PREFACE

This project has been submitted in the fulfillment of the requirements for the Educational Organisation. We the team members of this project, take pleasure in presenting the detail project report that reflects our efforts in academic year 2023-24.

Our project involves designing a Grid framework for executing complex applications where the process is divided into threads and accordingly the threads are executed by the executors. The outputs generated by the executors are given back to the manager which in turn gives the results to the owner. This is a dedicated in which the manager can select particular executors to run the application.

Initially manager is started by connecting it to a storage application. The executors are connected to the manager by providing the required credentials. Once the executors get connected to the manager the execution of the required can be started.

Additionally there is a Grid console which keeps track of the executors connected and the applications running. A record of all the operations performed by either of the logger is maintained in a log file.

ACKNOWLEDGEMENT

We have immense pleasure in presenting the report for our project entitled “Academic Flow”

We would like to take this opportunity to express our gratitude to a number of people who have been sources of help & encouragement during the course of this project.

We are very grateful and indebted to our project guide Ms. Ritu Bhatiya for providing their enduring patience, guidance & invaluable suggestions. They were the one who never let our morale down & always supported us through our thick & thin. They were the constant source of inspiration for us & took utmost interest in our project.

We would also like to thank all the staff members for their invaluable co-operation & permitting us to work in the computer lab.

We are also thankful to all the students for giving us their useful advice & immense co-operation. Their support made the working of this project very pleasant.

Group Members:

1. **Chaudhari Dhiraj.**
2. **Dadhich Akshat.**
3. **Patel Krishna.**

INDEX

CHAPTER-1

INTRODUCTION

Introduction

The College Management System (CMS) represents a pivotal advancement in the realm of educational administration, aiming to streamline and optimize various processes within academic institutions. In today's rapidly evolving educational landscape, colleges and universities face multifaceted challenges ranging from administrative tasks to academic management and student engagement. The CMS serves as a comprehensive solution designed to address these challenges by integrating technology with administrative processes to enhance efficiency, transparency, and effectiveness.

At its core, the CMS is a digital platform that centralizes and automates key administrative functions, including student enrollment, registration, scheduling, grading, and financial management. By digitizing and standardizing these processes, the CMS eliminates manual paperwork, reduces administrative burden, and minimizes the likelihood of errors or discrepancies. This not only improves operational efficiency but also frees up valuable time and resources that can be redirected towards more strategic initiatives and student-focused activities.

Moreover, the CMS facilitates data-driven decision-making by providing real-time access to accurate and up-to-date information. Administrators can generate comprehensive reports, analyze trends, and identify areas for improvement, enabling them to make informed decisions that enhance organizational performance and student outcomes. Additionally, the CMS promotes collaboration and communication among various stakeholders, including students, faculty, staff, and parents, by providing a centralized platform for sharing information, announcements, and updates.

In addition to administrative functions, the CMS also encompasses features designed to enhance the academic experience for students and faculty. For example, students can access course materials, submit assignments, and communicate with instructors through the CMS's online portal, fostering a more interactive and engaging learning environment. Similarly, faculty members can manage course materials, track student progress, and communicate with students more effectively, thereby enhancing the overall teaching and learning experience.

Furthermore, the CMS supports financial management processes, including budgeting, invoicing, and payroll, ensuring that financial resources are managed efficiently and transparently. By automating these processes and providing robust reporting capabilities, the CMS enables administrators to monitor expenditures, track revenue streams, and ensure compliance with financial regulations and policies.

Overall, the College Management System represents a paradigm shift in educational administration, offering a comprehensive and integrated solution to the complex challenges facing colleges and universities today. By harnessing the power of technology, the CMS empowers institutions to enhance efficiency, transparency, and effectiveness across all facets of their operations, ultimately enabling them to better serve their students, faculty, staff, and stakeholders

1.1 OBJECTIVES

To implements a College Management System in computerized by using MERN Stack.

Scope:

This project deals with the various functioning in College management process. The main idea is to implement a proper process to system .In our existing system contains a many operations registration, student search, fees, attendance, exam records, performance of the student etc. All these activity takeout manually by administrator.

