

BONAFIDE CERTIFICATE

Certified that this project **“COURSE RESULT MANAGEMENT SYSTEM”** is the bonafide work of **“PALACHURU NANDU PRIYA, RENA J ”** who carried out the project work under my supervision.

SIGNATURE

Dr. E.K.SUBRAMANIAN

ASSOCIATE PROFESSOR

Dept. of Artificial Intelligence and Data
Science,
Rajalakshmi Engineering College
Chennai

This mini project report is submitted for the viva voce examination to be held on

INTERNAL EXAMINER

EXTERNAL EXAMINER

ABSTRACT

The **Course Result Management System** is a web based application designed to simplify and automate the process of managing student academic records. The system provides separate login portals for administrators and students, ensuring data security and role-based access. Administrators can perform CRUD (Create, Read, Update, Delete) operations such as adding students, managing courses, and entering or updating results. Students can log in securely to view their individual results, grades, and overall performance in a user-friendly interface.

Developed using **HTML, CSS, Python (Flask framework), and SQLite**, this system ensures efficient data handling, quick retrieval, and accurate result computation. The project aims to reduce manual workload, minimize human errors, and enhance transparency in academic result management. Overall, the Course Result Management System serves as a reliable and efficient platform for educational institutions to manage student performance digitally.

ACKNOWLEDGEMENT

We express our sincere thanks to our beloved and honorable chairman **MR. S. MEGANATHAN** and the Chairperson **DR. M. THANGAM MEGANATHAN** for their timely support and encouragement.

We are greatly indebted to our respected and honorable principal **Dr. S.N. MURUGESAN** for his able support and guidance.

No words of gratitude will suffice for the unquestioning support extended to us by our Head of the Department **Dr. J.M. GNANASEKAR** for being ever supporting force during our project work

We also extend our sincere and hearty thanks to our internal guide **Dr. E.K. SUBRAMANIAN**, for her valuable guidance and motivation during the completion of this project.

Our sincere thanks to our family members, friends and other staff members of computer science engineering.

1.PALACHURU NANDU PRIYA

2. RENA J

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO
	ABSTRACT	iv
1	INTRODUCTION	1
1.1	INTRODUCTION	8
1.2	SCOPE OF THE WORK	8
1.3	PROBLEM STATEMENT	8
1.4	AIM AND OBJECTIVES OF THE PROJECT	8
2	SYSTEM SPECIFICATIONS	9
2.1	HARDWARE SPECIFICATIONS	9
2.2	SOFTWARE SPECIFICATIONS	9
3	MODULE DESCRIPTION	10
4	CODING	11
5	SCREENSHOTS	16
6	CONCLUSION AND FUTURE ENHANCEMENT	18
	REFERENCES	19

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
5.1	INTRODUCTION PAGE	15
5.2	STUDENT DETAILS	15
5.3	ADMIN/STUDENT LOGIN	16
5.4	ADD/VIEW STUDENTS	16
5.5	GRADE CARD	18

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

The Course Result Management System is developed to automate and streamline the process of managing academic results in educational institutions. Traditionally, maintaining student records, calculating grades, and generating report cards were done manually, which often led to errors, delays, and difficulty in data retrieval. To overcome these limitations, this system provides a digital platform that simplifies result management and enhances efficiency.

1.2 SCOPE OF THE WORK

The Course Result Management System focuses on automating student result processing and record management. It allows administrators to efficiently add, update, and view student and course details. Students can securely access their academic results online. The system ensures accuracy, saves time, and enhances transparency in academic evaluation.

1.3 PROBLEM STATEMENT

In many educational institutions, managing student results manually is time-consuming, error-prone, and inefficient. The lack of a digital system leads to data inconsistency and difficulty in retrieving records. Students often face delays in accessing their results. Hence, there is a need for an automated system to manage and display academic results accurately and efficiently.

1.3 AIM AND OBJECTIVES OF THE PROJECT

The aim of the project is to automate the management of student results and academic records. The system allows administrators to **securely**

add, update, and manage student and course data. Students can view their results and grades online instantly. It ensures accuracy, saves time, and reduces manual errors in result processing.

