

**RV COLLEGE OF ENGINEERING® , BENGALURU-
560059**

(Autonomous Institution Affiliated to VTU, Belagavi)

**DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING**



College Administration Management System

Mini - Project Report

Submitted by

**Pavan K R
Prasanna Bhat**

**1RV18CS112
1RV18CS116**

in partial fulfillment for the requirement of 5th Semester

DBMS Laboratory Mini Project (18CS53)

Under the Guidance of

**Dr. D Pratiba, Assistant Professor, Computer Science and Engineering,
RV College of Engineering**

Academic Year 2020- 2021

RV COLLEGE OF ENGINEERING®, BENGALURU - 560059
(Autonomous Institution Affiliated to VTU, Belagavi)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

Certified that the project work titled '**College Administration Management System**' is carried out by **Pavan K R and Prasanna Bhat (1RV18CS112, 1RV18CS116)**, who are bonafide students of RV College of Engineering®, Bengaluru, in partial fulfillment of the curriculum requirement of 5th Semester Database Design Laboratory Mini Project during the academic year **2020-2021**. It is certified that all corrections/suggestions indicated for the internal Assessment have been incorporated in the report deposited in the departmental library. The report has been approved as it satisfies the academic requirements in all respect laboratory mini-project work prescribed by the institution.

Signature of Faculty In-charge

Head of the Department
Dept. of CSE, RVCE

External Examination

Name of Examiners

Signature with date

ACKNOWLEDGEMENT

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends. A number of personalities, in their own capacities have helped me in carrying out this project work. I would like to take this opportunity to thank them all.

I deeply express my sincere gratitude to my guide **Internal Guide, Designation**, Department of CSE, RVCE, Bengaluru, for his able guidance, regular source of encouragement and assistance throughout this project

I would like to thank Dr.Ramakanth Kumar P, Head of Department, Computer Science & Engineering, R.V.C.E, Bengaluru, for his valuable suggestions and expert advice.

First and foremost I would like to thank **Dr. Subramanya. K. N**, Principal, R.V.C.E, Bengaluru, for his moral support towards completing my project work.

I thank my Parents, and all the Faculty members of Department of Computer Science & Engineering for their constant support and encouragement.

Last, but not the least, I would like to thank my peers and friends who provided me with valuable suggestions to improve my project.

Abstract

College administration has been traditionally using paper-based forms for various administrative purposes. Form digitization is essential feature that an administration system requires not only to efficiently store the collected forms, and restoring old forms, but also to design forms to be rendered.

This system has a main feature to digitalize the handwritten forms like student registration, exam registration etc. with minimal human work. The web application integrated these features with the college system where student counsellors can store digitized copy of the forms received from the students in a global database and also this can be used by administration department to digitalize plenty of handwritten forms, they receive every day. Also, students can be benefitted by the platform for receiving posts and notifications regarding notes, lecture videos, links, results, internship/placement offers, open consultancy projects etc.

The product aids in a college setting, not only by helping the students to access the materials shared by college and information regarding ongoing campus recruitment offers on an integrated platform, but also helps lecturers for easy creation of posts for their class and better maintenance of counselling records in the college. The OCR has helped the digitization process by identifying handwritten and printed text.

Table of Contents

Acknowledgement	i
Abstract	ii
Table of Contents	iii
List of Figures	vi
Glossary	

Table of Contents

Chapter 1	8
1.1 Motivation:	8
1.2 Existing System:.....	8
1.3 Proposed System:	8
Chapter 2	9
2.1 Software requirements.....	9
2.1 Hardware Requirements	9
2.2 Functional Requirements.....	9
2.2.1 Authentication Module:	9
2.2.2 Class Posts:	9
2.2.3 Department Posts:	9
2.2.4 Authorize users:	9
2.2.5 Apply for internship/placement offers	10
2.2.6 Form digitization.....	10
Chapter 3	11
3.1 Entities.....	11
3.2 Weak entities	11
3.3 Relations.....	11
Chapter 4	13
4.1 DFD 0.1	13
4.1.1 Processes:	13
4.1.2 Entities:	13
4.2 DFD Level 1	14
4.2.1 DFD 1.1.....	15
4.2.2 DFD 1.2.....	16
4.2.3 DFD 1.3.....	17
4.2.4 DFD 1.4.....	18

Chapter 5	19
5.1 Schema diagram	20
5.2 Normalization.....	21
5.2.1 Student	21
5.2.2 Department.....	21
5.2.3 Lecturer	21
5.2.4 Internship	21
5.2.5 Department Posts	22
5.2.6 Class Posts	22
5.2.7 Class.....	22
5.2.8 Course	22
5.2.9 Class Lecturer	23
5.2.10 SPC	23
5.2.11 Offer eligibility	23
5.2.12 Application.....	24
Chapter 6.....	25
6.1 Integrating NOSQL database and SQL Database	25
6.1.1 SQL Component:	25
6.1.2 NoSQL Component:	25
6.1.3 Integration	26
Chapter 7.....	28
Conclusion & Future Enhancement	28

List of Figures

Figure 1 ER diagram	12
Figure 2 DFD 0.1	13
Figure 3 DFD 1.1	14
Figure 4 DFD 1.2	16
Figure 5 DFD 1.3	17
Figure 6 DFD 1.4	18
Figure 7 Relational Schema	20
Figure 8 System architecture	25
Figure 9 snapshot of relational database	26
Figure 10 snapshot of NoSQL document	27
Figure 11 Pre-Login page	30
Figure 12 Admin-add student	30
Figure 13 Admin-add course	31
Figure 14 Admin-add teacher	31
Figure 15 Admin-register student for course	31
Figure 16 Lecturer(counselor)-digitize form	32
Figure 17 Lecturer-add class post	32
Figure 18 Lecturer(counselor)-get digitized forms	32
Figure 19 Student-home page	33
Figure 20 Student-login	33
Figure 21 Student-class notice board	33
Figure 22 Student-apply for internship	34
Figure 23 Student-department notice board	34
Figure 24 Student(SPC)-add company	34

GLOSSARY

OCR : Optical character recognition or optical character reader (OCR) is the electronic or mechanical conversion of images of typed, handwritten or printed text into machine-encoded text.

SRS : Software Requirement Specification

Chapter 1

Introduction

This system has a main feature to digitalize the handwritten forms like student registration, exam registration etc with minimal human work. The web application integrated these features with the college system where student counsellors can store digitized copy of the forms received from the students in a global database and also this can be used by administration department to digitalize plenty of handwritten forms, they receive every day. Also, students can be benefitted by the platform as all the important notifications about companies visiting for placement and internship and any other important notifications regarding open consultancy projects can be put here

1.1 Motivation:

This product is useful for college administration, which enables the college management to handle the functions with ease. The goal of the product is to improve existing mechanism in college for students to receive notifications, notes and other class related posts and internship/placement opportunities, results, etc. department level posts. Also, the process of form collection and maintaining which is tedious is eliminated by form digitization and OCR.

1.2 Existing System:

For getting the information from various forms to a proper database data entry is done which requires human resources. The information regarding various companies visiting the campus is communicated via WhatsApp but it can be made more centralized. All notifications and notes reach students via email or google classroom or quiklrn which can be made available from a single point which is easy for Lectures and Students to access.

1.3 Proposed System:

This product is mainly designed to replace an existing system of manual entry for forms and help students access notes, notifications, posts and apply for offers on campus easily. The form digitization feature involves recognition of elements of a form and identify the text. The application provides a platform for the lectures and department to share posts with the students. There by students no longer have to rely on several other portals and search for shared material and information.

Chapter 2

Software Requirement specification

The software requirements for the College Administration Management System are listed below. The software and hardware requirements specify the requirement for the working of the software and the functional requirements highlight the requirements for expected working of the software.

2.1 Software requirements

2.1.1 Operating system: Windows 7 and above, Linux, Mac OS

2.1.2 Programming languages: Python, JavaScript, NodeJS

2.1.3 Front End: React, HTML, CSS

2.1.4 Back End: NodeJS, Python

2.1.5 Databases: MySQL, MongoDB

2.1 Hardware Requirements

2.2.1 Processor: 64-bit, 2 cores, Pentium IV or higher

2.2.2 Processor Speed: 2.5GHz minimum per core

2.2.3 RAM: 2GB or higher

2.2.4 Hard disk: 2 GB free space for installation. For production use additional disk space for day-to-day operations.

2.2 Functional Requirements

2.2.1 Authentication Module:

2.2.1.1 User login: The users can login using the college email id and password provided by the admin.

2.2.1.2 Password: The users can change their passwords any time.

2.2.1.3 One-time login: Once the user logs in the server should provide a JWT storing it in the browser, the user should be able to access the portal without re-entering credentials.

