

**A REPORT**  
**ON**  
**Grading Management System**

**Prakhar Mundra**

**2021A7PS2694P**

**Nishant Singh**

**2021A7PS2689P**



**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE,**  
**PILANI (Rajasthan)**

For the fulfilment of the Assignment in the course- **Database**  
**Systems- CS F212**

April 2023

## Anti Plagiarism Statement

I declare in my honour that what has been written in this work has been written by me and that, with the exception of quotations, no part has been copied from scientific publications, the Internet or from research works already presented in the academic field by me or by other students. In the case of parts taken from scientific publications, from the Internet or from other documents, I have expressly and directly indicated the source at the end of the quotation or at the foot of the page. I also declare that I have taken note of the sanctions provided for in case of plagiarism by the current Study Regulations

Name	ID
Prakhar Mundra	2021A7PS2694P
Nishant Singh	2021A7PS2689P

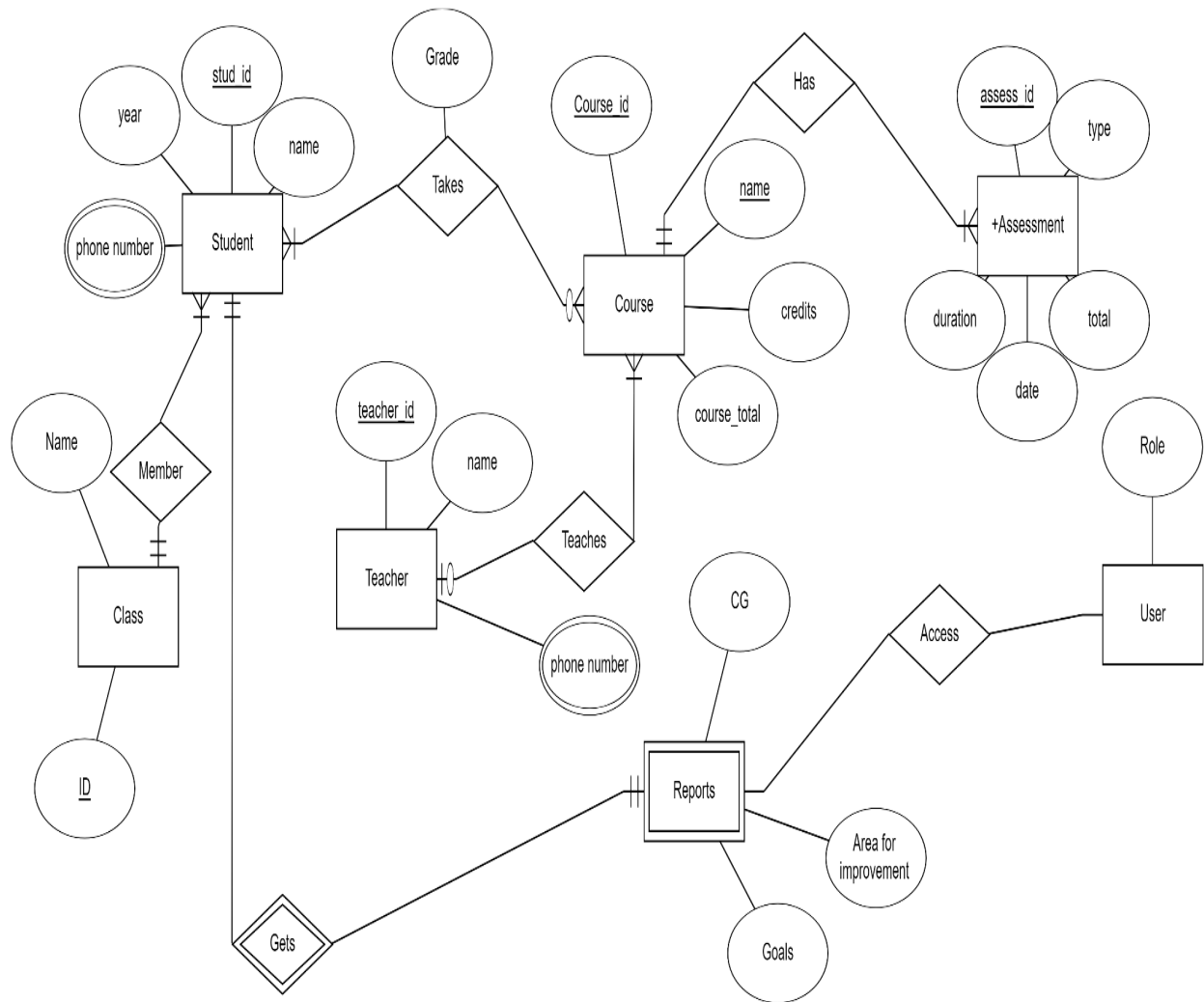
## Video Link:

[https://drive.google.com/drive/folders/1Wr\\_FEQSLvi8\\_0y2ULDU0YQ4jUi4o0GhV?usp=share\\_link](https://drive.google.com/drive/folders/1Wr_FEQSLvi8_0y2ULDU0YQ4jUi4o0GhV?usp=share_link)

## Github Link:

<https://github.com/PrakharMundra/DBMS->

## ER Diagram:



## **Documentation for ER diagram**

Grading Management System is a software for School and colleges that allows to effectively store, manage, and report grading data. Teachers will be able to enter and amend grades using the system's user-friendly interface. It will help to compile reports on student performance, see grade records for numerous classes, and monitor students' advancement over time thanks to the technology.

A thorough database with all the necessary data on students, classes, grades, and exams will be part of the system. Also, the system will have access restrictions to guarantee that only authorised users may access it. The required access controls will be in place to protect their privacy and guarantee data security, and the students will also have access to the database to view their performance and grades.

Also, the system will produce automated reports that include comprehensive data on each student's performance, development, and any areas that require improvement. This will make it easier for schools to monitor students' progress, spot areas for development, and make sure that they are achieving their academic objectives.

