RV College of Engineering®, Bengaluru – 59
Department of Computer Science and Engineering
DATABASE MANAGEMENT SYSTEMS (CD252IA)

Report

TITLE - Project Team Formation and Management System		
TEAM	**USN:**	**Name**
	1RV23CS172	PRAJWAL T A
	1RV23CS179	PRATEEK S D
	1RV23CS173	PRAKHAR PARASHAR
	1RV23CS166	PIYUSH KHERIA

1. Introduction

Project-based learning plays a crucial role in engineering education, encouraging students to apply theoretical knowledge to real-world problems through collaborative projects. As the number of students and project submissions increases every semester, managing team formation, ensuring fair collaboration, and maintaining originality in project ideas becomes increasingly challenging. To address these challenges, there is a need for a structured digital system that supports collaboration, enforces academic guidelines, and efficiently manages project information across semesters.

a. Problem Statement

In the current academic environment, team formation and project idea submission are largely handled through informal methods such as messaging groups and online forms. This manual approach often leads to difficulties in forming eligible teams, lack of transparency in project selection, and frequent repetition of project ideas across semesters. Faculty members also face challenges in tracking previous projects and evaluating originality due to the absence of a centralized repository. Therefore, there is a need for a structured, centralized, and intelligent system that automates team formation, ensures project novelty, and maintains an accessible archive of academic projects.

This project is a **Database Management System for engineering colleges** designed to streamline the process of project-based learning through two core modules: **Team Formation** and **Project Novelty & Collaboration**.

- **Team Formation**: Students join or create teams based on the type of project they wish to work on and their skills. The system matches students according to project interests and validates team eligibility as per college rules (branch, skills, and other criteria).

- **Project Novelty & Collaboration:** Teams submit project ideas, which are checked against previous submissions using AI-powered semantic search. The system highlights overlap with existing projects and enables students to note their novel contributions. Students may also showcase personal and hackathon projects for visibility and collaboration.

b. Objectives and Scope

The primary objective of this project is to develop a centralized **Academic Project Management System** that automates the lifecycle of student projects. The system aims to:

- **Automate Team Formation:** Enforce business rules (team size, department limits) during registration.
- **Streamline Approvals:** Digitize the workflow for abstract submission and mentor grading.
- **Enable Semantic Search:** Allow students and mentors to find archival projects using keyword/tag matching.
- **Centralize Data:** Maintain a single source of truth for student records and project marks.

Scope of Implementation: The system covers the core internal processes of the college, including user authentication, team creation, project submission, and archival reference. It explicitly excludes external integrations such as corporate recruitment portals or automated plagiarism detection engines.

2. Software Requirement Specification

a. Functional Requirements

The system functionality is divided by user role:

A. Student Module

1. **Registration & Profile:** Students must be able to register using their USN and manage their profile (skills, resume link).
2. **Team Formation:** Students can create a new team or send "Join Requests" to existing teams.

3. **Project Submission:** The team leader can submit the project title, abstract, and domain.
4. **Search Archives:** Students can search past projects by keywords or technology stack to gather references.

B. Faculty Module (Mentors)

1. **View Assigned Teams:** Faculty can view a dashboard of all teams they are guiding.
2. **Project Evaluation:** Mentors can view project submissions, refer archived projects and assign marks/grades.

C. Administrator Module

1. **Mentor Assignment:** Admins can view formed teams and assign faculty mentors based on domain expertise.
2. **System Oversight:** Admins have full access to view all users and project statuses.

b. Non-Functional Requirements

1. **Data Integrity:** The system must ensure that a student belongs to exactly one team and one department at any given time (enforced via database constraints).
2. **Scalability:** The database design must support the addition of new departments or changes in cluster logic without requiring schema restructuring.
3. **Security:** User authentication must be secure (hashed passwords), and authorization rules must prevent students from altering grades or admin settings.
4. **Performance:** The semantic search for archives should return results in under 2 seconds for a standard query load.

c. System Requirements

Software Requirements:

- **Front-End:** React.js / Javascript, Tailwind CSS
- **Back-End:** Node.js, Express.js
- **Database (Hybrid Approach):**
  - **MongoDB (NoSQL):** Used for User Authentication (Students, Faculty, Admin) and storing incomplete teams' details.
  - **SQLite (SQL):** Used for Core Relational Data (Teams, Projects, Departments, Marks) to ensure strict referential integrity.