2.2.2 Class Posts:

2.2.2.1 Post: Concerned class lecturers of the class handling the course can post.

2.2.2.2 Notify: All students belonging to that class will be notified.

2.2.2.3 Format: Supports different formats (links, videos, pdf's, doc, message, etc.)

2.2.3 Department Posts:

2.2.3.1 Post: HOD of the department and concerned department SPC's can post.

2.2.3.2 Notify: All students belonging to that class will be notified.

2.2.3.3 Link: Can link to Internship/Placement application

2.2.4 Authorize users:

2.2.5 Admin: can add students, courses, lectures, HOD's, Counsellors, SPC's.

2.2.4.1 Authorize: adds email id's and passwords to each user of the application.

2.2.5 Apply for internship/placement offers

2.2.5.1 Create: Offers can be put up by the placement department or the SPC's of the respective dept.

2.2.5.2 Filter: Should be received by students belonging to the eligible departments as specified by the company and meets all the basic criteria like minimum CGPA, backlogs, etc.

2.2.5.3 Apply: gives the flexibility for the students to apply or withdraw applications for any offer till the last date.

2.2.6 Form digitization

2.2.6.1 Image-input: User should be able to upload scanned images of documents.

2.2.6.2 Output: Digitizes the document and produces metadata in json form.

2.2.6.3 Counselling records: Organizes the student forms in the form of counselling records and store it in the database.

2.2.6.4 Design: Administration can design forms and create forms out of sketches.

Chapter 3

ER Diagram

The important [entities](#) are identified below in our problem statement. The associated [weak entities](#) are also identified. The [relations](#) identified among the entities are identified, along with their participation ratio and cardinality. The identified entities, weak entities and relations are transformed into the [ER diagram](#).

3.1 Entities

- 3.1.1 Department (Name varchar (50), Department_ID int(10), Department_CODE varchar(10))
- 3.1.2 Student (Name varchar (10), Student_ID int(10), Email varchar(10), Backlog float(5))
- 3.1.3 Lecturer (Name varchar (10), Lecturer_ID int(10), Email varchar(10))
- 3.1.4 Internship/Placement Offer (Date date, Company varchar(50), cutoff_CGPA float(5), Role varchar(20), Stipend int(10), Year int(10), Profile varchar(20), backlog float(5))
- 3.1.5 Post (Post_ID int(10), Date date, Title varchar(50), Year int(1), Department/Class_ID int(10), Message varchar(100))
- 3.1.6 Course (Course_ID int(10), Title varchar(50), CODE varchar(100))

3.2 Weak entities

- 1.3.1 Class (Class_ID int(10), Year int(1))

3.3 Relations

- 1.3.2 Class - Belongs_to – Department
- 1.3.3 Student - Belongs_to – Class
- 1.3.4 Student - Point_of_contact – Department
- 1.3.5 Lecturer - Belongs_to – Department
- 1.3.6 Department – Headed_by – Lecturer
- 1.3.7 Student – Counseled_by – Lecturer
- 1.3.8 Department – offers – Course
- 1.3.9 Student – Applies_for Internship/Placement_Offer
- 1.3.10 Department – Eligible_for Internship/Placement_Offer
- 1.3.11 Lecturer – teaches – Course
- 1.3.12 Lecturer, Course – class_posts – Post
- 1.3.13 Department – posts – Post

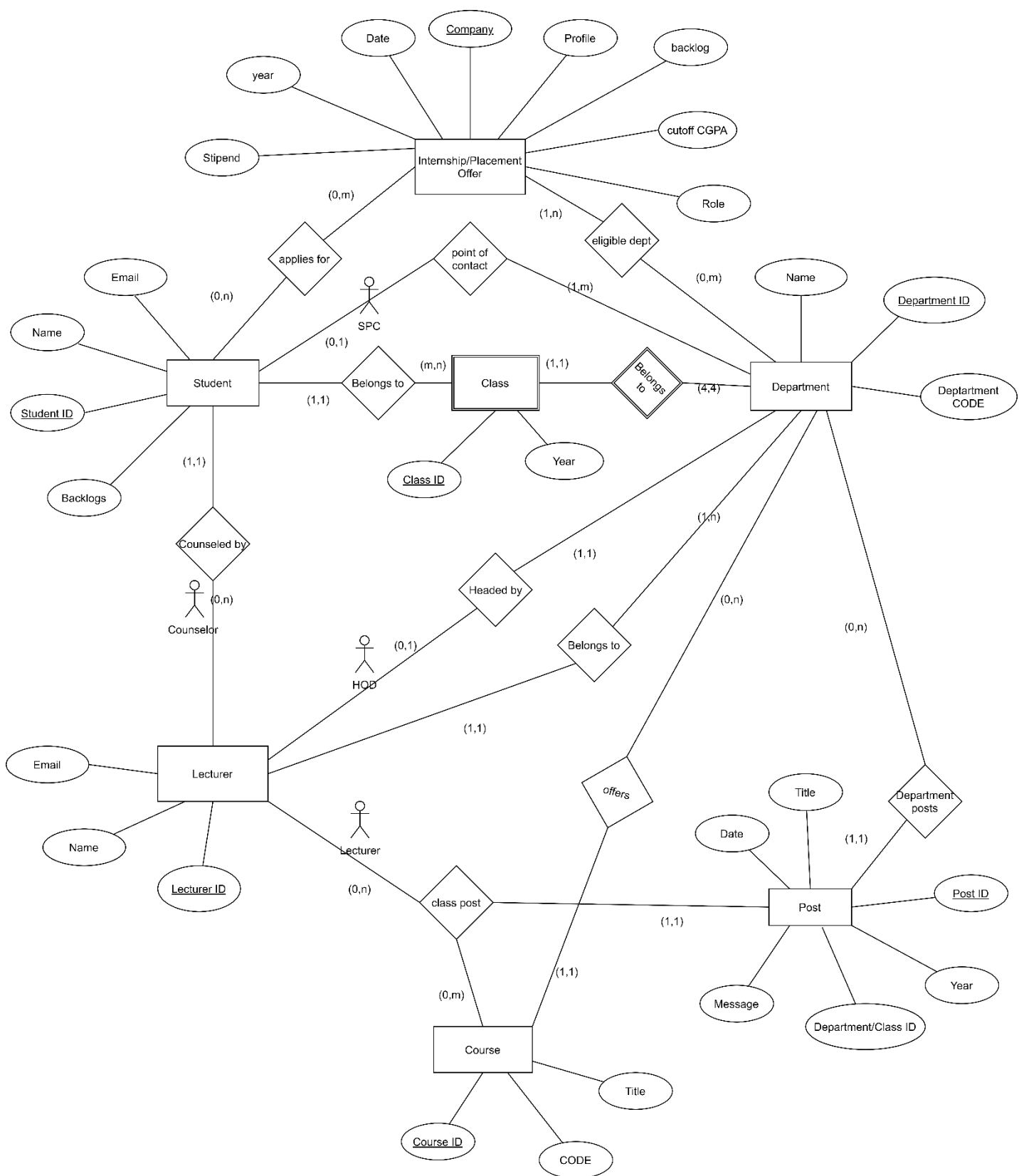


Figure 1 ER diagram

Chapter 4

Detailed Design

The design details of the College Administration Management System are described below. The [DFD Level 0](#) (context diagram) describes the abstract view. It shows the College Administration Management System as a single process College management with its relations with the external entities.

The [DFD Level 1](#) explains the context diagram decomposed into multiple processes. In this level, we highlight the main functions of the College Administration Management System and breakdown the high-level process of level-0 DFD into subprocesses. The level-1 DFD explains the role of different Entities, involved in various processes and their interaction with the database.

4.1 DFD 0.1

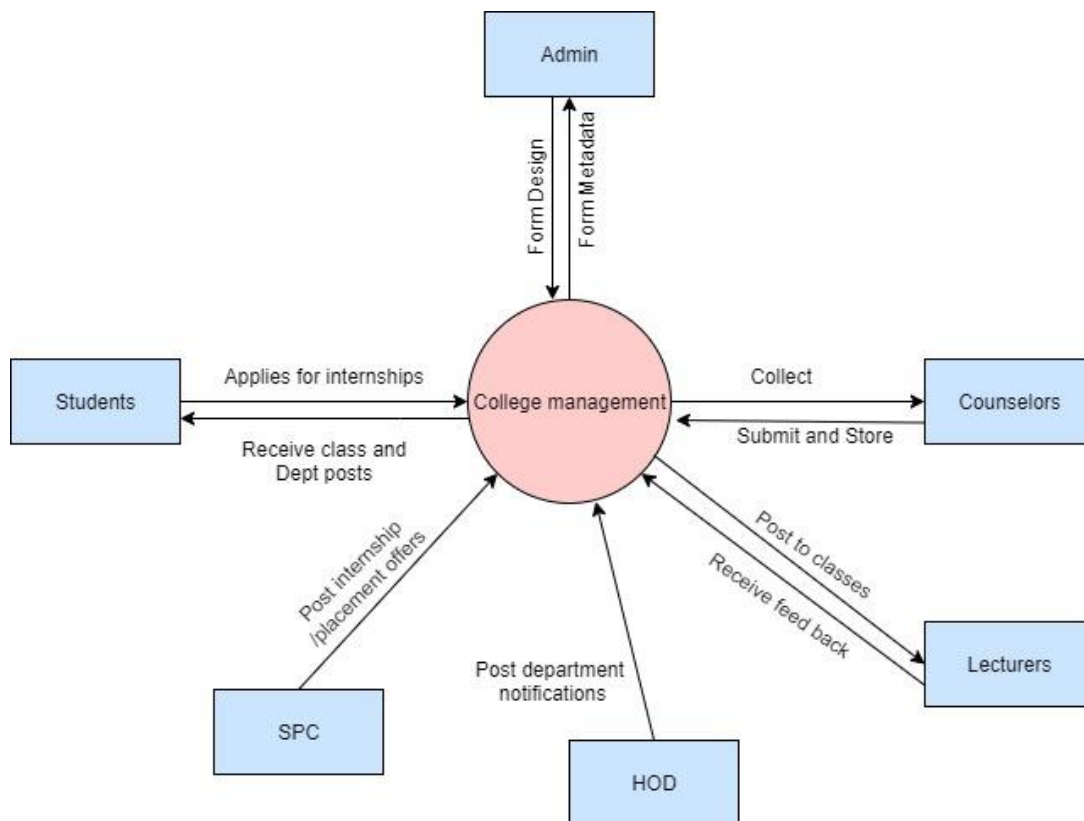


Figure 2 DFD 0.1

4.1.1 Processes:

4.1.1.1 College management: The process that handles the college management in association with admin, instructors and students.

