

Thakur College of Science & Commerce

Thakur Village, Kandivali East, Mumbai, Maharashtra 400101

A

PROJECT REPORT

ON

“Student Result Management System”

SUBMITTED BY

Enoch Sudhakar Sawant

[T. Y. B.SC. CS 386]

UNDER THE GUIDANCE OF

PROF. SIDDHI BIRWADKAR

Submitted in the partial fulfilment of the requirement for
qualifying B.Sc. (C.S), Semester-V Examination.

Mumbai University [2022-23]



Thakur Educational Trust's (Regd.)

THAKUR COLLEGE OF SCIENCE & COMMERCE

AUTONOMOUS COLLEGE, PERMANENTLY AFFILIATED TO UNIVERSITY OF MUMBAI

NAAC Accredited Grade 'A' (3rd Cycle) & ISO 9001: 2015 (Certified)

Best College Award by University of Mumbai for the Year 2018-2019



CELEBRATING
25 YEARS OF GLORY

DECLARATION BY LEARNER

I the undersigned, **Mr. Enoch Sawant** hereby, declare that the work embodied in this project work titled **“Student Result Management System”** forms my own contribution to the research work carried out under the guidance of Prof. **Siddhi Birwadkar** is a result of my own research work and has not been previously submitted to any other University for any other Degree/Diploma to this or any other University.

Wherever reference has been made to previous works of others, it has been clearly indicated as such and included in the bibliography.

I, hereby further declare that all information of this document has been obtained and presented in accordance with academic rules and ethical conduct.

Certified By
Siddhi Birwadkar

Learner
Enoch Sawant



Thakur Educational Trust's (Regd.)

THAKUR COLLEGE OF SCIENCE & COMMERCE

AUTONOMOUS COLLEGE, PERMANENTLY AFFILIATED TO UNIVERSITY OF MUMBAI

NAAC Accredited Grade 'A' (3rd Cycle) & ISO 9001: 2015 (Certified)

Best College Award by University of Mumbai for the Year 2018-2019

tcsc

CELEBRATING
25 YEARS OF GLORY

CERTIFICATE

This is to certify that **Mr. Enoch Sawant** has worked and duly completed his Project Work for the Degree of Bachelor under the Faculty of Science in Computer Science (T.Y.B.Sc CS) and the project is entitled, **Student Result Management System** towards the partial fulfilment of project work as prescribed by University of Mumbai for the year 2022-23.

I further certify that the entire work has been done by the learner and that no part of it has been submitted previously for any Degree or Diploma of any University.

Prof.Ashish Trivedi
Head of Department

Prof.Siddhi Birwadkar
Project Guide

Internal Examiner

External Examiner

ACKNOWLEDGEMENT

I would like to extend our heartiest thanks with a deep sense of gratitude and respect to all those who provides me immense help and guidance during my period.

I would like to thank my Project Guide **Prof.Siddhi Birwadkar** for providing a vision about the system. I have been greatly benefited from their regular critical reviews and inspiration throughout my work. I am grateful to them for their guidance, encouragement, understanding and insightful support in the development process.

I would also like to thank my college for giving required resources whenever I wanted and for giving the opportunity to develop the project.

I would like to express my sincere thanks to our Head of Department **Prof. Ashish Trivedi** for having facilitated us with the essential infrastructure & resources without which this project would not have seen light of the day.

I am also thankful to entire staff of **CS** for their constant encouragement, suggestions and moral support throughout the duration of my project.

Last but not the least I would like to mention here that I am greatly indebted to each and everybody my friends and who has been associated with my project at any stage but whose name does not find a place in this acknowledgement.

With sincere regards,
Enoch Sudhakar Sawant

CONTENTS

| | |
|--|-----------|
| CHAPTER 1: INTRODUCTION OF PROJECT..... | 2 |
| 1.1. Objective..... | 3 |
| 1.2. Introduction..... | 4 |
| 1.3. Scope..... | 5 |
| CHAPTER 2: HARDWARE & SOFTWARE REQUIREMENT..... | 6 |
| 1.1. Hardware Requirement..... | 7 |
| 1.2. Software Requirement..... | 7 |
| CHAPTER 3: TECHNICAL DESCRIPTION..... | 8 |
| 1.1. Front End Description..... | 9 |
| CHAPTER 4: SOFTWARE ANALYSIS & DESIGN..... | 10 |
| 4.1. Software Analysis..... | 11 |
| 4.1.1. Software Development Life cycle..... | 11 |
| 4.1.2. Description of Used Model..... | 12 |
| 4.2. Software Design..... | 14 |
| 4.2.1. Data Flow Diagram..... | 14 |
| 4.2.2. ER DIAGRAM..... | 15 |
| 4.2.3. USE CASE DIAGRAM..... | 16 |
| CHAPTER 5: DEVELOPMENT OF PROJECT..... | 18 |
| 5.1. Source Code..... | 19 |
| 5.2. Snapshots of Output..... | 24 |
| CHAPTER 6: TESTING..... | 27 |
| CHAPTER 7: BENEFITS..... | 30 |
| CHAPTER 8: LIMITATIONS..... | 32 |
| CHAPTER 9: FUTURE ENHANCEMENT..... | 34 |
| CHAPTER 10: CONCLUSION..... | 36 |
| CHAPTER 11: REFERENCES..... | 38 |

ABSTRACT

Our project Student Result Management system includes registration of students, storing their details into the system i.e. computerized the process. Our software has the facility to store the details of every student. It includes a search facility also – search by roll number. The data can be retrieved easily. The interface is very User-friendly. The data are well protected for personal use and makes the data processing very fast.

CHAPTER 1

INTRODUCTION

1.1 OBJECTIVE

Student Result Management System is software which is helpful for college as well as the school authorities. In the current system all the activities are done manually. It is very time consuming and costly. Our Student Result Management System deals with the various activities related to managing student records. Our Objective is computerizing the process of student records management.

1.2 INTRODUCTION

This project is aimed to automate the Student Result Management System. This project is developed mainly to administrate student records.

The purpose of the project entitled as STUDENT RESULT MANAGEMENT SYSTEM is to computerize the Front Office Management of student records in colleges, schools and coaching's, to develop software which is user friendly, simple, fast, and cost – effective.

Traditionally, it was done manually.

The main function of the system is to register and store student details, result details, retrieve these details as and when required, and also to manipulate these details meaningfully.

1.3 SCOPE

The main objective of the Project on Result Management Software is to manage the details of Result, Student, Progress, etc. It manages all the information about Result, Course, etc.

The project is totally built at administrative end and only the administrator is guaranteed the access to perform various task like Add, Update, Remove, Manage, Delete Data.

The purpose of the project is to build the program to reduce the manual work for managing the result work. It tracks all the details about the Result, Student, etc. It may help collecting perfect management in details.

It will also be easy to use for students to check their result, marks and performance. It also helps in current all works relative to Result Management. It will be also reduced the time of collecting the management & collection procedure will go on smoothly

CHAPTER 2

HARDWARE & SOFTWARE REQUIREMENT

SYSTEM REQUIREMENT

2.1 HARDWARE REQUIREMENT

- Pentium IV processor or higher
- 4 GB RAM (or above)
- 40 GB or more HARDDISK
- Mouse/Keyboard

2.2 SOFTWARE REQUIREMENT

- OS-Windows 7/8/10
- Python Interpreter
- Pycharm IDE or VS code
- Sqlite3

CHAPTER 3

TECHNICAL DESCRIPTION

3.1 FRONT END DESCRIPTION

LANGUAGE: - Python

Python is a widely used general-purpose, high level programming language. It was initially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code.

