

ERP Cell Web Application- Complete Workflow Report

Project: ERP Cell Management System

Stack: MERN (MongoDB, Express, React, Node.js)

Summary

This report outlines the complete workflow architecture for the ERP Cell web application, detailing how the user roles (Admin, Faculty, Student) interact with the system through their respective dashboards and modules. The system implements role-based access control (RBAC) with secure JWT authentication, allowing seamless management of academic data including attendance, marks, quizzes, and certificates.

Table of Contents

1. [System Overview](#system-overview)
2. [Architecture & Technology Stack](#architecture--technology-stack)
3. [User Roles & Permissions](#user-roles--permissions)
4. [Authentication Workflow](#authentication-workflow)
5. [Admin Workflow](#admin-workflow)
6. [Faculty Workflow](#faculty-workflow)
7. [Student Workflow](#student-workflow)

System Overview

The ERP Cell application is a centralized platform for managing educational institution operations. It provides three distinct user interfaces based on role-based access control:

- **Admin Role:** User creation, system configuration, data management
- **Faculty Role:** Quiz creation, attendance marking, grade assignment, certificate issuance
- **Student Role:** View personal attendance, marks, quizzes, certificates, and calculated attendance percentage

Key Features:

- Single unified login system
- Role-based access control
- Real-time data synchronization
- Secure password hashing
- JWT token authentication
- Dashboard customization per role

