be positive and required.
- CRUD operations for fee items and structures must be available.
- Data is stored in `fee_items` and `fee_structures` tables.
- There should be a page on the admin dashboard to insert bank details for bank transfer l

👩‍💼 **User Story:**

As a school admin,
I want to define fee structures for each class and term and have a total amount

So that I can generate consistent invoices for students and so that parents can clear their students debt

✅ v2.4 – Role-Based Access Control (RBAC) unique to each school

---

🔹 1. Feature: Define Roles(Role Creation) and permission assignment to role

✅ Acceptance Criteria:

- System allows creation of predefined roles: staffs-Admin, Teachers-manager, accountant , etc
- Roles can be managed (added, edited, removed) by Admins of each school only.
- Each role has a clear definition of responsibilities and access boundaries(permission).
- Roles stored in a roles table with unique role names.

🧑‍💼 **User Story:**
As a Super Admin,
I want to define and manage user roles
So that the system can enforce appropriate permission levels for different users.

🔹 2. Feature: Role Assignment to Users

✅ **Acceptance Criteria:**

- Admins can assign roles to users in the system.

- Users can have one or more roles assigned (start with one role per user for simplicity).
- Role assignment can be updated or revoked by Admins/Super Admins.
- Role assignments are stored and enforced throughout the system unique to each school.

🧑‍💼 **User Story:**
As an Admin,
I want to assign roles to users
So that users have access only to functions relevant to their responsibilities.

🔹 3. Feature: Role Permissions Logic (Backend & Frontend)

✅ **Acceptance Criteria:**

- Permissions are defined per module and action (view, create, edit,show, delete etc).
- Backend APIs check user role before allowing action.
- Frontend UI adapts based on user role, hiding or disabling unauthorized functions.
- Unauthorized access attempts return clear error messages or redirect to an error page.
- Permissions can be extended in the future for granular RBAC.

🧑‍💼 **User Story:**
As a user,
I want the system to restrict access to features based on my role
So that I only see and use functionality I'm authorized for.

---

v25 — Staff (Teachers) dashboard

*implement the **Staff (Teachers) Section** for our multi-school management system.*
Follow the structure and logic exactly as described below.

1. Dashboard Requirements

- *On their dashboard(v25/staff-dashboard), staff (teachers) should see the **classes** they have been assigned to and their corresponding **subjects**.*

2. Authentication

- *Each teacher or staff logs in using their **email** and their **default password**.*
- *They can only log in to the **specific school** they belong to.*

3. System Roles

- *System roles include: **Super Admin**, **School Admin**, **Teacher**, **Accountant**, etc.*
- *Support for **custom roles** created by the **school admin**.*

4. Role-Based Permissions

- *Each role should have **toggle permissions** for **View**, **Edit**, **Delete**, and **others** across system modules such as:*
  Session,Term,Classes,Results,Attendance and all others
- *On the **Role Permissions page**, there should be **easy toggle switches** for each module and action on the frontend.*
- *Permissions should directly control what a user can see or do on the dashboard (e.g., hide Edit/Delete buttons if permission is not granted).*

5. Teacher Access Scope

- *A teacher should only see and manage **students**, **attendance**, and **results** related to their assigned **classes** and **subjects**.*
- *Access to **results** should only be allowed **when permission is given by the admin**.*

6. Teacher Profile Page

*Each teacher should have a **profile page** where they can:*

- *Edit personal details (**name**, **phone number**, **address**, etc.)*
- *Change their **password***

7. Access Control Rules

Every request and page view should respect:

- *The **school** the user belongs to*
- *The **permissions** tied to their **role***
- *The **classes** and **subjects** assigned to them (for teachers)*

3.0 Online Tests (CBT)

✅ v3.1 – Multi-campus(branch) Architecture per school-

🔹 1. Feature: Multi-campus Support (One school-owner(admin), Multiple campus under a school)

✅ Acceptance Criteria:

- System supports managing multiple campuses under a schools from a single user account.
- User dashboard shows a add/list of campus under a schools the user has access to.
- User can switch between campuses under a schools easily within the application.

- campus-specific data (students, staff, classes, results) is isolated per campus under a school.

- Each campus under a school has its own profile and settings.