Python is a programming language that lets you work quickly and integrate systems more efficiently.

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

Tkinter

Tkinter is a Python binding to the Tk GUI toolkit. It is the standard Python interface to the Tk GUI toolkit, and is Python's *de facto* standard GUI. Tkinter is included with standard Linux , Microsoft Windows and Mac OS X installs of Python.

The name *Tkinter* comes from *Tk interface*. Tkinter was written by Fredrik Lundh.

CHAPTER 4

SOFTWARE ANALYSIS & DESIGN

4.1 Software Analysis

4.1.1 Software Development Life cycle

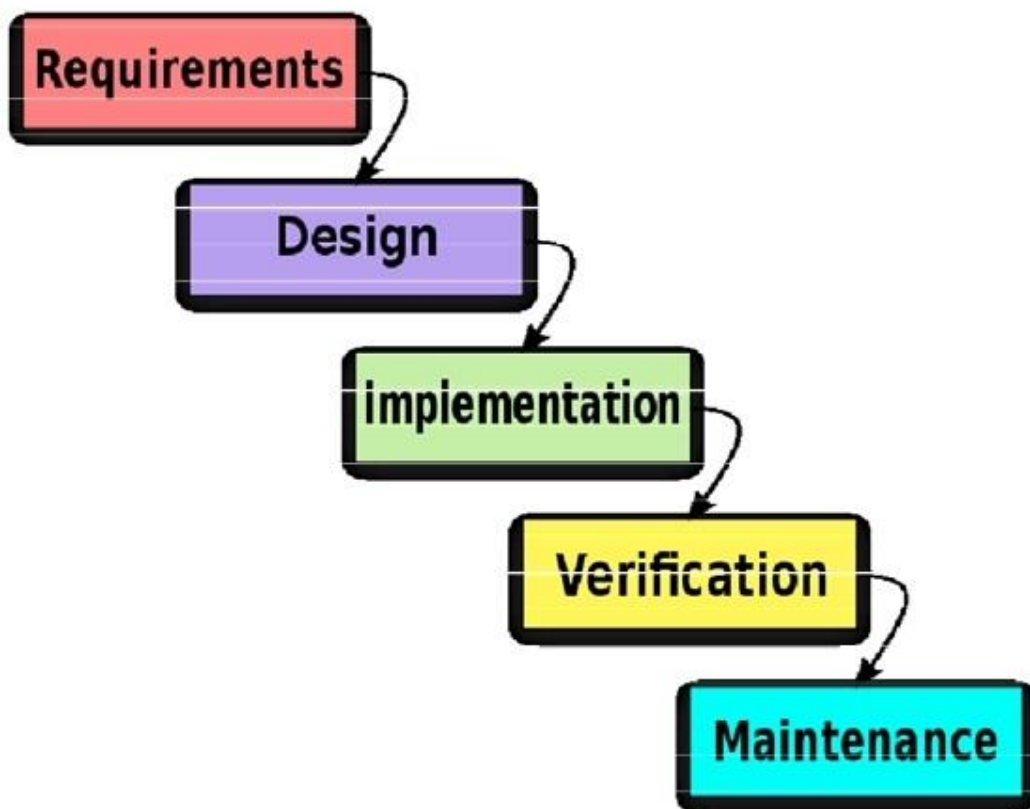


Fig: - SDLC

4.1.2 DESCRIPTION OF USED MODEL

The Waterfall Model

The Waterfall model is a sequential software development process, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design (validation), Construction, Testing and Maintenance.

To follow the waterfall model, one proceeds from one phase to the next in a sequential manner. For example, one first completes requirements specification, which after sign-off are considered "set in stone." When the requirements are fully completed, one proceeds to design. The software in question is designed and a blueprint is drawn for implementers (coders) to follow — this design should be a plan for implementing the requirements given. When the design is fully completed, an implementation of that design is made by coders. Towards the later stages of this implementation phase, separate software components produced are combined to introduce new functionality and reduced risk through the removal of errors.

Thus the waterfall model maintains that one should move to a phase only when its preceding phase is completed and perfected. However, there are various modified waterfall models (including Royce's final model) that may include slight or major variations upon this process. Time spent early in the software production cycle can lead to greater economy at later stages. It has been shown that a bug found in the early stages (such as requirements specification or design) is cheaper in terms of money, effort and time, to fix than the same bug found later on in the process. To take an extreme example, if a program design turns out to be impossible to implement, it is easier to fix the design at the design stage than to realize months later, when program components are being integrated, that all the work done so far has to be scrapped because of a broken design.

This is the central idea behind the waterfall model - time spent early on making sure that requirements and design are absolutely correct will save you much time and effort later. Thus, the thinking of those who follow the waterfall process goes, one should make sure that each phase is 100% complete and absolutely correct before proceeding to the next phase of program creation. Program requirements should be set in stone before design is started (otherwise work

put into a design based on incorrect requirements is wasted); the program's design should be perfect before people begin work on implementing the design (otherwise they are implementing the wrong design and their work is wasted), etc.

A further argument for the waterfall model is that it places emphasis on documentation (such as requirements documents and design documents) as well as source code. In less designed and documented methodologies, should team members leave, much knowledge is lost and may be difficult for a project to recover from. Should a fully working design document be present (as is the intent of Big Design Up Front and the waterfall model) new team members or even entirely new teams should be able to familiarize themselves by reading the documents.

Basic principles of the waterfall model are:

Project is divided into sequential phases, with some overlap and splash back acceptable between phases.

Emphasis is on planning, time schedules, target dates, budgets and implementation of an entire system at one time.

Tight control is maintained over the life of the project through the use of extensive written documentation, as well as through formal reviews and approval/signoff by the user and information technology management occurring at the end of most phases before beginning the next phase.

4.2 SOFTWARE DESIGN

4.2.1 Data Flow Diagram

The DFD takes an input-process-output view of a system. That is, data objects flow into the software, are transformed by the processing elements, and resultant data objects flow out of the software. Data objects are represented by labeled arrows and transformations are represented by circles. The DFD is represented in a hierarchical fashion. The first DFD represents the system as a whole. Subsequent data flow diagrams provide increasing detail with each subsequent level.

The data flow diagram enables the software engineer to develop models of the information domain and functional domain at the same time. As the DFD is refined into levels of greater detail, the analysts perform an implicit functional decomposition of the system. Also DFD refinement results in a corresponding refinement of data as it moves through the processes that embody the application.

