Documentation and Reporting

Student Names :

Harsh Doshi(92200133002) Krish Mamtora (92200133022) Rishit Rathod(92200133027)

Technical Overview of the System

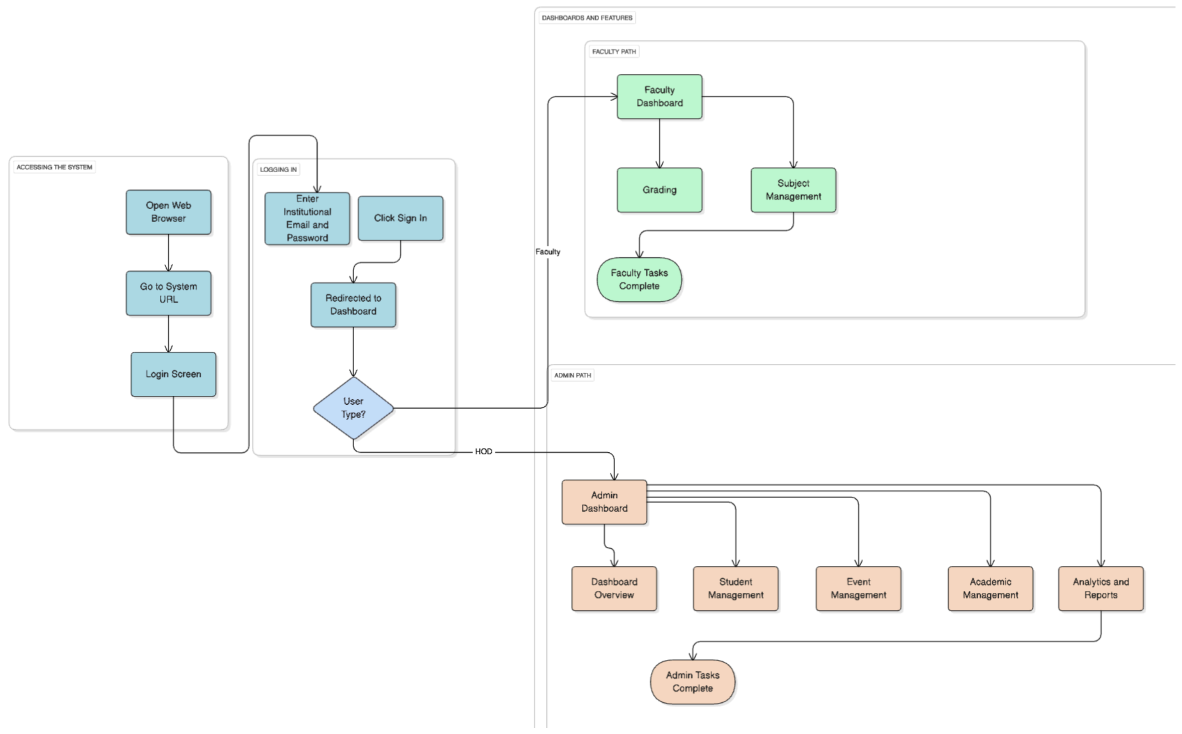
The Student Performance Analyzer is a system that helps HODs, faculties, students and parents to keep track of student performance easily. It combines academic results, co-curricular activities, and extra-curricular achievements in one place.

Frontend part : Students and parents have a mobile app made with Flutter where they can see marks and performance. Teachers and the HOD use a website made with React.js where they can add marks, manage subjects, and check reports.

Backend part : made with Node.js and Express. It does all the calculations, handles the data, and connects with the database. The APIs are checked in Postman.

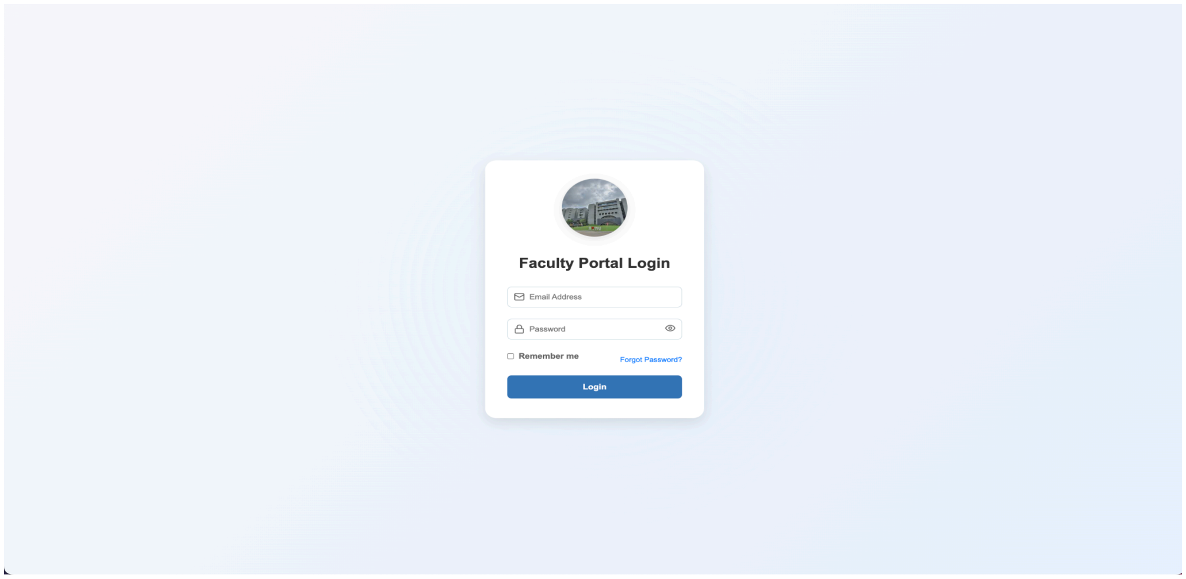
Database : We use MySQL to store things like student details, marks, subjects, and activities. The database is hosted on Aiven.

