1. **Synopsis**

**1.1 Introduction of the System**

The Project ***Student Result Processing System*** Software is a complete multi-user Web-Application. The admin accepts and then processes the scores of students and subsequently produces their report cards. The staff adds or updates student’s internal marks. The students can view and analyze their performances in the examination through graphical charts.

This project aimed at providing colleges to add, view and update student results and generate report cards and performance analysis charts, which can be viewed by the students through user friendly GUI based web pages.

* + 1. **Project Title**

"CANARA EXAM MANAGER”

*Student Result Processing System*

* + 1. **Category**

Web Application

* + 1. **Overview**

An individual report card of each student has to be displayed and printed at a keystroke according to standard format. Performance analysis charts are generated which can be viewed by the students through which he/she can analysis their performances in the examination. Student performance in a particular exam or all the exams must be expressed.

* 1. **Background**
     1. Brief note on Existing System

In the manual student result processing system, all the database commands have to be typed by the users. This procedure is very time consuming, and is limited to a single system. If someone want to get information about a particular subject score, student’s performance or any other content related to college, they have to contact the administrator handling the system.

In the proposed system, there are various controls to provide user friendliness. Details can be accessed over the internet, and huge amount of data, records and information can be stored. It provides high level of security, and there is no risk of data mismanagement. The overall result processing system is easier to use, flexible and with more features.

* 1. Objectives of the System
* The main objective of the system is to provide examination result to the student in a simple and user-friendly way.
* The privileges that are provided to student are to read his/her result and to compare and view his/her performance using graphical charts by providing user id and password for secure login and in case of new student the registration is done by the administrator (office) and the user id and password will be provided by the administrator.
* The privileges that are provided to staff are to read students results and to add or update internal marks to each individual students, by providing user id and password for secure login and in case of new staff the registration is done by the administrator (office) and the user id and password will be provided by the administrator.
* The privileges that are provided to administrator are to view and update the details of students and staff and also add or update students results, by providing the admin id and password for secure login.
  1. Scope of the System

In “Student Result Processing System” there are various controls to provide user friendliness. Details can be accessed over the internet, and huge amount of data, records and information can be stored. It provides high level of security and there is no risk of data mismanagement. The overall result processing system is easier to use, flexible and with more features.

* 1. Structure of the System
* Home Page
  + Student Login
  + Staff Login
  + Admin Login
* Login
  + User Name
  + Password
  + Login Button
* Admin Page
  + Home
  + Profile
  + Dashboard
  + Student Details
  + Satff Details
  + Programme and Courses
  + Classes
  + Student Result
  + Log Out
* Staff Page
  + Home
  + Profile
  + Dashboard
  + Student Results
  + Log Out
* Student Page
  + Home
  + Profile
  + Dashboard
  + Results
  + Performance
  + Log Out

Modules

Administrator

Login: Admin is provided with separate login page where he/she has to enter username and password that is provided to them by the Administrator. This module is for user authentication.

Dashboard: Administrator has a dashboard where he/she can manage all the functionality at one place.

Add or Remove User: He/she can add or remove any user or users i: e staff or students.

View Student Details: He/she can view the details of staff and students.

Update Student Details: He/she can update the details of students.

View Staff Details: He/she can view the details of staff.

Update Staff Details: He/she can update the details of staff.

View Programme and Courses: He/she can view the courses and subjects offered in the university.

Update Programme and Courses: He/she can add or remove any courses and subjects and also modify them.

Add or update student’s Semester marks: He/she can add or update semester marks of each individual students.

Logout: Admin can log out from his/her dashboard page which can then be accessed only after authorized login.

Staff

Login: Staff is provided with separate login page where he/she has to enter username and password that is provided to them by the Administrator. This module is for user authentication.

Dashboard: Staff has a dashboard where he/she can manage all the functionality at one place.

View Student Details: The staff can view the details of each student.

Check Examination Result: He/she can check the examination results of the students.

Add or update student’s internal marks: He/she can add or update internal marks of each individual students.

Logout: Staff can log out from his/her dashboard page which can then be accessed only after authorized login.

Student

Login: Students are provided with separate login page where he/she has to enter username and password that is provided to them by the Administrator. This module is for user authentication.

Dashboard: Student has a dashboard where he/she can manage all the functionality at one place.

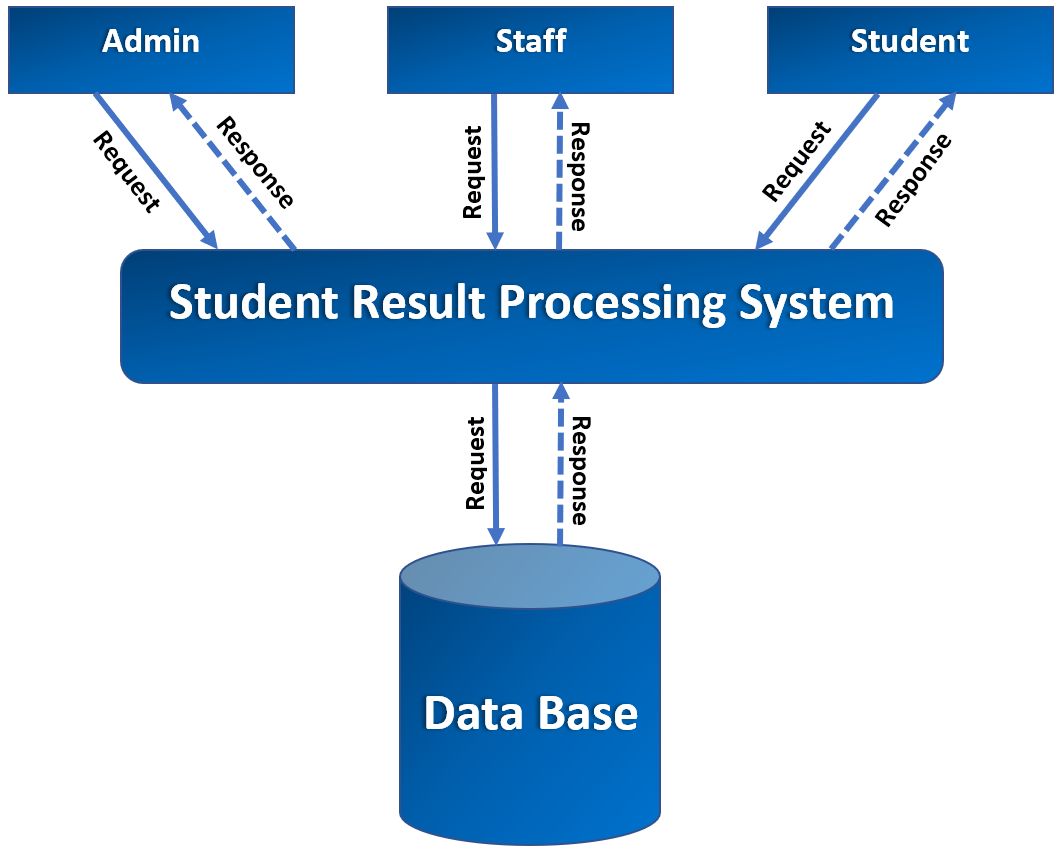
View Personal Details: He/she can view their personal details.