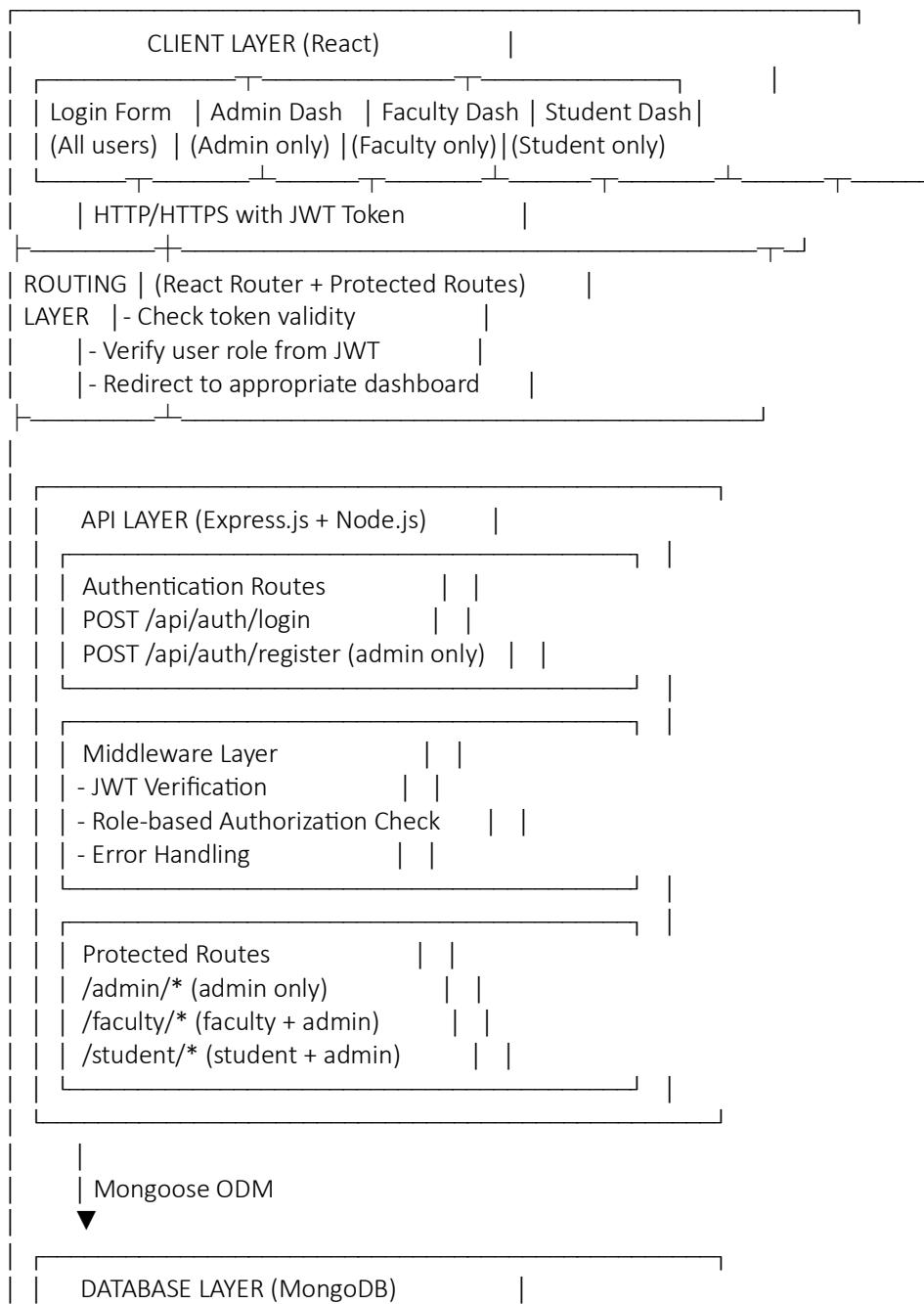
Architecture & Technology Stack

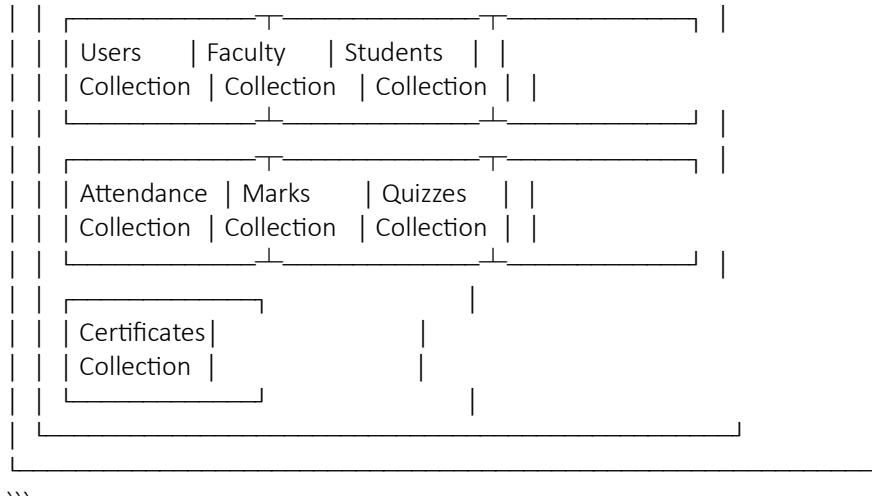
MERN Stack Components

Layer	Technology	Purpose
Frontend	React	Interactive user interfaces, real-time updates, form handling
Backend API	Express.js + Node.js	REST API endpoints, business logic, request routing
Database	MongoDB	Flexible document storage, collections for users, marks, attendance
Authentication	JWT (JSON Web Tokens)	Secure stateless authentication, role encoding

Architecture Diagram

...