Process flowchart

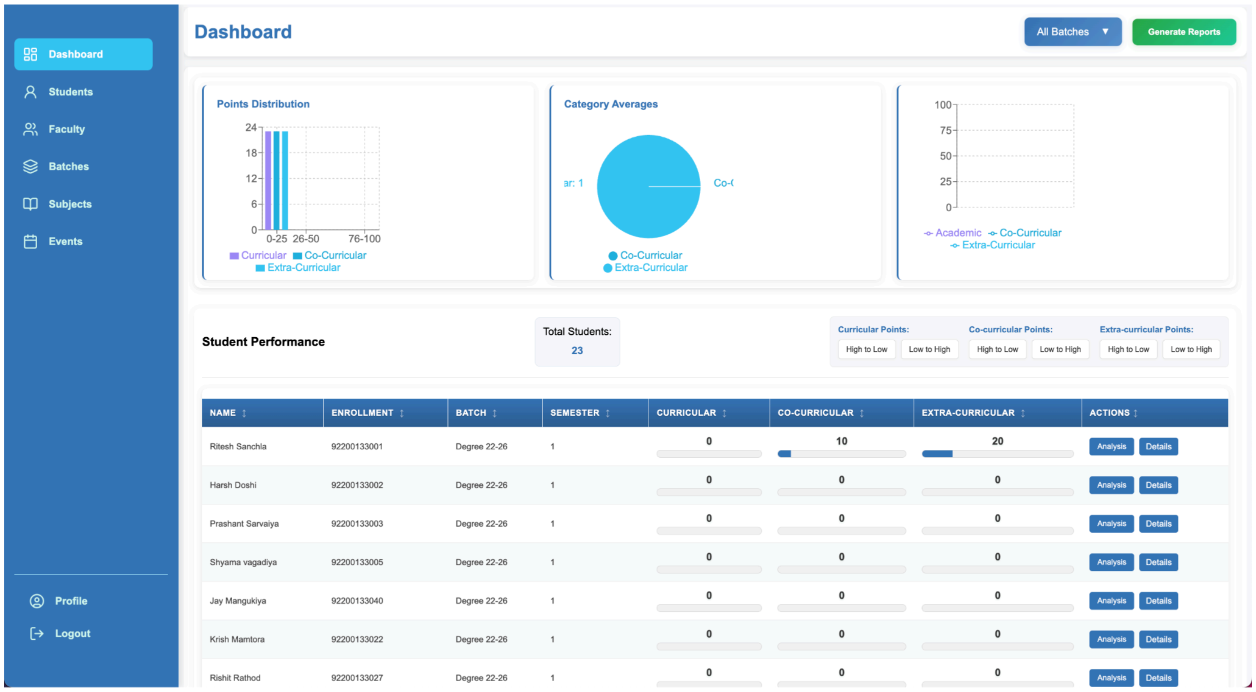


System Overview and User Guide

Accessing the System: Open a web browser and go to the system URL. Enter your institutional email and password, optionally and click Sign In. HOD users will see the Admin Dashboard, and faculty will see the Faculty Dashboard
Login page :



Admin Dashboard: The dashboard shows quick stats like the number of students and faculty, charts of overall performance, and recent activities or updates.



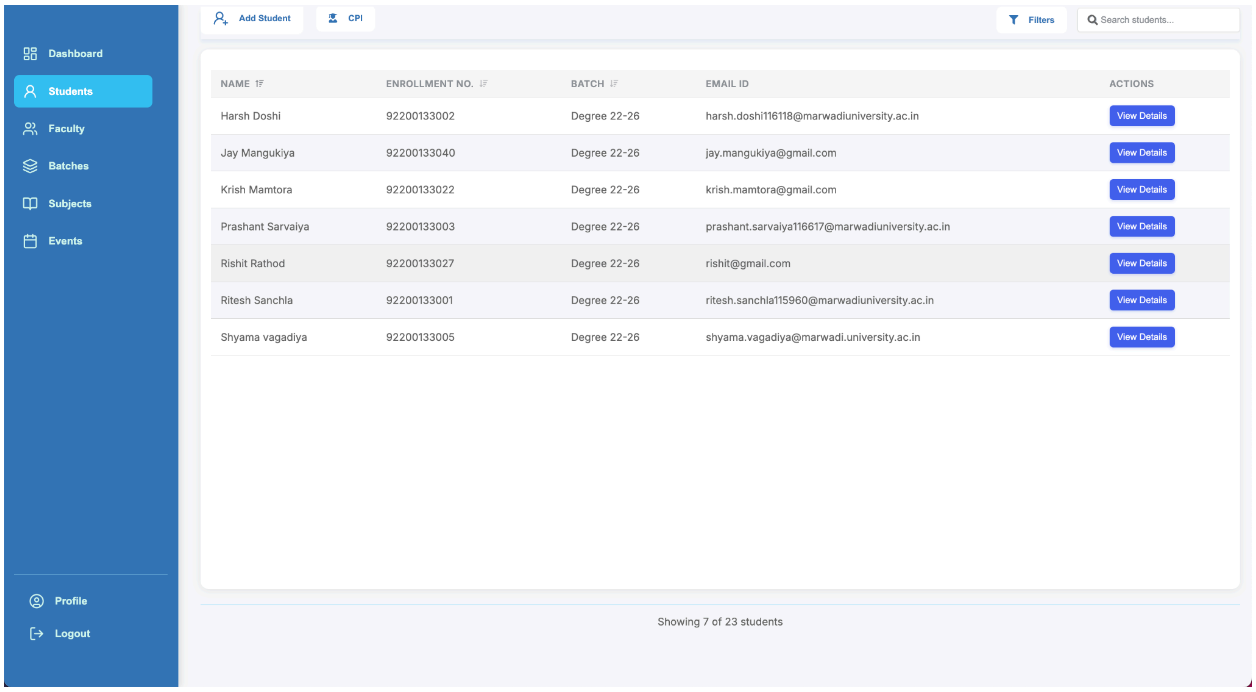
Student Performance

Total Students: 23

Curricular Points: High to Low | Low to High
Co-curricular Points: High to Low | Low to High
Extra-curricular Points: High to Low | Low to High