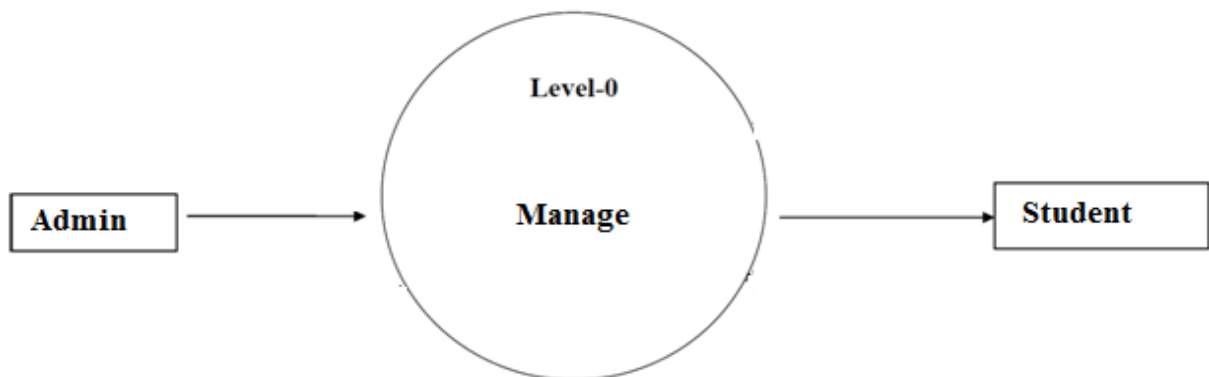


Fig: - Zero (0) Level

ER DIAGRAM

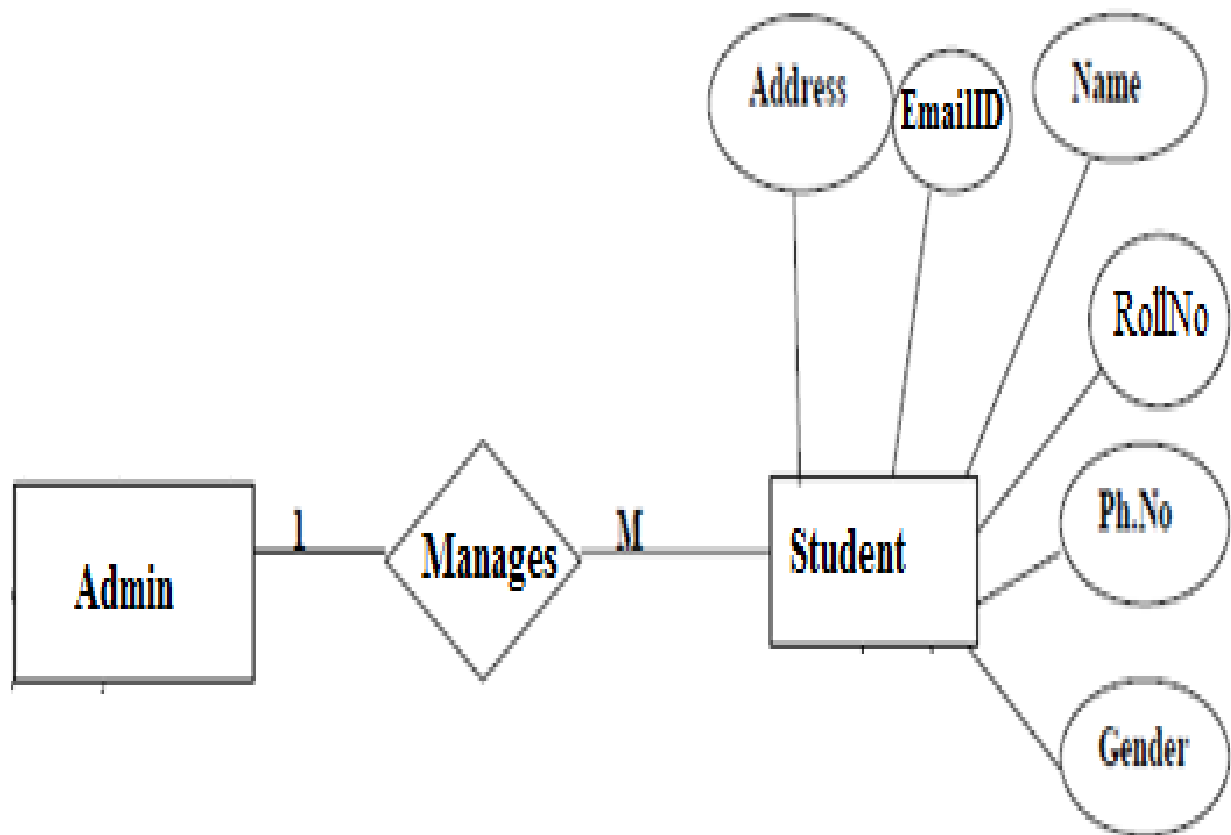
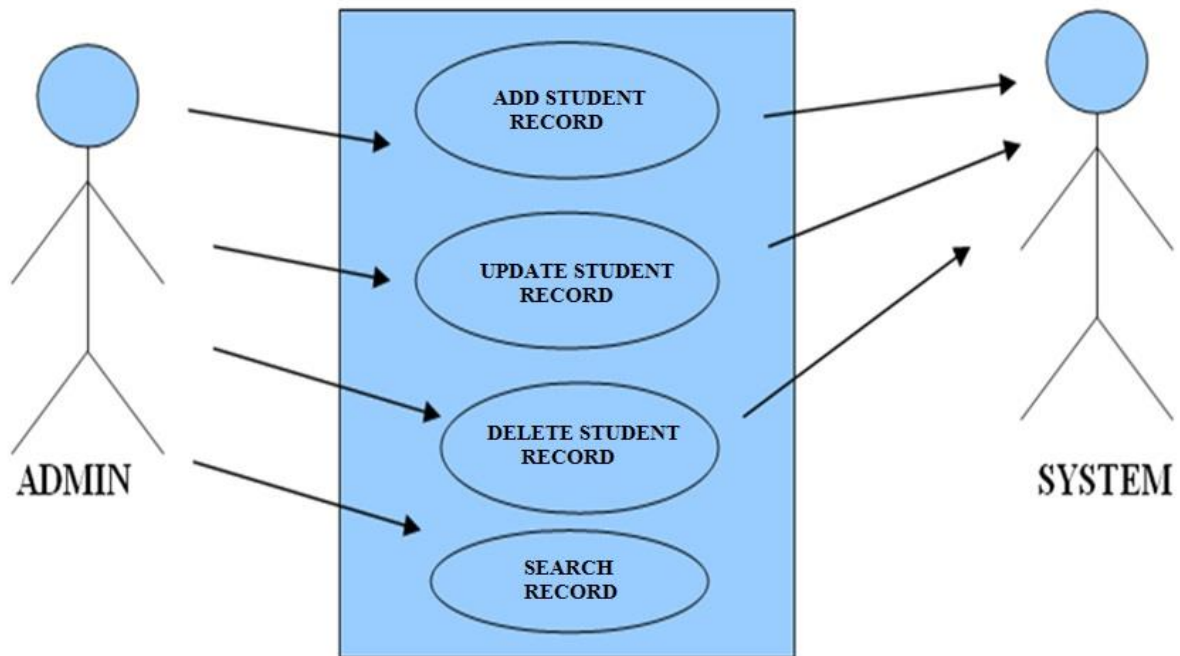
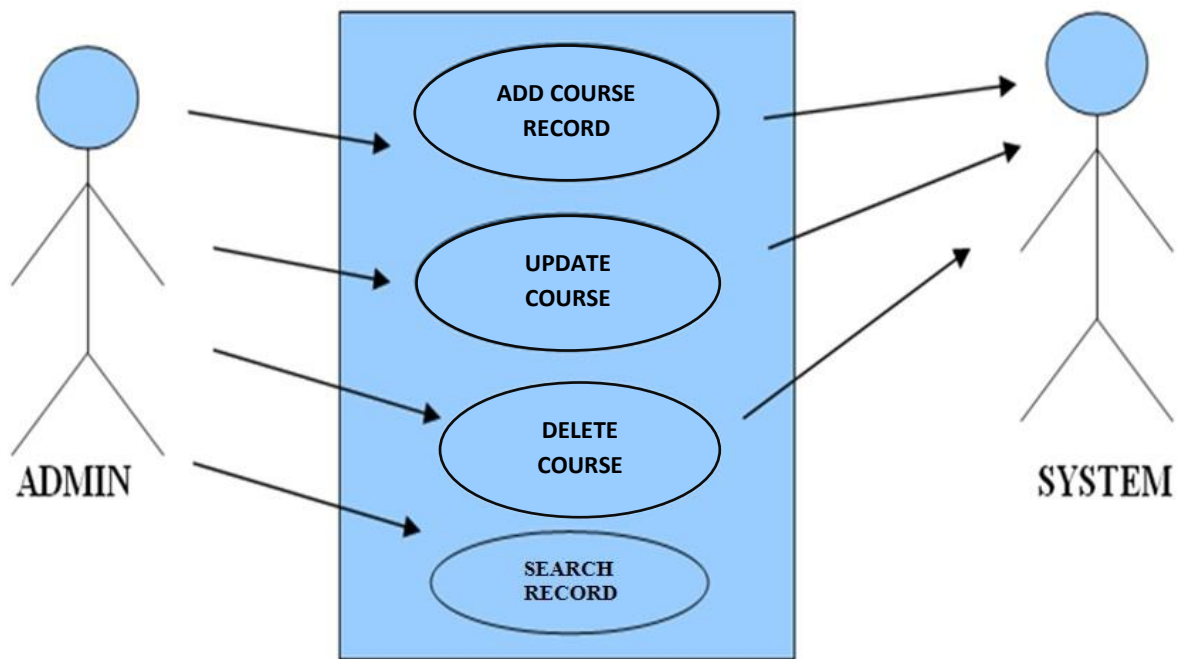