User Roles & Permissions

Role Matrix

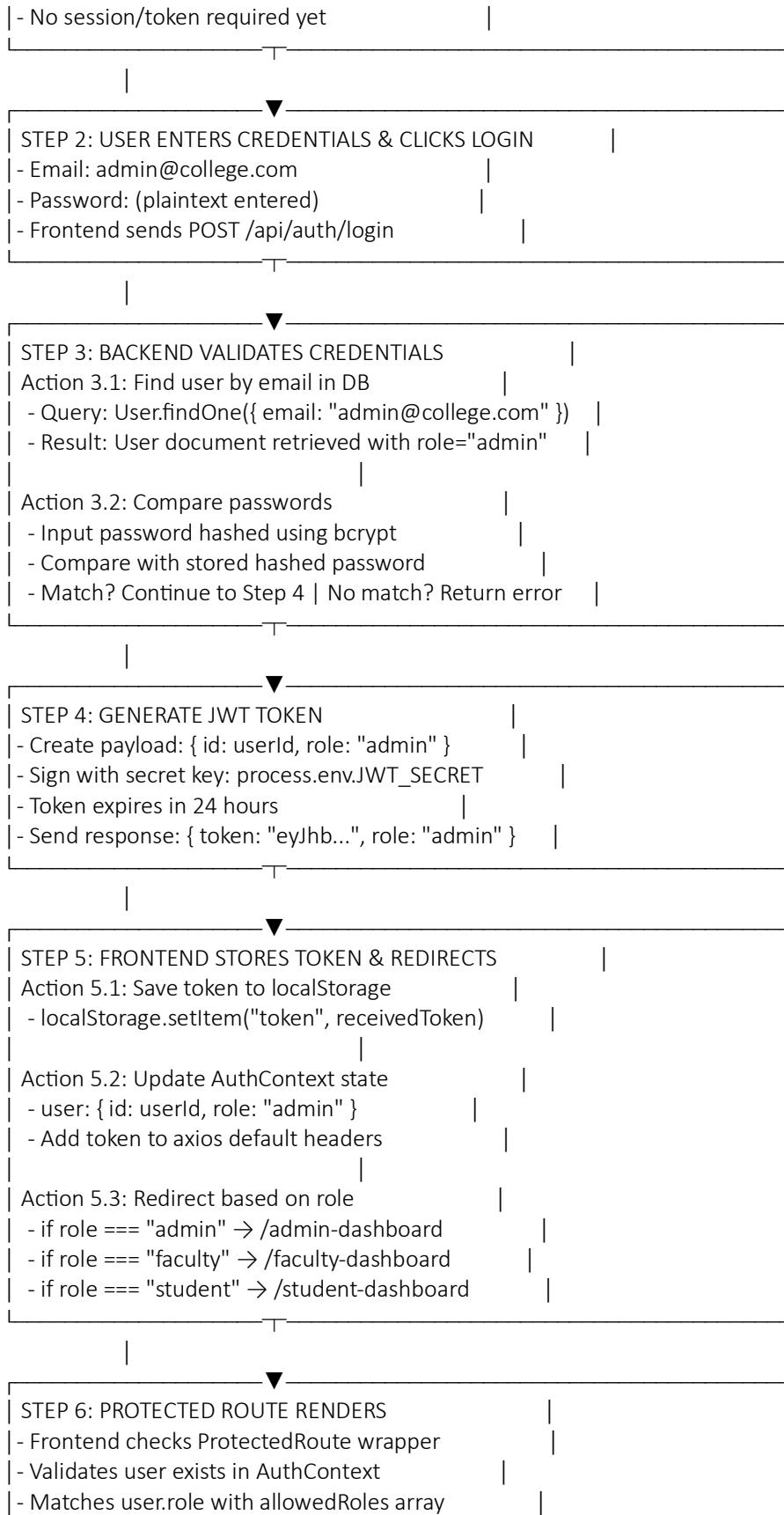
Feature	Admin	Faculty	Student
Create Faculty	✓	✗	✗
Create Student	✓	✗	✗
Create Quiz	✗	✓	✗
View Quiz	✗	✓	✓
Attempt Quiz	✗	✗	✓
Mark Attendance	✗	✓	✗
View Attendance	✗	✓	✓
Give Marks	✗	✓	✗
View Marks	✗	✓	✓
Issue Certificate	✗	✓	✗
View Certificate	✗	✓	✓
Calculate Attendance %	✗	✗	✓
View Reports	✓	✓	✓

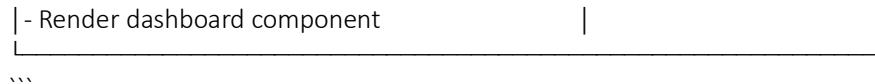
Authentication Workflow

Login Flow- Step by Step

...







Token Flow in Subsequent Requests

...