NAME	ENROLLMENT	BATCH	SEMESTER	CURRICULAR	CO-CURRICULAR	EXTRA-CURRICULAR	ACTIONS
Ritesh Sanchla	92200133001	Degree 22-26	1	0	10	20	Analysis Details
Harsh Doshi	92200133002	Degree 22-26	1	0	0	0	Analysis Details
Prashant Sarvaiya	92200133003	Degree 22-26	1	0	0	0	Analysis Details
Shyama vagadiya	92200133005	Degree 22-26	1	0	0	0	Analysis Details
Jay Mangukiya	92200133040	Degree 22-26	1	0	0	0	Analysis Details
Krish Mamtora	92200133022	Degree 22-26	1	0	0	0	Analysis Details
Rishit Rathod	92200133027	Degree 22-26	1	0	0	0	Analysis Details

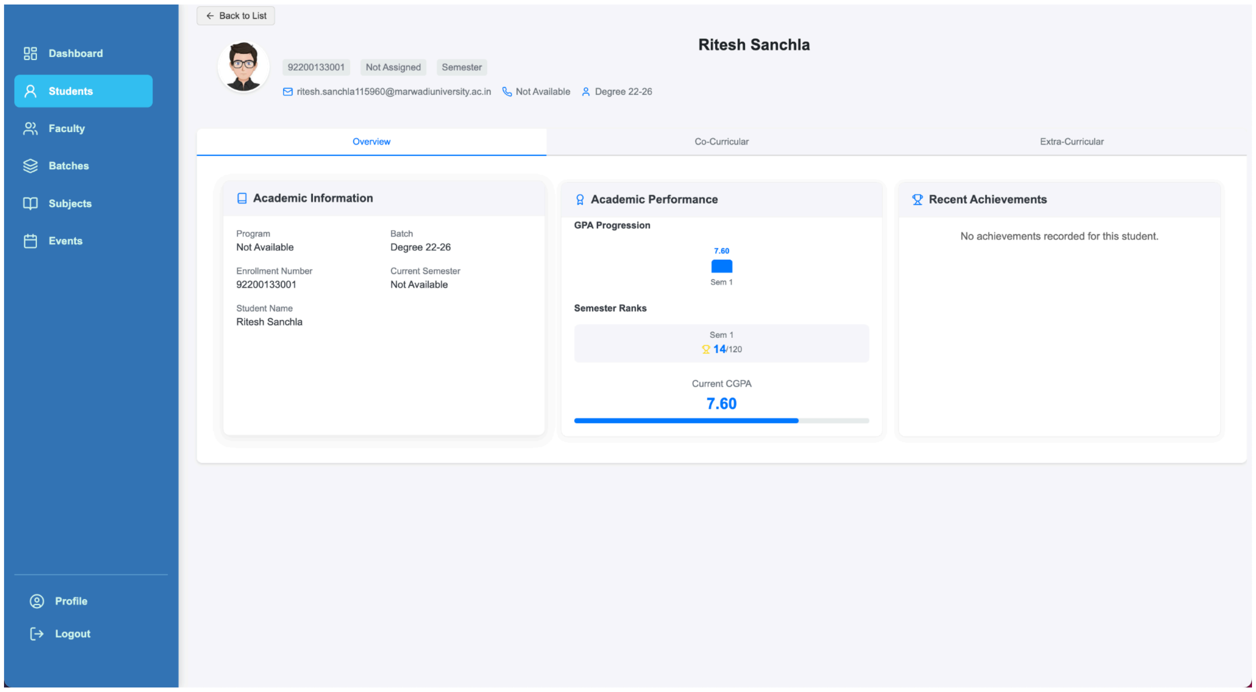
Student Management: Add new students with their enrollment details, contact info, batch, and semester. You can also search, view, and edit student profiles easily.



NAME	ENROLLMENT NO.	BATCH	EMAIL ID	ACTIONS
Harsh Doshi	92200133002	Degree 22-26	harsh.doshi116118@marwadiuniversity.ac.in	View Details
Jay Mangukiya	92200133040	Degree 22-26	jay.mangukiya@gmail.com	View Details
Krish Mamtora	92200133022	Degree 22-26	krish.mamtora@gmail.com	View Details
Prashant Sarvaiya	92200133003	Degree 22-26	prashant.sarvaiya116617@marwadiuniversity.ac.in	View Details
Rishit Rathod	92200133027	Degree 22-26	rishit@gmail.com	View Details
Ritesh Sanchla	92200133001	Degree 22-26	ritesh.sanchla115960@marwadiuniversity.ac.in	View Details
Shyama vagadiya	92200133005	Degree 22-26	shyama.vagadiya@marwadi.university.ac.in	View Details

Showing 7 of 23 students

Student Profile:



Ritesh Sanchla

92200133001 Not Assigned Semester

ritesh.sanchla115960@marwadiuniversity.ac.in Not Available Degree 22-26

Academic Information

Program: Not Available
Batch: Degree 22-26
Enrollment Number: 92200133001
Current Semester: Not Available
Student Name: Ritesh Sanchla