🧑‍💼 **User Story:**
As a Super Admin or Admin of a school,
I want to manage multiple campuses under a schools from one account
So that I can administer several campuses under a school without logging out or switching accounts.

🔹 4. Feature: User Access per School and campus(Branch)

✅ **Acceptance Criteria:**

- Users have permissions scoped to one or more campuses under a schools .

- On login, users/staffs sees only campuses under a schools they have access to.

- User roles and permissions enforced within each campuses under a school and branch context.

- Switching campuses under a schools or branches updates visible data accordingly.

🧑‍💼 **User Story:**
As a user with multiple campuses under a *school assignments,*
I want to switch between campuses under a *school and branches easily*
So that I can perform my duties in the correct organizational context.

🔹 v2.4.2 – Invoice & Payment Records

✅ **Acceptance Criteria:**

- Generate invoices based on defined fee structures.

- Each student must be linked to an invoice per term/session.

- Allow manual payment recording (cash, bank transfer, etc.).
- Show invoice status: paid, partial, unpaid.
- Track payment history with dates and methods.
- Send invoice summaries to parents via email (optional).
- UI for viewing invoice breakdown per student.
- Tables: `invoices`, `invoice_items`, `payments`.

🧑‍💼 **User Story:**

As a bursar or admin,
I want to generate invoices and track student payments
So that I can manage school finances accurately.

✅ v28 –print result in bulk

✅ v27 –batch result in upload

✅ v29 –s3 integration

✅ v2.5 – Timetable & Scheduling

🔹 v2.5.1 – Timetable Management

✅ **Acceptance Criteria:**

- Admin can define time slots (e.g., Mon 9–10 AM).
- Assign subjects and teachers to time slots per class.
- Ensure no teacher/class/subject conflict in the same time slot.
- Optional view: Weekly grid per class.
- Allow editing of timetables.
- Auto-suggestion for available teachers/time blocks.
- Tables: `timetables`, `time_slots`, `schedule_entries`.

👩‍🏫 **User Story:**

As a school admin or scheduler,
I want to assign teachers and subjects to time slots
So that each class has a structured academic timetable.

✅ v2.6 – Parent-Teacher Communication

🔹 v2.6.1 – In-App Announcements & Secure Messaging

✅ **Acceptance Criteria:**

- Admin/teachers can post announcements (with optional attachments).
- Parents receive notifications for new messages/announcements.
- Direct message functionality between parent and class teacher (or assigned staff).
- Message threading with timestamps.
- Messages are stored securely and only accessible to involved parties.
- Optional: email/SMS fallback if parent is offline.

- Tables: `messages`, `threads`, `announcements`.

🧑‍💼 **User Story:**

As a teacher or admin,
I want to send announcements or messages to parents
So that I can communicate academic and behavioral updates.

✅ v3.1 – Student Deactivation / Freezing

🔹 v3.1.1 – Student Deactivation/Freezing

✅ **Acceptance Criteria:**

- Admin can **deactivate** (temporarily freeze) a student without deleting their data.
- Deactivated students:
    - Do not appear in active class lists.
    - Cannot receive new results, assignments, or be promoted.
    - Are marked visually (e.g., with a status label).
- Deactivation status must be **reversible** (reactivation).
- Add `status` field to `students` table (e.g., `active`, `inactive`, `graduated`, `expelled`).
- Add filters to include/exclude inactive students in listings.
- History of status changes should be recorded.

🧑‍💼 **User Story:**

As an admin,
I want to deactivate a student without deleting them
So that I can temporarily freeze their participation while keeping their records.

✅ v3.3 – Audit Logs & User Activity

🔹 v3.3.1 – Audit Logs & User Activity

✅ Acceptance Criteria:

- Track user activity: login/logout, data changes (CRUD), permissions changes.
- Store logs in a secure, immutable audit table.
- Admins can view logs by:
    - Date range
    - User
    - Action (create, update, delete, etc.)
- Include IP address and device/browser info if available.
- Sensitive data (passwords, tokens) must not be logged.
- Export logs to CSV or PDF.
- Role-based access: Only Admins/Super Admins can view audit logs.

👩‍💻 User Story:

As a Super Admin,
I want to monitor user actions in the system
So that I can ensure accountability and trace data changes.

✅ That completes the **v1.0 to v3.3** user story map.