CHAPTER 2

SYSTEM SPECIFICATIONS

2.1 HARDWARE SPECIFICATIONS

Processor	:	Intel i7
Memory Size	:	8GB (Minimum)
HDD	:	1 TB (Minimum)

2.2 SOFTWARE SPECIFICATIONS

Operating System	:	WINDOWS 11
Front – End	:	Python(Flask)
Back - End	:	MySQL, Sqlite
Language	:	python, SQL

CHAPTER 3

MODULE DESCRIPTION

This application consists of two main modules. When the program runs, the user is prompted to log in either as an Administrator or as a Student. The modules are described as follows:

1. Admin Login

The administrator logs in using a secure username and password. The admin has the authority to add, update, view, and delete student information, course details, and student results. This module allows the admin to manage all academic data in the system efficiently.

2. Student Login

The student logs in with their unique credentials to securely access the system. After login, the student can view their results, grades, and overall performance in a clear and organized manner. This module ensures students have access to their academic records without affecting the database.

CHAPTER 4

SAMPLE CODING

```
from flask import Flask, render_template, request, redirect, session

import sqlite3

app = Flask(__name__)

app.secret_key = 'secretkey'

# --- Database Setup ---

def init_db():

    conn = sqlite3.connect('crms.db')

    cur = conn.cursor()

    # Students table

    cur.execute("""CREATE TABLE IF NOT EXISTS Student(

        id INTEGER PRIMARY KEY AUTOINCREMENT,

        roll_no TEXT,

        name TEXT,

        department TEXT,

        year INTEGER)""")
```

Courses table

```
cur.execute("""CREATE TABLE IF NOT EXISTS Course(  
  
            id INTEGER PRIMARY KEY AUTOINCREMENT,  
  
            code TEXT,  
  
            name TEXT,  
  
            credits INTEGER)""")
```

Results table

```
cur.execute("""CREATE TABLE IF NOT EXISTS Result(  
  
            id INTEGER PRIMARY KEY AUTOINCREMENT,  
  
            student_id INTEGER,  
  
            course_id INTEGER,  
  
            marks INTEGER,  
  
            FOREIGN KEY(student_id) REFERENCES Student(id),  
  
            FOREIGN KEY(course_id) REFERENCES Course(id))""")
```

Users table

```
cur.execute("""CREATE TABLE IF NOT EXISTS User(  
  
            id INTEGER PRIMARY KEY AUTOINCREMENT,  
  
            username TEXT UNIQUE,
```

```

        password TEXT,

        role TEXT,

        student_id INTEGER)'''

# Insert default admin if not exists

cur.execute("SELECT * FROM User WHERE username='admin'")

if not cur.fetchone():

        cur.execute("INSERT INTO User(username,password,role)
VALUES('admin','admin123','admin')")

conn.commit()

conn.close()

init_db()

# --- Routes ---

@app.route('/')

def home():

    return redirect('/login')

@app.route('/login', methods=['GET','POST'])

def login():

    if request.method == 'POST':

```

```
username = request.form['username']

password = request.form['password']

conn = sqlite3.connect('crms.db')

cur = conn.cursor()

    cur.execute("SELECT * FROM User WHERE username=? AND password=?",
(username,password))

user = cur.fetchone()

conn.close()

if user:

    session['username'] = username

    session['role'] = user[3]

    session['student_id'] = user[4] # None for admin

    if user[3] == 'admin':

        return redirect('/dashboard')

    else:

        return redirect('/student_dashboard')

else:

    return render_template('login.html', error="Invalid credentials")
```

```
return render_template('login.html')
```

Sample 1

```
def view_results():

    if session.get('role') != 'admin':

        return redirect('/login')

    conn = sqlite3.connect('crms.db')

    cur = conn.cursor()

    cur.execute("""SELECT r.id, s.name, c.name, r.marks

        FROM Result r

        JOIN Student s ON r.student_id=s.id

        JOIN Course c ON r.course_id=c.id""")

    data = cur.fetchall()

    conn.close()

    return render_template('view_results.html', results=data)

@app.route('/edit_result/<int:id>', methods=['GET','POST'])

def edit_result(id):
```

```
if session.get('role') != 'admin':

    return redirect('/login')

conn = sqlite3.connect('crms.db')

cur = conn.cursor()

cur.execute("SELECT id,name FROM Student")

students = cur.fetchall()

cur.execute("SELECT id,name FROM Course")

courses = cur.fetchall()

if request.method=='POST':

    student_id = request.form['student_id']

    course_id = request.form['course_id']

    marks = request.form['marks']

    cur.execute("UPDATE Result SET student_id=?, course_id=?, marks=? WHERE
id=?",

                (student_id, course_id, marks, id))

    conn.commit()

    conn.close()

    return redirect('/view_results')
```

```
cur.execute("SELECT * FROM Result WHERE id=?", (id,))

result = cur.fetchone()

conn.close()

return render_template('edit_result.html', result=result, students=students,
courses=courses)
```