Check Examination Result: He/she can check their examination results including the remarks provided by the staff.

Check Performance: He/she can analyze their performance through graphical charts.

Logout: Student can log out from his/her dashboard page which can then be accessed only after authorized login.

* 1. System Architecture



1

* 1. End Users

Administrator Role: Manipulates all the records. Provides authorization to the users (students/staff). An admin can add or remove users (students/staff). He/she can view and update the personal details of the students and staff. He/she can view and update the examination results of students.

Staff Role: Provides remarks to the student’s performance. He/she can view their own personal details and the personal details of student’s. He/she can view the examination results and analyze the performance of students through graphical charts.

Student Role: Can view their examination results. He/she can analyze their performance through graphical charts.

* 1. Software/Hardware used for development
     1. Software:

|  |  |
| --- | --- |
| IDE / Text Editor | Brackets, Visual Studio Code, Eclipse. |
| Web Contents | Java Servlets, JSP, HTML, CSS, Java Script. |
| Server-Side Programming | Java |
| Database | My SQL |
| Server | Apache Tomcat |
| Documentation Tool | MS Office |

* + 1. Hardware:

|  |  |
| --- | --- |
| Processor | i3 2.30 GHz |
| RAM | 8GB RAM |
| Memory | 70GB of Hard disk space or above |
| Graphics Card | 4GB Intel® UHD Graphics 620 |

* 1. Software/Hardware required for the implementation
     1. Client side:

Software:

|  |  |
| --- | --- |
| **Operating System** | Any Operating System that supports browser applications like MS Windows, Linux, Mac OS etc. |
| **Browser** | Any browser that supports html5, CSS and JavaScript like Microsoft Edge, Google Chrome, Mozilla Firefox, Brave etc. |

Hardware:

|  |  |
| --- | --- |
| Processor | Intel Pentium dual-core or above |
| RAM | 2GB RAM |
| Memory | 50GB of Hard disk space or above |
| Graphics Card | 2GB HD Graphics Card or above |
| Input Devices | Mouse, Keyboard |
| Output Devices | Monitor |

* + 1. Server side.

Software:

|  |  |
| --- | --- |
| Operating System | Any Operating System that supports Apace Tomcat server and JRE (Java Runtime Environment) like MS Windows, Linux, Mac OS etc. |
| Server | Apache Tomcat and MySQL server |
| Database | MySQL |

Hardware:

|  |  |
| --- | --- |
| Processor | Intel Pentium dual-core or above |
| RAM | 2GB RAM |
| Memory | 60GB of Hard disk space or above. |

1. **SOFTWARE REQUIREMENTS SPECIFICATION**
   1. **Introduction**

The introduction of the Software Requirements Specification (SRS) provides an overview of the entire SRS with purpose, scope, definitions acronyms, abbreviations and overview of the SRS. The aim of this document is to gather and analyze and give an in-depth insight of the online *Student Result Processing System* by defining the problem statement in detail. The detailed requirements of “CANARA EXAM MANAGER” system is provided in document.

* + 1. **Purpose**

This document mainly focuses on providing the information about the software requirements of the “CANARA EXAM MANAGER” system, which can be referred by the developers and will be basis for the validation of the final system. The project mainly focuses on exam management, where we can perform activities starting from registration of the students to generation of examination result and analyzing the performance of different examinations and comparing them.

* + 1. **Scope**

The scope of this document is to provide sufficient details on the functional and non-functional requirements are properly implemented. This project has an enormous scope in the upcoming days when the institutions can make their work of generating examination results automatic rather than manually doing all the calculations, which will save a lot of time.

* + 1. **Definition, Acronyms, Abbreviations**

This particular software is defined as automated software for the examination department of Universities/Institutions.

|  |  |
| --- | --- |
| **SRS** | Software Requirements Specification |
| **SRPS** | Student Result Processing System |
| **CEM** | Canara Exam Manager |
| **DFD** | Data Flow Diagram |
| **ERD** | Entity Relationship Diagram |
| **CFD** | Context Flow Diagram |
| **DBMS** | Database Management System |
| **RDBMS** | Relational Database Management System |
| **GUI** | Graphical User Interface |
| **ADMIN** | The Administrator |
| **JSP** | Java Server Page |
| **SQL** | Structured Query Language |
| **HTML** | Hyper Text Markup Language |
| **CSS** | Cascading Style Sheets |
| **JS** | Java Script |

* + 1. **Overview**

This SRS provides the overall description and specific requirements of the project. The general description will describe the requirements of the *“Student Result Processing System”.* The particular requirement section describes the details of the system. The system provides an easy solution to process the results of the students which reduces some of the unnecessary manual calculations. This web application can be used by any user who has basic knowledge of using a computer system without need of any special skills.

* 1. **Overall Description**

This examination software mainly focuses on the automation of the examination process, which starts from the registration of students to generation of examination result and analyzing the performance of different examinations and comparing them. Here Admin registers every detail of staff and students. The admin uploads the marks after getting the results of the examination (which includes all the examinations i: e from semester exams to internal exams) of the students and the result will be publish which can be viewed by the students.

* + 1. **Product Perspective**

The main objective of this **CEM** is to provide examination result to the student in a simple and user-friendly way. It is a web-based application with 3 interfaces Admin, staff and students. All the information is stored in the database, which can be retrieved. The website works with desktops, laptops and mobile with the help of the browser application.

* + 1. **Product Functions**
* Registering staffs and students.
* Ability to modify and upload data.
* Marks of the entire semester can be published.
* Allow admin to take control of the entire application.
* Details of each student’s profile are viewable by admin and staff.
* Staffs can view each student profile and exam results.
* Staffs can give remarks to each student based on their performance.
* Students can view their profile.
* Students can view their marks.
* Students can analyze their performance using enhanced graphs.
  + 1. **User Characteristics**

1. **Admin:** Admin is the core user. He/she has a complete access to the web-application. The admin manipulates all the records. Provides authorization to the users (students/staff). An admin can add or remove users (students/staff). He/she can view and update the personal details of the students and staff. He/she can view and update the examination results of students.
2. **Staff:** These are the users who have access to the information regarding the examination results and profile information of each student. He/she can add or update internal marks of each individual students.
3. **Student:** These are the users who have access to the information regarding the examination results and their profile information. He/she can view their examination results. He/she can analyze their performance through enhanced graphical charts.
   * 1. **General Constraints**

* The system must be capable of doing parallel operations and must not have scalability issues with regard to number of computers, mobiles, tablets and monitors connected at one time.
* There is a small chance of miscalculations in examination results and performance analysis.
* The developed software should run on any platform (Linux, Mac, Windows etc. which supports Java) that contains a web browser.
* The end user (University) needs a hosting space so that it can be accessed from any remote location.
  + 1. **Assumptions**
* The code should be error-free.
* Roles and tasks are predefined.
* The Administrator is created in the system already.
  1. **Special Requirements**
     1. **User Interface:**

The user interface of the web-application should be compatible with any browser such as Google Chrome, Microsoft Edge, Mozilla Firefox, Brave etc.