Academic Performance

GPA Progression: 7.60 Sem 1

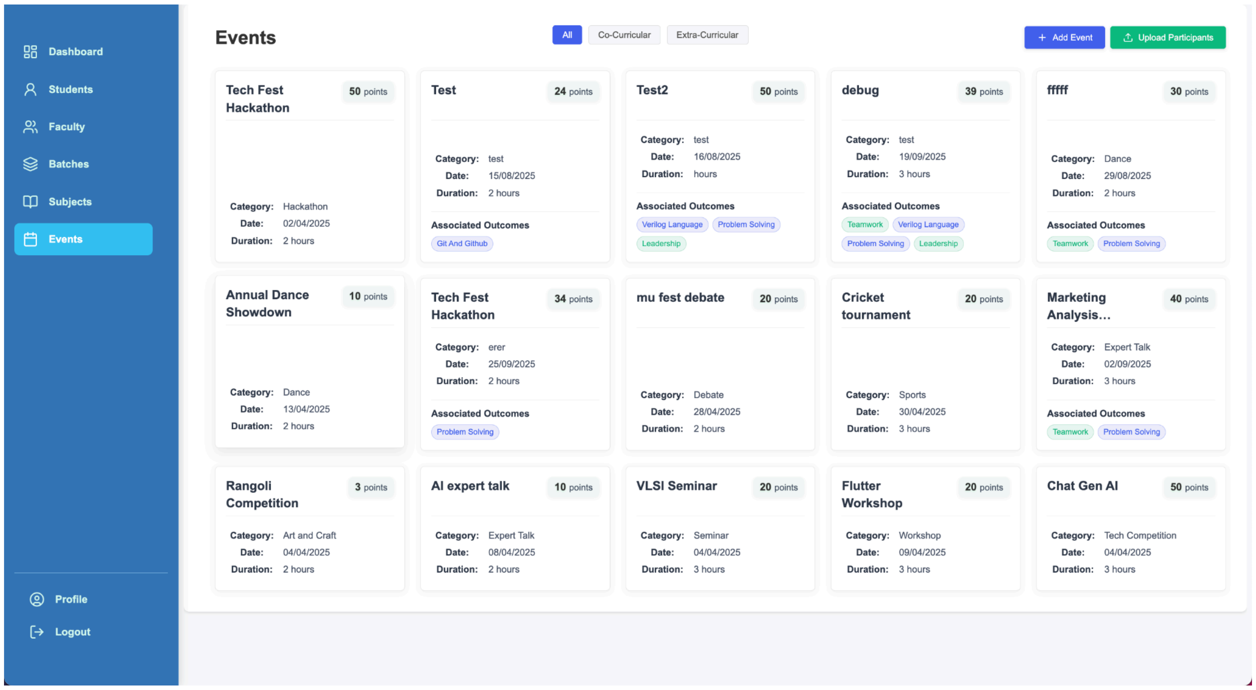
Semester Ranks: Sem 1 14/120

Current CGPA 7.60

Recent Achievements

No achievements recorded for this student.

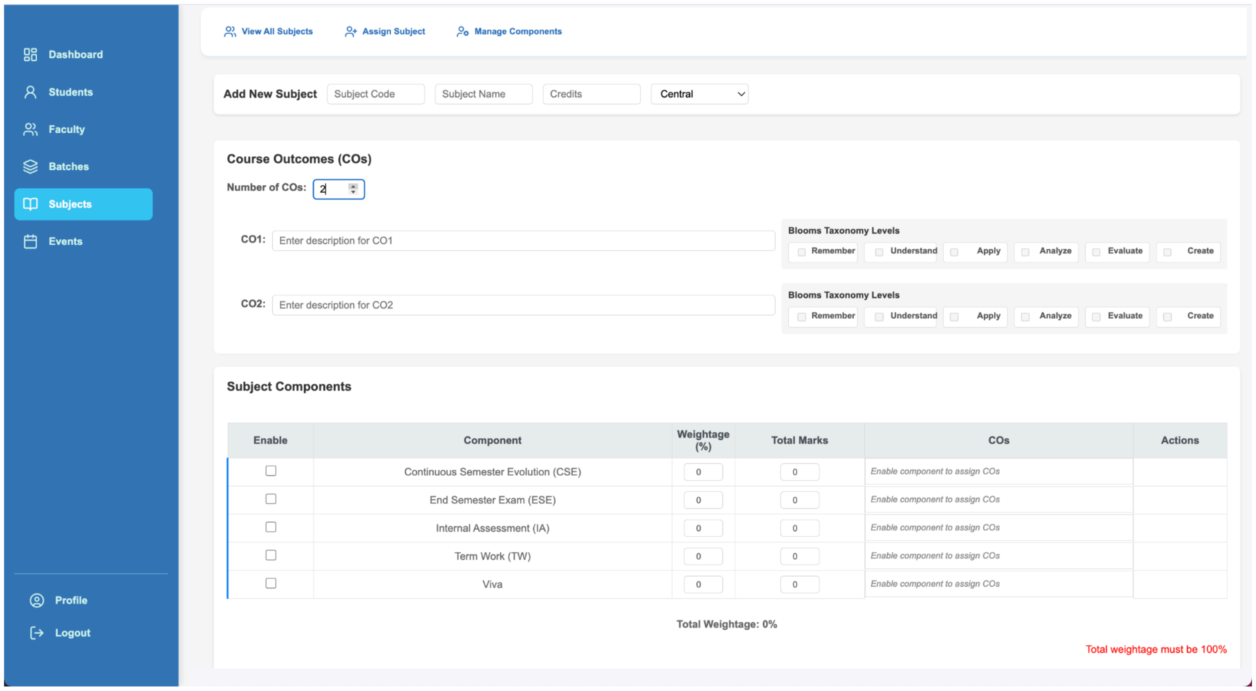
Event Management: Create co-curricular or extracurricular events with details like name, type, date, duration, points, and outcomes. Existing events can be edited or deleted.


Dashboard
Students
Faculty
Batches
Subjects
Events
Profile
Logout

Events — All | Co-Curricular | Extra-Curricular — + Add Event | Upload Participants

Event	Points	Category	Date	Duration	Associated Outcomes
Tech Fest Hackathon	50 points	Hackathon	02/04/2025	2 hours	
Test	24 points	test	15/08/2025	2 hours	Git And Github
Test2	50 points	test	16/08/2025	hours	Verilog Language, Problem Solving, Leadership
debug	39 points	test	19/09/2025	3 hours	Teamwork, Verilog Language, Problem Solving, Leadership
fffff	30 points	Dance	29/08/2025	2 hours	Teamwork, Problem Solving
Annual Dance Showdown	10 points	Dance	13/04/2025	2 hours	
Tech Fest Hackathon	34 points	erer	25/09/2025	2 hours	Problem Solving
mu fest debate	20 points	Debate	28/04/2025	2 hours	
Cricket tournament	20 points	Sports	30/04/2025	3 hours	
Marketing Analysis...	40 points	Expert Talk	02/09/2025	3 hours	Teamwork, Problem Solving
Rangoli Competition	3 points	Art and Craft	04/04/2025	2 hours	
AI expert talk	10 points	Expert Talk	08/04/2025	2 hours	
VLSI Seminar	20 points	Seminar	04/04/2025	3 hours	
Flutter Workshop	20 points	Workshop	09/04/2025	3 hours	
Chat Gen AI	50 points	Tech Competition	04/04/2025	3 hours	

Academic Management: Create and manage batches, add subjects with codes, credits, and assessments, and assign faculty to subjects with defined roles.
Subject Creation:


Dashboard
Students
Faculty
Batches
Subjects
Events
Profile
Logout

View All Subjects | Assign Subject | Manage Components

Add New Subject: Subject Code | Subject Name | Credits | Central

Course Outcomes (COs)
Number of COs: 2

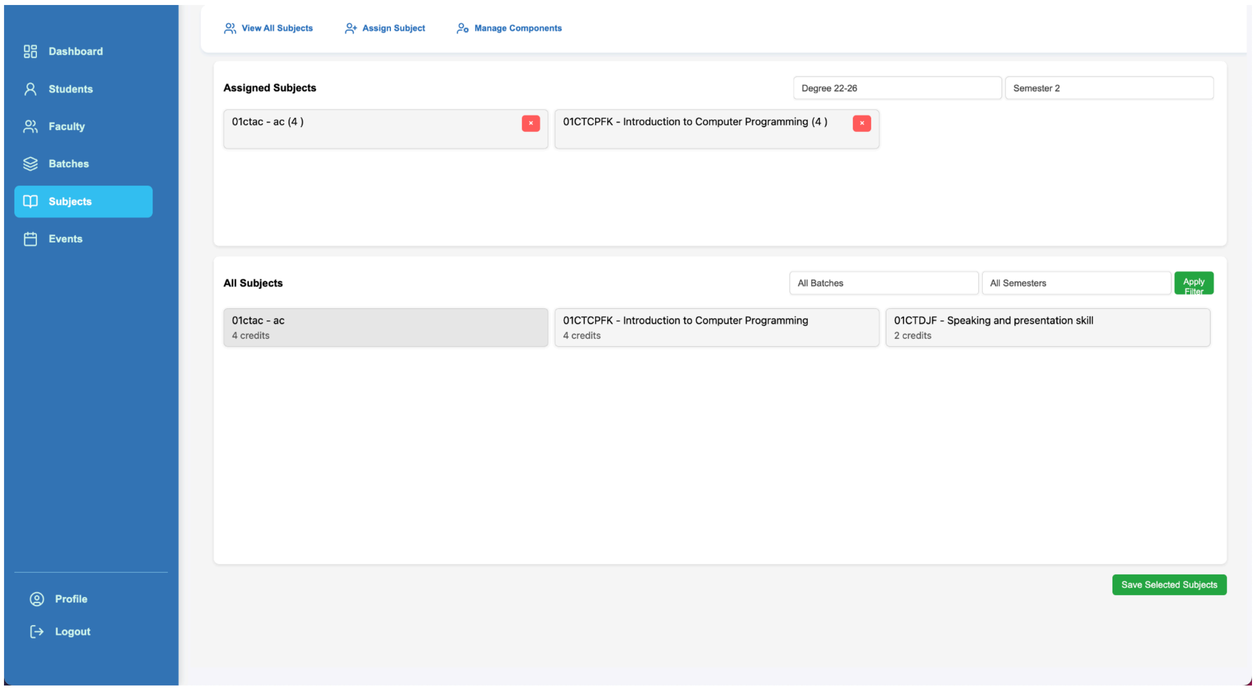
CO1: Enter description for CO1 — Blooms Taxonomy Levels: ☐ Remember ☐ Understand ☐ Apply ☐ Analyze ☐ Evaluate ☐ Create
CO2: Enter description for CO2 — Blooms Taxonomy Levels: ☐ Remember ☐ Understand ☐ Apply ☐ Analyze ☐ Evaluate ☐ Create

Subject Components

Enable	Component	Weightage (%)	Total Marks	COs	Actions
☐	Continuous Semester Evolution (CSE)	0	0	Enable component to assign COs	
☐	End Semester Exam (ESE)	0	0	Enable component to assign COs	
☐	Internal Assessment (IA)	0	0	Enable component to assign COs	
☐	Term Work (TW)	0	0	Enable component to assign COs	
☐	Viva	0	0	Enable component to assign COs	

Total Weightage: 0%
Total weightage must be 100%

Subject Allocation in Class:



Assigned Subjects — Degree 22-26 — Semester 2

01ctac - ac (4)

01CTCPFK - Introduction to Computer Programming (4)

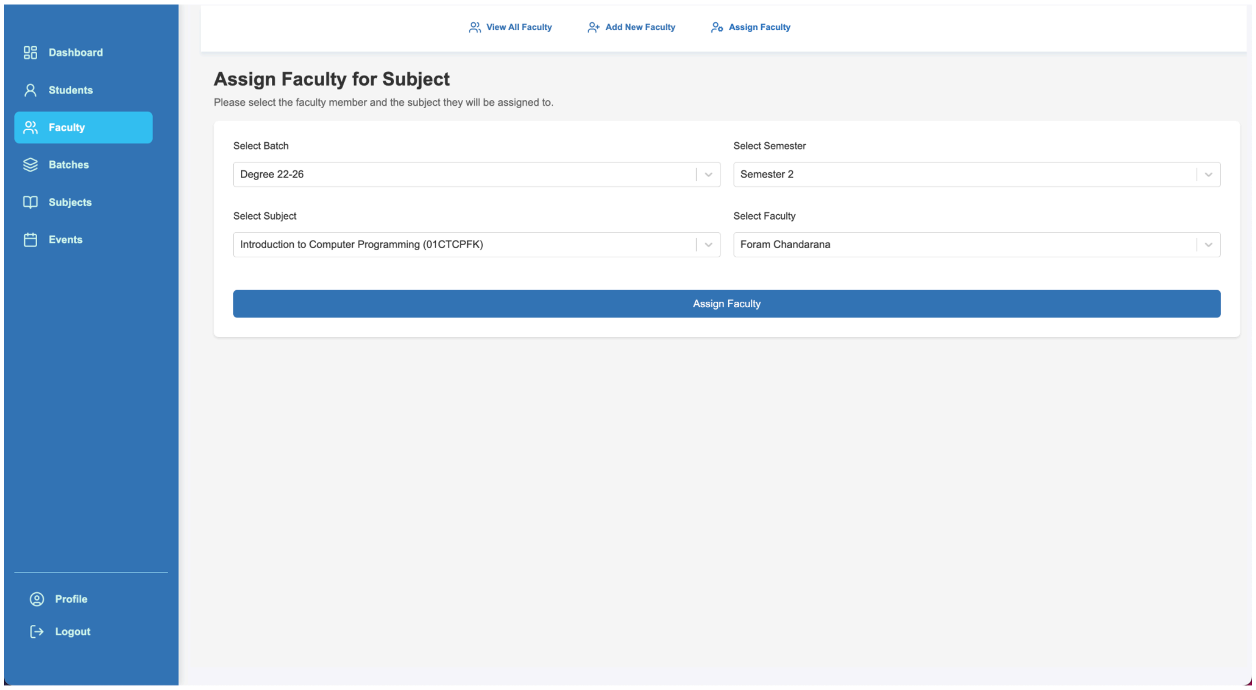
All Subjects — All Batches — All Semesters

01ctac - ac
4 credits

01CTCPFK - Introduction to Computer Programming
4 credits

01CTDJF - Speaking and presentation skill
2 credits

Subject Assignment to Faculty:



Assign Faculty for Subject

Please select the faculty member and the subject they will be assigned to.