CHAPTER 5

SCREEN SHOTS

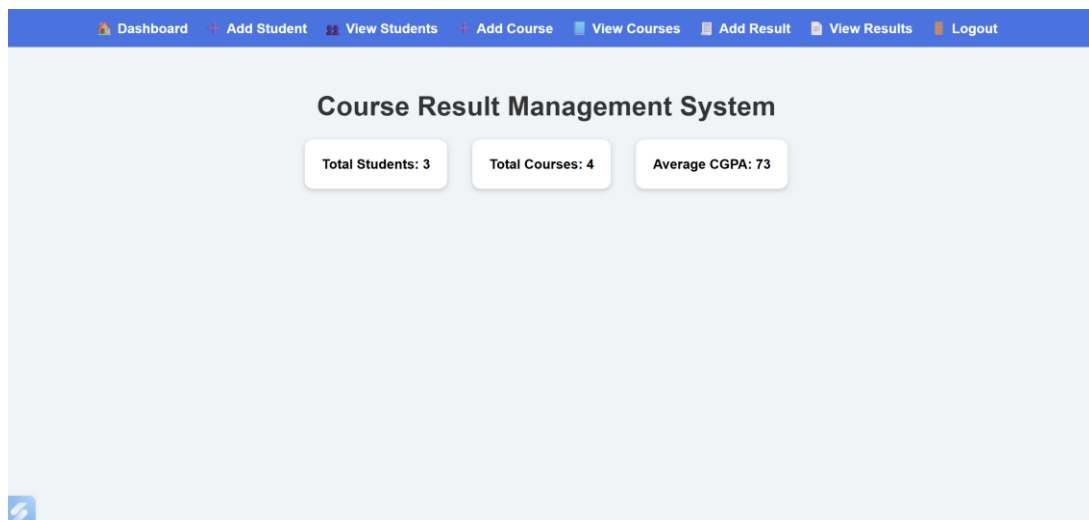


Fig 5.1 Dashboard

Add Student

Roll Number

Name (will be username)

Department

Year

Add Student

Fig 5.2 Student details

Student List

ID	Roll No	Name	Department	Year	Actions
1	1	Rena	AI&DS	2	Edit Delete
2	5	Ram	CSE	2	Edit Delete
3	29	John	IT	2	Edit Delete

Fig 5.3 View Students

Results				
ID	Student	Course	Marks	Actions
1	Rena	DBMS	93	Edit Delete
2	Rena	JAVA	87	Edit Delete
3	Rena	DAA	91	Edit Delete
4	Ram	DBMS	79	Edit Delete
5	Ram	DAA	65	Edit Delete
6	Ram	JAVA	74	Edit Delete
7	John	DBMS	96	Edit Delete

Fig 5.4 View Result

Grade Card		
Name: Rena Roll No: 1 Department: AI&DS Year: 2		
Course	Marks	Grade
DBMS	93	A+
JAVA	87	A
DAA	91	A+
CGPA: 9.67		

Fig 5.5 Grade Card

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

The Course Result Management System automates the management of student results, reducing manual work and errors. Administrators can efficiently add, update, and manage student and course data, while students can securely view their results online. The system ensures accuracy, transparency, and quick access to academic records. In the future, features like automated grade analysis, notifications via email or SMS, faculty login, and mobile access can be added. Integration with cloud databases can improve scalability and data backup. These enhancements will make the system more user-friendly and efficient. Overall, it provides a reliable digital solution for academic result management.

REFERENCES

1. <https://www.w3schools.com/sql/>
2. <https://www.tutorialspoint.com/sqlite/index.htm>
3. <https://www.wikipedia.org/>
4. <https://www.learnpython.org/>
5. <https://www.codecademy.com/learn/learn-python>