**University management system**

**Final report**

**Group member:**

Chathuni Noyali Wahalathantri

Mandeep Singh

Gagandeep Singh

Vivek Jayeshbhai Talpada(leader)

**Table of Contents**

Table of Contents

[Executive summary 3](#_Toc163330907)

[Design Analysis process 4](#_Toc163330908)

[Functional and non-functionality 7](#_Toc163330909)

[Software Development life Cycle 8](#_Toc163330910)

[Entity relationship Diagram 11](#_Toc163330911)

[Table design: data dictionary 12](#_Toc163330912)

[Contribution of team for the report 16](#_Toc163330913)

[References 17](#_Toc163330914)

## Executive summary

In the project, the aim is to create a database for our system using SQLite. This report contains the foundation for creating a database, which includes the design analysis process, entity relationship diagram and table designs.

The design analysis process encompasses user stories, use cases, a use case diagram, actors, functional and non-functional requirements and a brief explanation for selecting this system. The entity relationship diagram section comprises logical and physical ER diagrams with a brief explanation of the relationships with cardinalities between the entities. Within the table designs section, there are data dictionaries for each table in our database.

## Design Analysis process

User stories

Student

As a student, I want to be able to view detailed information about each course, including the course description, instructors. So that I can make informed decisions when selecting courses.

As a student, I want to have easy access to my academic records that includes grades, and progress reports, so that I can track my performance and academic progress.

As a student I want to register for a course so that I can get involves in new courses.

As a student I want to view payment detail so that I can get to know about financial information.

Lecturer

As a lecturer, I want to mark grades for each student, so that student track their academic progress.

As a lecturer I want to view my assigned courses so that I can be up to date for my weekly schedule for class.

As a lecturer, I want to view student records so that I can get to know about student information.

As a lecturer, I want to edit student grades so that I can correct my mistakes in marking the student so that student can get accurate grades.

As a lecturer, I want to generate the report card for a student so that admin can track student progress.

ADMIN

As an admin, I want to view student detail because I can be up to date for maintaining student information.

As an admin, I want to register a new student in a course so that new students can get enrolled in a course.

As an admin, I want to update payment detail so that admin can view every payment detail of the student

**List of cases**

Student:

 Register for a course: Students can register for a course they are interested in taking.  
View grade: Students can view their grades for a particular course they are enrolled in.  
View courses: Students can view all the courses that are available at the university.  
View payment detail: Students can view the payment details for the courses they are enrolled in.  
View Student detail: Students can view their details.

Lecturer:

 View courses: Lecturers can view the courses they are assigned to teach.  
Grade student in a course: Lecturers can grade students in a particular course they are assigned to teach.  
Generate the report card: Lecturers can generate the report card for a particular student in a course they are assigned to teach.

View student records:  Lecturers can view the records of all the students enrolled in the course.

Update student grade: lecturer can update the grade of a student in their course if there is any mistake in grading.

Admin:

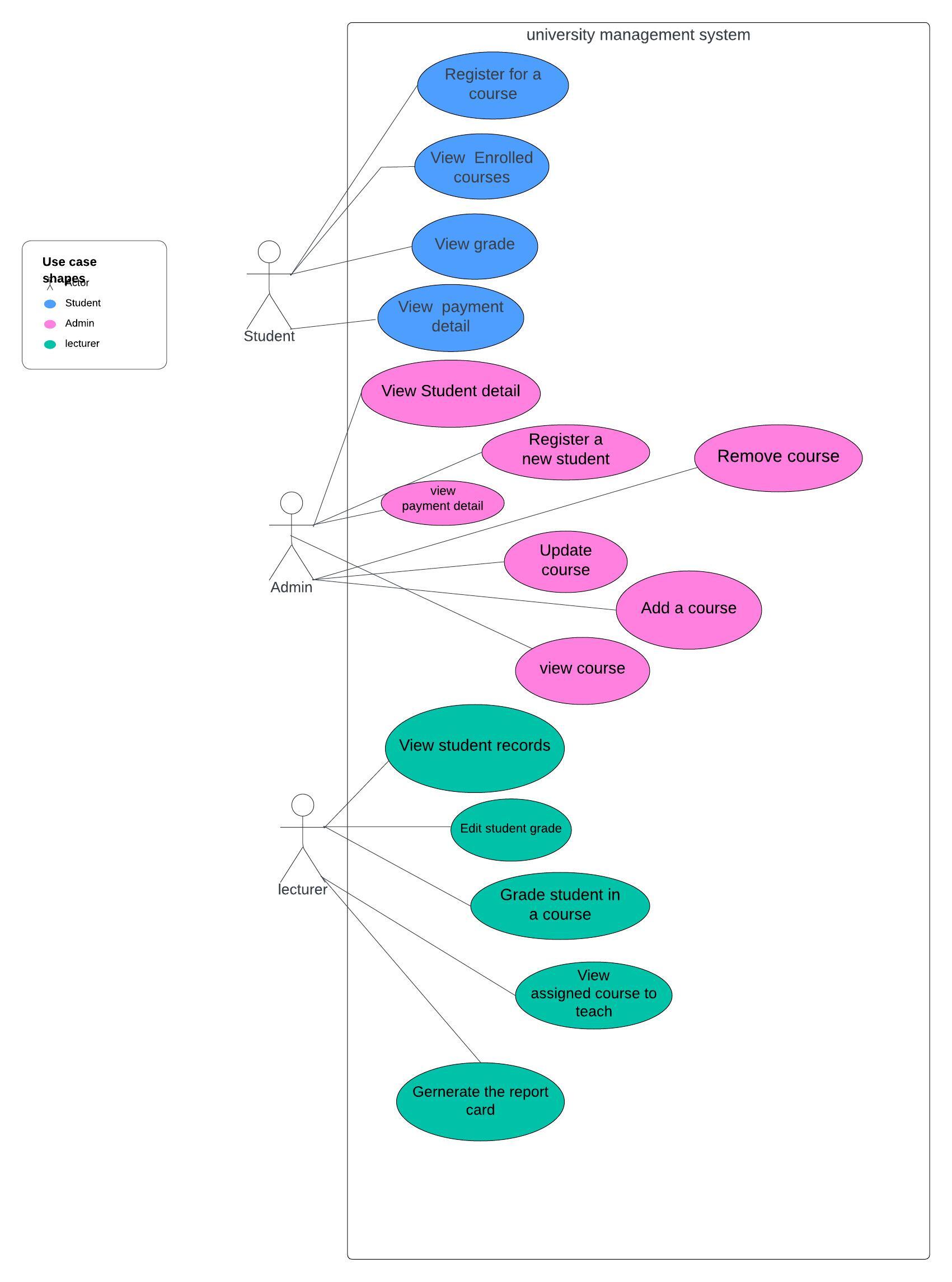
Register a new student: Administrators can register a new student at the university.  
View student records: Administrators can view the records of all the students enrolled in the university.  
View courses: Administrators can view all the courses that are available at the university.  
Update course: Administrators can update the details of a particular course.  
Add a course: Administrators can add a new course to the university.  
Remove course: Administrators can remove a course that is no longer offered by the university.

View payment details: administrator can view payment details of every student.

**Actor**

* Administrator
* Lecturer
* student

Use case diagram



## Functional and non-functionality

Functional Requirements

* The students and lecturers need to be able to view the course information from the system.
* A class should be assigned to each student from the system.
* The students need to be able to pay the fees for each year.
* The admin should be able to receive the fees paid by each student.

Non-Functional Requirements

* Security – The system should be able to protect students, lecturers and admin personal information.
* Scalability – The system should be able to manage the student capacity of each class.
* Reliability – The system should be available to use by students, lecturers and admin.

‌

## Software Development life Cycle

We are using the agile methodology for our system since we are constantly changing our system without a proper plan. This will make our project flexible, and our progress was repetitive. We worked on smaller sprints instead of trying to complete all the tasks in one sprint. At the end of each sprint, our team will provide feedback for the task completed and check whether each sprint goal is achieved. Based on the feedback, our team adjusted the plan for the next sprint. From agile methodology, we were able to provide feedback early, our team was able to adapt to changes quickly, it reduced the risk of unexpected problems and with continuous feedback, it helped our team to improve the project over time.