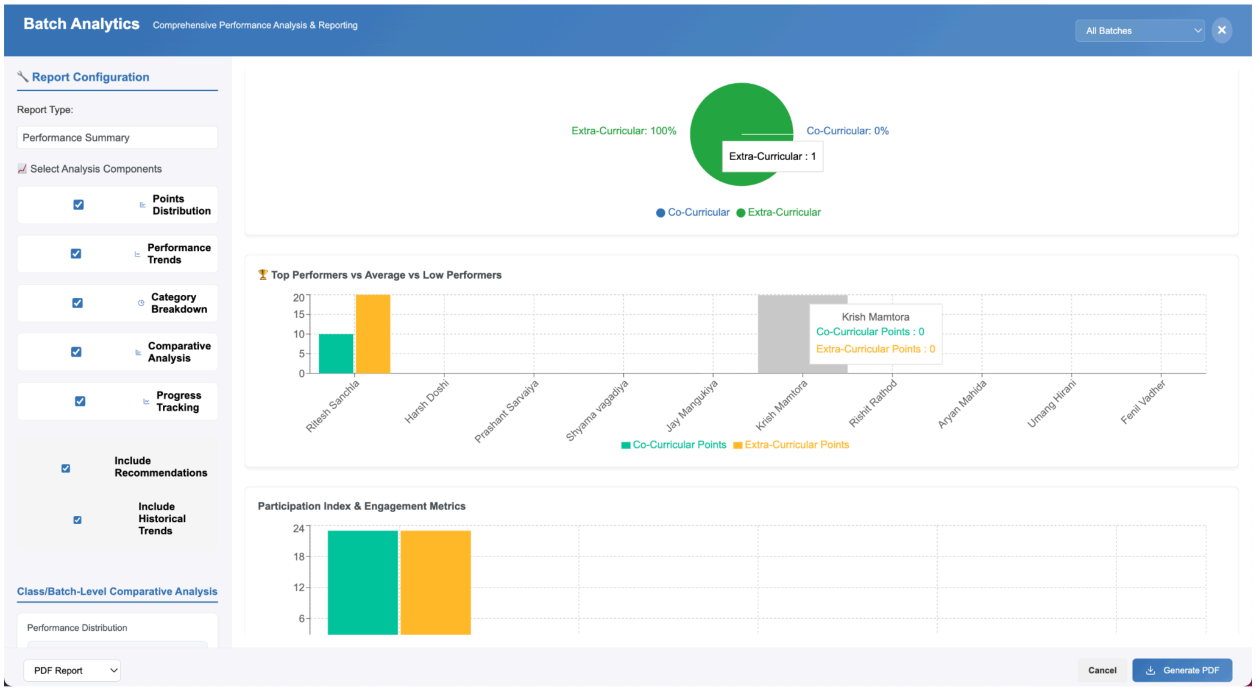
Select Batch: Degree 22-26

Select Semester: Semester 2

Select Subject: Introduction to Computer Programming (01CTCPFK)

Select Faculty: Foram Chandarana

Analytics & Reports: View student performance, grades, activity history, and Bloom's Taxonomy analysis. Generate reports for students, batches, or faculty in PDF or Excel format.


Batch Analytics Comprehensive Performance Analysis & Reporting

All Batches

Report Configuration

Report Type: Performance Summary

Select Analysis Components

- ✓ Points Distribution
- ✓ Performance Trends
- ✓ Category Breakdown
- ✓ Comparative Analysis
- ✓ Progress Tracking
- ✓ Include Recommendations
- ✓ Include Historical Trends