There are 3 user interfaces for 3 different users they are follows: -

* Admin
* Staff
* Student
  + 1. **Hardware Interface:**
* Processors: Intel Pentium dual-core or above.
* RAM: 2 GB and above.
* Hard Disk Utilization: 60 GB and above.
* Input Devices: Mouse, Keyboard.
* Output Devices: Monitor.
  + 1. **Software Interface:**
  + Browser: Any browser that supports html5, CSS and JavaScript like Microsoft Edge, Google Chrome, Mozilla Firefox, Brave etc.
  + Server: Apache Tomcat.
  + IDE’s: Eclipse Enterprise Edition, Brackets, Visual Studio Code.
  + Web Components: Java Servlets, JSP, HTML, CSS.
  + Language: Java
  + Scripting Language: Java Script.
  + Database: MySQL.
    1. **Communication Interface:**

This is a web-based system, and communication is done through internet and internet protocols are used (HTTP Protocols).

* 1. **Functional Requirements**

Student Result Processing system is a dynamic web-application that is connected to the database. The students (users) can view their results and analyze their performance. There are 3 modules in this web-application they are: -

* + 1. **Login Module:** This module is used for authentication of the users. Admin, staff and students have different login page where the users have to provide username id and password. Only if the username id and password evaluate to true then he/she can access their profile page.
    2. **Admin Module:** The Administrator is the core user of the system. This module has the following functionalities.
* The Administrator should Login into the system with unique his/her username and password.
* Admin adds or removes users (student/staff).
* Admin views staff Personal details.
* Admin updates staff Personal details.
* Admin views student Personal details.
* Admin updates student Personal details.
* Admin views Programme and courses.
* Admin updates Programme and courses.
* Admin views Results of students.
* Admin updates Results of students.
  + 1. **Staff Module:** Staffs are the observers, although they do not have the privileges to change any data or information of the students including examination results. This module has the following functionalities.
* The Staff should login into the system with unique his/her username and password.
* Staff views student’s Personal details.
* Staff views his/her own Personal details.
* Staff views Results of students.
* Staff updates Internal marks of the students.
  + 1. **Student Module:** Students are the basic users of this system. He/she don’t have the privileges to modify the data and information regarding his/her own personal details, examination result details, examination schedules any other details. This module has the following functionalities.
* The student should login into the system with unique his/her username and password.
* Student views his/her own Personal details.
* Student views Results.
* Student views Schedules.
* Student views his/her performance in the examination using enhanced graphical charts.
  1. **Design Constraints**

The system is designed to allow web usability. The system is designed in such a way that it will be visible and easy to use on most of the web browsers. The system is secure and does not allow malicious or unauthorized users to access particular resources.

* 1. **System Attributes**

1. **Compatibility:** This system will be compatible with almost all the web servers.
2. **Flexibility:** The system keeps updating the data according to the changes takes place.
3. **Portability:** The system can be run in any operating system and browser.
4. **Reliability:** This system is designed to have very simple database just to extract the details of every user. It is tested for all the constraints at development stage.
5. **Security:** This system provides with authentication without which no user can pass. So only the legitimate users are allowed to use the application. If the legitimate users share the authentication information, then the system is open to outsiders.
6. **Timelines:** The system carries out all operations with consumptions of very less time.
   1. **Other Requirements**
      1. **Safety Requirements:** There are several users in CEM software. Access to the various subsystems will be protected by a user log in screen that requires a username id and password. This gives different views and accessible function of user levels through the system. Maintaining backups ensure the system database security. System can be restored in any case of emergency.
      2. **Security Requirements:**

* Depending upon the category of user the access rights are decided.
* Admin has the maximum privileges to all subsystems.
* Only authenticated users can access the systems.
  + 1. **Performance Requirements:** In order to maintain an acceptable speed at maximum numbers of upload allowed from a particular user as any number of users can access to the system anytime. Also, the connections to the server will be based on the attributes of the user like location and server should be working 24X7.

1. **SYSTEM DESIGN**
   1. **Introduction**

System design is the process of defining the components, modules, interfaces and data for a system to satisfy specified requirements. System design involves designing a new system that will meet the requirements identified during system analysis. The focus of system design is on deciding which modules are needed for system, the specifications of these modules and how the modules should be interconnected.

* 1. **Assumptions and Constrains**

Project assumptions and constraints are identified at the beginning of the project. Throughout the project life cycle, they will be refined and re-analysed. Project assumptions and constraints are key to many processes in the PMBOK Guide. An essential aspect of your project, assumptions, and constraints are not managed like requirements or risks. However, documenting them helps to protect you from future aggravation. You should outline your project’s assumptions and constraints in the project scope statement.

Project constraints are limitations imposed on the project, like the budget, schedule, or resources. The recognizes six project constraints: scope, quality, schedule, budget, resources, and risk. Out of these six, scope, schedule, and budget are known as the triple constraints.

* 1. **Functional Decomposition**

Functional decomposition is a method of breaking down a problem into smaller and smaller functions. Each function is subdivided until it is manageable. Functional decomposition refers broadly to the process of resolving a functional relationship into its constituent parts in such a way that the original can be reconstructed from those parts by function components. In general, this process of decomposition is undertaken either for the purpose of gaining insight into the identity of the constituent component or for the purpose of obtaining a compressed representation of the global function, A task which is feasible only when the constituent processes possess a certain level of modularity.