In our model, it deals with the Operations in system. For example, when students fill the registration form the record is stored in the database. And display the details of student is perform by retrieving information from database table.

Description of the project:

We identify several problems including unauthorized privilege escalation, incorrect use of cryptography, vulnerabilities to network threats, and poor software development processes. We show that only college administrator can start the system. administrator can search the particular student by his/her enrollment number or student id And we are adding notification module where administrator should add the notification with start and end date.

CHAPTER-2

SYSTEM ANALYSIS

2.1 INTRODUCTION

The main function of the analysis phase is to look carefully at the requested features with an eye towards the issue that each may create in the actual coding. This phase is the time during which reasonably deliverable thoughts of each team member can decide.

User requirements:

The following requirements are raised during the analysis of the needs of the users:

Ø A Person Should be able to login to the system through the first page of the Application.

Ø The Administrator can create users as per user requirement.

Ø A general user will have access to see the status of particular Student id number.

Ø Student (user) can use all the facilities, same as which are provided to him in the college.

Ø Student can see attendance, notices, grades and other facilities in updated manner.

Ø There will be a separate page for every student as his account in which he can get notices, attendance, grades, assignments etc.

Ø Faculty can give the attendances and notices for the students.

After analyzing the requirements of the task to be performed, the next step is to analyze the problem and understand its context. The first activity in the phase is studying the existing system and other is to understand the requirements and domain of the new system. Both the activities are equally important, but the first activity serves as a basis of giving the functional specifications and then successful design of the proposed system. Understanding the properties and requirements of a new system is more difficult and requires creative thinking and understanding of existing running system is also difficult, improper understanding of present system can lead diversion from solution.

2.2 ROLE OF SYSTEM ANALYSIS

- **Memory constraints:** There may be some memory constraints due to advance detailed database, which has many primary, foreign, unique keys and large datasets.

- **Software interfaces:** Frontend is in REACT JS/NATIVE and Backend is in NODE JS using MONGO DB Database.

- **System features:** Simple GUI, easy to use, no special training required, cost should be feasible, protected through authentication process

CHAPTER-3

APPLICATION DESIGN

3.1 FORM DESIGNING

- **Login Form:-**

  This is login form of the Software, it ask for the id and password of the user, access is given to the authorized persons only. types of logins are there Admin only.

  .

- **MDI form:**

  This screen acts as the home screen for this software. Through this MDI form we can access its child forms. This form gives the user further 5 options:-

  - **Login Page** :it contains LOGIN, LOG OUT,and EXIT option to exit from software.
  - **Registration:**This module is designed for registration

- **Form list and details**

1. **Registration Entry Form:** registration entry form can be used to enter the new entry of student.

2. **Student Search Form:** student search form can be used by admin, for searching particular student related entering branches and year.

3. **Notification Form:** notification form can be used to add notification related to the exams and schedule of the college. Full authority is given to the admin.

4. **Fees Form:** this can be generated the fees report of the students.

3.2 FRONT END - BACK END CONNECTIVITY

Connecting the frontend and backend using the MERN (MongoDB, Express.js, React.js, Node.js) stack involves setting up communication between the client-side application built with React.js and the server-side application built with Node.js and Express.js. Here's a step-by-step guide on how to achieve frontend and backend connectivity using the MERN stack:

1. Setting Up the Backend (Node.js and Express.js):

- Install Node.js and npm (Node Package Manager) on your system if you haven't already.

- Create a new directory for your backend application and navigate into it.

- Initialize a new Node.js project by running npm init -y command.

- Install Express.js framework by running npm install express command.

- Create a new file (e.g., server.js) and set up your Express.js server.

- Define routes for handling incoming requests and serving data to the frontend.

- Connect to your MongoDB database using Mongoose or any other MongoDB driver.

2. Setting Up the Frontend (React.js):

- Install create-react-app globally if you haven't already by running npm install -g create-react-app command.

- Create a new React.js project by running npx create-react-app frontend command.

- Navigate into the newly created frontend directory (cd frontend) and start the development server by running npm start.

- Create React components for different parts of your frontend application, such as UI components, forms, and pages.