4.1.2 Entities:

4.1.2.1 Students: One of the main stakeholders for the application, that uses the application for accessing the class and department posts, getting updates and applying for internship/placement offers on campus.

- 4.1.2.2 SPC:** Their responsibility is to update students regarding the internship/placement offers on campus, handling applicants.
- 4.1.2.3 HOD:** They are department instructors who are designated as head of department, who can post department notifications.
- 4.1.2.4 Lecturers:** They handle courses for students, they can post course related posts.
- 4.1.2.5 Counselors:** They handle counseling records from students. They can digitize, save and retrieve forms.
- 4.1.2.6 Admin:** They enter the student, course and instructor details to the database and also use the form designing feature.

4.2 DFD Level 1

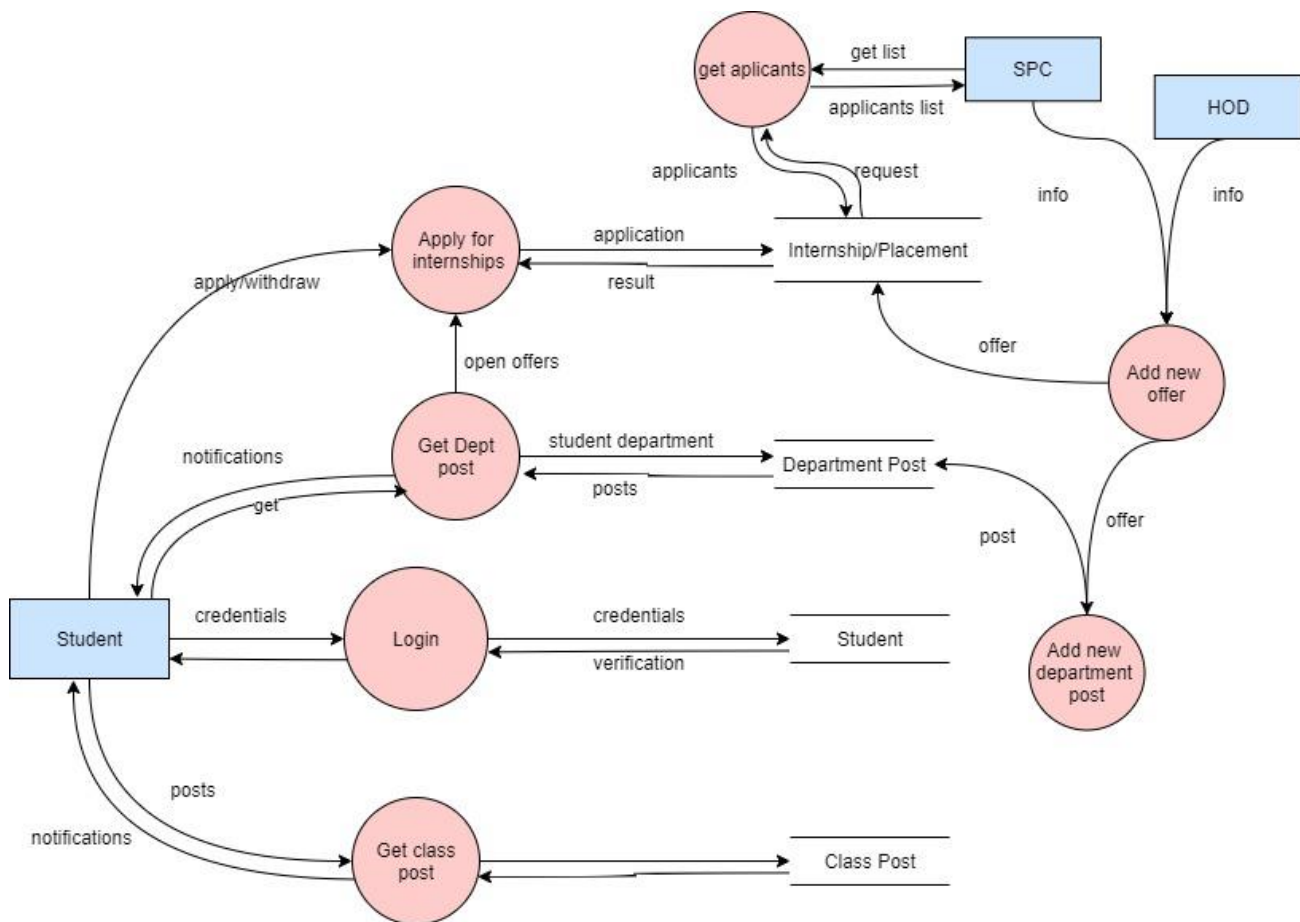


Figure 3 DFD 1.1

4.2.1 DFD 1.1

4.2.1.1 Processes:

- 4.2.1.1.1 **Apply for internships:** accept student applications and withdrawals, gets information from offers database and inserts/deletes student applications.
- 4.2.1.1.2 **Get Dept post:** retrieves student's department related posts from the datastore and notifies students.
- 4.2.1.1.3 **Login:** authorizes students to the application with their college email id and password.
- 4.2.1.1.4 **Get class post:** retrieves student's class/course related posts from the datastore and notifies students.
- 4.2.1.1.5 **Add new department post:** adds new posts to department posts.
- 4.2.1.1.6 **Add new offer:** adds new offer to placement/internship datastore.

4.2.1.2 Entities:

- 4.2.1.2.1 **Student:** The user can access class and department posts, apply for internship/placement offers.
- 4.2.1.2.2 **SPC:** Adds internship/placement offers using offer information.
- 4.2.1.2.3 **HOD:** They are department instructors who are designated as head of department, who can post department notifications.

4.2.1.3 Datastore:

- 4.2.1.3.1 **Student:** stores all student information including authorization, grades, class etc.
- 4.2.1.3.2 **Class Post:** contains class post information.
- 4.2.1.3.3 **Department Post:** contains department post information.
- 4.2.1.3.4 **Internship/Placement:** contains campus offer information including applicants list.

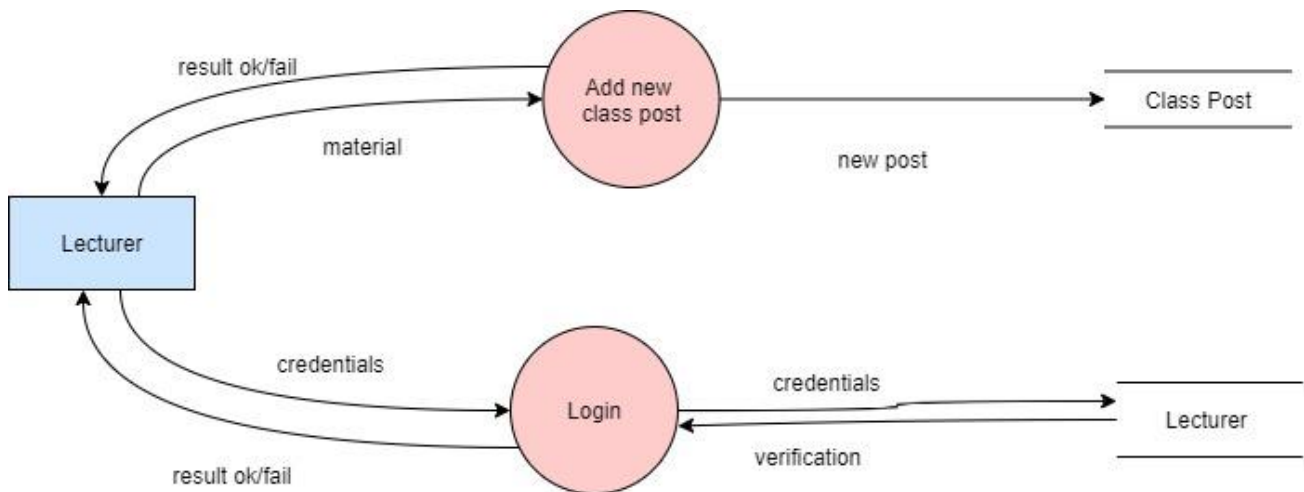


Figure 4 DFD 1.2

4.2.2 DFD 1.2

4.2.2.1 Processes:

4.2.2.1.1 **Login:** authorizes students to the application with their college email id and password.