* + 1. **System Software Architecture**
    2. **System Technical Architecture**

Presentation layer

Database

Web Server Layer

Business Logic layer

HMTL, CSS, JS

MY SQL

JSP, SERVELET

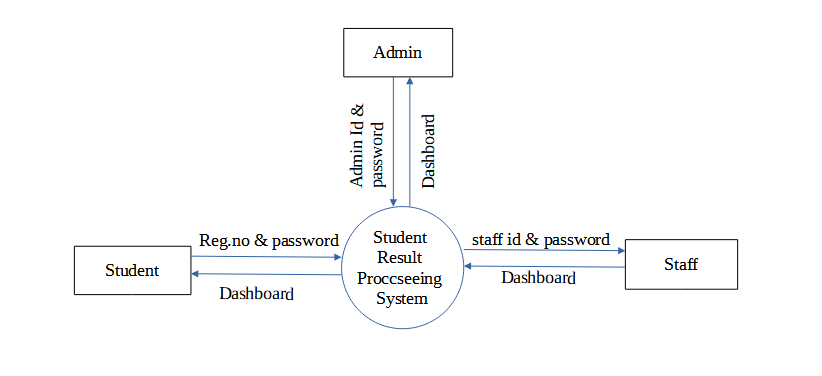
JAVA

The system is made up of four layers:

1. The presentation layer: includes user interfaces built using the XML tools. The interface is designed using GUI components which are user-friendly.
2. Business Logic layer: is used for the retrieval of data obtained from user interfaces or acquired form databases. Activities for calculating, managing and making decisions and communicating with the front-end application and back-end database are managed by business logic written using Java language are appearing here.
3. Web Server layer: is to serve websites on the internet. It acts as a middleman between the server and client machines. It pulls content from the server on each user request and delivers it to the App.
4. Database layer: is used to store data forwarded by the business layer and retrieve data on demand to the business layer. Firebase database software is used as a back-end database of the system.
   * 1. **System Hardware Architecture**
     2. **External Interface**

* Name of the Application: Canara Exam Manager
* Details of Interface: User Interface
* Type of Interface: Graphical User Interface, Menu driven Interface.
  1. **Description of Programs**
     1. **Context Flow Diagram**

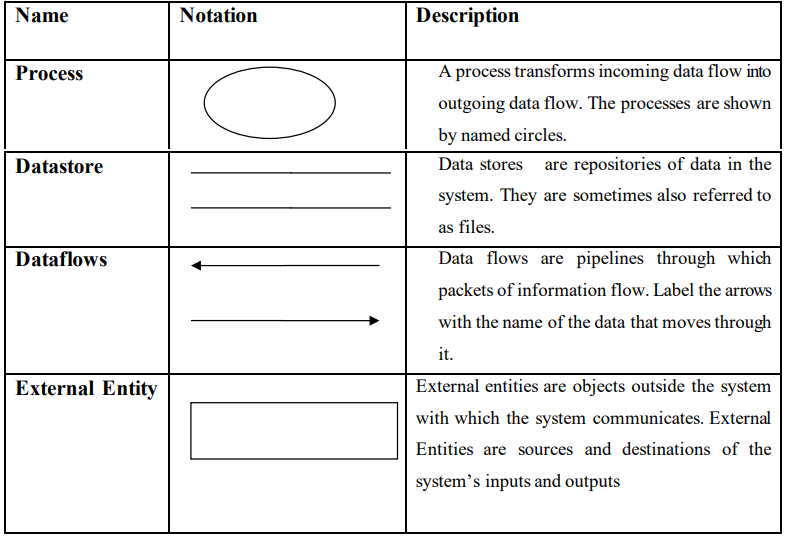
It defines the flows of information between the system and external entities. The entire software system is shown in single process. It’s designed to be an abstraction view, showing the system as a single process with its relationship to external entities. It represent the entire system as single bubble with input and output data indicated by incoming/outgoing arrows.



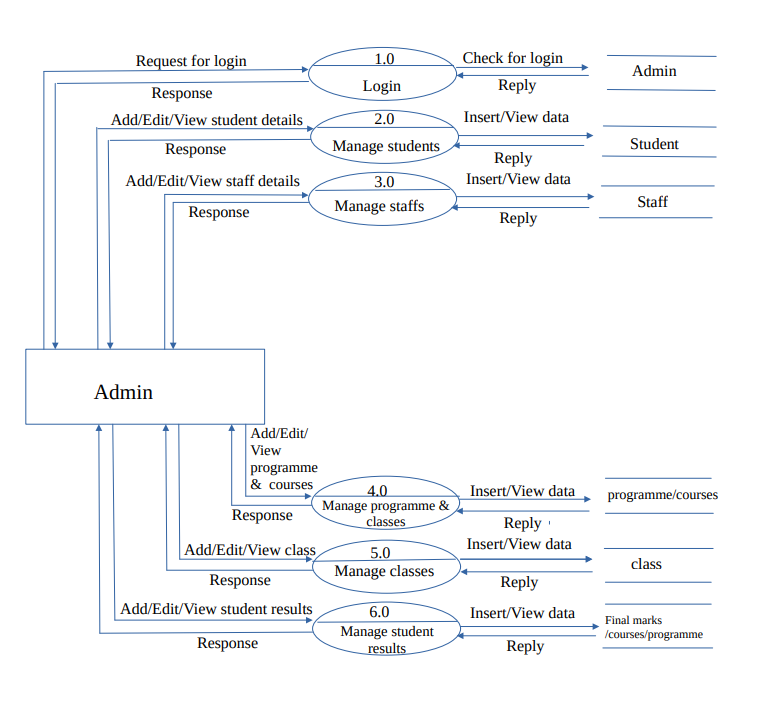
* + 1. **Data Flow Diagram**

Data Flow Diagram are the graphical way of showing the flow of data through an information system. It is the common practice for a designer to draw a context level DFD. A complete set of DFD provide a compact top down representation of the system. It also expresses the requirement of the system and shows how the current system is implemented. Data Flow Diagram are commonly used during problem analysis. It views a system as a function that perform the input into the desired output. DFD shows the information moves through and how it is modified by a series of transformation. DFD may be used to represent the system at any level of abstraction. DFD can be used to provide the end users with the physical idea of where the data they input ultimately