‌Final report vs proposal.

Scope and Purpose:

The proposal of creating a university management system is outlined by the main objectives.

The report elaborates on the design analysis process, and development methodology, and the concrete details would consist of data dictionaries and entity relationship diagrams.

Content Detail:

The suggestion provides just a generic presentation of functions and the anticipated benefits.

The report concludes detailing the user stories, use cases, functional and non-functional requirements, software development life cycle, data dictionary.

User Stories and Use Cases:

I have no idea how the proposal is very short that refers to user stories and use cases is there is use case and the user story where not referring to our system because many of the use story and cases cannot be implemented in the system at this stage.

The report comes with industry users' stories and use cases about the student's, lecturer's, and administrator's system, on which users could have a good understanding of the system.

Entity Relationship Diagram (ERD):

Proposal: the proposal includes the logical diagram that have course coordinator entity seperaty but as per the feedback we had put lecturer as course coordinator.

Final Report: the report includes the improved physical diagram that covers the changes and addition of new entities and attributes. .

Data Dictionary:

Proposal: The proposal looks vague because there are no data dictionaries.

Final Report: The final report comprises informative data dictionaries of entities, e.g. students, classes, payment schedules, courses, administration , lecturers, and departments; these dictionaries will help to understand the design of the database schema more profoundly.

Link of the repository

https://github.com/vivektalpada07/University\_Management\_System.git

## Entity relationship Diagram

University management system

A student is identified by the student ID, which is unique for them; and a name, email, and contact number also belong to them. This is done through the department ID of each student that belongs to a particular department.

The department is identified through it's Department ID and by it's Name.

A class has a number that classifies it as a class, the class has a capacity, and the class belongs to a block year. Each class is linked with a certain course, and this course is referred to as course ID. Firstly, sever classes can be linked to several students, and each can be linked to multiple classes.

An award is identified by a particular award id, and that student and admin are linked to it. Each fee differs by the amount and the date by which it is due.

A unique admin ID and a name and email address authenticate an admin. Every admin will be paired with many fees and the courses they can offer.

The course is identified by the course ID, and it has the specific name, credit, wet, duration, and level. Every course is assigned by one or more classes which are lectured by a particular lecturer from the class called lecturer ID.

A lecturer is indicated by a particular lecturer ID, the lecturer has a title, contact info, and email address, and is affiliated with a specific department. Lecturers can be equally can teach more than one course.