4.2.2.1.2 **Add new class post:** adds new class materials as a class post.

4.2.2.2 Entities:

4.2.2.2.1 **Lecturer:** The user can access the application to send lecture materials and other information to the students.

4.2.2.3 Datastore:

4.2.2.3.1 **Lecturer:** stores all lecturer information including authorization details and course details.

4.2.2.3.2 **Class Post:** contains class post information including class identity and material information.

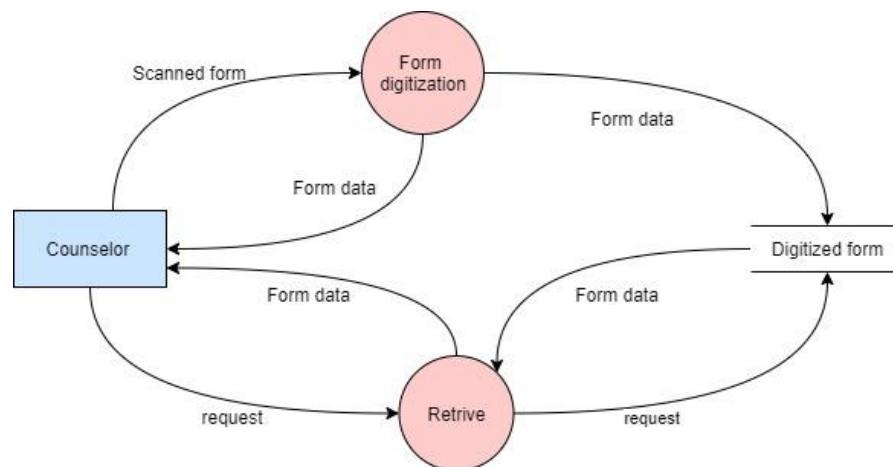


Figure 5 DFD 1.3

4.2.3 DFD 1.3

4.2.3.1 Processes:

4.2.3.1.1 **Form digitization:** The counselors can collect printed forms and digitize the documents to convert the data into digital form and store them.

4.2.3.1.2 **Retrieve:** Get digitized forms from the database.

4.2.3.2 Entities:

4.2.3.2.1 **Counselor:** Counselor of each student collects forms from students and maintains counseling records.

4.2.3.3 Datastore:

4.2.3.3.1 **Digitized forms:** Stores digitized forms corresponding to each lecturer and retrieves it when required.

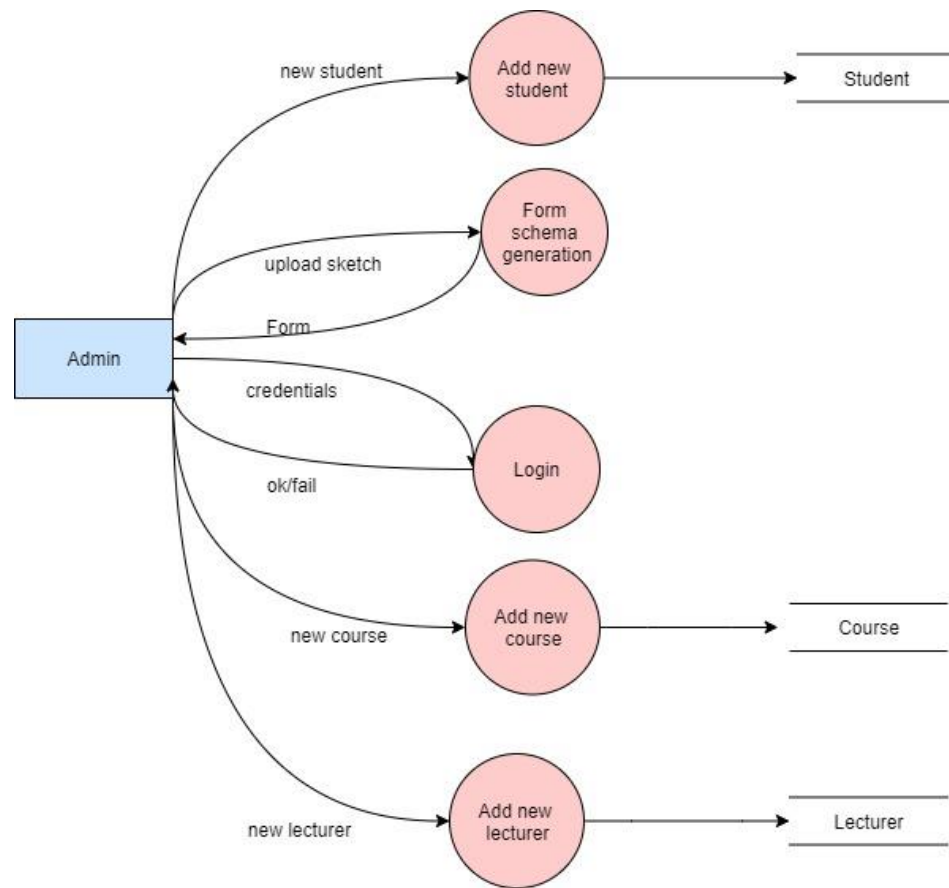


Figure 6 DFD 1.4

4.2.4 DFD 1.4

4.2.4.1 Processes:

- 4.2.4.1.1 **Add new student:** adds student information and assign credentials.
- 4.2.4.1.2 **Form schema generation:** generates form from drawn design.
- 4.2.4.1.3 **Login:** authorizes admin user through password.
- 4.2.4.1.4 **Add new course:** adds new course information.
- 4.2.4.1.5 **Add new lecturer:** adds lecturer information and provide credentials.

4.2.4.2 Entities:

- 4.2.4.2.1 **Admin:** The user can access the application to add user data, course information and for form designing.

4.2.4.3 Datastore:

- 4.2.4.3.1 **Student:** stores all student information.
- 4.2.4.3.2 **Course:** stores all course information.
- 4.2.4.3.3 **Lecturer:** stores all lecturer information.

Chapter 5

Relational schema and Normalization

The section describes the schema mapped from the [ER diagram](#) explained in previous sections. The Relational Schema is the result of 7 step mapping of the ER diagram. The Schema diagram can be found in section [5.1](#).

Further, in section [5.2](#), we identify functional dependencies of each relation in the Relational schema and normalize the relations with respect to the identified functional dependencies, to a maximum of BCNF. Thus, the Relational Schema of the Database satisfies 1NF, 2NF, 3NF and BCNF. The justification for each normal form can be found below.