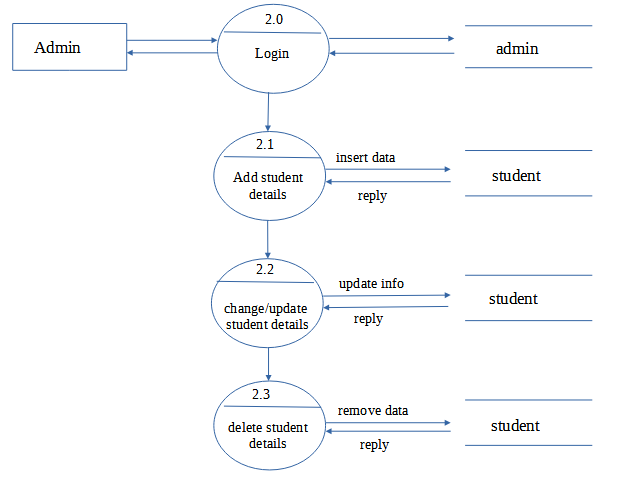
* + 1. **Data Flow Symbols**



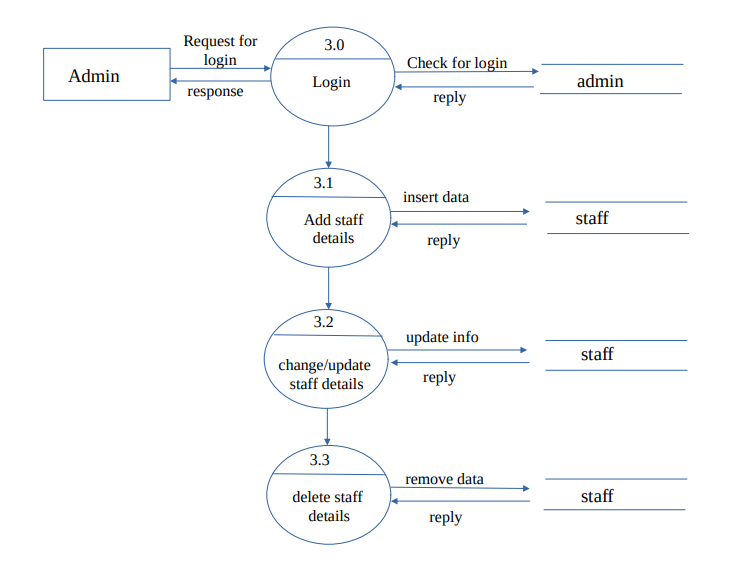
* + 1. **DFD Level 1(Admin)**



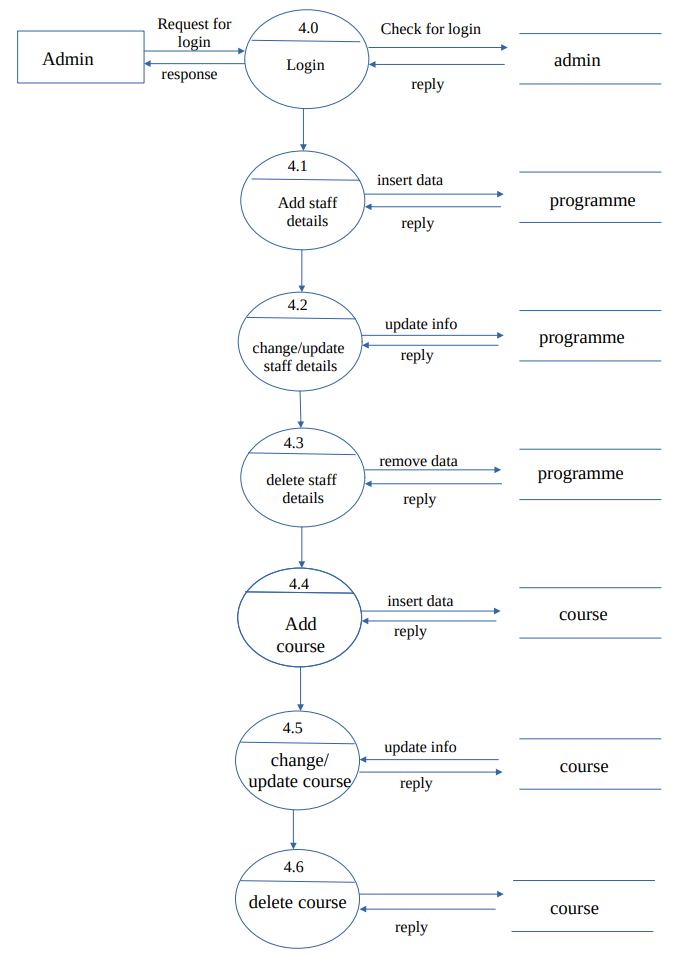
* + 1. **DFD Level 2(Admin)**



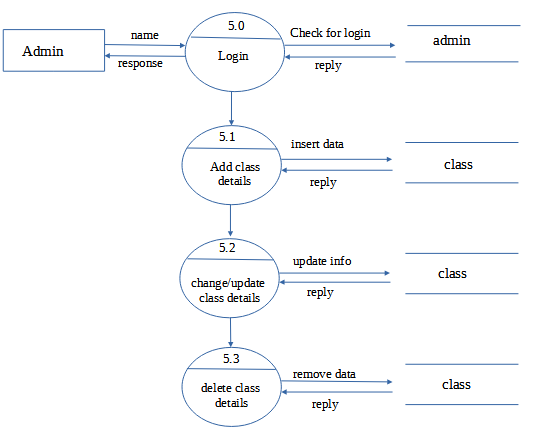
* + 1. **DFD Level 3(Admin)**



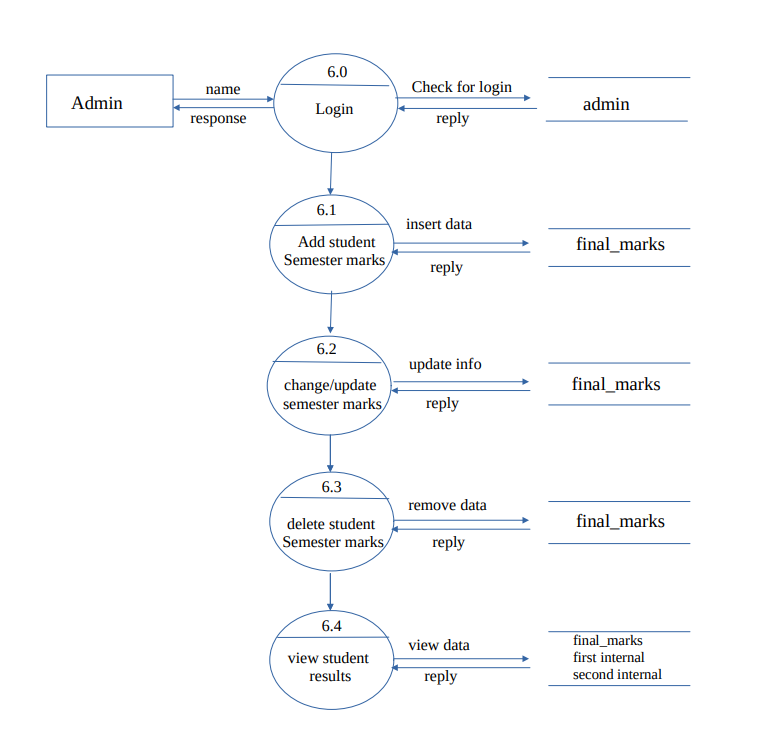
* + 1. **DFD Level 4(Admin)**



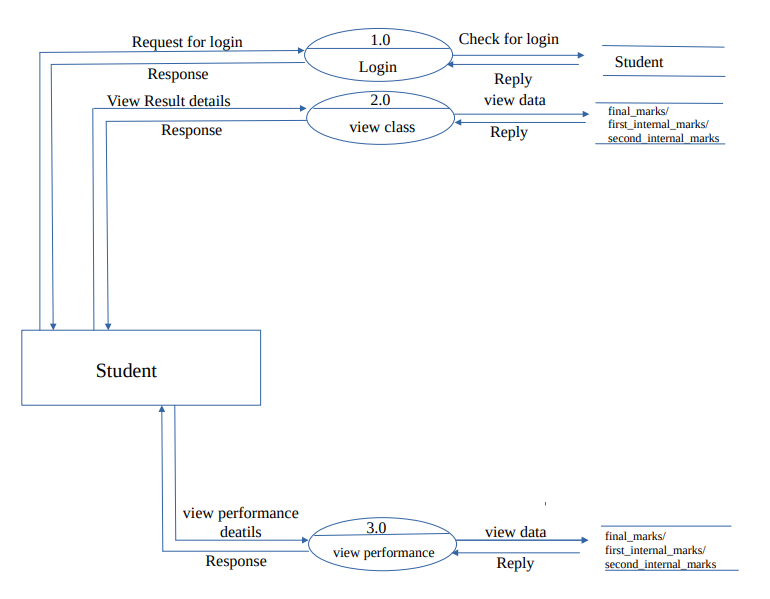
* + 1. **DFD Level 5(Admin)**



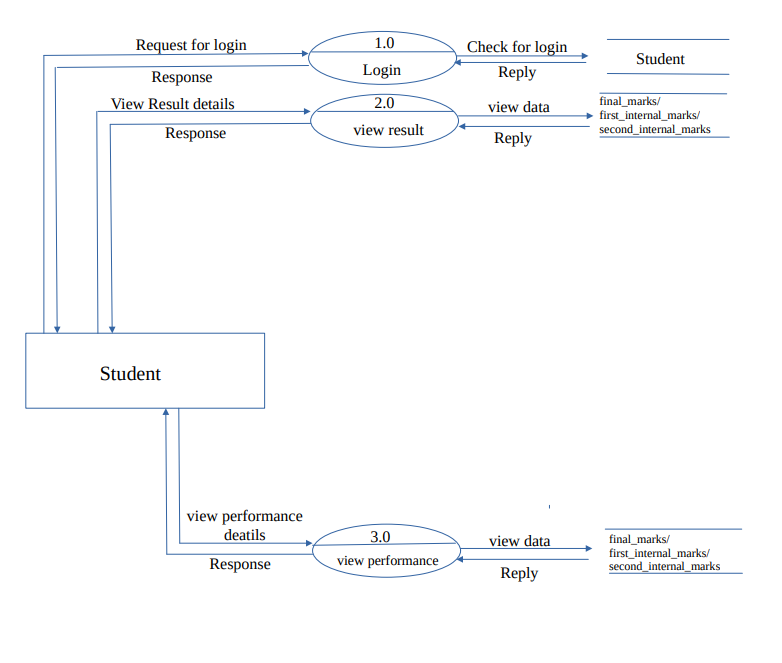
* + 1. **DFD Level 6(Admin)**



* + 1. **DFD Level 1(Student)**



* + 1. **DFD Level 1(Staff)**



1. **DATABASE DESIGN**
   1. **Introduction**

Database design is an important place in designing a system. The word “database” used to describe everything from a single set of data, to a complex set of tools, such as SQL server and a whole lot in between. The term data model to mean the conceptual description of the problem space. This includes the definition of entities, their attributes and the entity constraints. The data model also includes a description of the relationships between entities and any constraints on those relationships. The main advantage of this software is to reduce the manual work. During this phase care should be taken to avoid redundancy of information storing into a database since it leads to wastage of memory space.

* 1. **Purpose And Scope**
  2. **Database Identification**

Naming Conventions:

* The identifier should not have quotes.
* Identifier names should not have whitespace.
* Identifiers should be written entirely in lower case.
* Database object names, particularly column names, should be a noun describing the field or object.

Table Conventions:

* Database table names and column names are defined without leaving space.
* Lowercase used to create database tables and columns.
* Primary key and foreign key defined with the same name.
  1. **Schema Information**

A schema is a structure behind data organization. It is a visual representation of how different table relationships enable the schema’s underlying mission business rules for which the database is created. In a schema diagram, all database tables are designed with unique columns and special features, e.g., primary/foreign keys or not null, etc. Formats and symbols for expression are University understood, eliminating the possibility of confusion. The table relationships also are expressed via the parent table’s primary key lines when joined with the child tables corresponding to foreign keys. Schema diagrams have an important function because they force database developers to transpose ideas to paper. This provides an overview of the entire database while facilitating future database administrator work.