### Entities:

1. Student entity : Stores student information and it has the following attributes:
  - stud\_id(Primary Key)
  - name
  - Phone number(multivalued
  - year
2. Teacher entity : Stores teacher information and it has the following attributes:
  - teacher\_id(Primary Key)
  - name
  - Phone number(multivalued)
3. Course entity : Stores information about a particular course. It has following attributes:
  - course\_id(Primary Key)

- name(Unique)
  - Credits
  - course\_total
4. Class entity : Comprises of many students and has the following attributes:
    - ID (Primary Key)
    - name
  5. Assessment : Provides information about assessments of a particular course. It has attributes:
    - assess\_id(Primary Key)
    - type
    - Total
    - Date
    - duration
  6. Reports entity(Weak) : Comprises of students report card and has the following attributes:
    - CG
    - Area of Improvement
    - Goals
  7. Users entity: Comprises of parents and admin and has the following attributes:
    - role

### Relationships:

1. Takes : Describes a many-to-many relationship between student entity and course entity. A student can take part in many courses and a course can have many students. It also has total participation of student as a student must take atleast one course. It has course\_id and stud\_id as the primary key. It also has grade attribute.
2. Teaches: Describes one-to-many relationship between teacher entity and course entity. A teacher can teach many courses but a course can be taught by only one teacher. Also it has total participation of a course as a course must be taught by atleast one teacher.
3. Has : Describes a one-to-many relationship between course entity and assessment entity. A course can have many assessments and an assessment can belong to exactly one course. Also it has total participation of both entities as a course must have assessment and an assessment must be a part of a course.
4. Member: Describes one-to-many relationship between class entity and student entity. One class can have many student but a student can belong to only one class. Also it involves

total participation of both entities as a class must have atleast one student and a student must belong to a class.

5. Gets: It is a weak one one relationship.A student has a report and a report belongs to a student.Also a student must have a report and a report must belong to a student.
6. Access: A user has access to a report of a particular student by entering student credentials and their role example parent,admin etc.

## **Relational Schema before normalization**

Student

<u>stud_id</u>	name	year	phone number	ID
----------------	------	------	--------------	----

Teacher

<u>teacher_id</u>	name	phone number
-------------------	------	--------------

Course

<u>course_id</u>	<u>name</u>	credits	course_total	teacher_id
------------------	-------------	---------	--------------	------------

Assessment

<u>assess_id</u>	type	total	date	duration	course_id
------------------	------	-------	------	----------	-----------

User

role
------

Reports

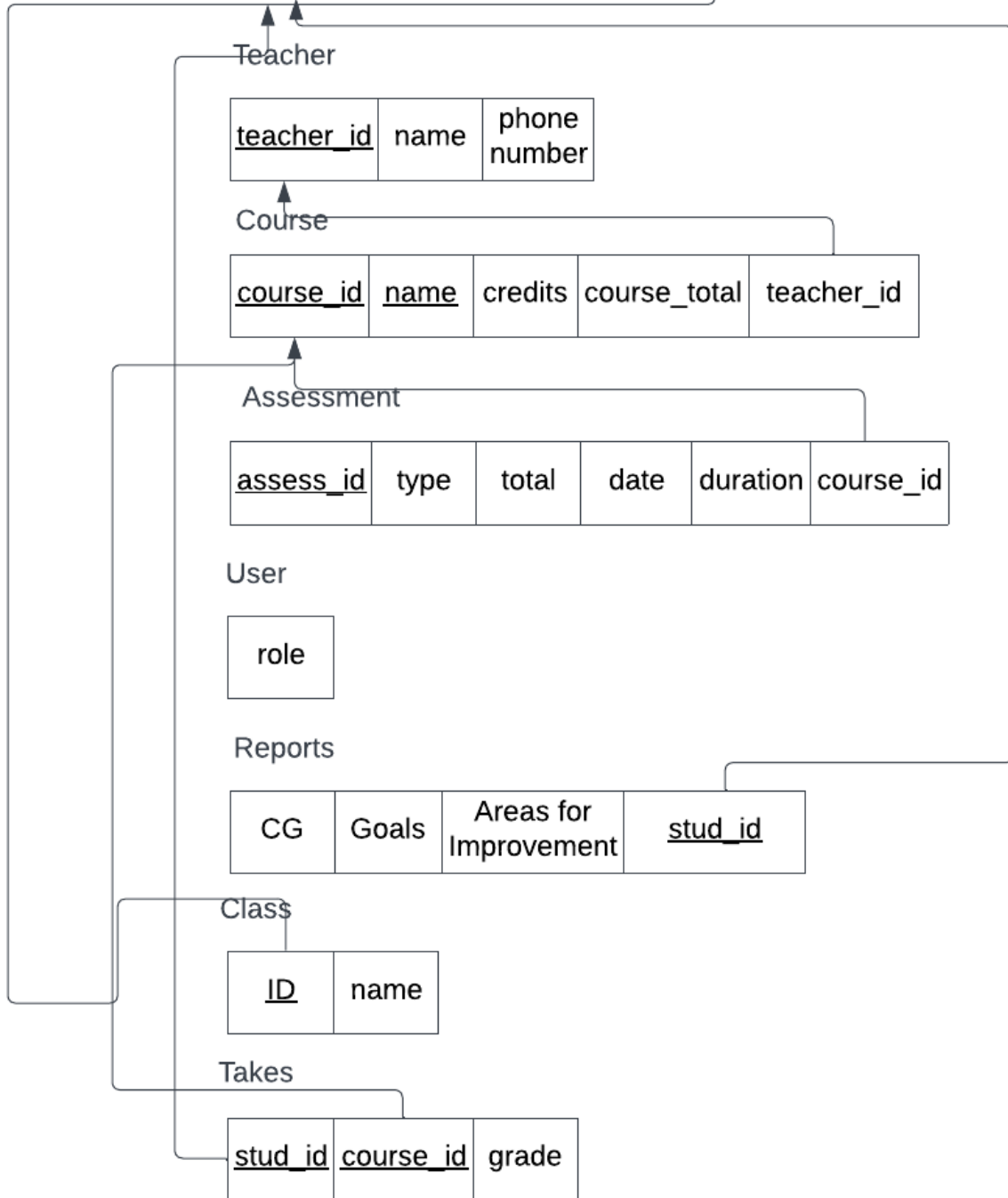
CG	Goals	Areas for Improvement	<u>stud_id</u>
----	-------	-----------------------	----------------