5.1 Schema diagram

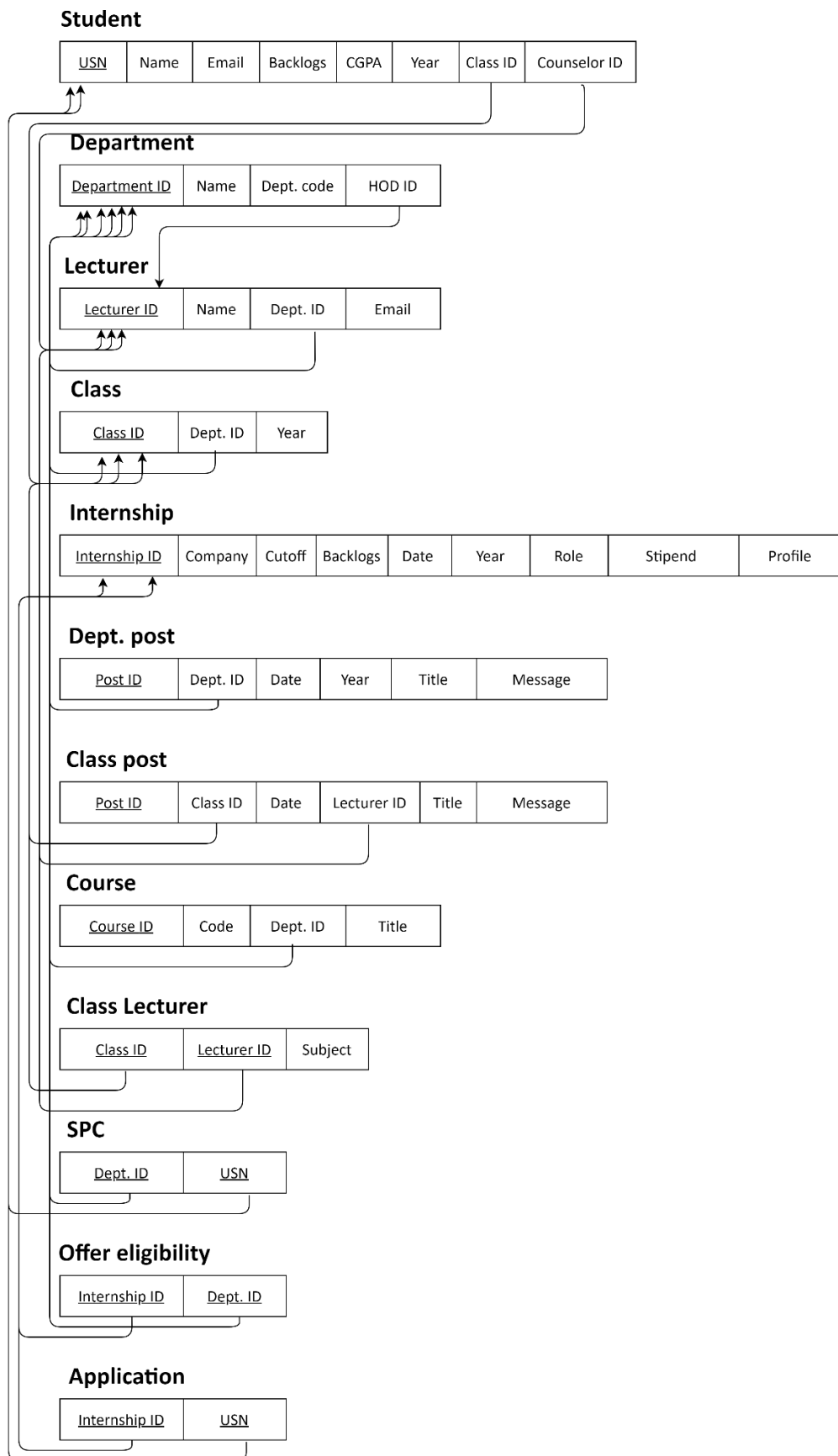


Figure 7 Relational Schema

5.2 Normalization

5.2.1 Student

Functional dependencies:

USN \rightarrow Name, Email, Backlogs, CGPA, Year, Class ID, Counselor ID

1NF: The relation does not contain any multivalued or composite attribute. Hence it is in first normal form.

2NF: The relation is in 1NF and there is no partial dependency, i.e, no non-prime attribute is dependent on any proper subset of any candidate key of the table as only one attribute is present in LHS.

3NF: The relation is in 2NF and there is no transitive dependency for non-prime attributes.

Also, USN is the super key for the relation.

BCNF: The relation is in 3NF and USN is the super key.

5.2.2 Department

Functional dependencies:

Department ID \rightarrow Name, Dept code, HOD ID

1NF: The relation does not contain any multivalued or composite attribute. Hence it is in first normal form.

2NF: The relation is in 1NF and there is no partial dependency, i.e, no non-prime attribute is dependent on any proper subset of any candidate key of the table as only one attribute is present in LHS.

3NF: The relation is in 2NF and there is no transitive dependency for non-prime attributes.

Also, USN is the super key for the relation.

BCNF: The relation is in 3NF and USN is the super key.

5.2.3 Lecturer

Functional dependencies:

Lecturer ID \rightarrow Dept. ID, Name, Email

1NF: The relation does not contain any multivalued or composite attribute. Hence it is in first normal form.

2NF: The relation is in 1NF and there is no partial dependency, i.e, no non-prime attribute is dependent on any proper subset of any candidate key of the table as only one attribute is present in LHS.

3NF: The relation is in 2NF and there is no transitive dependency for non-prime attributes.

Also, USN is the super key for the relation.

BCNF: The relation is in 3NF and USN is the super key.

5.2.4 Internship

Functional dependencies:

Internship ID \rightarrow Company, Cutoff, Date, Role, Backlog, Stipend, Profile

1NF: The relation does not contain any multivalued or composite attribute. Hence it is in first normal form.

2NF: The relation is in 1NF and there is no partial dependency, i.e, no non-prime attribute is dependent on any proper subset of any candidate key of the table as only one attribute is present in LHS.

3NF: The relation is in 2NF and there is no transitive dependency for non-prime attributes.

Also, USN is the super key for the relation.

BCNF: The relation is in 3NF and USN is the super key.

5.2.5 Department Posts

Functional dependencies:

Post ID \rightarrow Dept. ID, Year, Date, Title, Message

1NF: The relation does not contain any multivalued or composite attribute. Hence it is in first normal form.

2NF: The relation is in 1NF and there is no partial dependency, i.e, no non-prime attribute is dependent on any proper subset of any candidate key of the table as only one attribute is present in LHS.

3NF: The relation is in 2NF and there is no transitive dependency for non-prime attributes.

Also, USN is the super key for the relation.

BCNF: The relation is in 3NF and USN is the super key.

5.2.6 Class Posts

Functional dependencies:

Post ID \rightarrow Class ID, Lecturer, Date, Title, Message

1NF: The relation does not contain any multivalued or composite attribute. Hence it is in first normal form.

2NF: The relation is in 1NF and there is no partial dependency, i.e, no non-prime attribute is dependent on any proper subset of any candidate key of the table as only one attribute is present in LHS.

3NF: The relation is in 2NF and there is no transitive dependency for non-prime attributes.

Also, USN is the super key for the relation.

BCNF: The relation is in 3NF and USN is the super key.

5.2.7 Class

Functional dependencies:

Class ID \rightarrow Dept_ID, Year

1NF: The relation does not contain any multivalued or composite attribute. Hence it is in first normal form.

2NF: The relation is in 1NF and there is no partial dependency, i.e, no non-prime attribute is dependent on any proper subset of any candidate key of the table as only one attribute is present in LHS.

3NF: The relation is in 2NF and there is no transitive dependency for non-prime attributes.

Also, USN is the super key for the relation.

BCNF: The relation is in 3NF and USN is the super key.

5.2.8 Course

Functional dependencies:

Course ID \rightarrow Dept_ID, Code, Title

1NF: The relation does not contain any multivalued or composite attribute. Hence it is in first normal form.

2NF: The relation is in 1NF and there is no partial dependency, i.e, no non-prime attribute is dependent on any proper subset of any candidate key of the table as only one attribute is present in LHS.

3NF: The relation is in 2NF and there is no transitive dependency for non-prime attributes. Also, USN is the super key for the relation.

BCNF: The relation is in 3NF and USN is the super key.

5.2.9 Class Lecturer

Functional dependencies:

Class ID, Lecturer ID \rightarrow Subject

1NF: The relation does not contain any multivalued or composite attribute. Hence it is in first normal form.

2NF: The relation is in 1NF and there is no partial dependency, i.e, no non-prime attribute is dependent on any proper subset of any candidate key of the table as only one attribute is present in LHS.

3NF: The relation is in 2NF and there is no transitive dependency for non-prime attributes. Also, USN is the super key for the relation.

BCNF: The relation is in 3NF and USN is the super key.

5.2.10 SPC

Functional dependencies:

Department ID, USN \rightarrow {}

1NF: The relation does not contain any multivalued or composite attribute. Hence it is in first normal form.

2NF: The relation is in 1NF and there is no partial dependency, i.e, no non-prime attribute is dependent on any proper subset of any candidate key of the table as only one attribute is present in LHS.

3NF: The relation is in 2NF and there is no transitive dependency for non-prime attributes. Also, USN is the super key for the relation.

BCNF: The relation is in 3NF and USN is the super key.

5.2.11 Offer eligibility

Functional dependencies:

Internship ID, Department ID \rightarrow {}

1NF: The relation does not contain any multivalued or composite attribute. Hence it is in first normal form.

2NF: The relation is in 1NF and there is no partial dependency, i.e, no non-prime attribute is dependent on any proper subset of any candidate key of the table as only one attribute is present in LHS.

3NF: The relation is in 2NF and there is no transitive dependency for non-prime attributes. Also, USN is the super key for the relation.

BCNF: The relation is in 3NF and USN is the super key.

5.2.12 Application

Functional dependencies:

Internship ID, USN $\rightarrow \{\}$

1NF: The relation does not contain any multivalued or composite attribute. Hence it is in first normal form.

2NF: The relation is in 1NF and there is no partial dependency, i.e., no non-prime attribute is dependent on any proper subset of any candidate key of the table as only one attribute is present in LHS.

3NF: The relation is in 2NF and there is no transitive dependency for non-prime attributes. Also, USN is the super key for the relation.

BCNF: The relation is in 3NF and USN is the super key.

Chapter 6

NOSQL

This section describes the integration of SQL and NOSQL database with the application.