- Use React Router to set up client-side routing if your application requires multiple pages.

- Fetch data from your backend API using the fetch API, Axios, or any other HTTP client library supported by React.

3. Connecting Frontend to Backend:

- Update your React components to make HTTP requests to the backend API endpoints.

- Use fetch or Axios to send GET, POST, PUT, DELETE, etc., requests to the appropriate backend routes.

- Handle the responses received from the backend in your React components and update the UI accordingly.

- Consider using asynchronous functions, promises, or async/await syntax for handling asynchronous operations in your frontend code.

- Test the connectivity between the frontend and backend by sending requests and verifying the responses.

4. Handling Authentication and Authorization:

- Implement user authentication and authorization mechanisms in your backend API using JWT (JSON Web Tokens), OAuth, or any other authentication protocol.

- Secure your API routes by validating authentication tokens and authorizing users based on their roles and permissions.

- Implement login/logout functionality in your frontend application and store authentication tokens securely (e.g., in local storage or cookies).

By following these steps, you can establish connectivity between the frontend and backend of your MERN stack application, allowing data to be exchanged seamlessly between the client-side and server-side components.

CHAPTER-4

SYSTEM DESIGN

4.1 INTRODUCTION

Purpose of **College Management System for a College (CMS)** Design Document is to describe the design and the architecture of **CMS**. The design is expressed in sufficient detail so as to enable all the developers to understand the underlying architecture of **CMS**

Logical View:

it provides the user with an abstract view of the overall system functionality

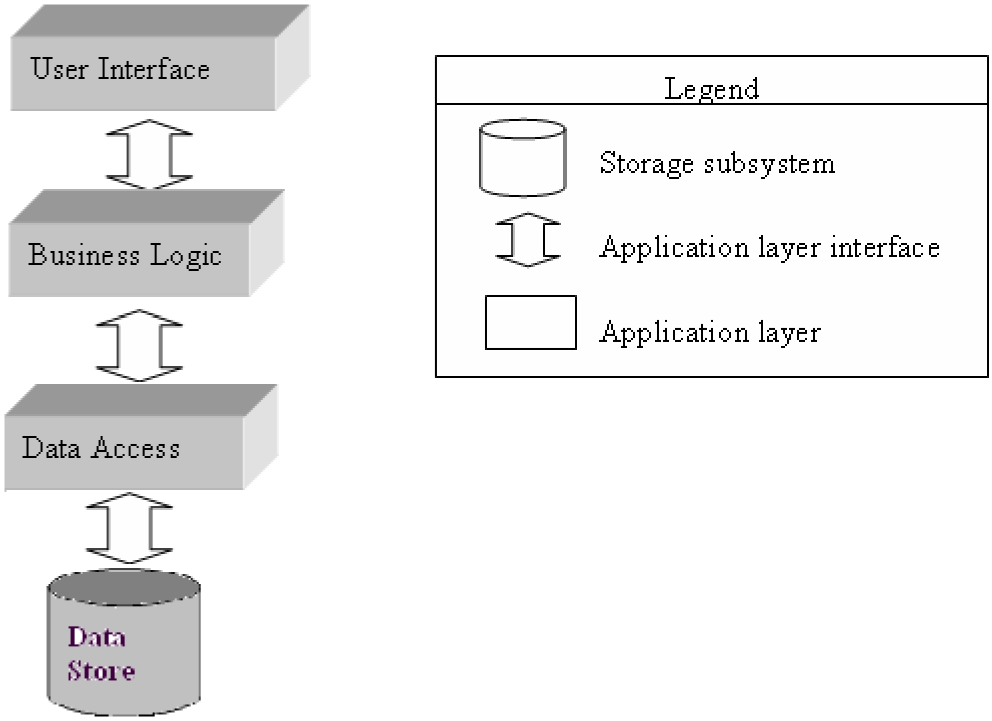


fig: abstract view of CMS(College Management System)

4.2 EVALUATION OF GOOD DESIGN SOFTWARE (LIFE CYCLE MODEL)

System Definition

Scope and Boundaries

Scope of this system can be described as follows:-

i. It is a standalone module, available for desktop application used by computer Administrator to store student's data.
ii. The storing and retrieval of data is quick
iii. It is an online application.
iv. Access is given to only authorize person and no other person can access this software.
v. Data security is maintained properly by authentication of users.
vi. Large amount of data can be processed quickly with ease

User view

I. **Admin:** It is the administrator account having rights to govern wholesoftware. He can create and delete student and details ,create and change password, he also have rights to add and delete record, various criteria and notification.