- **AI / Search Module:** Sentence-BERT (S-BERT), FastAPI

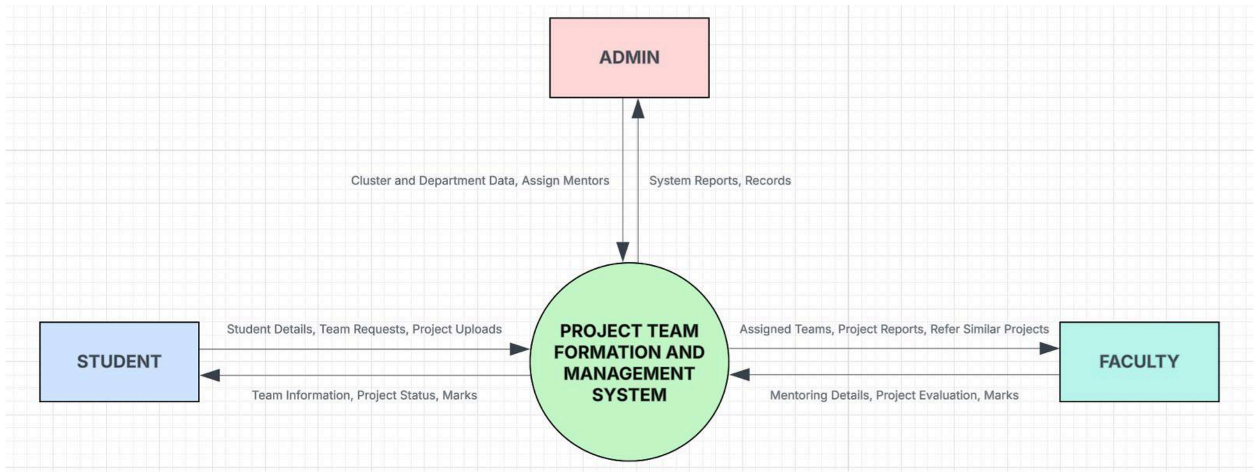
3. System Analysis

Data Flow Diagrams are used to visualize the movement of information within the system, tracing inputs from external entities to the final outputs.

a. Level 0 DFD (Context Diagram)

The Level 0 DFD depicts the **Academic Project Management System** as a single high-level process interacting with three external entities: **Student**, **Faculty**, and **Admin**.

- **Admin:** Inputs Cluster and Department configurations and receives system reports.
- **Student:** Submits registration details, team requests, and project work; receives team status and marks.
- **Faculty:** Submits mentoring details and marks; receives assigned team lists and project reports.

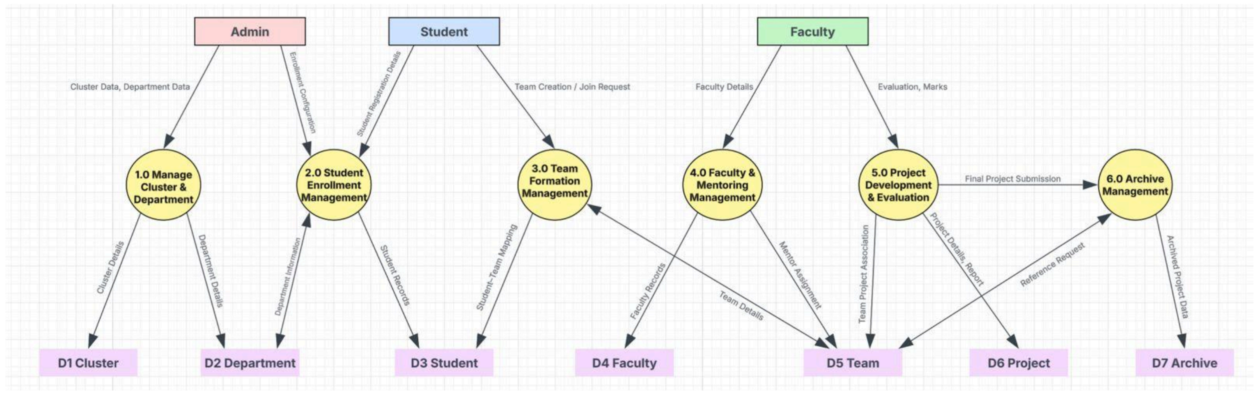

ADMIN
Cluster and Department Data, Assign Mentors
System Reports, Records
STUDENT
Student Details, Team Requests, Project Uploads
Team Information, Project Status, Marks
PROJECT TEAM FORMATION AND MANAGEMENT SYSTEM
Assigned Teams, Project Reports, Refer Similar Projects
Mentoring Details, Project Evaluation, Marks
FACULTY