Class

<u>ID</u>	name
-----------	------

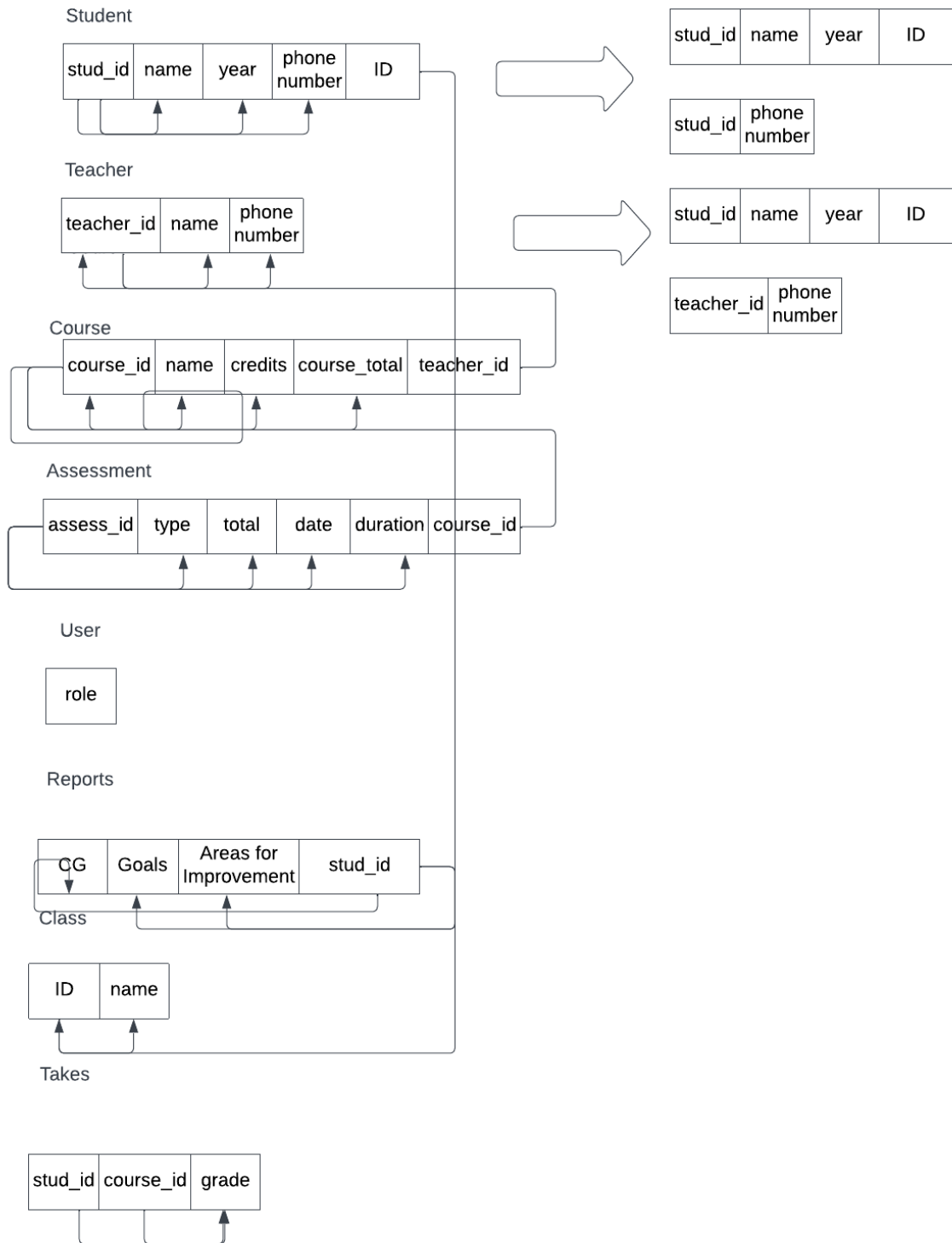
Takes

<u>stud_id</u>	<u>course_id</u>	grade
----------------	------------------	-------





## Functional Dependencies



## **Conversion to 3NF**

Converting to 1NF:

- On looking at the relational schema, we observe that the student phone number and teacher phone number are multivalued attributes
- Hence, we have to create separate tables which consist of their individual ids and phone numbers

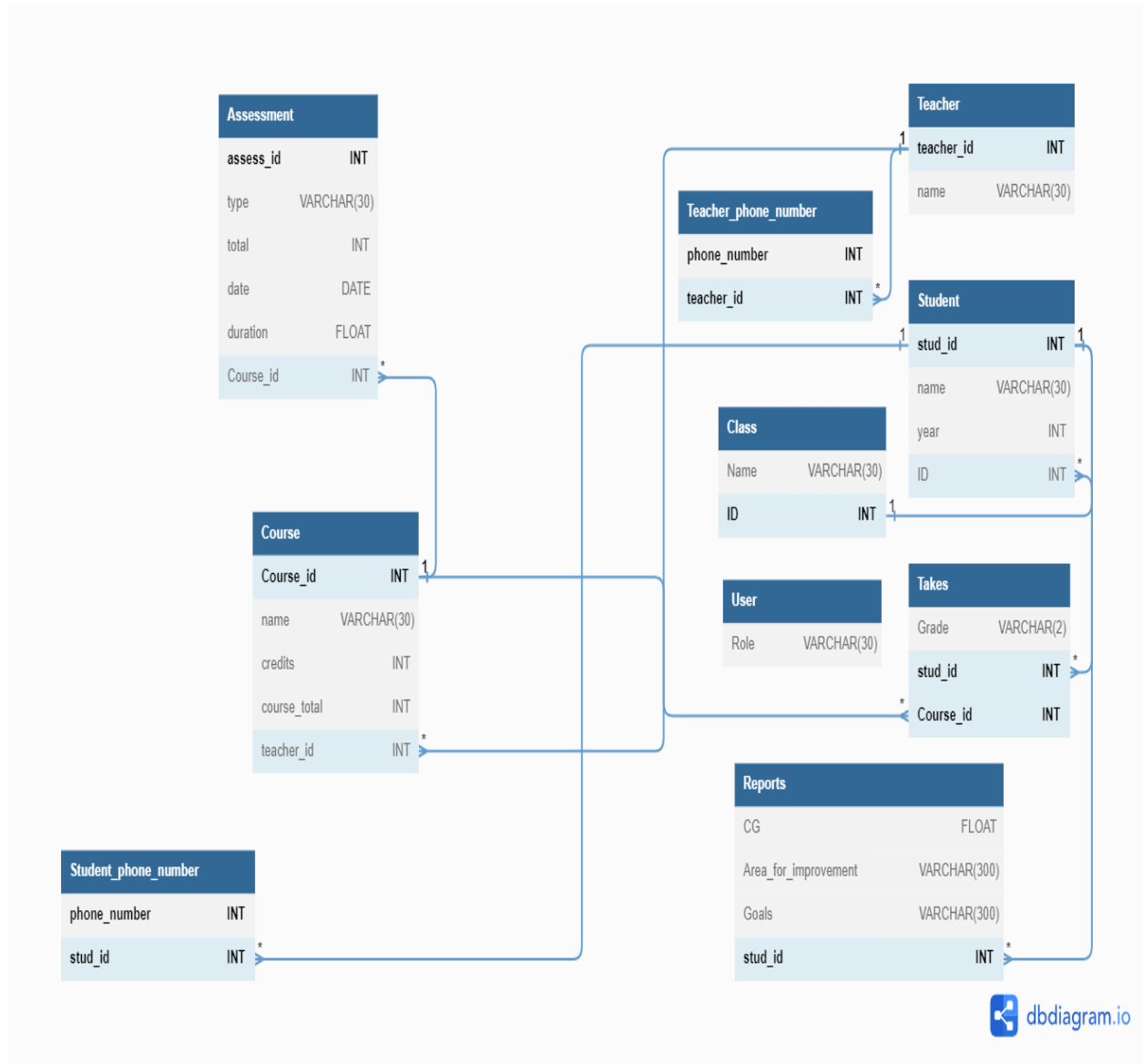
Converting to 2NF:

- On looking at the relational schema, we observe that no sub-part of any candidate key of any table could uniquely identify any other attributes i.e every non-prime attribute is fully functional dependent.
- Hence, we can conclude that the relational schema is already in 2NF.

Converting to 3NF:

- On looking at the relational schema, we observe that there are no transitive dependencies. That is, if any non-key attribute depends on another non-key attribute, it should be moved to a separate table with the attribute on which it depends as the primary key.
- Hence, we can conclude that the relational schema is already in 3NF.

## Relational Schema after normalization



## SQL Queries

Q1. Create all the necessary tables, such as student, teacher, course, class, etc.

```
create database grading;
use grading;
CREATE TABLE IF NOT EXISTS Teacher
(
    teacher_id INT NOT NULL,
    name VARCHAR(30) NOT NULL,
    PRIMARY KEY (teacher_id)
);
```

```
CREATE TABLE IF NOT EXISTS Class
(
    Name VARCHAR(30) NOT NULL,
    ID INT NOT NULL,
    PRIMARY KEY (ID)
);
```

```
CREATE TABLE IF NOT EXISTS User
(
    Role VARCHAR(30) NOT NULL
);
```

```
CREATE TABLE IF NOT EXISTS Teacher_phone_number
(
    phone_number INT NOT NULL ,
    teacher_id INT NOT NULL,
    PRIMARY KEY (phone_number, teacher_id),
    FOREIGN KEY (teacher_id) REFERENCES Teacher(teacher_id) ON DELETE
    CASCADE
);
```

```
CREATE TABLE IF NOT EXISTS Student
(
    stud_id INT NOT NULL,
    name VARCHAR(30) NOT NULL,
    year INT NOT NULL,
```

```
ID INT NOT NULL,  
PRIMARY KEY (stud_id),  
FOREIGN KEY (ID) REFERENCES Class(ID) ON DELETE CASCADE  
);
```

```
CREATE TABLE IF NOT EXISTS Course  
(  
    Course_id INT NOT NULL,  
    name VARCHAR(30) NOT NULL unique,  
    credits INT NOT NULL,  
    course_total INT NOT NULL,  
    teacher_id INT,  
    PRIMARY KEY (Course_id),  
    FOREIGN KEY (teacher_id) REFERENCES Teacher(teacher_id) ON DELETE  
CASCADE  
);
```

```
CREATE TABLE IF NOT EXISTS Assessment  
(  
    assess_id INT NOT NULL,  
    type VARCHAR(30) NOT NULL,  
    total INT NOT NULL,  
    date DATE NOT NULL,  
    duration FLOAT NOT NULL,  
    Course_id INT NOT NULL,  
    PRIMARY KEY (assess_id),  
    FOREIGN KEY (Course_id) REFERENCES Course(Course_id) ON DELETE  
CASCADE  
);
```

```
CREATE TABLE IF NOT EXISTS Reports  
(  
    CG FLOAT CHECK(CG >= 2.0 AND CG <= 10.0),  
    Area_for_improvement VARCHAR(300) ,  
    Goals VARCHAR(300),  
    stud_id INT NOT NULL,  
    PRIMARY KEY (stud_id),  
    FOREIGN KEY (stud_id) REFERENCES Student(stud_id) ON DELETE CASCADE  
);
```

```

CREATE TABLE IF NOT EXISTS Takes
(
    Grade VARCHAR(2) ,
    stud_id INT NOT NULL,
    Course_id INT NOT NULL,
    PRIMARY KEY (stud_id, Course_id),
    FOREIGN KEY (stud_id) REFERENCES Student(stud_id),
    FOREIGN KEY (Course_id) REFERENCES Course(Course_id) ON DELETE
    CASCADE
);

```

```

CREATE TABLE IF NOT EXISTS Student_phone_number
(
    phone_number INT NOT NULL,
    stud_id INT NOT NULL,
    PRIMARY KEY (phone_number, stud_id),
    FOREIGN KEY (stud_id) REFERENCES Student(stud_id) ON DELETE CASCADE
);

```

- The “teacher” table contains information about id and name of the teacher with id being the primary key.
- The “class” table contains information about the name and id of the class with id being the primary key.
- The “user” table can have different kinds of roles. For example-parent and admin.
- The “teacher\_phone\_number” table contains the phone numbers of teachers with the combination of phone number and teacher id being the primary key. This table is created as a result of phone number of teacher being multivalued.
- The “student” table contains information about the name, student id, year and class id of the student with student id being the primary key.
- The table “course” contains the information about the course id, name of the course, credits, total of the course and id of the teacher with course id being the primary key.
- The table “assessment” contains the information about the assess id (to recognize the type of assessment), type of assessment, total of the assessment, date of conduction and the duration of the assessment, and the course id to recognize the course it is associated to, with assess id being the primary key.
- The table “reports” contains the CG of the student, Area for improvement and goals for the student. There is also student id to recognize the student and it acts as both foreign key and primary key of the table.
- The table “takes” contains the id of both the student and course such that the student gets enrolled in the course. Apart from that there is also a grade attribute to show the grade

achieved by the student in the respective course. The id of both the student and teacher acts as the primary key and the foreign key to this table.