* 1. **Table Definition**
     1. **Table Name: admin**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sl.No | Field Name | Data Type | Field Size | Constrain | Description |
| 1 | admin\_id | int | 3 | Primary Key | Admin Id |
| 2 | first\_name | varchar | 64 | Not Null | First Name |
| 3 | last\_name | varchar | 64 | Not Null | Last Name |
| 4 | phone | bigint | 10 | Not Null | Phone Number |
| 5 | email | varchar | 225 | Not Null | Email |
| 6 | password | varchar | 64 | Not Null | Password |

* + 1. **Table name: staff**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sl.No | Field Name | Data Type | Field Size | Constrain | Description |
| 1 | staff\_id | int | 3 | Primary Key | Staff Id |
| 2 | first\_name | varchar | 64 | Not Null | First Name |
| 3 | last\_name | varchar | 64 | Not Null | Last Name |
| 4 | gender | varchar | 6 | Not Null | Gender |
| 5 | email | varchar | 225 | Not Null | Email |
| 6 | phone | bigint | 10 | Not Null | Phone |
| 7 | address | varchar | 225 | Not Null | Address |
| 8 | password | varchar | 64 | Not Null | Password |
| 9 | staff\_status | varchar | 5 | Not Null | Status |
| 10 | role | varchar | 60 | Not Null | Role |
| 11 | profile\_image | longblob |  | Null | Profile Image |
| 12 | program\_id | int | 3 | Foreign Key | Program Id |

* + 1. **Table name: student**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sl.No** | **Field Name** | **Data Type** | **Field Size** | **Constrain** | **Description** |
| 1 | reg\_no | int | 9 | Primary Key | Register No |
| 2 | first\_name | varchar | 64 | Not Null | First Name |
| 3 | last\_name | varchar | 64 | Not Null | Last Name |
| 4 | gender | varchar | 6 | Not Null | Gender |
| 5 | dob | date |  | Not Null | Dob |
| 6 | email | varchar | 225 | Not Null | Email |
| 7 | phone | bigint | 10 | Not Null | Phone |
| 8 | address | varchar | 225 | Not Null | Address |
| 9 | birth\_place | varchar | 64 | Not Null | Birth Place |
| 10 | birth\_district | varchar | 64 | Not Null | Birth District |
| 11 | birth\_state | varchar | 64 | Not Null | Birth State |
| 12 | pincode | int | 6 | Not Null | Pincode |
| 13 | password | varchar | 64 | Not Null | Password |
| 14 | programme\_id | int | 3 | Foreign Key | Programme Id |
| 15 | class\_id | int | 3 | Foreign Key | Class Id |
| 16 | joining\_year | int | 4 | Not Null | Joining Year |
| 17 | semester | int | 2 | Not Null | Semester |
| 18 | status | varchar | 5 | Not Null | Status |
| 19 | profile\_image | longblob |  | Null | Profile Image |