Fig: ER DIAGRAM

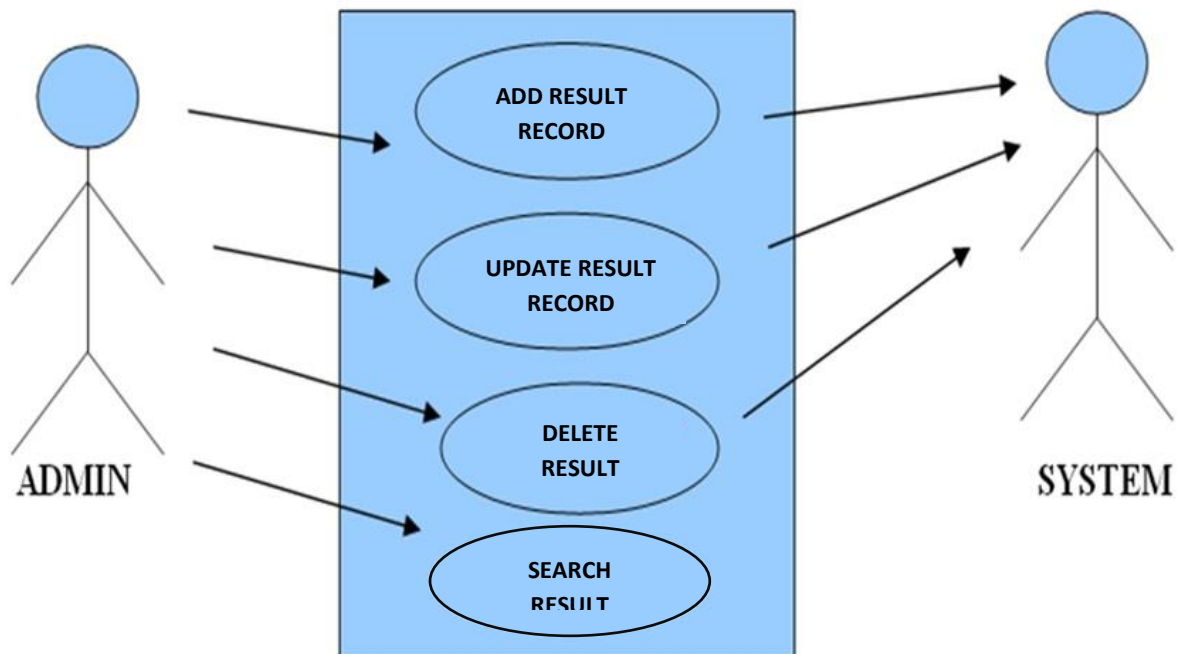
USE CASE DIAGRAM



Use Case Diagram between ADMIN and SYSTEM:

