**Student Management System Report**

**Branch: MCA(AIML) Section/Group:1(A)**

**Performed by : Anuj ,Mannat Mahajan Semester:1st**

**UID:24MCI10020,24MCI10032**

**Subject Name: PL\SQL LAB(24CAP-602)**

**Introduction**

This report describes the development of a Student Management System (SMS) using Python's Tkinter library for the graphical user interface and MySQL for database management. The system aims to facilitate user registration, student record management, and course selection, providing a simple interface for both administrators and students.

**Table of Contents**

1. **Overview of the System**
2. **Technologies Used**
3. **Features of the System**
   * **User Authentication**
   * **User Registration**
   * **Student Record Management**
4. **Code Implementation**
5. **Conclusion and Future Work**

**1. Overview of the System**

The Student Management System allows users to:

* Register and log in.
* Manage student records, including adding, retrieving, updating, and deleting information.
* Select courses based on the department.

This system helps educational institutions keep track of student information efficiently.

**2. Technologies Used**

* **Python**: The primary programming language for developing the application.
* **Tkinter**: A standard GUI toolkit in Python for building the user interface.
* **MySQL**: A relational database management system for storing user and student data.
* **mysql.connector**: A library used to connect Python to MySQL.

**3. Features of the System**

**User Authentication**

The login interface allows users to enter their username and password. Upon successful login, users can access the student management interface.

**Code for User Login:**

def logmein():

uid = e1.get().strip() # Get username

pwd = e2.get().strip() # Get password

db = mysql.connector.connect(host="localhost", user="root", password="", db="pgcdharamshala")

cursor = db.cursor()

RetPass = "SELECT upass FROM users WHERE uname='" + uid + "'"

cursor.execute(RetPass)

myresult = cursor.fetchall()

if myresult: # Check if user exists

passw = myresult[0][0].strip()

if passw == pwd:

messagebox.showinfo("LOGIN", "Welcome to Student Administration System")

top.destroy() # Close login window

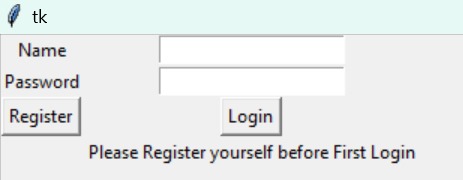
import pgcformstu # Open student management window

else:

messagebox.showinfo("LOGIN", "Wrong Password")

else:

messagebox.showinfo("LOGIN", "User not found")



**User Registration**

Users can register through a dedicated interface. The system checks if the passwords match before storing user details.

**Code for User Registration:**

def Registerme():

snm = e1.get().strip() # Get name

sunm = e2.get().strip() # Get username

spass = e3.get().strip() # Get password

scpass = e4.get().strip() # Confirm password

if spass == scpass: # Check if passwords match

db = mysql.connector.connect(host="localhost", user="root", password="", db="pgcdharamshala")

qry = "INSERT INTO registration VALUES(%s,%s,%s,%s)"

values = (snm, sunm, spass, scpass)

cursor = db.cursor()

cursor.execute(qry, values)

db.commit()

qry = "INSERT INTO users VALUES(%s,%s)"

cursor.execute(qry, (sunm, spass))

db.commit()

messagebox.showinfo("REGISTER", "Registration successful")

top.destroy() # Close registration window

else:

messagebox.showinfo("REGISTER", "Passwords do not match")

A screenshot of a computer login box

Description automatically generated

**Student Record Management**

The system allows for managing student records, including:

* **Adding New Students**: Automatically generates the next roll number based on the department.
* **Retrieving Student Records**: Fetches student details using roll number and department.
* **Updating Student Information**: Modifies existing records based on user input.
* **Deleting Student Records**: Removes records from the database.

**Adding New Student Record**

def insert():

dnm = dpts.get() # Get selected department

rl = e1.get().strip() # Get roll number

name = e2.get().strip() # Get student name

age = e3.get().strip() # Get student age

course = crsnm # Selected course

fee = e5.get().strip() # Get fee

db = mysql.connector.connect(host="localhost", user="root", password="", db="pgcdharamshala")

qry = "INSERT INTO stuinfo VALUES(%s,%s,%s,%s,%s,%s)"

values = (dnm, rl, name, age, course, fee)

cursor = db.cursor()

cursor.execute(qry, values)

db.commit()

messagebox.showinfo("SUCCESS", "Student record inserted successfully")

A screenshot of a computer

Description automatically generated

**4. Code Implementation**

**Full Code**

Below is the complete code for the Student Management System, integrating all functionalities discussed.

from tkinter import \*

import mysql.connector

from tkinter import messagebox

top = Tk()

top.title("Student Management System")

# Function for user login

def logmein():

uid = e1.get().strip()

pwd = e2.get().strip()

db = mysql.connector.connect(host="localhost", user="root", password="", db="pgcdharamshala")

cursor = db.cursor()

RetPass = "SELECT upass FROM users WHERE uname='" + uid + "'"

cursor.execute(RetPass)

myresult = cursor.fetchall()

if myresult:

passw = myresult[0][0].strip()

if passw == pwd:

messagebox.showinfo("LOGIN", "Welcome to Student Administration System")

top.destroy()

import pgcformstu

else:

messagebox.showinfo("LOGIN", "Wrong Password")

else:

messagebox.showinfo("LOGIN", "User not found")

# Function for user registration

def Registerme():

snm = e1.get().strip()

sunm = e2.get().strip()

spass = e3.get().strip()

scpass = e4.get().strip()

if spass == scpass:

db = mysql.connector.connect(host="localhost", user="root", password="", db="pgcdharamshala")

qry = "INSERT INTO registration VALUES(%s,%s,%s,%s)"

values = (snm, sunm, spass, scpass)

cursor = db.cursor()

cursor.execute(qry, values)

db.commit()

qry = "INSERT INTO users VALUES(%s,%s)"

cursor.execute(qry, (sunm, spass))

db.commit()

messagebox.showinfo("REGISTER", "Registration successful")

top.destroy()

else:

messagebox.showinfo("REGISTER", "Passwords do not match")

# Student record management functions (insert, update, delete, retrieve) go here...

# User interface layout (labels, entries, buttons) go here...

top.mainloop()

**User Interface Design**

The interface consists of:

* Labels for fields (e.g., Name, Username, Password).
* Entry widgets for user input.
* Buttons for actions (e.g., Login, Register, Add Student, Retrieve Record).

**5. Conclusion and Future Work**

The Student Management System successfully integrates essential features for managing student records, including user authentication, registration, and CRUD operations.

**Future Work**

1. **Security Enhancements**: Implement password hashing to secure user credentials.
2. **Improved User Interface**: Design a more user-friendly interface with better aesthetics.
3. **Reporting Features**: Add functionalities to generate reports on student performance and attendance.
4. **Role Management**: Differentiate user roles (e.g., admin, student) for tailored access.