- The table “student\_phone\_number” contains the phone numbers of students with the combination of phone number and student id being the primary key. This table is created as a result of phone number of student being multivalued.

### Result of the Query

#	Time	Action	Message	Duration / Fetch
✓ 8	21:26:05	CREATE TABLE IF NOT EXISTS Student ( stud_id INT NOT NULL, name VARC...	0 row(s) affected	0.016 sec
✓ 9	21:26:05	CREATE TABLE IF NOT EXISTS Course ( Course_id INT NOT NULL, name VAR...	0 row(s) affected	0.015 sec
✓ 10	21:26:05	CREATE TABLE IF NOT EXISTS Assessment ( assess_id INT NOT NULL, type V...	0 row(s) affected	0.016 sec
✓ 11	21:26:05	CREATE TABLE IF NOT EXISTS Reports ( CG FLOAT CHECK(CG >= 2.0 AND CG...	0 row(s) affected	0.016 sec
✓ 12	21:26:05	CREATE TABLE IF NOT EXISTS Takes ( Grade VARCHAR(2) , stud_id INT NOT...	0 row(s) affected	0.031 sec
✓ 13	21:26:05	CREATE TABLE IF NOT EXISTS Student_phone_number ( phone_number INT N...	0 row(s) affected	0.016 sec

### Insertion of values into the table

Inserting teacher into the database

```
INSERT INTO Teacher (teacher_id,name) VALUES (1,'Ms. Sinha');
INSERT INTO Teacher (teacher_id,name) VALUES (2,'Mr. Singh');
INSERT INTO Teacher (teacher_id,name) VALUES (3,'Dr. Sejpal');
INSERT INTO Teacher (teacher_id, name) VALUES (4, 'Mrs. Nayyar');
INSERT INTO Teacher (teacher_id, name) VALUES (5, 'Mr. Khatri');
INSERT INTO Teacher (teacher_id, name) VALUES (6, 'Ms. Sood');
INSERT INTO Teacher (teacher_id, name) VALUES (7, 'Dr. Patel');
INSERT INTO Teacher (teacher_id, name) VALUES (8, 'Mr. Sharma');
INSERT INTO Teacher (teacher_id, name) VALUES (9, 'Ms. Pratap');
INSERT INTO Teacher (teacher_id, name) VALUES (10, 'Mrs. Agarwal');
```

Using the above query we insert the values of the attributes of the teachers into the table teachers with attributes valued at NULL.

Inserting class into the database

```
INSERT INTO Class (ID,name) VALUES (1,'CS');
INSERT INTO Class (ID,name) VALUES (2,'Math');
INSERT INTO Class (ID,name) VALUES (3,'Electrical');
INSERT INTO Class (ID, name) VALUES (4, 'English');
INSERT INTO Class (ID, name) VALUES (5, 'Science');
```

```
INSERT INTO Class (ID, name) VALUES (6, 'History');
INSERT INTO Class (ID, name) VALUES (7, 'Physical Education');
INSERT INTO Class (ID, name) VALUES (8, 'Art');
INSERT INTO Class (ID, name) VALUES (9, 'Music');
INSERT INTO Class (ID, name) VALUES (10, 'Foreign Language');
```

Using the above query we add different classes to the database.

Inserting different roles in the user table

```
INSERT INTO User (Role) VALUES ('Admin');
INSERT INTO User (Role) VALUES ('Student');
I  INSERT INTO User (Role) VALUES ('Teacher');
INSERT INTO User (Role) VALUES ('Parent');
INSERT INTO User (Role) VALUES ('Guardian');
```

Using the above query we can give different roles to the user which can help further in giving limited access to a particular role.

Inserting phone numbers of teachers into the database

```
INSERT INTO Teacher_phone_number (phone_number, teacher_id) VALUES (123456789, 1);
INSERT INTO Teacher_phone_number (phone_number, teacher_id) VALUES (234567890, 2);
INSERT INTO Teacher_phone_number (phone_number, teacher_id) VALUES (345678901, 3);
INSERT INTO Teacher_phone_number (phone_number, teacher_id) VALUES (456789012, 4);
INSERT INTO Teacher_phone_number (phone_number, teacher_id) VALUES (567890123, 5);
INSERT INTO Teacher_phone_number (phone_number, teacher_id) VALUES (678901234, 6);
INSERT INTO Teacher_phone_number (phone_number, teacher_id) VALUES (789012345, 7);
INSERT INTO Teacher_phone_number (phone_number, teacher_id) VALUES (890123456, 8);
INSERT INTO Teacher_phone_number (phone_number, teacher_id) VALUES (901234567, 8);
INSERT INTO Teacher_phone_number (phone_number, teacher_id) VALUES (123450987, 10);
```

Inserting 10 students into the database

```
INSERT INTO Student (stud_id, name, year, ID) VALUES (1, 'Aditya', 1, 1);
INSERT INTO Student (stud_id, name, year, ID) VALUES (2, 'Rahul', 4, 2);
INSERT INTO Student (stud_id, name, year, ID) VALUES (3, 'Suhana', 3, 3);
INSERT INTO Student (stud_id, name, year, ID) VALUES (4, 'Nidhi', 1, 4);
```



```
INSERT INTO Student (stud_id, name, year, ID) VALUES (5, 'Shahrukh', 1, 5);
INSERT INTO Student (stud_id, name, year, ID) VALUES (6, 'Ishika', 3, 6);
INSERT INTO Student (stud_id, name, year, ID) VALUES (7, 'Rishi', 2, 7);
INSERT INTO Student (stud_id, name, year, ID) VALUES (8, 'Manan', 2, 5);
INSERT INTO Student (stud_id, name, year, ID) VALUES (9, 'Vedant', 2, 6);
INSERT INTO Student (stud_id, name, year, ID) VALUES (10, 'Parth', 4, 7);
```