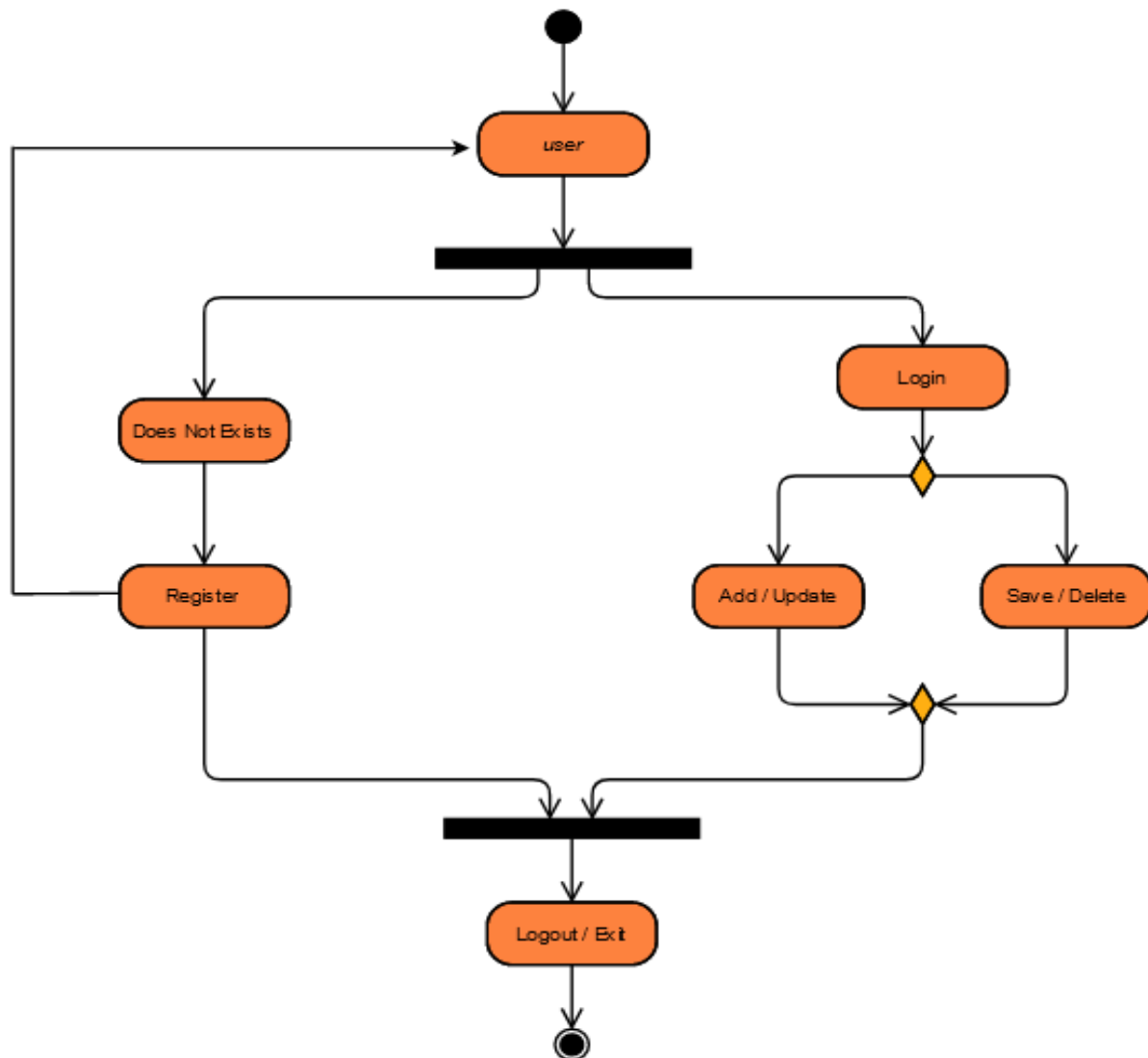


Use Case Diagram between ADMIN and SYSTEM:

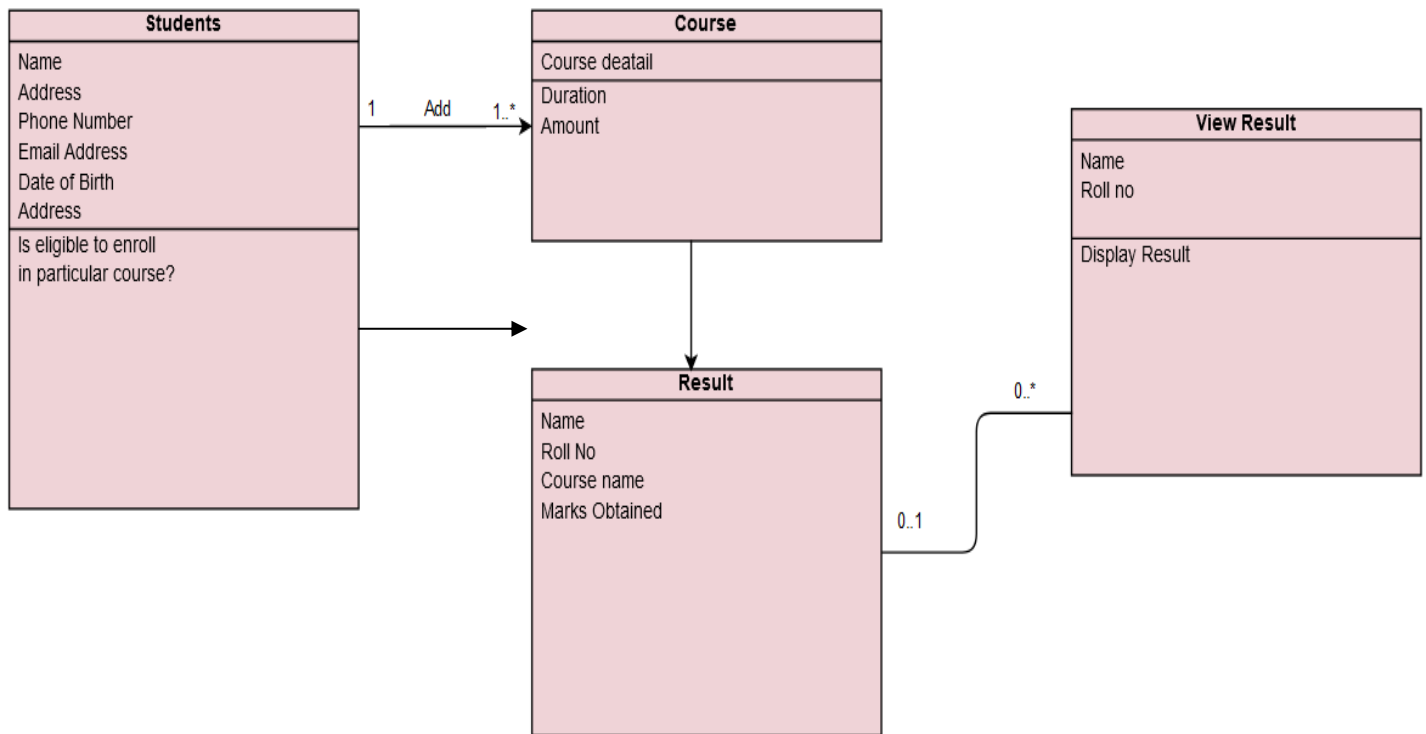


Use Case Diagram between ADMIN and SYSTEM:

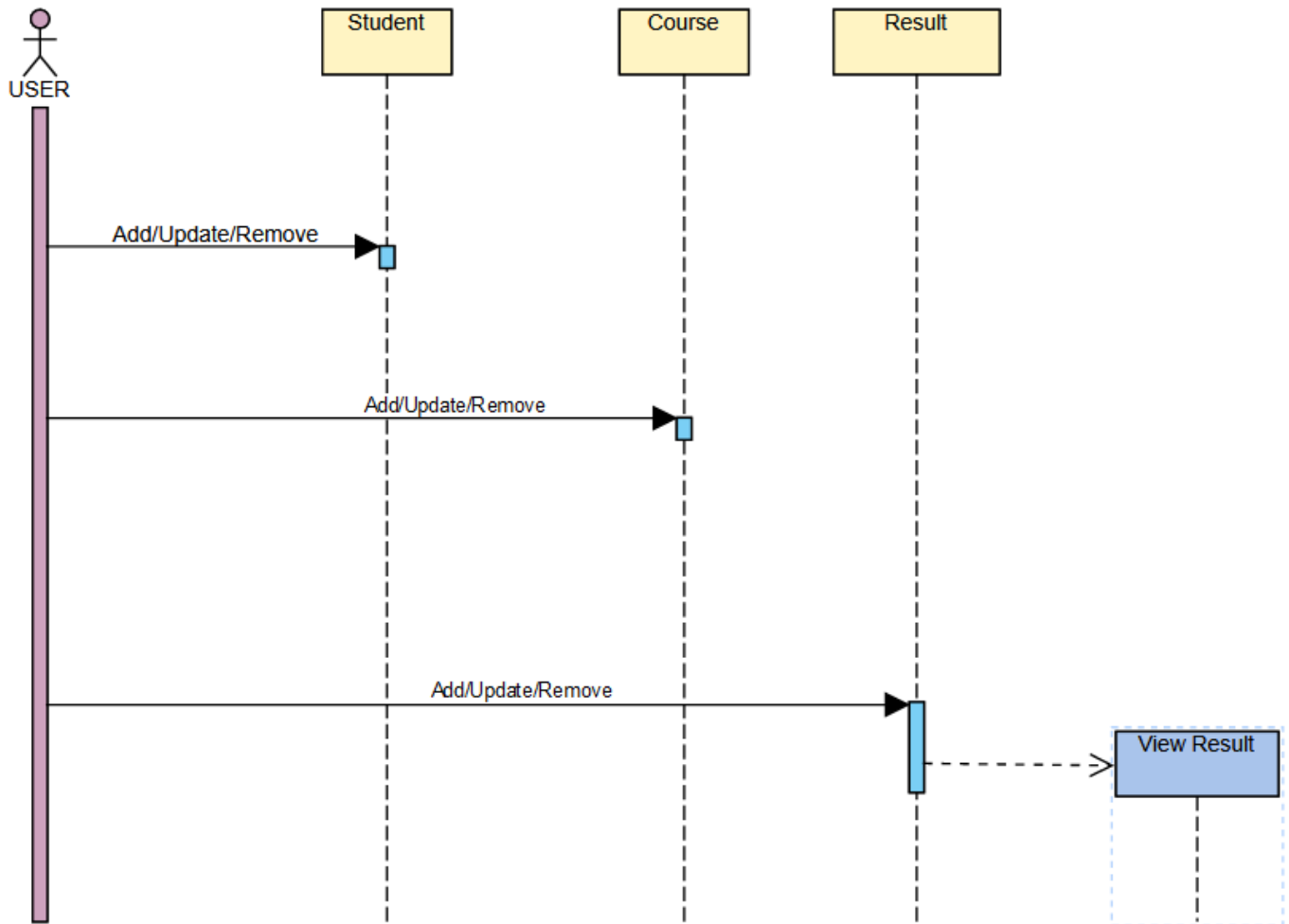
ACTIVITY DIAGRAM



CLASS DIAGRAM



SEQUENCE DIAGRAM



CHAPTER 5

DEVELOPMENT

5.1 SOURCE CODE

Dashboard.py: -

```
from tkinter import*
from PIL import Image,ImageTk
from course import CourseClass
from student import studentClass
from result import resultClass
from report import reportClass
from tkinter import messagebox
import os
import sqlite3 as db

class RMS:
    def __init__(self,root):
        self.root=root
        self.root.title("Student Result Management Software")
        width= self.root.winfo_screenwidth()
        height= self.root.winfo_screenheight()
        self.root.geometry("%dx%d" % (width, height))
        self.root.config(bg="white")

        #--icon--
        self.logo_dash=ImageTk.PhotoImage(file="images/logo_p.png")

        #--title--
        title=Label(self.root,text="Student Result Management System",
        compound= LEFT,padx=20 ,image=self.logo_dash,font=("goudy old
        style",30,
        "bold"),bg="orange",fg="blue").place(x=0,y=0,relwidth=1,height=50)

        #--menu--
        M_Frame=LabelFrame(self.root,text="Menu",font=("times new roman",15),
        bg="white", fg="Black")
        M_Frame.place(x=10,y=60,width=1515,height=80)

        btn_course=Button(M_Frame,text="Course",font=("goudy old
        style",15,"bold"),
        bg="darkblue",fg="white",cursor="hand2",command=self.add_course).place
        (x=20,y=5,width=200,height=40)
```

```

btn_Student=Button(M_Frame,text="Student",font=("goudy old
style",15,"bold"
),bg="darkblue",fg="white",cursor="hand2",command=self.add_student).pl
ace
(x=270,y=5,width=200,height=40)

btn_Result=Button(M_Frame,text="Result",font=("goudy old
style",15,"bold"),
bg="darkblue",fg="white",cursor="hand2",command=self.add_result).place
(x=520,y=5,width=200,height=40)

btn_View=Button(M_Frame,text="View Student Results",font=("goudy old
style",
15,"bold"),bg="darkblue",fg="white",cursor="hand2",command=self.add_re
port)
.place(x=770,y=5,width=200,height=40)

btn_Logout=Button(M_Frame,text="Logout",font=("goudy old
style",15,"bold"),
bg="darkblue",fg="white",cursor="hand2",command=self.logout)
.place(x=1020,y=5,width=200,height=40)

btn_Exit=Button(M_Frame,text="Exit",font=("goudy old
style",15,"bold"),
bg="darkblue",fg="white",cursor="hand2",command=self.exit_).place(x=12
70,y=5,width=200,height=40)

#--content window---
self.bg_image=Image.open("images/bg.png")
self.bg_image=self.bg_image.resize((1000,400),Image.ANTIALIAS)
self.bg_image=ImageTk.PhotoImage(self.bg_image)

self.lbl_bg=Label(self.root,image=self.bg_image).place(x=450,y=180,
width=1000,height=400)

#--update details--
self.lbl_course=Label(self.root,text="Total Courses\n[0]",font=("goudy
old
style",20),bd=10,relief=RIDGE,bg="#e43b06",fg="white")
self.lbl_course.place(x=450,y=600,width=300,height=100)

self.lbl_Students=Label(self.root,text="Total
Students\n[0]",font=("goudy old
style",20),bd=10,relief=RIDGE,bg="#e43b06",fg="white")
self.lbl_Students.place(x=800,y=600,width=300,height=100)

```

```

self.lbl_result=Label(self.root,text="Total Results\n[0]",font=("goudy
old style",20),bd=10,relief=RIDGE,bg="#e43b06",fg="white")
self.lbl_result.place(x=1148,y=600,width=300,height=100)

#--footer--
footer=Label(self.root,text="SMRS- Student Result Management Software\n
Contact Us for any Tecnical Issue: 7666XXX53 ",font=("goudy old
style",20,) ,bg="black",fg="white").pack(side=BOTTOM,fill=X)

self.update_course_details()
self.update_student_details()
self.update_result_details()

def update_course_details(self):
    con=db.connect(database="rms.db")
    cur=con.cursor()

    try:
        cur.execute("select * from course")
        cr=cur.fetchall()
        self.lbl_course.config(text=f"Total Courses\n[{str(len(cr))}]")
        self.lbl_course.after(200,self.update_course_details)

    except Exception as ex:
        messagebox.showerror("Error",f"Error due to {str(ex)}")

def update_student_details(self):
    con=db.connect(database="rms.db")
    cur=con.cursor()

    try:
        cur.execute("select * from student")
        cr=cur.fetchall()
        self.lbl_Students.config(text=f"Total
Students\n[{str(len(cr))}]")
        self.lbl_Students.after(200,self.update_student_details)

    except Exception as ex:
        messagebox.showerror("Error",f"Error due to {str(ex)}")

def update_result_details(self):
    con=db.connect(database="rms.db")
    cur=con.cursor()