Request → Browser reads localStorage token
→ Adds to Authorization header: "Bearer eyJhb..."
→ Axios intercepts and includes in all requests
 ↓

Backend receives request with token in header
→ authMiddleware extracts & verifies token
→ Decodes JWT to get userId and role
→ Finds user in DB, attaches to req.user
→ roleCheckMiddleware checks if role allowed
→ If allowed: proceed to controller
→ If denied: return 403 Forbidden
 ↓

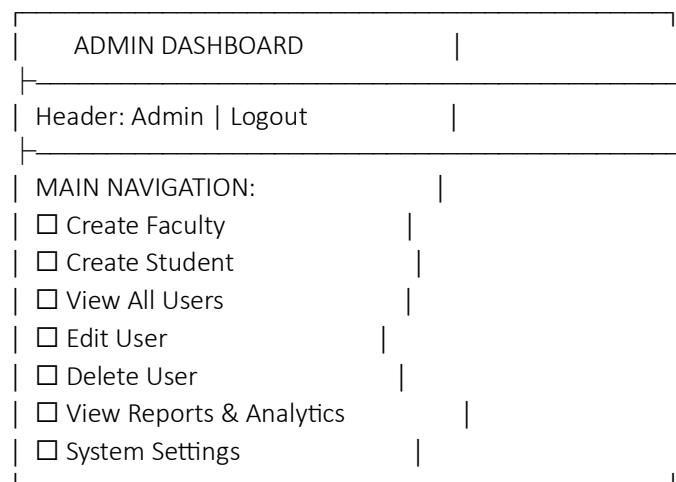
Response sent with updated data or error
→ Frontend updates state and re-renders
...

Admin Workflow

Admin Dashboard Overview

Admin has access to user management and system-wide controls.

...



...

Create Faculty Workflow

...

STEP 1: Admin clicks "Create Faculty" button



STEP 2: Form appears with fields:

- Full Name (text)
- Email (email)
- Password (auto-generated or entered)
- Department (dropdown)
- Specialization (text)
- Subject (dropdown/multi-select)



STEP 3: Admin fills form and clicks "Create"



STEP 4: Frontend validates:

- All required fields present
- Email format valid
- Email not already in DB



STEP 5: Frontend sends POST /api/admin/users

```
Body: {  
    name: "Dr. Sharma",  
    email: "sharma@college.com",  
    password: "hashed_pass",  
    role: "faculty",  
    dept: "CSE",  
    specialization: "AI/ML"  
}
```



STEP 6: Backend authMiddleware checks token

- Token valid? Continue | Invalid? Return 401



STEP 7: Backend roleCheckMiddleware checks role

- Role = "admin"? Continue | Else? Return 403



STEP 8: Controller creates faculty:

- Hash password with bcrypt
- Create User document
- Create Faculty linked document
- Save both to MongoDB



STEP 9: Return success with faculty ID



STEP 10: Frontend displays success message

Refreshes faculty list

Form resets

...

Create Student Workflow

...

Same as Create Faculty, but:

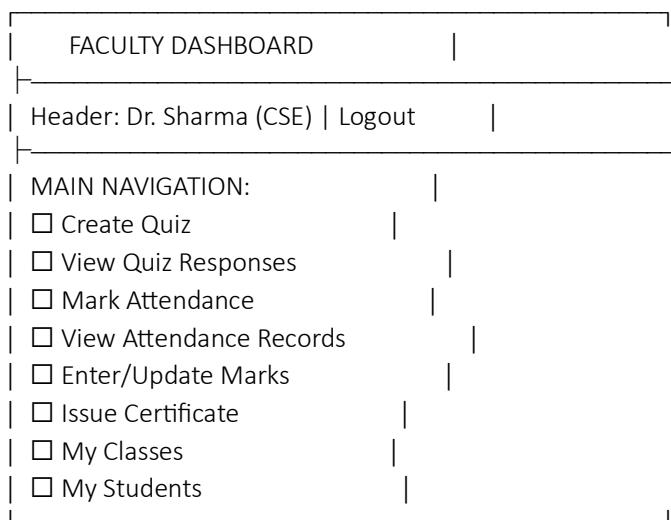
- Form fields: Name, Email, Password, Roll Number, Semester, Department
- Creates Student linked document instead of Faculty
- POST /api/admin/students endpoint

...