Inserting 10 courses into the database

```
INSERT INTO Course (Course_id, name, credits, course_total, teacher_id)
VALUES (1, 'Database Systems', 4, 200, 1),
(2, 'Data Structures', 3, 150, 2),
(3, 'Operating Systems', 4, 250, 3),
(4, 'Algorithms', 4, 200, 1),
(5, 'Web Development', 3, 200, 2),
(6, 'Software Engineering', 3, 200, 3),
(7, 'Computer Networks', 4, 250, 1),
(8, 'Programming Languages', 3, 150, 2),
(9, 'Computer Graphics', 4, 250, 3),
(10, 'Artificial Intelligence', 4, 300, 1);
```

Inserting 10 assessments of courses into the database

```
INSERT INTO Assessment (assess_id, type, total, date, duration, Course_id)
VALUES (1, 'Mid-term Exam', 50, '2022-04-25', 2.5, 1),
(2, 'Final Exam', 100, '2022-06-20', 3.0, 1),
(3, 'Quiz 1', 20, '2022-03-15', 1.0, 2),
(4, 'Quiz 2', 25, '2022-04-05', 1.5, 2),
(5, 'Mid-term Exam', 50, '2022-04-25', 2.5, 3),
(6, 'Final Exam', 100, '2022-06-20', 3.0, 3),
(7, 'Quiz 1', 20, '2022-03-15', 1.0, 4),
(8, 'Quiz 2', 25, '2022-04-05', 1.5, 4),
(9, 'Mid-term Exam', 50, '2022-04-25', 2.5, 5),
(10, 'Final Exam', 100, '2022-06-20', 3.0, 5);
```

Inserting reports of 10 students into the database

```
INSERT INTO Reports (CG, Area_for_improvement, Goals, stud_id)
VALUES (8.5, 'Improvement in writing skills', 'To improve coding skills', 1),
(7.0, 'Need to improve time management', 'To get better grades', 2),
(9.0, 'Need to improve communication skills', 'To work on team projects', 3),
(6.5, 'Need to improve understanding of concepts', 'To study regularly', 4),
(8.0, 'Need to improve presentation skills', 'To improve problem-solving skills', 5),
(7.5, 'Need to improve analytical skills', 'To learn new technologies', 6),
(9.5, 'Need to improve coding skills', 'To work on open-source projects', 7),
(8.0, 'Need to improve debugging skills', 'To participate in coding competitions', 8),
(7.5, 'Need to improve understanding of algorithms', 'To practice more problems', 9),
(9.0, 'Need to improve team management skills', 'To work on group projects', 10);
```

Inserting the values into the takes table of the database

```
('A', 1, 1),
('B', 2, 1),
('C', 3, 1),
('A-', 4, 4),
('B+', 5, 5),
('B', 6, 6),
('A+', 7, 1),
('C-', 8, 8),
('B-', 9, 1),
('A-', 10, 1);
```

Inserting the phone number of 10 students into the database

```
(123457890, 1),
(234568901, 2),
(345689012, 3),
(456890123, 4),
(568901234, 5),
(689012345, 6),
(890123456, 7),
(901234567, 8),
(012345678, 9),
(234567809, 10);
```

## Results of the above query

#	Time	Action	Message	Duration / Fetch
58	21:47:24	INSERT INTO Student (stud_id, name, year, ID) VALUES (10, 'Parth', 4, 7)	1 row(s) affected	0.000 sec
59	21:47:24	INSERT INTO Course (Course_id, name, credits, course_total, teacher_id) VALUES ...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.015 sec
60	21:47:24	INSERT INTO Assessment (assess_id, type, total, date, duration, Course_id) VALUE...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.000 sec
61	21:47:24	INSERT INTO Reports (CG, Area_for_improvement, Goals, stud_id) VALUES (8.5, 1...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.000 sec
62	21:47:24	INSERT INTO Takes (Grade, stud_id, Course_id) VALUES ('A', 1, 1), ('B', 2, 1), ('C', ...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.016 sec
63	21:47:24	INSERT INTO Student_phone_number (phone_number, stud_id) VALUES (1234578...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.000 sec

Q2. Insert a new student into the students table.

```
INSERT INTO Student (stud_id, name, year, ID)
VALUES (11, 'Prakhar Mundra', 2023, 1);
```

This inserts a new student with a student ID of 11, name of Prakhar Mundra, year of 2023, and class ID of 1

## Result of the query

Select \* from student where stud\_id=11

	stud_id	name	year	ID
▶	11	Prakhar Mundra	2023	1

Q3. Insert a new teacher into the teachers table.

```
INSERT INTO Teacher (teacher_id, name)
VALUES (11, 'Dr.Amit Dua');
```

This inserts a new teacher with a teacher ID of 11 and name of Dr.Amit Dua.

## Result of the query

Select \* from teacher where teacher\_id=11

	teacher_id	name
▶	11	Dr.Amit Dua

Q4. Update a student's information in the students table.

```
UPDATE Student
SET year = 2024
WHERE stud_id = 11;
```

This updates the year of the student with a student ID of 11 to 2024.

#### **Result of the query**

```
select year from student
where stud_id=11;
```

	year
▶	2024

Q5. Delete a teacher from the teachers table.

```
DELETE FROM Teacher
WHERE teacher_id = 11;
```

This deletes the teacher with a teacher ID of 11.

#### **Result of the query**

```
Select * from teacher where teacher_id=11
```

	teacher_id	name
*	NULL	NULL

Q6. Retrieve all the information about a specific student.

```
SELECT *
FROM Student
WHERE stud_id = 1;
```

This retrieves all the information about the student with a student ID of 1.

**Result of the query**

	stud_id	name	year	ID
▶	11	Prakhar Mundra	2024	1

Q7. Retrieve the number of students who scored above a certain grade for a specific course for a specific class.

```
SELECT COUNT(*)  
FROM Takes  
WHERE Grade > 'B' AND Course_id = 1 AND stud_id IN  
(SELECT stud_id FROM Student WHERE ID = 1);
```

This retrieves the number of students who scored above a B in the course with a course ID of 1 and in the class with a class ID of 1

**Result of the query**

	COUNT(*)
▶	1

8. Retrieve the lowest grade for a specific course for a specific class.

```
SELECT MIN(Grade)  
FROM Takes  
WHERE Course_id = 1 AND stud_id IN  
(SELECT stud_id FROM Student WHERE ID = 1);
```

This retrieves the lowest grade in the course with a course ID of 1 and in the class with a class ID of 1.