```

```

try:
    cur.execute("select * from result")
    cr=cur.fetchall()
    self.lbl_result.config(text=f"Total Results\n[{str(len(cr))}]")
    self.lbl_result.after(200,self.update_result_details)

except Exception as ex:
    messagebox.showerror("Error",f"Error due to {str(ex)}")

def add_course(self):
    self.new_win=Toplevel(self.root)
    self.new_obj=CourseClass(self.new_win)

def add_student(self):
    self.new_win=Toplevel(self.root)
    self.new_obj=studentClass(self.new_win)

def add_result(self):
    self.new_win=Toplevel(self.root)
    self.new_obj=resultClass(self.new_win)

def add_report(self):
    self.new_win=Toplevel(self.root)
    self.new_obj=reportClass(self.new_win)

def logout(self):
    op=messagebox.askyesno("Confirm","Do you really want to logout?",
    parent = self.root)
    if op==True:
        self.root.destroy()
        os.system("python login.py")

def exit_(self):
    op=messagebox.askyesno("Confirm","Do you really want to exit?",
    parent=self.root)

    if op==True:
        self.root.destroy()
        os.system("python login.py")

if __name__=="__main__":
    root=Tk()
    obj=RMS(root)
    root.mainloop()

```

Database: -

Create_db.py: -

```
import sqlite3 as db
def create_db():

    con=db.connect(database="rms.db")
    cur=con.cursor()
    cur.execute("CREATE TABLE IF NOT EXISTS course(cid INTEGER PRIMARY KEY
AUTOINCREMENT, name text, duration text, charges text, description text)")
    con.commit()

    cur.execute("CREATE TABLE IF NOT EXISTS student(roll INTEGER PRIMARY KEY
AUTOINCREMENT, name text, email text, gender text, dob text, contact text,
admission text, course text, state text, city text, pin text, address text)")
    con.commit()

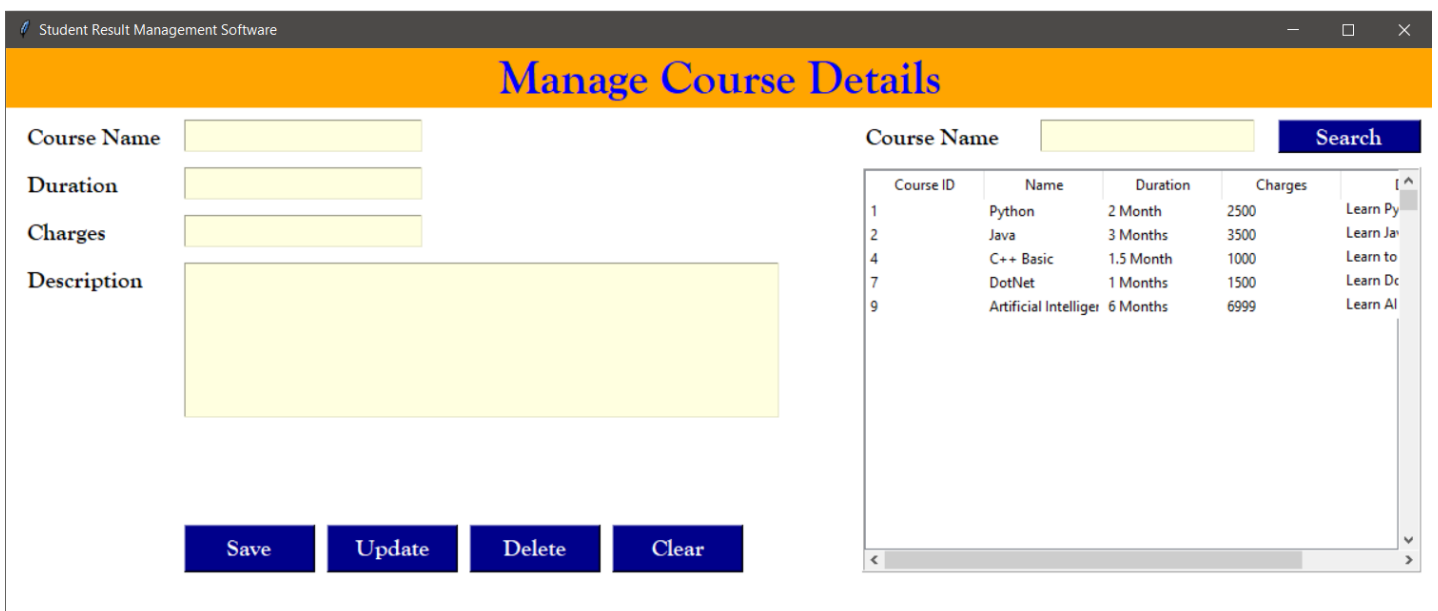
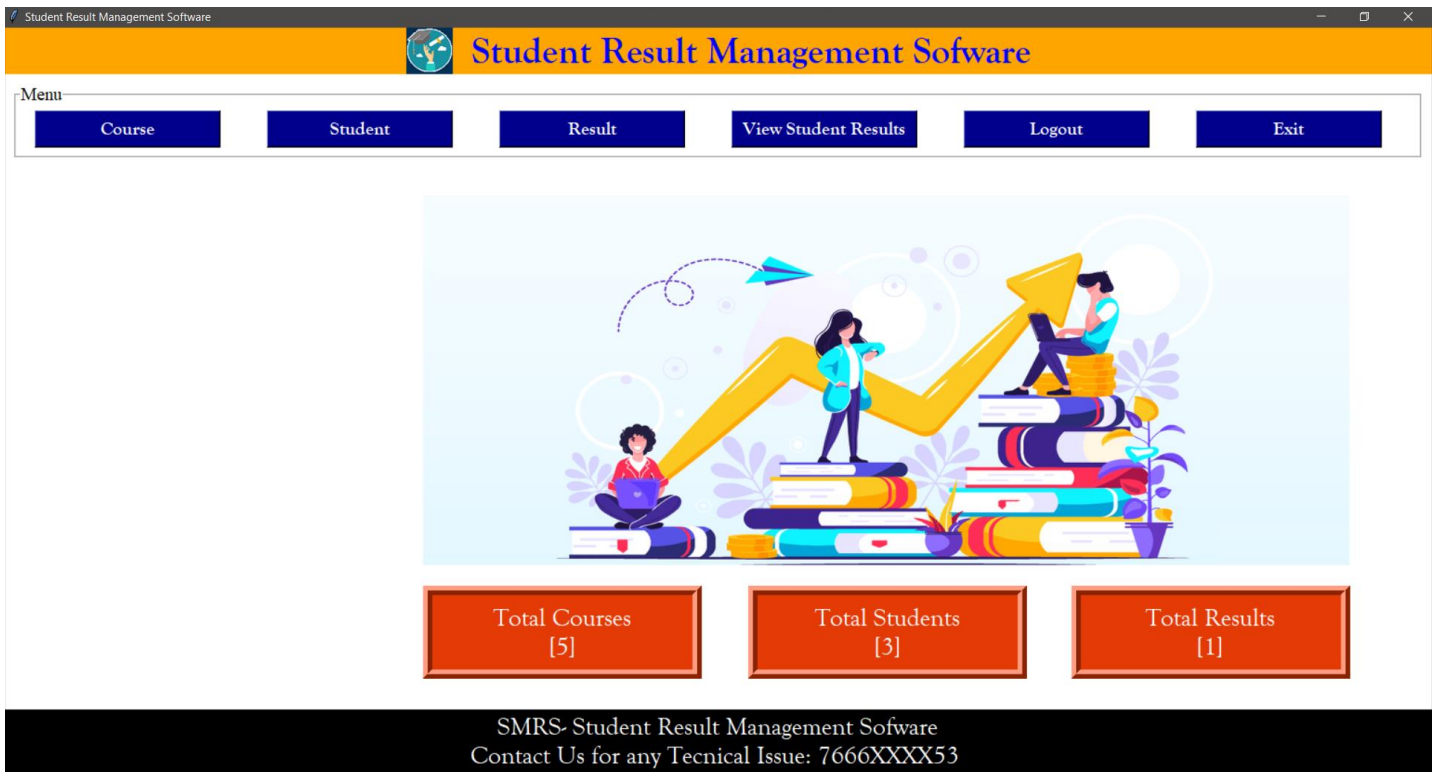
    cur.execute("CREATE TABLE IF NOT EXISTS result(rid INTEGER PRIMARY KEY
AUTOINCREMENT, roll text, name text, course text, marks_ob text, full_marks text,
per text)")
    con.commit()

    cur.execute("CREATE TABLE IF NOT EXISTS employee(eid INTEGER PRIMARY KEY
AUTOINCREMENT, f_name text, l_name text, contact text, email text, question text,
answer text, password text)")
    con.commit()

    con.close()

create_db()
```