Faculty Workflow

Faculty Dashboard Overview

...



...

Quiz Creation & Management Workflow

Create Quiz:

...

STEP 1: Faculty clicks "Create Quiz" button



STEP 2: Quiz form appears:

- Quiz Title
- Subject
- Class/Semester
- Start Date & Time
- End Date & Time
- Total Marks
- Questions Section:
 - * Question Text
 - * Question Type (MCQ/True-False/Short Answer)
 - * Options (for MCQ)

- * Correct Answer
 - * Marks per question
- ↓
- STEP 3: Faculty adds multiple questions:
- Click "Add Question" button repeatedly
 - Each question added to array in form state
- ↓
- STEP 4: Faculty clicks "Create Quiz"
- ↓
- STEP 5: Frontend validation:
- All required fields present
 - At least one question exists
 - Start time < End time
- ↓
- STEP 6: Frontend sends POST /api/faculty/quiz
- ```
Body: {
 title: "Database Quiz 1",
 subject: "Database Management",
 semester: 4,
 startDate: "2026-01-10T10:00:00",
 endDate: "2026-01-10T11:00:00",
 totalMarks: 50,
 facultyId: ObjectId,
 questions: [
 {
 text: "What is normalization?",
 type: "short-answer",
 marks: 5
 },
 {
 text: "SQL stands for?",
 type: "mcq",
 options: ["Structured Query Language", ...],
 correctAnswer: 0,
 marks: 2
 }
]
}
```
- ↓
- STEP 7: Backend:
- Verify token and role="faculty"
  - Create Quiz document in DB
  - Quiz available only during specified time window
  - Return quiz ID
- ↓
- STEP 8: Frontend displays success
- Redirects to quiz list page
- ...

#### Student Attempts Quiz:

...

STEP 1: Student views list of quizzes

- Only shows quizzes in "open" status
- Current time between startDate and endDate

↓

STEP 2: Student clicks "Attempt Quiz" button

↓

STEP 3: Quiz interface loads:

- Question displayed
- Student selects/enters answer
- Timer shows remaining time
- Option to "Next Question" or "Previous"
- "Submit Quiz" button

↓

STEP 4: Student answers all questions

↓

STEP 5: Student clicks "Submit Quiz"

↓

STEP 6: Frontend confirmation dialog:

- "Are you sure? You cannot change answers after submit"

↓

STEP 7: Frontend sends POST /api/student/quiz/:id/submit

```
Body: {
 studentId: ObjectId,
 answers: [
 { questionId: id, answer: selectedOption },
 ...
],
 submittedAt: timestamp,
 timeSpent: seconds
}
```

↓

STEP 8: Backend:

- Verify student is in class for this quiz
- Calculate score by comparing answers with correct answers
- Create entry in Marks collection
- Return score and feedback

↓

STEP 9: Frontend shows:

- "Quiz submitted successfully"
- Score obtained
- Option to view previous questions/answers
- Redirect to dashboard

...

### ### Attendance Marking Workflow

...

STEP 1: Faculty clicks "Mark Attendance"

- ↓
- STEP 2: Attendance form appears:
- Date picker (defaults to today)
  - Class/Semester selector
  - Subject selector
- ↓
- STEP 3: Faculty selects values and clicks "Load Students"
- ↓
- STEP 4: Frontend sends GET /api/faculty/class/:classId/students
- ↓
- STEP 5: Backend:
- Find all students in this class
  - Check if attendance already marked for this date
  - Return list of students
- ↓
- STEP 6: Frontend displays:
- List of students with checkboxes
  - [✓] Student 1- Present
  - [ ] Student 2- Absent
  - [ ] Student 3- Absent
  - "Save Attendance" button
- ↓
- STEP 7: Faculty checks "Present" boxes
- ↓
- STEP 8: Faculty clicks "Save Attendance"
- ↓
- STEP 9: Frontend sends POST /api/faculty/attendance
- ```
Body: {
  date: "2026-01-10",
  subject: "Database Management",
  classId: ObjectId,
  facultyId: ObjectId,
  attendance: [
    { studentId: id, status: "present" },
    { studentId: id, status: "absent" },
    ...
  ]
}
```
- ↓
- STEP 10: Backend:
- Verify faculty can mark attendance for this class
 - Create/update Attendance documents for each student
 - Save to DB
- ↓
- STEP 11: Frontend displays:
- "Attendance saved successfully"
 - Shows summary: "35 present, 5 absent"
- ...