b. Level 1 DFD (Process Decomposition)

The Level 1 DFD decomposes the system into six core functional modules, illustrating how data moves between processes and the seven specific data stores (D1 to D7).

- **1.0 Manage Cluster & Department:** Handles administrative setup of clusters (D1) and departments (D2).
- **2.0 Student Enrollment Management:** Processes student registration data into the Student table (D3).

- **3.0 Team Formation Management:** Facilitates team creation and joining requests, linking Students (D3) to Teams (D5).
- **4.0 Faculty & Mentoring Management:** Manages faculty records (D4) and assigns mentors to teams (D5).
- **5.0 Project Development & Evaluation:** Tracks project submissions in the Project table (D6) and records faculty evaluations.
- **6.0 Archive Management:** Moves completed project data from active status to the Archive table (D7).

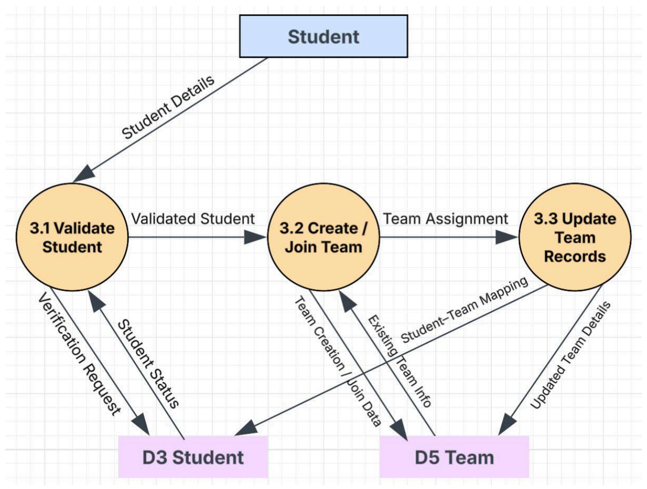


c. Level 2 DFD (Detailed Process Logic)

The Level 2 DFDs provide a granular view of specific complex processes identified in Level 1.

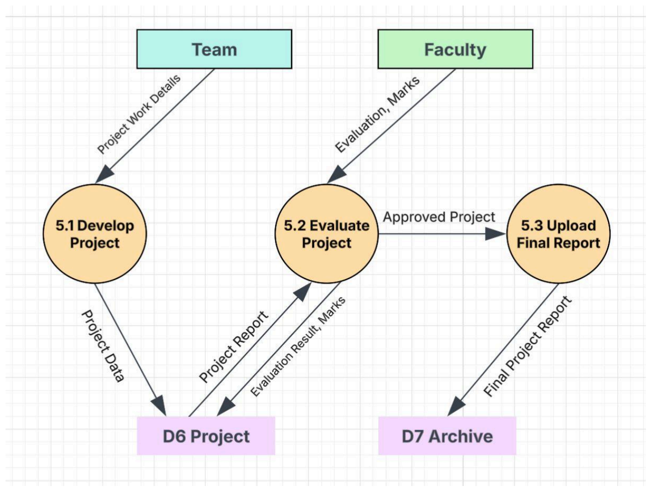
A. Team Formation Logic (Expansion of Process 3.0): This diagram details the specific steps a student takes to become part of a team:

1. **3.1 Validate Student:** The system checks D3 Student to verify eligibility (ensuring the student is not already in a team).
2. **3.2 Create / Join Team:** The validated student data is processed to either generate a new Team ID or request entry to an existing one in D5 Team.
3. **3.3 Update Team Records:** The system finalizes the association, updating the student-team mapping in both data stores.