6.1 Integrating NOSQL database and SQL Database

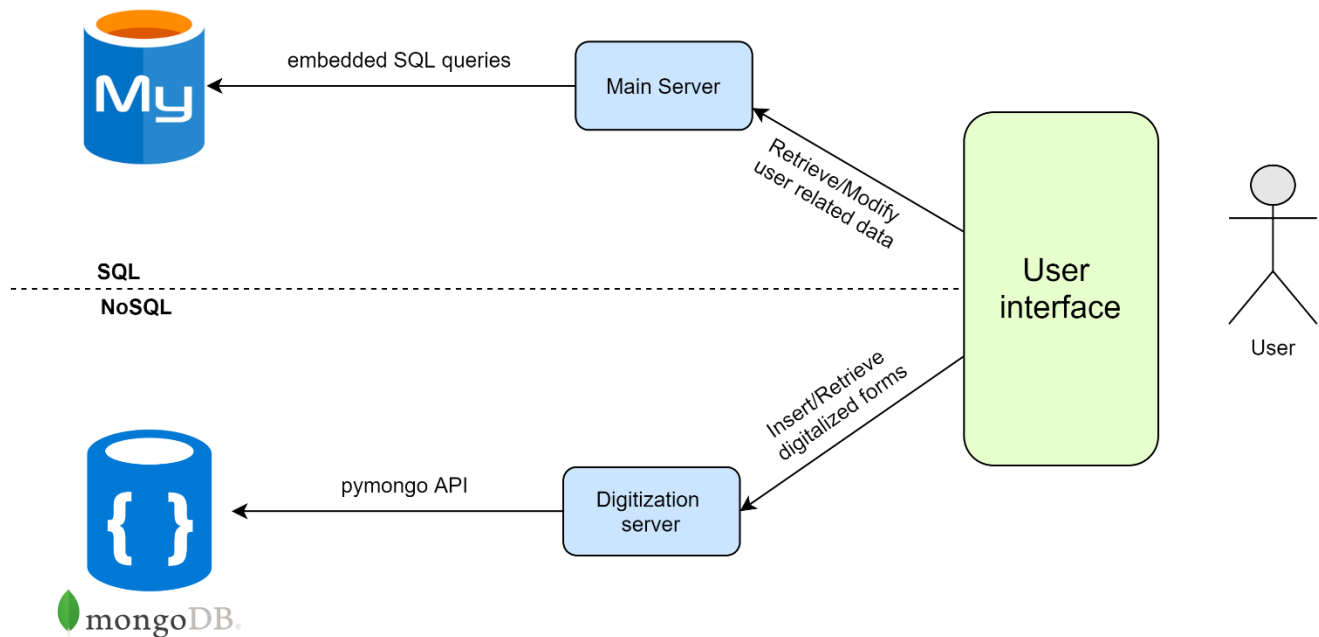


Figure 8 System architecture

6.1.1 SQL Component:

This component has a storage engine that manages data storage in MySQL DB. MySQL has stand-alone clients that allow users to interact directly with a MySQL database using SQL, but more often MySQL is used with other programs to implement applications that need relational database capability. MySQL is written in C and C++. Its SQL parser is written in yacc, but it uses a home-brewed lexical analyzer. It has also been tested to be a "fast, stable and true multi-user, multi-threaded SQL database server".

The storage engine is composed of a transactional log file and data filegroups that can be orderly divided into data files, tables, indexes, extent, and page. The transaction log file is utilized to attain data integrity and data recovery. The beginning and end of each operation and all modifications done are recorded in the transaction log file.

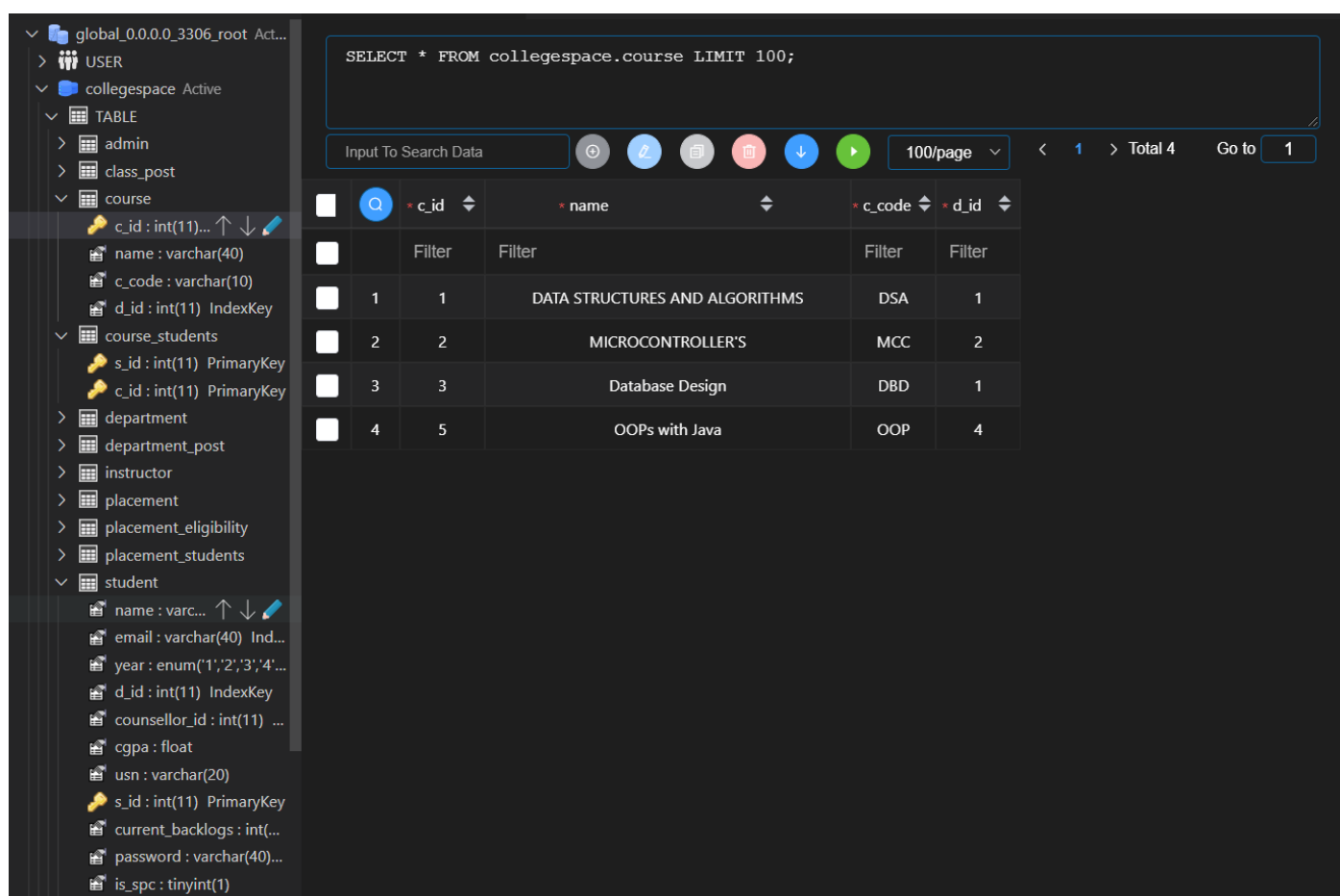
6.1.2 NoSQL Component:

This component is managed by MongoDB. It is one of the most powerful NoSQL systems and databases around, today. It does not use the usual rows and columns that you so much associate with the relational database management. It is an architecture that is built on collections and documents. The basic unit of data in this database consists of a set of key-value pairs. It allows documents to have different fields and structures.

This database uses a document storage format called BSON which is a binary style of JSON documents. The data model that MongoDB follows is a highly elastic one that lets you combine and store data of multivariate types without having to compromise on the powerful indexing options, data access, and validation rules.

6.1.3 Integration

The system architecture is illustrated in System architecture. The MySQL database is integrated with the application with the help of a backend server. The backend server being written in NodeJS, uses MySQL with the help of library “mysql” written in NodeJS, which supports embedded SQL queries for querying the MySQL database. The application uses the MySQL database for storing the relational database. The relational database consists of tables where each table represents a relation from the relational mapping explained in previous sections. Upon starting the application, the frontend sends GET requests to the backend which in turn queries the database to populate the data into the application. When data has to be inserted the front send post requests to the database with the data to be insert, which is then inserted into the database with the help of SQL queries. Modification of the data also works in similar fashion. A snapshot of relational database state is attached below.



The screenshot shows a database management interface. On the left is a sidebar with a tree view of the database structure, including tables like 'course', 'course_students', 'department', etc. The main area displays a SQL query: `SELECT * FROM collegespace.course LIMIT 100;`. Below the query is a search bar and a table of results. The table has columns: `c_id`, `name`, `c_code`, and `d_id`. The results show 4 rows of data.

<code>c_id</code>	<code>name</code>	<code>c_code</code>	<code>d_id</code>
1	DATA STRUCTURES AND ALGORITHMS	DSA	1
2	MICROCONTROLLER'S	MCC	2
3	Database Design	DBD	1
4	OOPs with Java	OOP	4

Figure 9 snapshot of relational database

The MongoDB is integrated with the application for the purpose of storing and retrieving digitized forms. Digitized forms have no definite schema and thus a NoSQL DB like MongoDB will be appropriate.

For digitization, when a user uploads a scanned image of the form, the application sends a post request with the image data to flask server, which has exposed a digitization API. The server processes the image and extracts the required form data and sends it to the MongoDB with the help of pymongo, a python library, which provides API's to query the MongoDB. At the same time, it also sends a success message to the client.