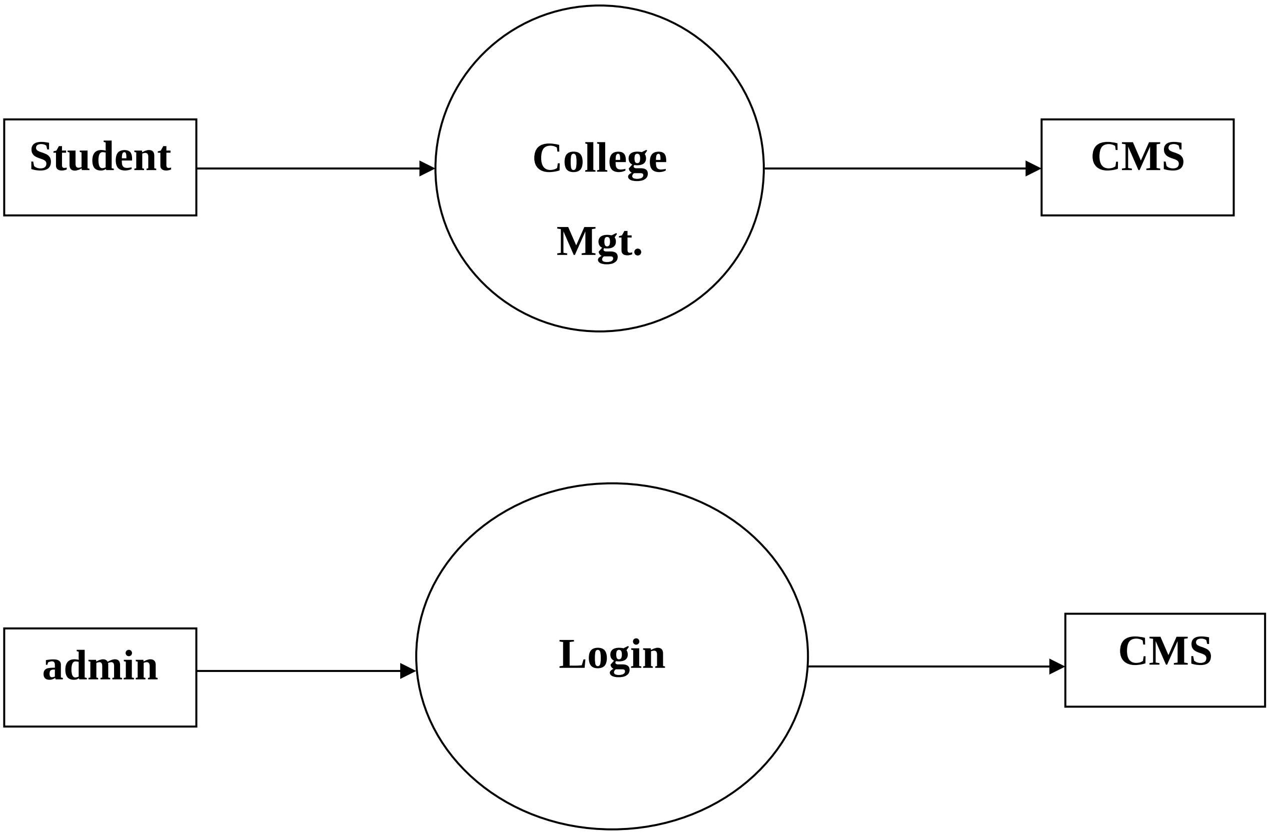
II. **User:** users are having only read permission; user type can only see his details, he will not be able to manipulate or change any information ,it also have same privileages as that of admin.