B. **Project Evaluation & Archival Logic (Expansion of Process 5.0)** This diagram illustrates the workflow from project submission to final archival:

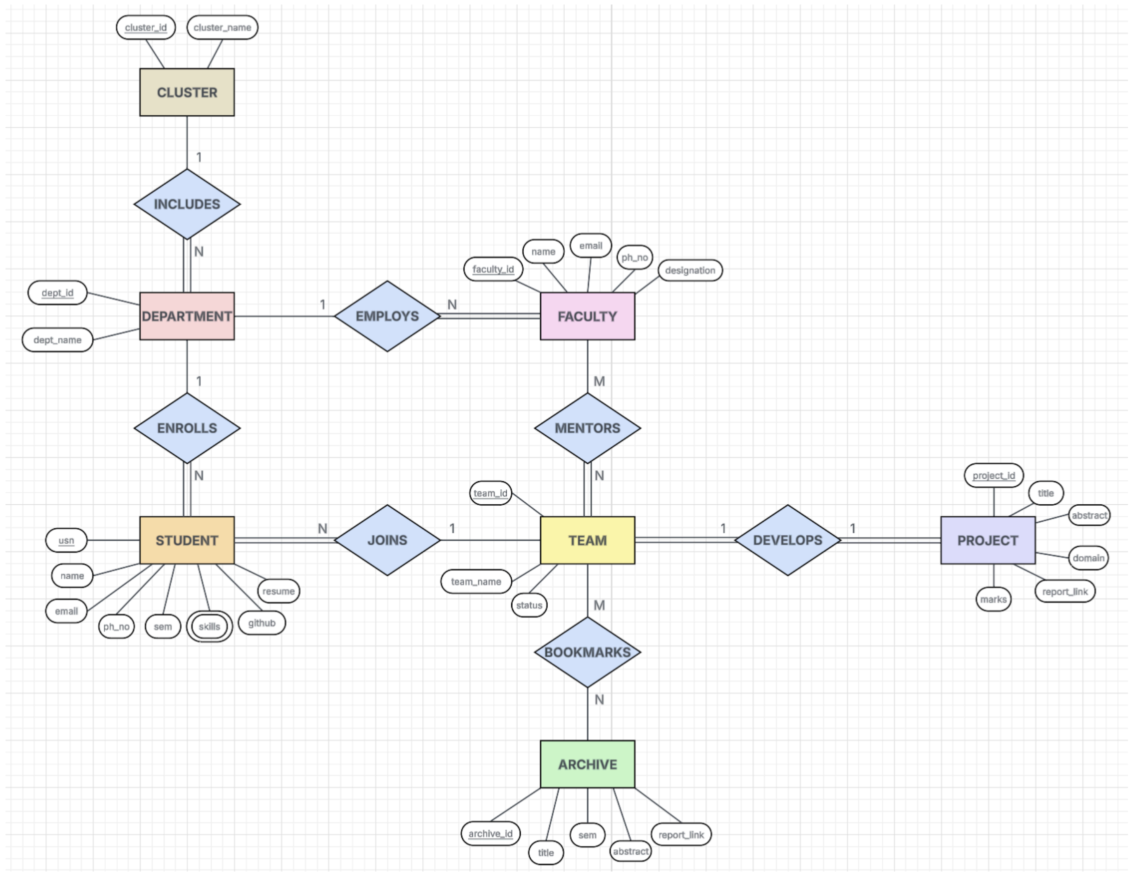
1. **5.1 Develop Project:** Teams submit their project work details, which are stored in D6 Project.
2. **5.2 Evaluate Project:** Faculty access the project data, perform evaluations, and submit marks back to D6 Project.
3. **5.3 Upload Final Report:** Upon successful completion and approval, the final report is transferred to D7 Archive for future reference.



4. Conceptual Design

a. E-R Diagram

The Entity-Relationship (E-R) diagram represents the logical structure of the database, visualizing the entities, attributes, and the relationships between them.



b. Entity Descriptions

The system consists of the following key entities:

- **CLUSTER:** Represents the high-level grouping of related branches (e.g., "Computer Science Cluster" or "Electronics Cluster").
- **DEPARTMENT:** Represents the specific academic branches (e.g., CSE, ISE) responsible for enrolling students and employing faculty.
- **STUDENT:** Stores student details. It includes a multi-valued attribute for **skills** to assist in team formation.
- **FACULTY:** Represents the teaching staff who act as project mentors.
- **TEAM:** The central entity representing a group of students working together.
- **PROJECT:** Represents the active work submitted by a team, including details like title, abstract, and marks.