For retrieving the forms from the MongoDB, the server exposes another API for requesting stored digitized form of a particular counselor, through a get request. The server further requests for data from the MongoDB and returns it to the client end. The snapshot of MongoDB online server is attached below.

```
QUERY RESULTS 1-3 OF 3

{
  "_id": ObjectId("5fe82c1a5f662e795cebae6b"),
  "counselor_id": 2,
  "data": Object
}

{
  "_id": ObjectId("5fe838435f662e795cebae71"),
  "counselor_id": 1,
  "data": Object
}

{
  "_id": ObjectId("5fe839355f662e795cebae72"),
  "counselor_id": 1,
  "data": Object
  {
    "Name of the Student (In Capitals)": "Prasanna Bhat",
    "Branch (In Capitals)": "Computer Science and Engineering",
    "USN": "IRV18CS116",
    "Mobile Number": "6362474863",
    "Gender": "Male",
    "Blood Group": "A+",
    "Category": "GM",
    "Name of the Counselor(In Capitals)": "Sandhya S",
    "RVCE Mail ID": "sandhya.sampangi@rvce.edu.in",
    "Father Name & Contact Number": "Mahesh Bhat",
    "Mother Name & Contact Number": "Uma Bhat",
    "Permanent Address (In Capitals)::": "F7 Malhar Apartment Mahadwar Road 5th cross Belgaum",
    "Pincode": "590001"
  }
}
```

Figure 10 snapshot of NoSQL document

Chapter 7

Conclusion & Future Enhancement

This system aims at digitizing documents which helps save time and redundant manual effort. It uses the organized nature of a form by which an ML algorithm can detect its components. It also uses OCR to detect computer printed text as well as handwritten data.

Further the system can be extended by integrating with other college management systems like the Quiklrn, there by making the notes, tests, quizzes, etc., all available on an integrated platform to the students easily. Also grading and maintaining student records becomes easier.

With respect to the enhancement of existing features, the OCR can be relaced by a commercial grade OCR and the ML model used for detecting objects could be extended for custom object detection. The posts can be more elegant by displaying the contents as image. The placement section can also be improved to show the statistics and live news.

References

1. <https://www.w3schools.com/sql/>
2. [How to Query a MongoDB Database using PyMongo in Python? \(analyticsvidhya.com\)](https://analyticsvidhya.com/how-to-query-a-mongodb-database-using-pymongo-in-python/)
3. [Welcome to Flask — Flask Documentation \(1.1.x\) \(palletsprojects.com\)](https://palletsprojects.com/en/1.1.x/welcome/)
4. [MongoDB Tutorial - Tutorialspoint](https://www.tutorialspoint.com/mongodb/index.htm)
5. Fundamentals of Database Systems, 7th Edition, Ramez Elmasri, University of Texas at Arlington, Shamkant B. Navathe, University of Texas at Arlington

Appendix

Screenshots with descriptions

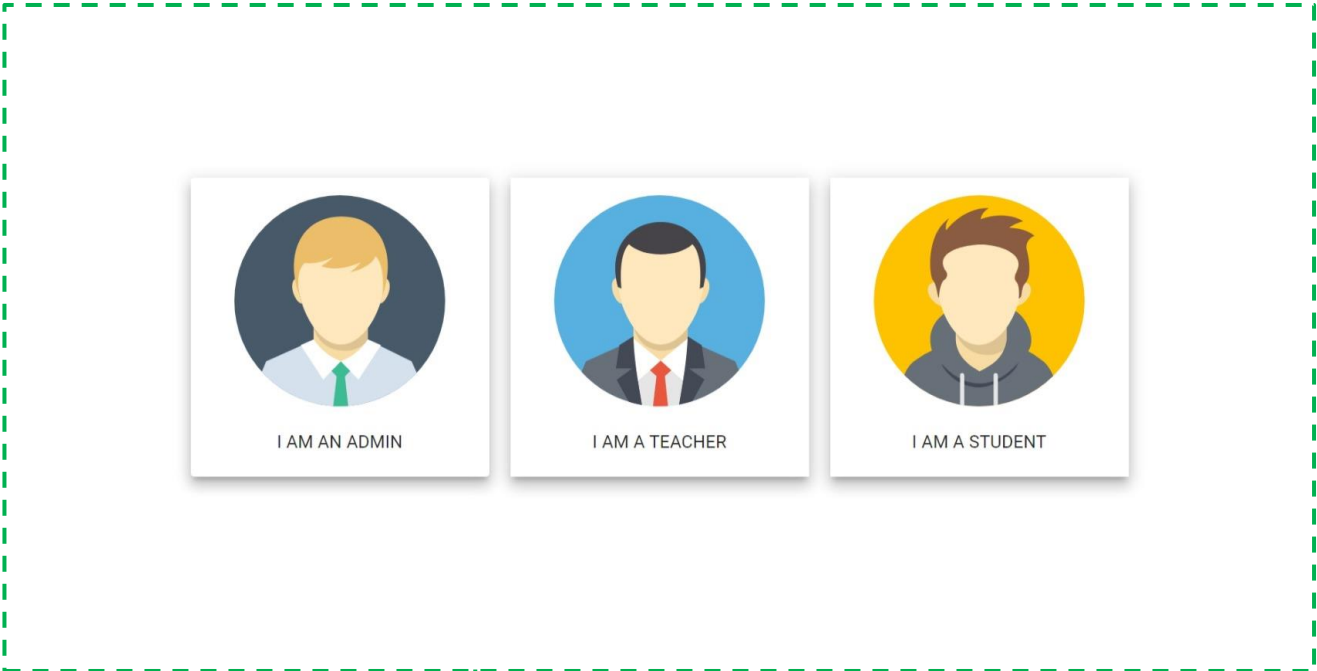












Figure 11 Pre-Login page

Admin

College Space admin1

STUDENT											
Name	Email Id	Department	Dept Id	USN	Year	Backlogs	SPC	CGPA	Counsellor Id	Password	Actions
Prasanna Bhat	prasannabhat.cs18@rvce.edu.in	Computer Science	1	1RV18CS116	2	0	✓	9.73	1		 
Pavan KR	pavankr.cs18@rvce.edu.in	Electronics and Communication	2	1RV18CS110	3	0	✓	9.95	1		 
student1	student1.ec19@rvce.edu.in	Electronics and Communication	2	1RV19EC005	2	0	—	8.5	1		 
student2	student2@gmail.com	Electronics and Communication	2	1RV19CS116	2	0	—	7.7	1		 
harish	harish.cs20@rvce.edu.in	Computer Science	1	1RV19CS05	1	0	—	0	3		 

5 rows |< > 1-5 of 5 |>

Figure 12 Admin-add student

Lecturer

Student

College Space

admin1

TEACHER							
Name	Email Id	Department	Department Id	Is HOD	Password	Actions	
instructor1	instructor1@gmail.com	Electronics and Communication	2	✓			
instructor2	instructor2@gmail.com	Electrical and Electronics	3	—			
CSinstructor1@gmail.com	CSinstructor1	Computer Science	1	—			
instructor3	instructor3@gmail.com	Electronics and Communication	2	—			
instructor4	instructor4@gmail.com	Computer Science	1	✓			

5 rows | 1/1 | 1/1 of 5

Figure 14 Admin-add teacher

College Space

admin1

Student Table

Teacher Table

Register for Course

Add Course

Log Out

Add New Course

Course Name *

XYZ

Name of this course

Course Code *

X

Set Course Code

Department *

Computer Science

Electronics and Communication

Electrical and Electronics

Information Science

Figure 13 Admin-add course

College Space

admin1

Student Table

Teacher Table

Register for Course

Add Course

Log Out

Register Student For Course

Student *

Choose Student

DATA STRUCTURES AND ALGORITHMS

MICROCONTROLLERS

Database Design

OOPs with Java

Figure 15 Admin-register student for course

Lecturer

Student

College Space

instructor1

Add Class Post
Add Department Post
Department Posts
Scan Document
Get Objects
Log Out

Create Class Post

Course*
Enter the Course for which you want to put the message

Title*
Enter the title for the post

Message*
Enter the message

POST


Figure 17 Lecturer-add class post

College Space

instructor1

Add Class Post
Add Department Post
Department Posts
Scan Document
Get Objects
Log Out

Upload Image to be Processed


Drag or click to select file (Image or pdf)