* + 1. **Table name: programme**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sl.No** | **Field Name** | **Data Type** | **Field Size** | **Constrain** | **Description** |
| 1 | Programme\_id | int | 3 | Primary Key | Programme Id |
| 2 | Programme\_name | varchar | 64 | Unique Key | Programme Name |
| 3 | Programme\_duration | int | 1 | Not Null | Duration |
| 4 | Progrsmme\_sem | int | 2 | Not Null | Total Semester |

* + 1. **Table name: class**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sl.NO** | **Field Name** | **Data Type** | **Field Size** | **Constrain** | **Description** |
| 1 | class\_id | int | 3 | Primary Key | Class Id |
| 2 | class\_name | varchar | 64 | Not Null | Class Name |
| 3 | programme\_id | int | 3 | Foreign Key | Programme Id |
| 4 | class\_year | int | 1 | Not Null | Class Year |

* + 1. **Table name: course**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sl.NO** | **Field Name** | **Data Type** | **Field Size** | **Constrain** | **Description** |
| 1 | course\_code | varchar | 9 | Primary Key | Course Code |
| 2 | course\_name | varchar | 64 | Not Null | Course Name |
| 3 | course\_type | varchar | 9 | Not Null | Course Type |
| 4 | course\_group | varchar | 64 | Not Null | Course Group |
| 5 | course\_group\_number | float | 2 | Not Null | Course Group Number |
| 6 | course\_sem | int | 2 | Not Null | Course Sem |
| 7 | max\_marks | int | 3 | Not Null | Max Marks |
| 8 | min\_marks | int | 3 | Not Null | Min Marks |
| 9 | max\_ia | int | 3 | Not Null | Max IA |
| 10 | programme\_id | int | 3 | Foreign Key | Programme Id |

* + 1. **Table name: exam\_details**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sl.NO** | **Field Name** | **Data Type** | **Field Size** | **Constrain** | **Description** |
| 1 | reg\_no | int | 9 | Foreign Key | Register Number |
| 2 | exam\_type | varchar | 8 | Not Null | Exam Type |
| 3 | result\_date | date |  | Not Null | Result Date |

* + 1. **Table name: first\_internal\_marks**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sl.NO** | **Field Name** | **Data Type** | **Field Size** | **Constrain** | **Description** |
| 1 | reg\_no | int | 9 | Foreign Key | Register No |
| 2 | course\_code | varchar | 9 | Foreign Key | Course Code |
| 3 | program\_id | int | 3 | Foreign Key | Program Id |
| 4 | max\_marks | int | 3 | Not Null | Max Marks |
| 5 | min\_marks | int | 3 | Not Null | Min Marks |
| 6 | obtained\_marks | int | 3 | Not Null | Obtained Marks |
| 7 | grade | varchar | 4 | Not Null | Grade |
| 8 | result | varchar | 10 | Not Null | Result |

* + 1. **Table name: second\_internal\_marks**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sl.NO** | **Field Name** | **Data Type** | **Field Size** | **Constrain** | **Description** |
| 1 | reg\_no | int | 9 | Foreign Key | Register No |
| 2 | course\_code | varchar | 9 | Foreign Key | Course Code |
| 3 | program\_id | int | 3 | Foreign Key | Program Id |
| 4 | max\_marks | int | 3 | Not Null | Max Marks |
| 5 | min\_marks | int | 3 | Not Null | Min Marks |
| 6 | obtained\_marks | int | 3 | Not Null | Obtained Marks |
| 7 | grade | varchar | 4 | Not Null | Grade |
| 8 | result | varchar | 10 | Not Null | Result |

* + 1. **Table name: final\_marks**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sl.NO** | **Field Name** | **Data Type** | **Field Size** | **Constrain** | **Description** |
| 1 | reg\_no | int | 9 | Foreign Key | Register No |
| 2 | course\_code | varchar | 9 | Foreign Key | Course Code |
| 3 | program\_id | int | 3 | Foreign Key | Program Id |
| 4 | obtained\_marks | int | 3 | Not Null | Obtained Marks |
| 5 | IA\_marks | int | 3 | Not Null | IA Marks |
| 6 | total\_marks | int | 3 | Not Null | Total Marks |
| 7 | grade | varchar | 4 | Not Null | Grade |
| 8 | credit | int | 2 | Not Null | Credit |
| 9 | grade\_point | int | 2 | Not Null | Grade Point |
| 10 | grade\_point\_weightage | int | 2 | Not Null | Grade Point Weightage |
| 11 | result | varchar | 4 | Not Null | Result |

* 1. **Physical Design**

The physical design of your database optimizes performance while ensuring data integrity by avoiding unnecessary data redundancies. During physical design, you transform the entities into tables, the instances into rows, and the attributes into columns. You need to continually monitor the performance and data integrity characteristics of the database as time passes. Many factors necessitate periodic refinements to the physical design

* 1. **Data Dictionary**

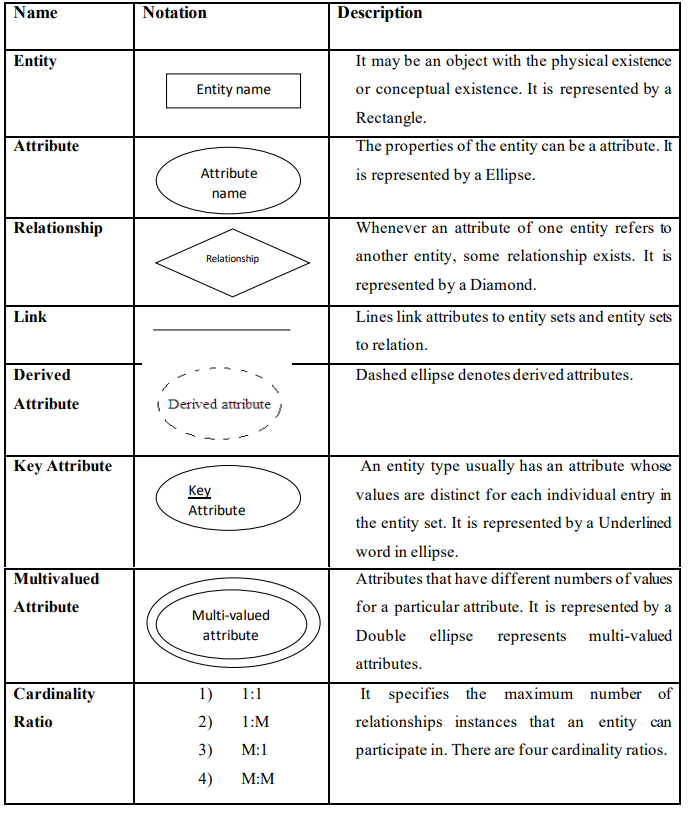
A Data Dictionary is a collection of names, definitions, and attributes about data elements that are being used or captured in a database, information system, or part of a research project. It describes the meanings and purposes of data elements within the context of a project, and provides guidance on interpretation, accepted meanings and representation. A Data Dictionary also provides metadata about data elements. The metadata included in a Data Dictionary can assist in defining the scope and characteristics of data elements, as well the rules for their usage and application.