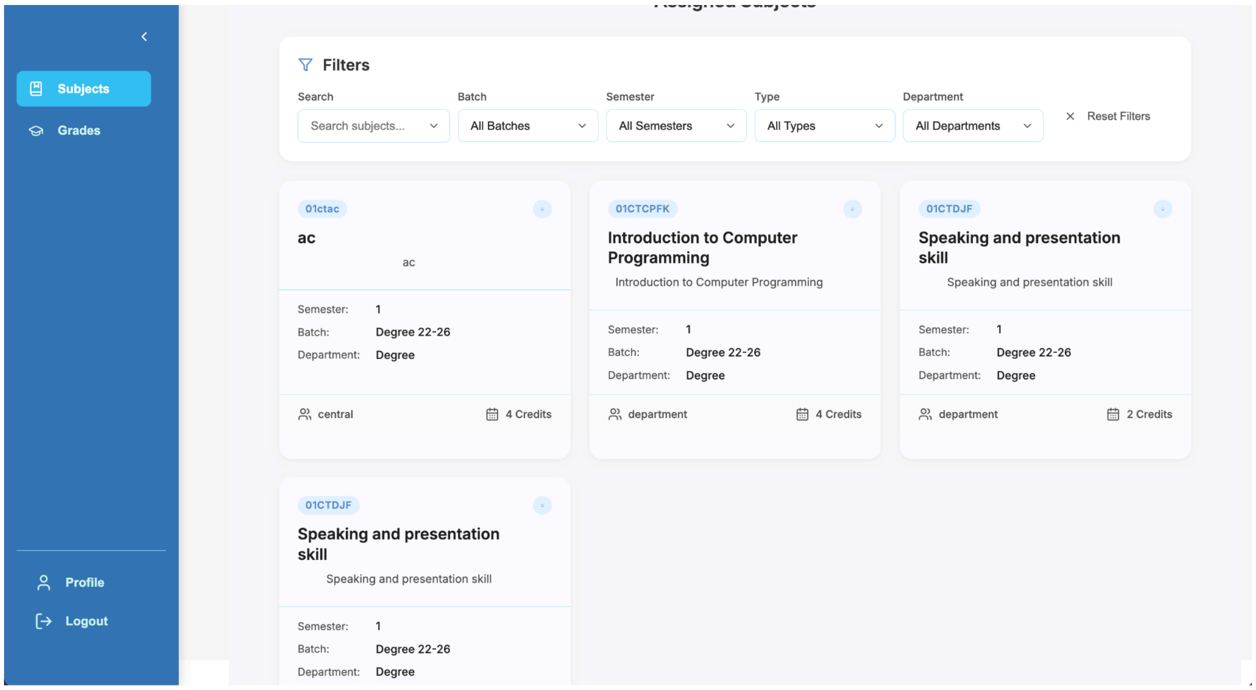
Class/Batch-Level Comparative Analysis

Performance Distribution

PDF Report

Cancel | Generate PDF

Faculty Dashboard & Grading: Select your subject and batch, choose the assessment type (CA, IA, ESE, TW, VIVA), enter marks, and the system automatically calculates percentages. Save the grades to complete the process.



01ctac
ac
ac

Semester: 1
Batch: Degree 22-26
Department: Degree
central — 4 Credits

01CTCPFK
Introduction to Computer Programming
Introduction to Computer Programming

Semester: 1
Batch: Degree 22-26
Department: Degree
department — 4 Credits

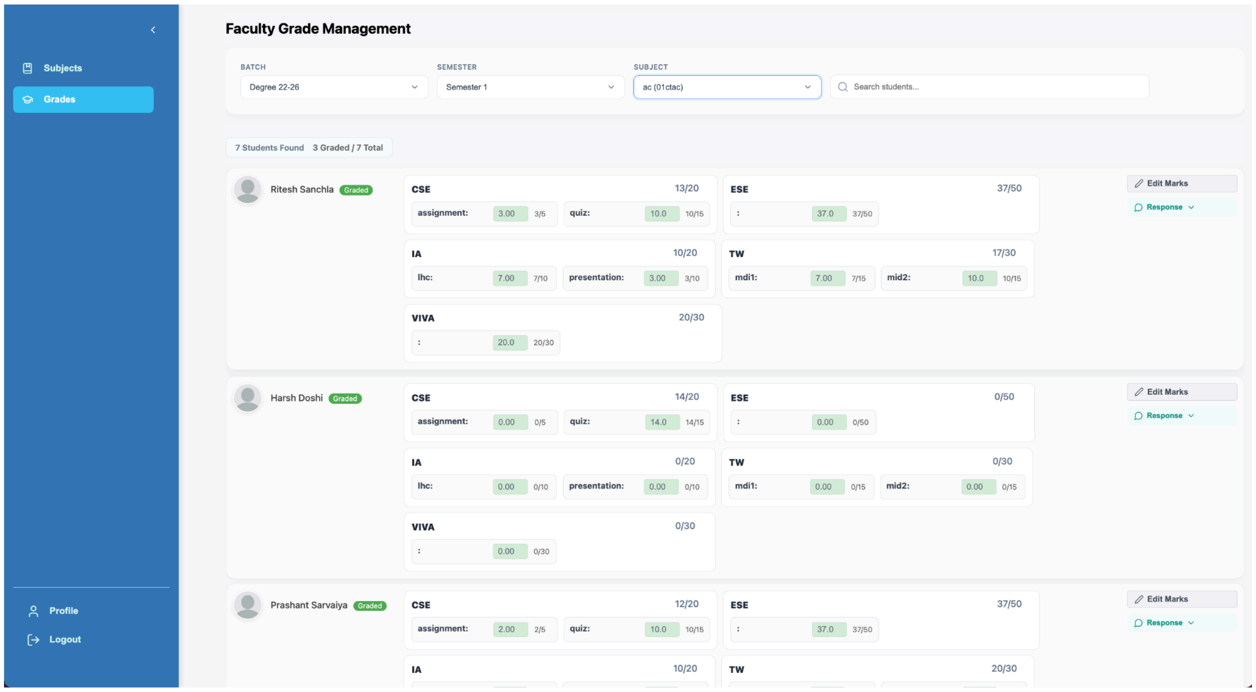
01CTDJF
Speaking and presentation skill
Speaking and presentation skill

Semester: 1
Batch: Degree 22-26
Department: Degree
department — 2 Credits

01CTDJF
Speaking and presentation skill
Speaking and presentation skill

Semester: 1
Batch: Degree 22-26
Department: Degree

Student Grading by Faculty



Faculty Grade Management

BATCH: Degree 22-26
SEMESTER: Semester 1
SUBJECT: ac (01ctac)
Search students...

7 Students Found 3 Graded / 7 Total

Ritesh Sanchla Graded
CSE 13/20 — assignment: 3.00 3/5; quiz: 10.0 10/15
ESE 37/50 — : 37.0 37/50
IA 10/20 — lhc: 7.00 7/10; presentation: 3.00 3/10
TW 17/30 — mdi1: 7.00 7/15; mid2: 10.0 10/15
VIVA 20/30 — : 20.0 20/30

Harsh Doshi Graded
CSE 14/20 — assignment: 0.00 0/5; quiz: 14.0 14/15
ESE 0/50 — : 0.00 0/50
IA 0/20 — lhc: 0.00 0/10; presentation: 0.00 0/10
TW 0/30 — mdi1: 0.00 0/15; mid2: 0.00 0/15
VIVA 0/30 — : 0.00 0/30

Prashant Sarvaiya Graded
CSE 12/20 — assignment: 2.00 2/5; quiz: 10.0 10/15
ESE 37/50 — : 37.0 37/50
IA 10/20
TW 20/30

System Architecture & Technical Implementation

1. Frontend Architecture (React.js)

- Framework: React 19.0.0 with Vite build tool
- Routing: React Router DOM for navigation
- State Management: React Hooks (useState, useEffect)
- UI Components: Custom components with Lucide React icons
- Charts & Visualizations: Chart.js and Recharts
- Styling: CSS modules with responsive design