A diagram of a computer program

Description automatically generated with medium confidence

## Table design: data dictionary

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| entity Name | DescriptION | | | | |
| student | The student entity includes student personal information like name, email and contact with department information | | | | |
| Field Name | **Description** | **Data type** | **Key field** | **Constraints** | **Example** |
| student\_id | Unique id for each student | Text | PK | NOT NULL | S1, S2 |
| student\_name | Name of each student | Text |  | NOT NULL | John, Mary |
| student\_email | Email of each student | Text |  | NOT NULL | john@123.com |
| student\_contact | Contact number of each student | Integer |  | NOT NULL | 0245367891 |
| dept\_id | This is from Department table | Text | FK | NOT NULL | D1, D2 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| entity Name | DescriptION | | | | |
| class | The class entity includes the capacity of the class, student information and course information | | | | |
| Field Name | **Description** | **Data Type** | **Key Field** | **Constraints** | **Example** |
| class\_id | Unique id for each class | Text | PK | NOT NULL | C1, C2 |
| capacity | Capacity of students in a class | Integer |  | NOT NULL | 10, 15 |
| student\_id | This is from student table | Text | FK | NOT NULL | S1, S2 |
| course\_id | This is from Course table | Text | FK | NOT NULL | C1, C2 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| entity name | **DESCRIPTION** | | | | |
| fee | **This fee entity store fee information for each student.** | | | | |
| FIELD NAME | **Description** | **Data Type** | **Key Field** | **Constraints** | **Example** |
| fee\_id | **Unique code to identify the fee** | **Text** | **PK** | **NOT NULL** | F1 |
| amount | Amount need to pay | real |  | not null | 3455.00 |
| due\_date | Due date of the fees | **Text** |  | not null | 12-1-23-2024 |
| year | **Fees for each year.** | **Text** |  | **Not null** | **2024** |
| student\_id | **Student\_id is taken from a student table** | **Text** | f.k | **Not null** | **Sf23** |
| admin\_id | **Admin\_id is taken from a Admin table** | **Text** | f.k | **Not nul** | **f233** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| entity name | **course** | | | | |
| Course | **This fee entity store fee information for each student.** | | | | |
| FIELD NAME | **Description** | **Data Type** | **Key Field** | **Constraints** | **Example** |
| course\_id | **Unique code to identify the course** | text | PK | NOT NULL | Bit2@45 |
| course\_name | Name of the course | text |  | not null | Bit1 |
| credit | **Credit of the course** | integer |  | not null | 12 |
| duration | **Duration of the course** | **text** |  | Not null | **3 years** |
| level | **Level of the course** | **int** |  | Not null | **6** |
| lecturer\_id | **Lecturer is taken from lecturer table** | **Text** | f.k | Not null | **Bit23** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ENTITY NAME** | **DESCRIPTION** | | | | |
| **admin** | **This admin entity stores the information of admin** | | | | |
| **Field\_name** | **Description** | **Data Type** | **Key Field** | **Constraints** | **Example** |
| **Admin\_id** | **Unique code to identify the bank** | text | **PK** | **Not null** | **a1** |
| **admin\_name** | **Name of the admin** | **Text** |  | not nulls | **avi** |
| **Admin\_email** | **email from the each Admin** | **Text** |  | not nulls | **admin123@gmail.com** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ENTITY NAME** | **DESCRIPTION** | | | | |
| **Lecturer** | **This admin entity stores the information related to the lecturer** | | | | |
| **Field\_name** | **Description** | **Data Type** | **Key Field** | **Constraints** | **Example** |
| **lecturer\_id** | **Lecturer id is a unique id to identify the lecturer** | **Text** | **PK** | **Not null** | **L1** |
| **Lecturer\_email** | **email of the each lecturer** | **Text** |  | not null | **Tanvir123@gmail.com** |
| **lecturer\_contact** | **Contact of each lecturer** | **Text** |  | not nulls | **+61 0224281919** |
| **dept\_id** | **dept\_id is taken from the department entity** | **text** | Fk | **not nulls** | **D1** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ENTITY NAME** | **DESCRIPTION** | | | | |
| **Department** | This departemnt entity stores the information of department related to the course. | | | | |
| **Field name** | **Description** | **Data Type** | **Key Field** | **Constraints** | **Example** |
| **Dept\_id** | **Unique code to identify the bank** | text | **PK** | **Not null** | **D1** |
| **Dept\_name** | **Name of the admin** | **Text** |  | not nulls | **Bachelor of it** |

## Contribution of team for the report

**Vivek**

* **Write the use cases for the report.**
* **Rewrite the user stories for the report.**
* **Design physical ER Diagram and Use case diagram.**
* **Data dictionary for fee and Course**
* **Write Actors.**
* **Write final report vs proposal**
* **Made the Database (Combined all the queries to make the database).**
* **Finalise the report.**

**Chathuni**

* **Write the software development life cycle.**
* **Design the logical ERD.**
* **Data dictionary for the student and class**
* **Write the Function and non-Function.**
* **Write Executive Summary**

**Mandeep**

* **Data Dictionary for the Department.**
* **Write the user story but not related to the system so the leader rewrite it.**

**Gagandeep**

* **Data Dictionary for Admin and Lecturer**
* **Help leader to finalise use cases**

## References

Singla, C. (2020, April 28). Functional vs Non-Functional Requirements. GeeksforGeeks. <https://www.geeksforgeeks.org/functional-vs-non-functional-requirements/>