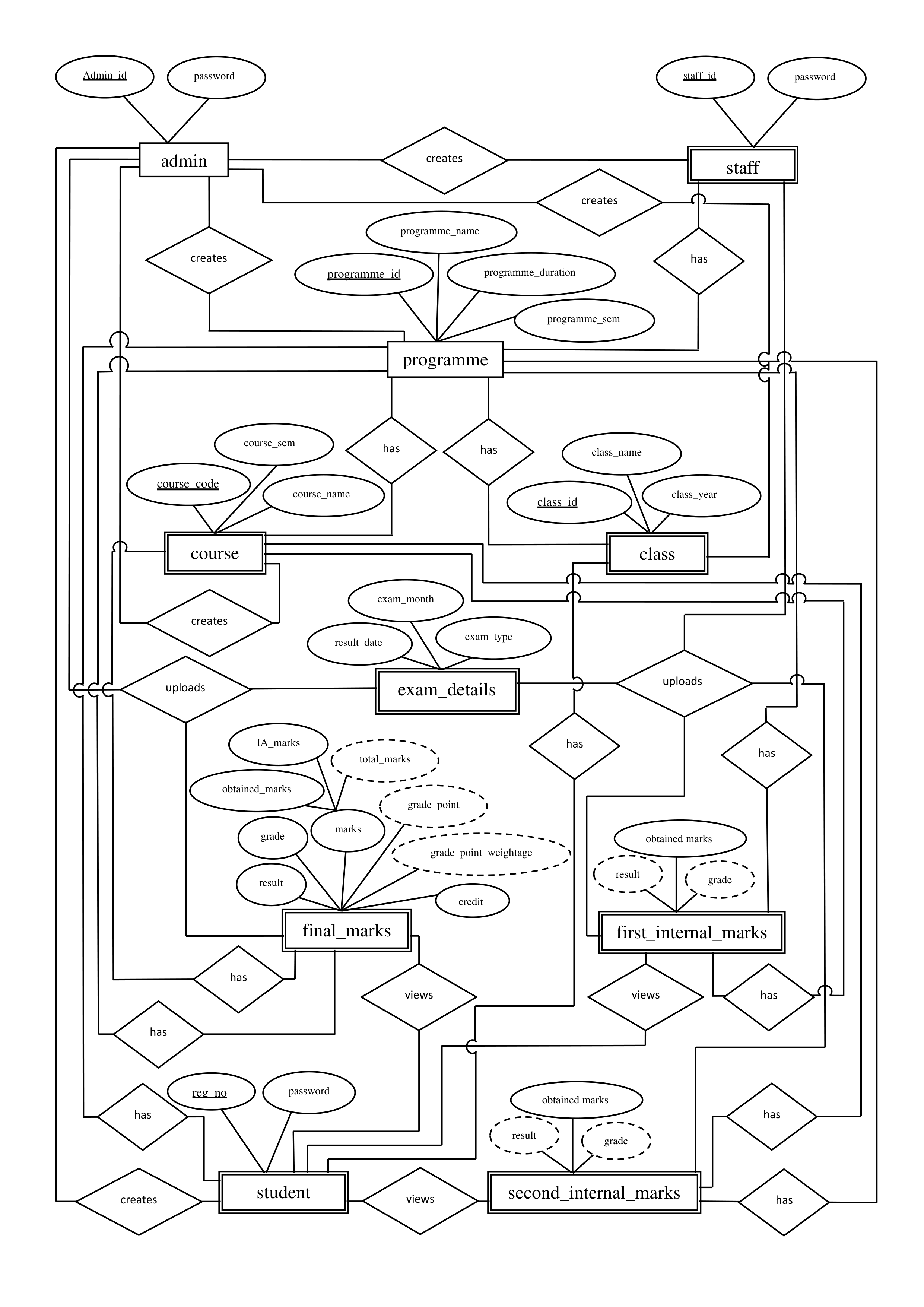
Data Dictionaries are useful for a number of reasons. In short, they:

* Assist in avoiding data inconsistencies across a project
* Help define conventions that are to be used across a project
* Make data easier to analyse
* Enforce the use of Data Standards
  1. **Entity Relationship Diagram**

An Entity Relation of (ER) Diagram is a specialized graphics that illustrates the interrelationship between entities in a database. Entities are physical items or aggregations of data items that are important to the business that we analysis to the system. It is a framework using specialized symbols to define the relationship between entities. ER diagram is created based on three main components entities, attributes and relationship. ER helps us conceptualize the database and help us know which fields need to be embedded for a particular entity. ER Diagram gives a better understanding of the information to be stored in a database. Reduces completely and saves time which allows you to build database quickly

* + 1. **ER Diagram symbol’s**





* 1. **Database Administration**

**4.10 Description of components**

1. **DETAILED DESIGN**
   1. Introduction

Detailed design can be generally defined as a collection of tasks or processes that enhance the designing, development, implementation, and maintenance of enterprise data management system. Designing a proper database reduces the maintenance cost thereby improving data consistency and the cost-effective measures are greatly influenced in terms of disk storage space. Therefore, there has to be a brilliant concept of designing a database. The designer should follow the constraints and decide how the elements correlate and what kind of data must be stored.

The main objectives behind database designing are to produce physical and logical design models of the proposed database system. To elaborate this, the logical model is primarily concentrated on the requirements of data and the considerations must be made in terms of monolithic considerations and hence the stored physical data must be stored independent of the physical conditions. On the other hand, the physical database design model includes a translation of the logical design model of the database by keep control of physical media using hardware resources and software systems such as Database Management System (DBMS).

* 1. Structure of the software package

The functional components are:

* Admin module
* Staff module
* Student module

* 1. Modular decomposition of the system
     1. Admin Module
        1. Inputs
        2. Procedural Details
        3. File I/O Interface
        4. Outputs
        5. Implementation Aspects
     2. Staff Module
        1. Inputs
        2. Procedural Details
        3. File I/O Interface
        4. Outputs
        5. Implementation Aspects
     3. Student Module
        1. Inputs
        2. Procedural Aspects
        3. File I/O Interface
        4. Outputs
        5. Implementation Aspects