2. Backend Architecture (Node.js / Express)

- Framework: Express.js 4.21.2
- Database: MySQL with Sequelize ORM 6.37.5
- Authentication: JWT (JSON Web Tokens) with bcrypt for hashing
- File Processing: Multer (uploads), XLSX (Excel file handling)
- Email: Nodemailer for notifications
- Security: CORS enabled, password encryption with bcrypt

Dependencies

Backend Dependencies

```json
{
"express": "^4.21.2",
"sequelize": "^6.37.5",
"mysql2": "^3.12.0",
"jsonwebtoken": "^9.0.2",
"bcrypt": "^5.1.1",
"cors": "^2.8.5",
"multer": "^1.4.5-lts.1",
"xlsx": "^0.18.5",
"nodemailer": "^7.0.3",
"dotenv": "^16.4.7"
}
```

Frontend Dependencies

```
{
"react": "^19.0.0",
"react-router-dom": "^7.1.5",
"chart.js": "^4.4.8",
"recharts": "^2.15.2",
"axios": "^1.7.9",
"lucide-react": "^0.475.0",
"jspdf": "^3.0.3",
"react-toastify": "^11.0.5",
}
```

Database Models

- Students → Stores student details and enrollment data
- Faculty → Faculty information and subject assignments
- Subjects → Courses with components and weightages
- EventMaster → Event definitions (co-curricular/extra-curricular)
- StudentMarks → Assessment records
- BloomsTaxonomy → Learning outcome classifications

Key Relationships

- Students ↔ Batches (Many-to-One)
- Faculty ↔ Subjects (Many-to-Many)
- Events ↔ Outcomes (Many-to-Many)
- Students ↔ Marks (One-to-Many)

Security Features

1. **JWT-Based Authentication with Role-Based Access :** Token generation on login and Role-based navigation (HOD → Admin panel, Faculty → Dashboard, etc.)

```
const token = jwt.sign(
  { id: user.id, role: user.role },
  process.env.JWT_SECRET || 'your_secret_key',
  { expiresIn: '1h' }
);
```

```
res.status(200).json({
  message: 'Login successful',
  token,
  user: {
    id: user.id,
    name: user.name,
    email: user.email,
    role: user.role
  }
});
```

2. **Password Security :** Bcrypt used for password hashing and verification and Only hashed passwords stored in DB

```
const salt = await bcrypt.genSalt(10);
const hashedPassword = await bcrypt.hash(password, salt);

const user = await User.create({
  name,
  email,
  password: hashedPassword,
  role
});

const isMatch = await bcrypt.compare(password, user.password);
if (!isMatch) {
  console.log('Invalid password for email:', email);
  return res.status(400).json({
    message: 'Invalid email or password',
    error: 'Invalid password'
  });
}
```

3. **Input Validation :** Server-side validation for required fields and Error handling with detailed messages

```
const createStudent = async (req, res) => {
  try {
    const { name, email, batchID, enrollment, currentSemester, currentClassName } = req.body;
    if (!name || !email || !batchID || !enrollment || !currentSemester) {
      console.error("Missing required fields:", { name, email, batchID, enrollment,
currentSemester });
      return res.status(400).json({
        error: "All fields are required",
        received: { name, email, batchID, enrollment, currentSemester }
      });
    }

    const batch = await Batch.findOne({
      where: { batchName: batchID },
      attributes: ['id', 'batchName']
    });

    if (!batch) {
      console.error("Batch not found:", batchID);
      return res.status(400).json({
        error: "Batch not found",
        receivedBatchName: batchID
      });
    }

  } catch (error) {
    res.status(500).json({
      error: error.message,
      type: error.name,
      details: error.errors?.map(e => e.message) || []
    });
  }
};
```

4. **SQL Injection Prevention :** Sequelize ORM uses parameterized queries and Sensitive data (like passwords) excluded from queries

```
const user = await User.findOne({
  where: { email }
});

const users = await User.findAll({
  attributes: { exclude: ['password'] }
});
```

Data Flow

1. **Frontend → Backend**: HTTP requests made using Axios
2. **Backend → Database**: Sequelize ORM executes queries safely
3. **Authentication**: JWT tokens passed in request headers
4. **File Processing**: Multer handles file uploads, XLSX parses Excel sheets
5. **Email Notifications**: Automated alerts sent with Nodemailer

Backend folder structure

```
backend/
├── package.json
├── server.js
├── app.js
│
├── controller/
│   ├── authController.js
│   ├── studentController.js
│   ├── eventController.js
│   ├── facultyController.js
│   ├── studentAnalysisController.js
│   └── excelUploadController.js
│
├── models/
│   ├── index.js
│   ├── students.js
│   ├── faculty.js
│   ├── users.js
│   ├── EventMaster.js
│   ├── studentMarks.js
│   └── batch.js
│
├── routes/
│   ├── auth_routes.js
│   ├── student_routes.js
│   ├── events.js
│   ├── studentAnalysis_routes.js
│   └── faculty_routes.js
│
└── uploads/
```

Frontend folder structure

```
frontend/
├── package.json
├── vite.config.js
├── index.html
│
└── src/
    ├── main.jsx
    ├── App.jsx
    │
    ├── components/
    │   │
    │   ├── login/
    │   │   ├── Login.jsx
    │   │   └── Login.css
    │   │
    │   ├── HOD/
    │   │   ├── dashboard/
    │   │   │   ├── Dashboard.jsx
    │   │   │   ├── StudentAnalysis.jsx
    │   │   │   ├── PerformanceOverview.jsx
    │   │   │   └── ReportGenerator.jsx
    │   │   │
    │   │   ├── events/
    │   │   │   └── AddEventForm.jsx
    │   │   │
    │   │   ├── addstudent/
    │   │   ├── managefaculty/
    │   │   ├── managebatches/
    │   │   ├── sidebar/
    │   │   └── navbar/
    │   │
    │   └── Faculty/
    │       ├── dashboard/
    │       ├── grading/
    │       ├── assignedSubjects/
    │       ├── sidebar/
```

```
|         └── navbar/
|
├── utils/
|    └── apiConfig.js
|
└── assets/
```