5.2 SNAPSHOTS



Student Result Management Software

Manage Student Details

Roll No.
D.O.B

Name
Contact

Email
Admission

Gender
Course

State
City
Pin

Address

Save

Update

Delete

Clear

Roll No.

Search

| Roll No. | Name | Email | Gender | D.O.B |
|----------|--------------|------------------|--------|----------|
| 1018 | Virat Kohli | virat18@gmail.co | Male | 05-01-20 |
| 1111 | Salman | pmobvtetf@spcn | Select | XX-XX-X |
| 1234 | Enoch Sawant | enochsawant12@ | Male | 01-09-20 |

Student Result Management Software

Add Student Results

Select Student

Search

Name


Course

Marks Obtained

Full Marks

Submit

Clear



Student Result Management Software

View Student Results

Search by Roll No.

Search

Clear

| Roll No | Name | Course | Marks Obtained | Total Marks | Percentage |
|---------|-------------|--------|----------------|-------------|------------|
| 1018 | Virat Kohli | Java | 80 | 100 | 80.0 |

Delete

Registration Window

Thakur College of Science and Commerce

REGISTERE HERE

| | |
|-------------------------------------|----------------------|
| First Name | Last Name |
| <input type="text"/> | <input type="text"/> |
| Contact No. | Email |
| <input type="text"/> | <input type="text"/> |
| Security Question | Answer |
| <input type="text" value="Select"/> | <input type="text"/> |
| Password | Confirm Password |
| <input type="text"/> | <input type="text"/> |


☐ I Agree to the Terms & Conditions

[REGISTER NOW](#) [SIGN IN](#)

Login Window

Thakur College of Science and Commerce

LOGIN HERE



| |
|--------------------------|
| EMAIL ADDRESS |
| <input type="text"/> |
| PASSWORD |
| <input type="password"/> |

[Register New Accout](#) [Forget Password?](#)

[LOGIN](#)

CHAPTER 6

TESTING

6.1 TESTING

Testing is more than just debugging. The purpose of testing can be quality assurance, verification and validation, or reliability estimation. Correctness testing and reliability testing are two major areas of testing. Software testing is a trade-off between budget, time and quality.

Software Testing

Software Testing is the process of executing a program or system with the intent of finding errors. Or, it involves any activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required results. Software is not unlike other physical processes where inputs are received and outputs are produced. Where software differs is in the manner in which it fails. Unlike most physical systems, most of the defects in software are design errors, not manufacturing defects.

To improve quality

As computers and software are used in critical applications, the outcome of a bug can be severe. Bugs can cause huge losses.

For Verification & Validation (V&V)

Another important purpose of testing is verification and validation (V&V). It is heavily used as a tool in the V&V process. Testers can make claims based on interpretations of the testing results, which either the product works under certain situations, or it does not work.

Software Testing Types

Black-box testing

The black-box approach is a testing method in which test data are derived from the specified functional requirements without regard to the final program structure. It is also termed data-driven, input/output driven or requirements-based testing. A testing method emphasized on executing the functions and examination of their input and output data.

White-box testing

Contrary to black-box testing, software is viewed as a white-box, or glass-box in white-box testing, as the structure and flow of the software under test are visible to the tester. This testing is based on knowledge of the internal logic of an application's code. Testing plans are made according to the details of the software implementation, such as programming language, logic, and styles. Test cases are derived from the program structure. White-box testing is also called glass-box testing, logic-driven testing or design-based testing.

Unit testing

This involves testing of individual software components or modules. Typically done by the programmer and not by testers, as it requires detailed knowledge of the internal program design and code.

System testing

Entire system is tested as per the requirements. Black-box type testing that is based on overall requirements specifications, covers all combined parts of a system.

End-to-end testing

Similar to system testing, involves testing of a complete application environment in a situation that mimics real-world use, such as interacting with a database, using network communications, or interacting with other hardware, applications, or systems if appropriate.

Usability testing

User-friendliness check. Application flow is tested, Can new user understand the application easily, Proper help documented whenever user stuck at any point. Basically system navigation is checked in this testing.

Install/uninstall testing

Tested for full, partial, or upgrade install/uninstall processes on different operating systems under different hardware, software environment.

Recovery testing

Testing how well a system recovers from crashes, hardware failures, or other catastrophic problems.

Security testing

Can system be penetrated by any hacking way. Testing how well the system protects against unauthorized internal or external access. Checked if system, database is safe from external attacks.

Compatibility testing

Testing how well software performs in a particular hardware/software/operating system/network environment and different combinations of above.

Comparison testing

Comparison of product strengths and weaknesses with previous versions or other similar products.

Alpha testing

In house virtual user environment can be created for this type of testing. Testing is done at the end of development. Still minor design changes may be made as a result of such testing.

Beta testing

Testing typically done by end-users or others. Final testing before releasing application for commercial purpose.

CHAPTER 7

BENEFITS

7.1 BENEFITS

- Software provides easy management of student records.
- Software has very user-friendly interface which is very easy to handle and understand.
- Software provides security to private data by hiding them.
- Software uses very less memory and takes less time to startup.

CHAPTER 8

LIMITATIONS

8.1 LIMITATIONS

- Software is limited to Desktop only.
- System requires python interpreter installed on the system.
- All options of student management are not included in current version.
- Security options provide only low-level security against beginner attackers.
- GUI is in English only.

CHAPTER 9

FUTURE ENHANCEMENT

9.1 FUTURE ENHANCEMENT

- This Software can be made for all OS
- Higher Security features can be included in this software.
- Program scheduling can also be included in this software.
- This software can be developed to use as tutorial to teach basic concepts of OS to new users.
- This system can be implemented with OS to reduce overhead of installing and running interface of each and every tool at different place.
- Automatic Shutdown through SMS service can be implemented in this.

CHAPTER 10

CONCLUSION

10.1 CONCLUSION

The project entitled “Student Result Management System” is developed using Python Tkinter as front end and Sqlite3 database in back end to computerize the process of management of student records. This project covers only the basic features required.

CHAPTER 11

REFERENCES

11.1 REFERENCES

- Wikipedia
- <https://www.geeksforgeeks.org/python-gui-tkinter/>
- <https://www.javatpoint.com/python-tkinter>
- <https://www.python.org/>
- <https://www.youtube.com>