File
pavan.jpg

SUBMIT

Figure 16 Lecturer(counselor)-digitize form

College Space

instructor1

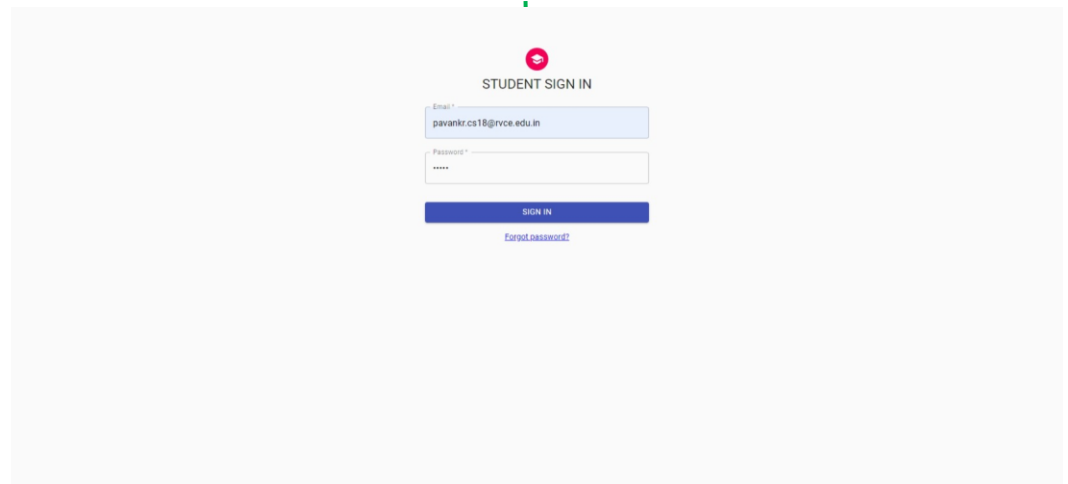
Add Class Post
Add Department Post
Department Posts
Scan Document
Get Objects
Log Out

See Information Stored

{
Name of the Student (In Capitals) : Prasanna Bhat
Branch (In Capitals) : Computer Science and Engineering
USN : RV18CS116
Mobile Number : 6362474863
Gender : Male
Blood Group : A+
Category : GM
Name of the Counselor(In Capitals) : Sandhya S
RVCE Mail ID : sandhya.sampangi@rvce.edu.in
Father Name & Contact Number : Mahesh Bhat
Mother Name & Contact Number : Uma Bhat
Permanent Address (In Capitals) : F7 Malhar Apartment Mahadwar Road 5th cross Belgaum Pincode:590001
}

{
Name of the Student (In Capitals) : Pavan KR
Branch (In Capitals) : Computer Science and Engineering
USN : RV18CS112
Email Id : pavankr.cs18@rvce.edu.in
Mobile Number : 8310233896
Gender : Male
Blood Group : B+
Category : GM
Name of the Counselor(In Capitals) : SANDHYA SAMPANGIRAMIAH
RVCE Mail ID : sandhya.sampangi@rvce.edu.in
Father Name & Contact Number : Raghavendra KV
Mother Name & Contact Number : Sugatha G
Permanent Address (In Capitals) : #298, 8th main, CK A, Bengaluru Pincode:560085.
}

Figure 18 Lecturer(counselor)-get digitized forms



STUDENT SIGN IN

Email*

Password*

SIGN IN

[Forgot password?](#)

Figure 20 Student-login

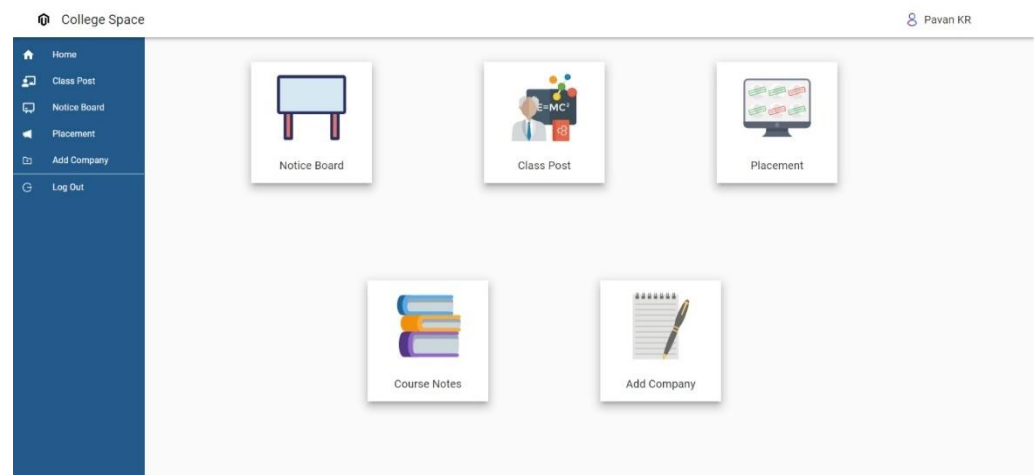


Figure 19 Student-home page

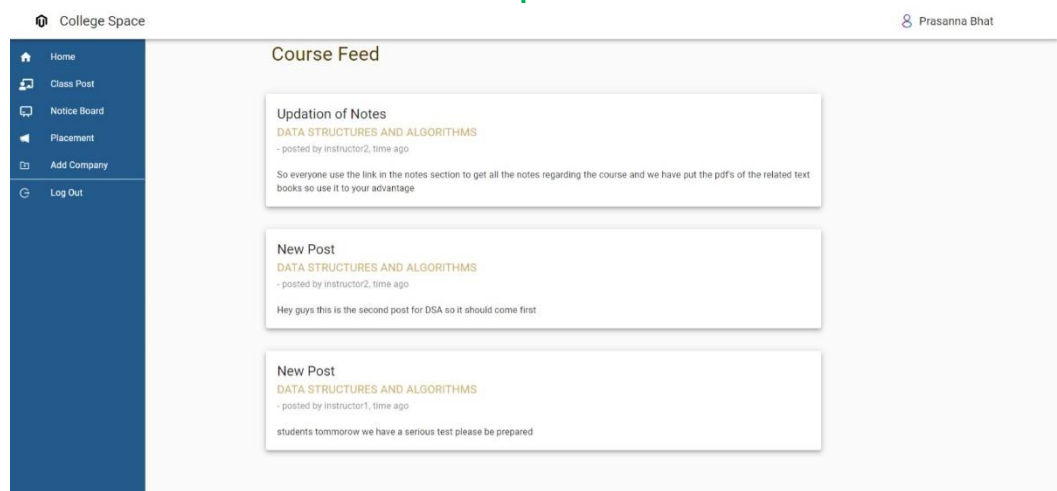


Figure 21 Student-class notice board

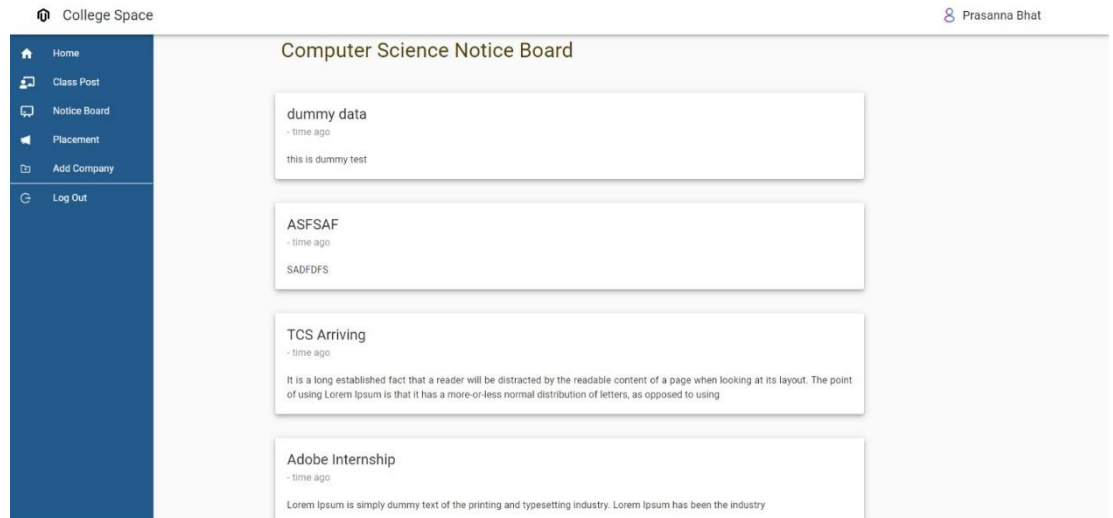


Figure 23 Student-department notice board

College Space

Prasanna Bhat

Placement/Internship

Company	Role	Job Profile	Stipend/CTC	Category	Actions
GOOGLE	INTERN	ide	11111	open dream	✓
AFAP	INTERN	ASFASDF	50000	AFSADFASDF	✓
TCS	INTERN	SDET	10000	Dream	📄
Abode	INTERN	MTS2	60000	Open Dream	✓
Adobe	INTERN	Member of Technical Staff	60000	Open Dream	✓

5 rows | 1-5 of 14

Figure 22 Student-apply for internship

College Space

Prasanna Bhat

Entry For Placement

Company *

Enter the Name of the Company

Job Profile *

Enter the Job Profile

Role *

Enter if the role is Intern or FTE

Min CGPA *

Enter the minimum CGPA for qualification (Decimal between 0-10)

Max Backlogs *

Maximum Backlogs (Number between 0-10)

Stipend/CTC *

Enter Stipend for Internship and CTC for FTE (Enter a Number)

Year *

Enter Eligibility year for students

Category *

Enter which category of the company dream /open dream etc

Entry For Post

Title *

Title for Department Post Regarding Internship

Description *

Message for the Department Post

SUBMIT

Figure 24 Student(SPC)-add company