Mark Entry Workflow

...

STEP 1: Faculty clicks "Enter Marks"



STEP 2: Form options:

- Assessment Type: Quiz / Assignment / Mid-Exam / End-Exam
- Subject selector
- Class selector



STEP 3: Faculty selects and clicks "Load Students"



STEP 4: Frontend displays:

Table:

Roll No	Name	Marks	Total	Grade
101	Student A	45	50	A
102	Student B	38	50	B



STEP 5: Faculty can:

- Click on marks cell to edit
- Enter new marks
- Grades auto-calculated
- Click "Save All Marks"



STEP 6: Frontend sends POST /api/faculty/marks

```
Body: {
  assessmentType: "quiz",
  subject: "Database",
  marks: [
    { studentId: id, score: 45, total: 50 },
    ...
  ]
}
```



STEP 7: Backend:

- Create/update Marks documents
- Store in DB



STEP 8: Student can immediately view updated marks in their dashboard

...

Certificate Issuance Workflow

...

STEP 1: Faculty clicks "Issue Certificate"



STEP 2: Form appears:

- Select Certificate Type: Completion / Achievement / Course
- Select Class/Students
- Certificate Template selector

- Issue Date
- Additional notes (optional)

↓

STEP 3: Faculty selects options and multi-select students

↓

STEP 4: Faculty clicks "Issue"

↓

STEP 5: Frontend sends POST /api/faculty/certificate

Body: {
 type: "completion",
 students: [id1, id2, ...],
 template: "template_id",
 issueDate: "2026-01-15",
 facultyId: ObjectId
}

↓

STEP 6: Backend:

- Create Certificate documents for each student
- Set status: "issued"
- Add to student's certificate collection

↓

STEP 7: Frontend displays:

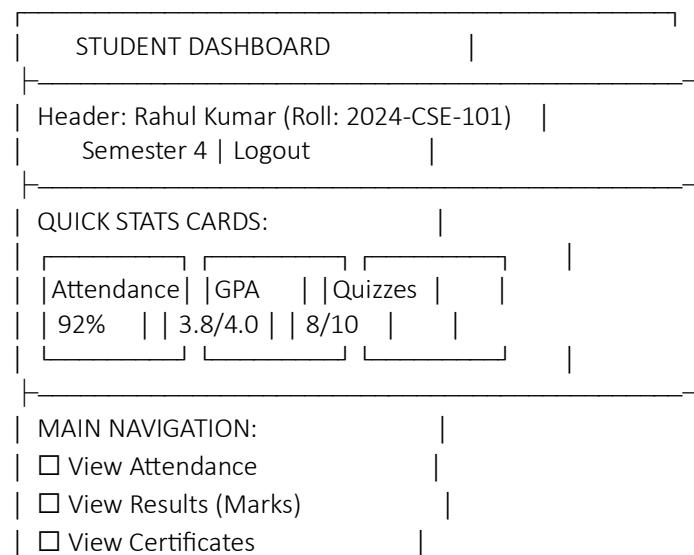
- "Certificate issued to 35 students"
- Show list of issued certificates

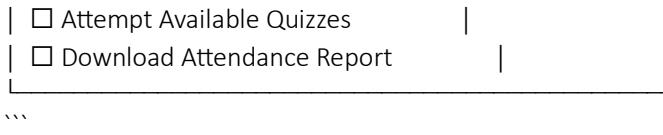
...