CHAPTER-5

DATA FLOW DIAGRAM

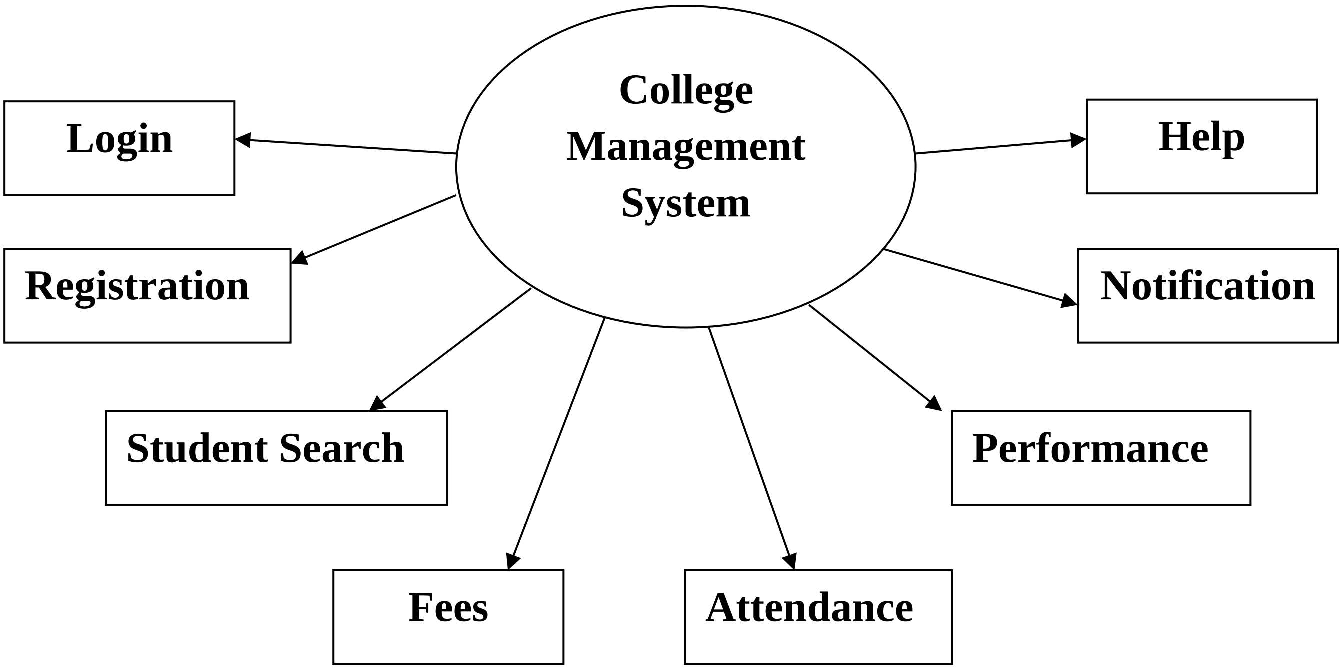
5.1 CONTEXT LEVEL DFD

Level-0



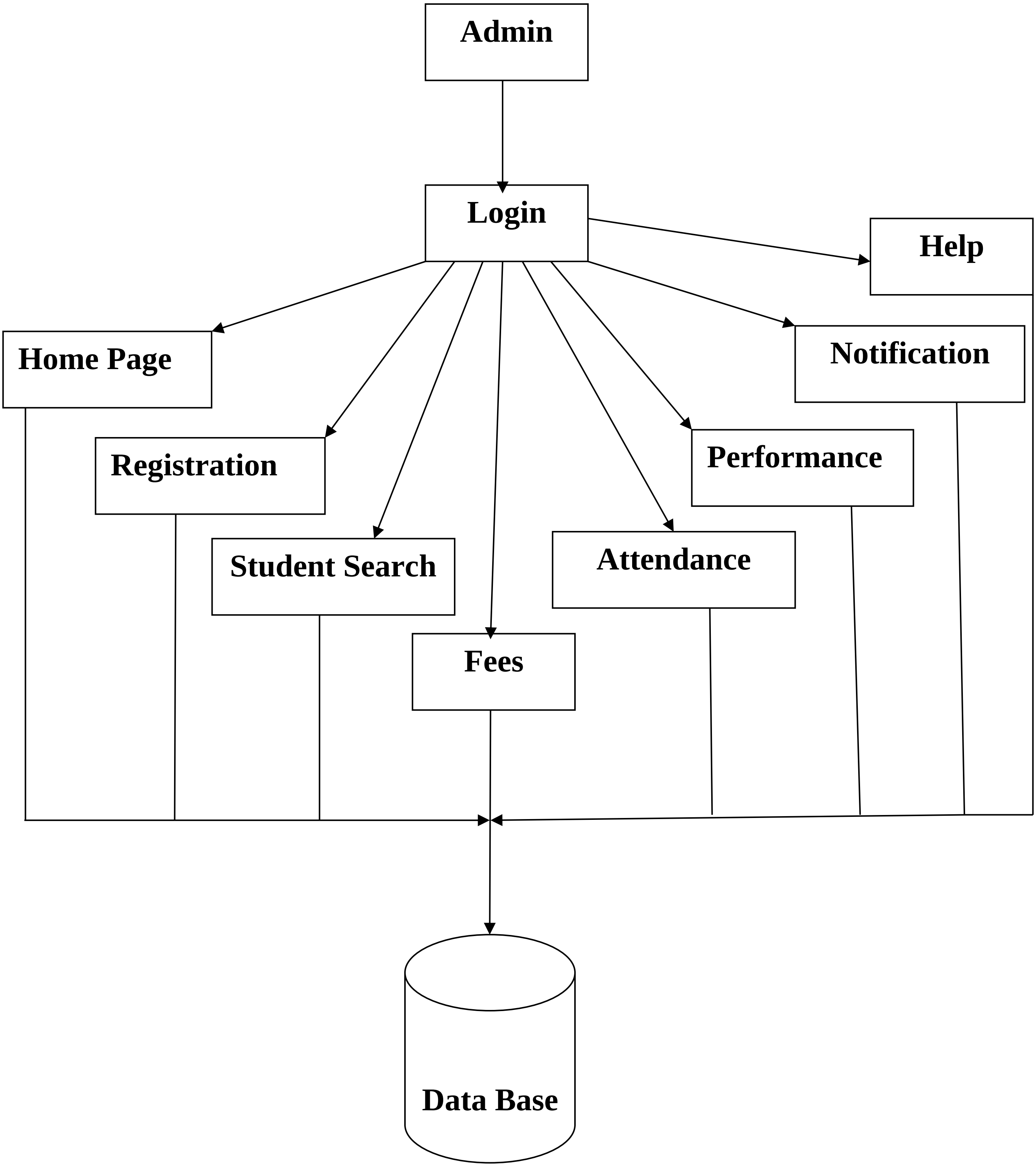
5.2 LEVEL- 1 DFD

Level-1



5.3 LEVEL- 2 DFD

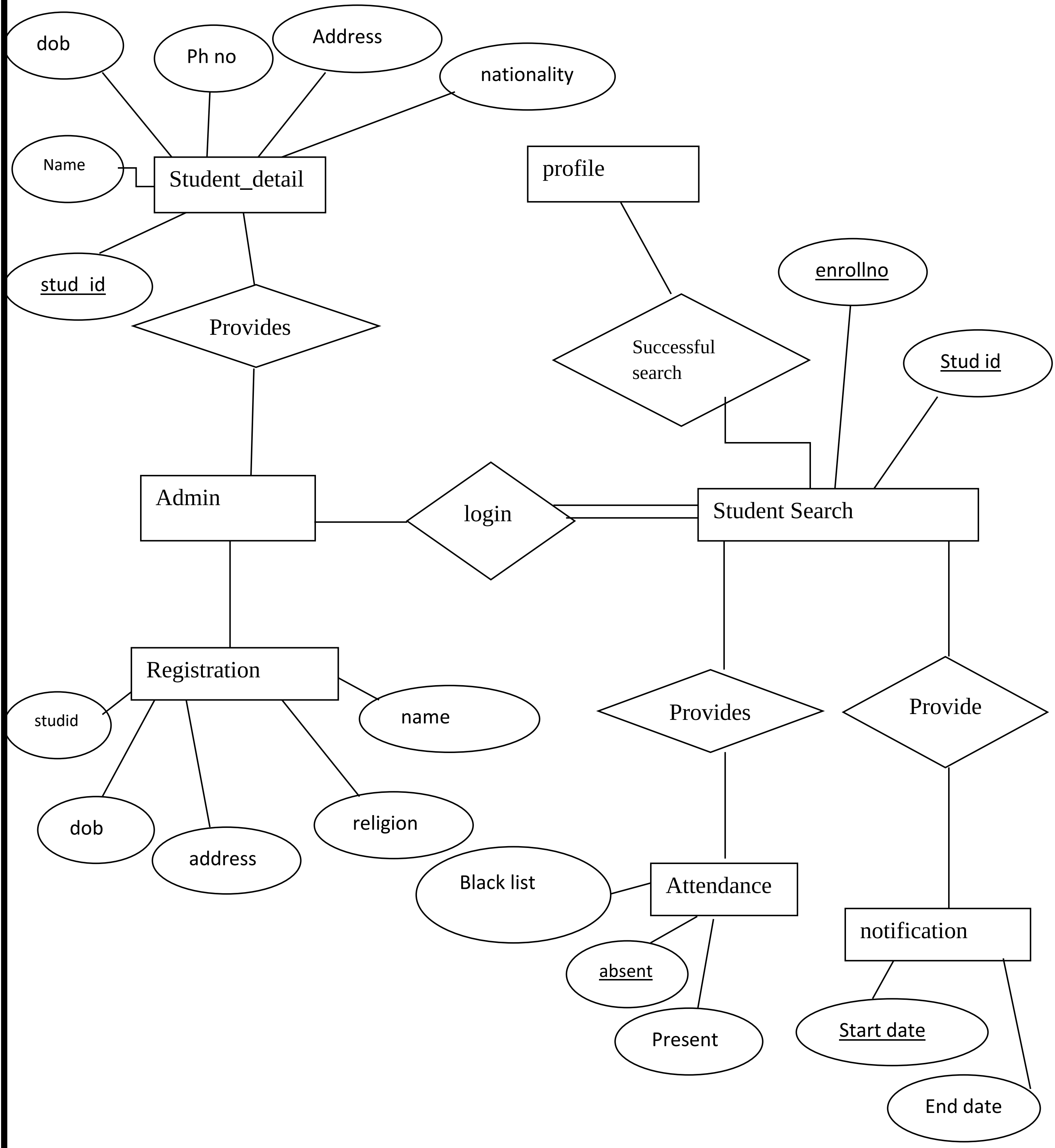
Level -2



CHAPTER-6

ENTITY RELATIONSHIP DIAGRAM

Entity Relationship Diagram



6.1 MODULE DESCRIPTION

Login:

The login page only used by the authority or admin.

Admission:

The admission form can be filled by the student who wants to register him/herself. And after admission complete, the details can be stored in database.

Student Search:

The student search form is only designed for the searching particular student or details about the student.

Fees:

The fees module only accessible for the admin .The details about the fees detail is contain in this module.

Attendance:

This module displays the attendance record of the student and also displays the defaulter’s list.

Notification:

Notification module only used for adding notification,start date and time. This is acts as a small alarm system for any task i.e. submission, examination time- table etc.

Help:

The help can be provided any application help.

6.2 DATABASE TABLE

Table name	Purpose
1.admin	Admin table contain password and user name field.
2.admission	Admission table contain student details like name, dob, address etc.
3.notification	Notification table contain notification detail.
4.performance	Performance table contain the performance details about the student.
5.student table	Student table table hold the stud id.
6.fees	Fees table contain the fees details of the students.

Future Enhancement:

- Online examination module would be introduced to conduct online examination,
- Scheduling of the staff. i.e. , time table setting of the staff
- Further, the faculty can upload the videos of their **LIVE & Recorded** lectures on to this site and students who had missed those classes can view those videos.