- **ARCHIVE:** A repository of past projects used for reference and similarity checking.

c. Relationships and Cardinalities

The diagram defines the following constraints and associations:

- **INCLUDES (Cluster ←→ Department): 1:N**.
  - One Cluster includes one or more Departments. A Department belongs to exactly one Cluster.
- **ENROLLS (Department ←→ Student): 1:N**.
  - A Department enrolls multiple Students. Each Student belongs to exactly one Department.
- **EMPLOYS (Department ←→ Faculty): 1:N**.
  - A Department employs multiple Faculty members.
- **JOINS (Student ←→ Team): N:1**.
  - Multiple Students join a single Team.
- **MENTORS (Faculty ←→ Team): M:N**.
  - Faculty members mentor Teams. The M:N cardinality implies that a team could theoretically have co-mentors, or a faculty member mentors multiple teams.
- **DEVELOPS (Team ←→ Project): 1:1**.
  - One Team develops exactly one Project. This is a strict one-to-one relationship.
- **BOOKMARKS (Team ←→ Archive): M:N**.
  - Teams bookmark multiple Archived projects for reference (based on semantic search), and an Archived project can be bookmarked by multiple Teams.

d. Attributes and Keys

Key attributes identified in the design include:

- **Primary Keys (Underlined):** *cluster_id*, *dept_id*, *usn* (Student), *faculty_id*, *team_id*, *project_id*, *archive_id*.
- **Multi-valued Attributes (Double Oval):**
  - **Student Skills:** Captures multiple skills (e.g., Python, React) for a single student.
- **Derived/Link Attributes:**
  - *report_link* in PROJECT and ARCHIVE stores the URL to the document rather than the file itself, optimizing database performance.

5. Logical Database Design

a. Relational Schema

The logical schema represents the final database structure derived from the E-R diagram.

The system is implemented using a relational model with the following table definitions:

- **CLUSTER** (cluster_id [PK], cluster_name)
- **DEPARTMENT** (dept_id [PK], dept_name, cluster_id [FK])
- **STUDENT** (usn [PK], name, email, ph_no, sem, github, resume, dept_id [FK], team_id [FK])
- **FACULTY** (faculty_id [PK], name, email, ph_no, designation, dept_id [FK])
- **TEAM** (team_id [PK], team_name, status)
- **PROJECT** (project_id [PK], title, abstract, domain, report_link, marks, team_id [FK])
- **ARCHIVE** (archive_id [PK], title, sem, abstract, report_link)
- **SKILLS** (usn [FK], skill) *(Composite PK)*
- **MENTORS** (team_id [FK], faculty_id [FK]) *(Composite PK)*
- **BOOKMARKS** (team_id [FK], archive_id [FK]) *(Composite PK)*

b. Normalization Analysis

The database schema has been verified against standard normalization forms:

- **1NF:** Multi-valued attributes (Student Skills) were separated into the SKILLS table to ensure atomicity.
- **2NF:** All tables possess single-attribute primary keys or are junction tables (MENTORS, BOOKMARKS) containing only keys, eliminating partial dependencies.
- **3NF:** Transitive dependencies (e.g., cluster_name depending on Student) were removed by chaining STUDENT → DEPARTMENT → CLUSTER.

6. Conclusion

The **Academic Project Management System** has been successfully designed to address the limitations of manual project tracking in academic institutions. By establishing a structured database schema and defining clear data flows, this project lays the foundation for a centralized platform that streamlines team formation, project submission, and evaluation.

The system analysis, supported by **Level 0, 1, and 2 Data Flow Diagrams (DFDs)**, confirms that the proposed logic effectively handles the complexities of cluster constraints and mentor assignments. Furthermore, the database design has been rigorously normalized to **Third Normal Form (3NF)**, ensuring a scalable and redundancy-free architecture. This design phase provides a robust blueprint for the subsequent implementation of the system, ensuring data integrity and operational efficiency for all stakeholders.