Student Workflow

Student Dashboard Overview

...





```  
### View Attendance & Calculate Percentage

```

STEP 1: Student clicks "View Attendance"



STEP 2: Frontend sends GET /api/student/attendance/:studentId



STEP 3: Backend:

- Find all Attendance records for this student
- Group by subject
- Count present/total sessions per subject



STEP 4: Backend calculation:

For each subject:

$$\text{Attendance \%} = (\text{Total Present Days} / \text{Total Possible Days}) \times 100$$

Example:

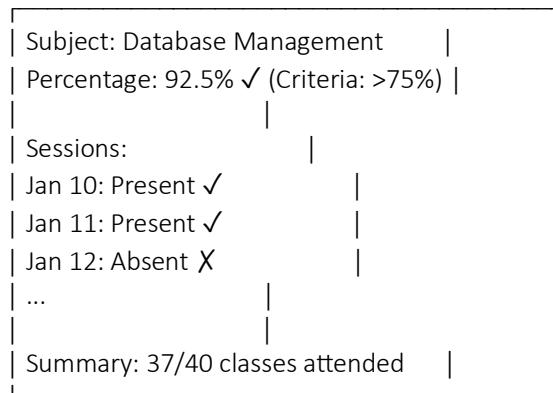
Database Management:

- Total classes: 40
- Present: 37
- Absent: 3
- Percentage: $(37/40) \times 100 = 92.5\%$

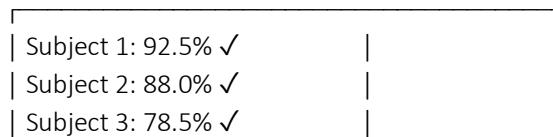


STEP 5: Frontend displays:

ATTENDANCE RECORD



OVERALL SUMMARY:



Subject 4: 72.0% X (Below 75%)	
Subject 5: 95.0% ✓	
Overall: 85.2%	

Export Options:

- Download PDF Report
- Download Excel Sheet

...

View Marks/Results

...

STEP 1: Student clicks "View Results"



STEP 2: Frontend sends GET /api/student/marks/:studentId



STEP 3: Backend:

- Find all Marks for this student
- Group by subject/assessment type
- Calculate subject-wise and overall GPA



STEP 4: Frontend displays:

ACADEMIC RESULTS

DATABASE MANAGEMENT			
Assessment Type	Marks	Total	Grade
Quiz 1	45	50	A
Quiz 2	48	50	A+
Assignment	18	20	A
Mid-Exam	38	50	B+
<hr/>			
Subject Average: 87.5% (A)			

Similar cards for all other subjects

OVERALL ACADEMIC SUMMARY:

Current GPA: 3.8 / 4.0	
Current CGPA: 3.75 / 4.0	
Overall Average: 86.2%	
Academic Status: Good Standing ✓	

...

View Certificates

...

STEP 1: Student clicks "View Certificates"



STEP 2: Frontend sends GET /api/student/certificates/:studentId



STEP 3: Backend:

- Find all Certificate records for student
- Status: "issued"



STEP 4: Frontend displays:

MY CERTIFICATES

Certificate 1	
Title: Database Management Course	
Issued By: Dr. Sharma	
Date: January 15, 2026	
[View PDF] [Download] [Print]	

Certificate 2	
Title: Machine Learning Specialization	
Issued By: Prof. Gupta	
Date: December 20, 2025	
[View PDF] [Download] [Print]	

More certificates...

...

Attempt Quiz

...

STEP 1: Student clicks "View Quizzes"



STEP 2: Frontend sends GET /api/student/quizzes/:studentId



STEP 3: Backend returns quizzes where:

- Student's class matches quiz class
- Current time is within quiz time window OR
- Student has already attempted and result available



STEP 4: Frontend displays:

AVAILABLE QUIZZES

Quiz 1: Database Fundamentals	
Subject: Database Management	
Opens: Jan 15, 2026 at 10:00 AM	

Closes: Jan 15, 2026 at 11:00 AM	
Marks: 50	
[Attempt] [Info]	

COMPLETED QUIZZES

Quiz 2: SQL Queries	
Your Score: 42/50 (84%) ✓	
Completed On: Jan 10, 2026 at 10:45 AM	
[Review] [Download Result]	

(Quiz attempt process detailed in Faculty Workflow section)