**Result of the query**

	MIN(Grade)
▶	A

Q9. Retrieve the progress of a student.

```
SELECT Reports.CG, Reports.Area_for_improvement, Reports.Goals
FROM Reports
WHERE stud_id = 1;
```

This retrieves the CG (cumulative grade) of the student with a student ID of 1, as well as their areas for improvement and goals.

**Result of the query**

	CG	Area_for_improvement	Goals
▶	8.5	Improvement in writing skills	To improve coding skills

Q10. Update a student's grade in a course.

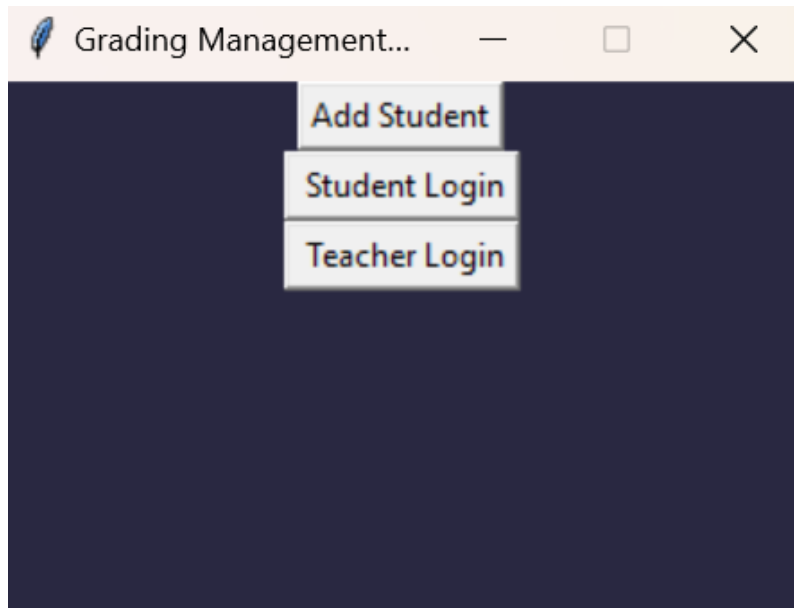
```
UPDATE Takes
SET Grade = 'A'
WHERE stud_id = 1 AND Course_id = 1;
```

This updates the grade of the student with a student ID of 1 in the course with a course ID of 1 to an A.

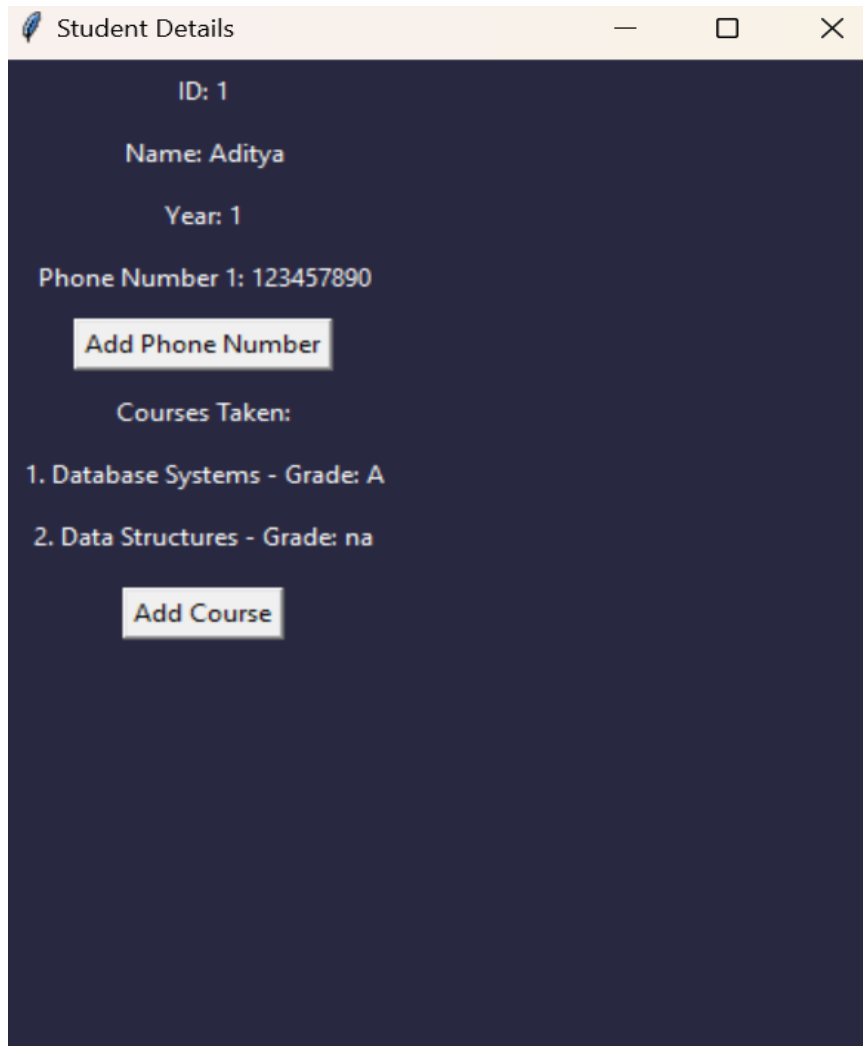
**Result of the query**

```
select grade from takes where stud_id=1 and course_id=1;
```

	grade
▶	A



This is the main and starting window of the program. In the above window we can either add students or login in either as a student or teacher using the credentials with which they have been registered in the database.



A screenshot of a web application window titled "Student Details". The window has a dark blue background and a light orange header bar with a feather icon, a minus sign, a square icon, and a close button. The content area displays the following information: "ID: 1", "Name: Aditya", "Year: 1", and "Phone Number 1: 123457890". Below the phone number is a button labeled "Add Phone Number". Underneath is the section "Courses Taken:" followed by a list: "1. Database Systems - Grade: A" and "2. Data Structures - Grade: na". At the bottom of the list is a button labeled "Add Course".

Student Details

ID: 1

Name: Aditya

Year: 1

Phone Number 1: 123457890

Add Phone Number

Courses Taken:

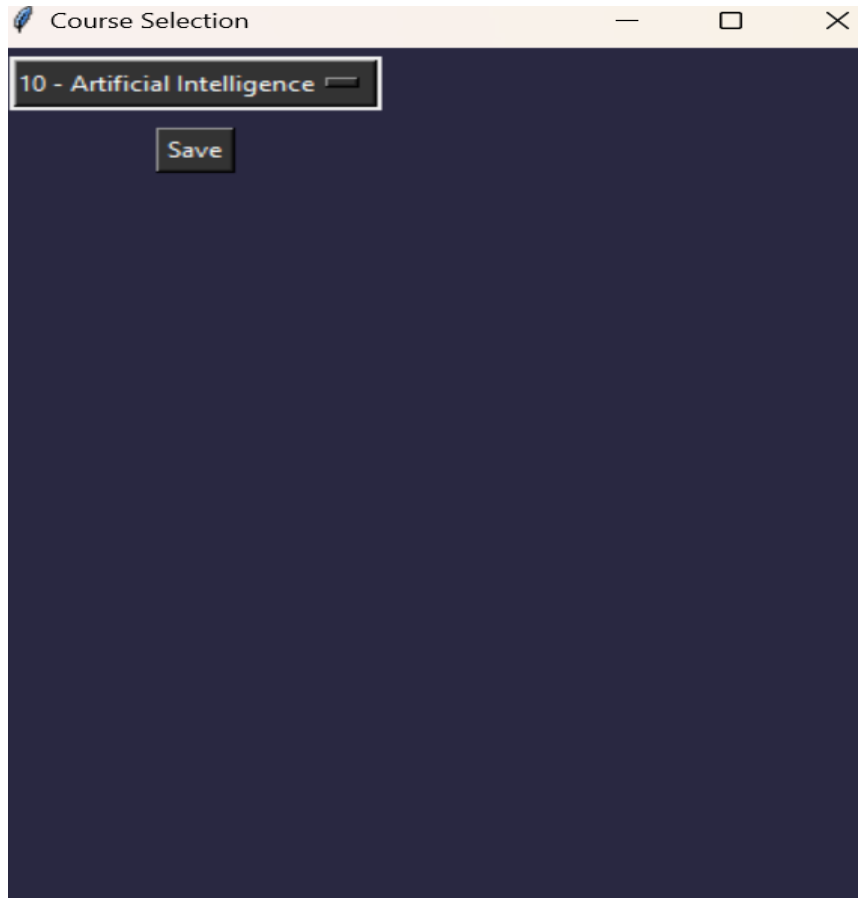
1. Database Systems - Grade: A

2. Data Structures - Grade: na

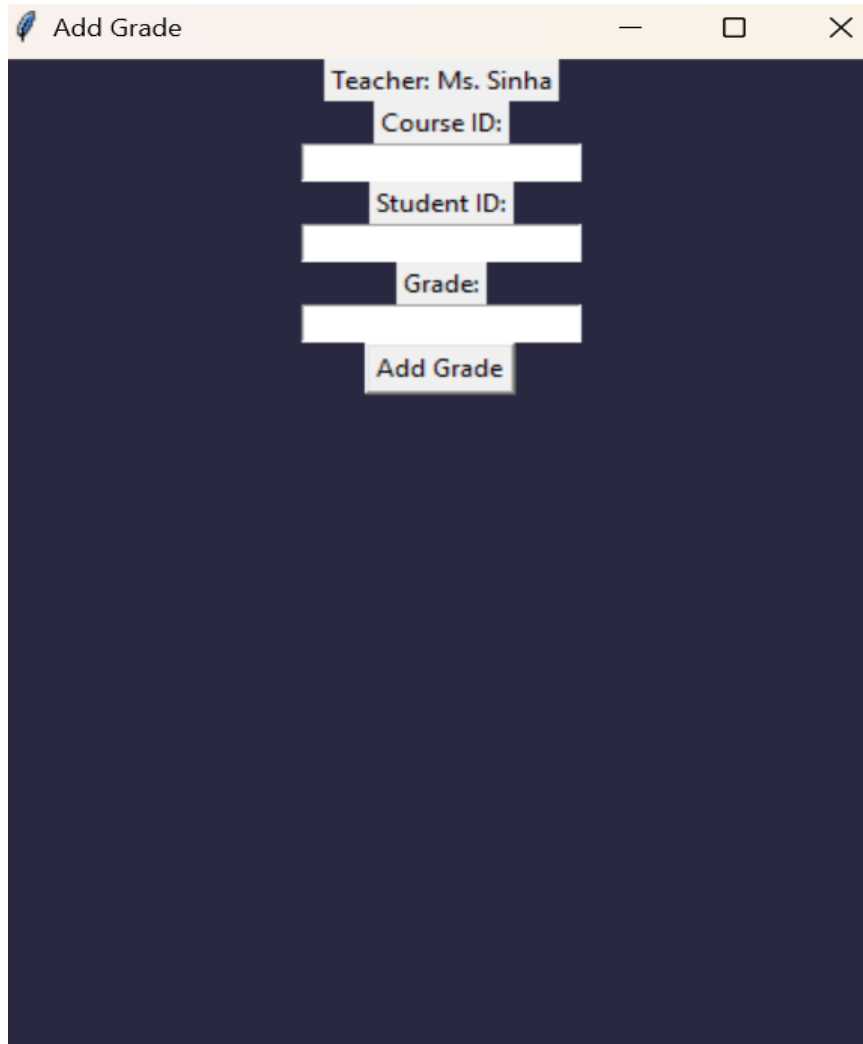
Add Course

When the user logs in as a student this window appears and details of the students are displayed. Apart from that there are two buttons which can be used to add phone number or to add courses.





When the add courses button is pressed the above window appears and a dropdown menu is given where all the available options are given and the student is made to choose from it. After pressing the save button the course is added to the student details and the student is added with the respective student id and course id is added to the takes table.

A screenshot of a web application window titled "Add Grade". The window has a dark blue background. In the center, there is a vertical stack of white input fields and a button. The first field is labeled "Teacher: Ms. Sinha". The second field is labeled "Course ID:". The third field is labeled "Student ID:". The fourth field is labeled "Grade:". The fifth field is a button labeled "Add Grade".

Teacher: Ms. Sinha

Course ID:

Student ID:

Grade:

Add Grade

After logging in as a teacher they are given access to give the grade of a particular course and the grade is then updated in the students details window which the student can see after logging in.