KISHKINDA UNIVERSITY

MINI PROJECT REPORT

PROJECT TITLE: FILM PROJECT FUNDING PLATFORM(POC)

PROJECT TEAM MEMBERS

MEGHASHREE	**TEMPBTech-CSE056**
POORNIMA C	**TEMPBTech-CSE061**
BHUVANESHWARI M	**TEMPBTech-CSE036**
RAKSHITHA.K	**KUB23-CSE113**
SNEHA .k	**KUB23-CSE138**

Overview

1 .Introduction.

2.Create.

3.Read.

4.Update.

5.Delete.

6.Alter

7.Code Implementation

8.Output

9.Conclusion

INTRODUCTION

1. A to-do list is a list of tasks that need to be completed, either for a specific time or project.

2. To-do lists can help you to organize your tasks and stay on track.

3. A "To-Do" program is a simple application designed to help users manage and organize tasks or activities they need to complete. It typically provides fuctionalities such as adding new tasks, marking tasks as complete, editing existing tasks, and deleting tasks. The purpose is to enhance productivity and keep track of daily goals.

For Example: On student point of view, a to-do list helps students organize their tasks based on subjects, assignments, deadlines, or importance

CREATE

Create: This operation allows users to add new tasks to their to-do list.

Users can input details such as the task description, due date, and priority level.

Example: A user inputs a task with a pid, name, and status.

Query:

INSERT INTO table_name (column1, column2) VALUES (value1, value2);

READ

Read: This function retrieves and displays the current tasks in the to-do list.

Example: Users can view all tasks, filter by status (completed or pending), or search for specific tasks.

Query:

SELECT * FROM table_name WHERE condition;

UPDATE

Update: Users can modify existing tasks to reflect changes.

(Example: This includes editing the task description, changing the name, or marking a status as completed.)

DELETE

Delete: This operation enables users to remove tasks from their list.

Example: Users can select tasks they no longer need, effectively managing their to-do list.

Query:

UPDATE table_name SET column1 = value1 WHERE condition;

ALTER

In a to-do list application, the "ALTER TABLE" command is used to modify the structure of a database table.

This can include :-

Adding new columns, Removing columns, Renaming columns ...

Query:

DELETE FROM table_name WHERE condition;

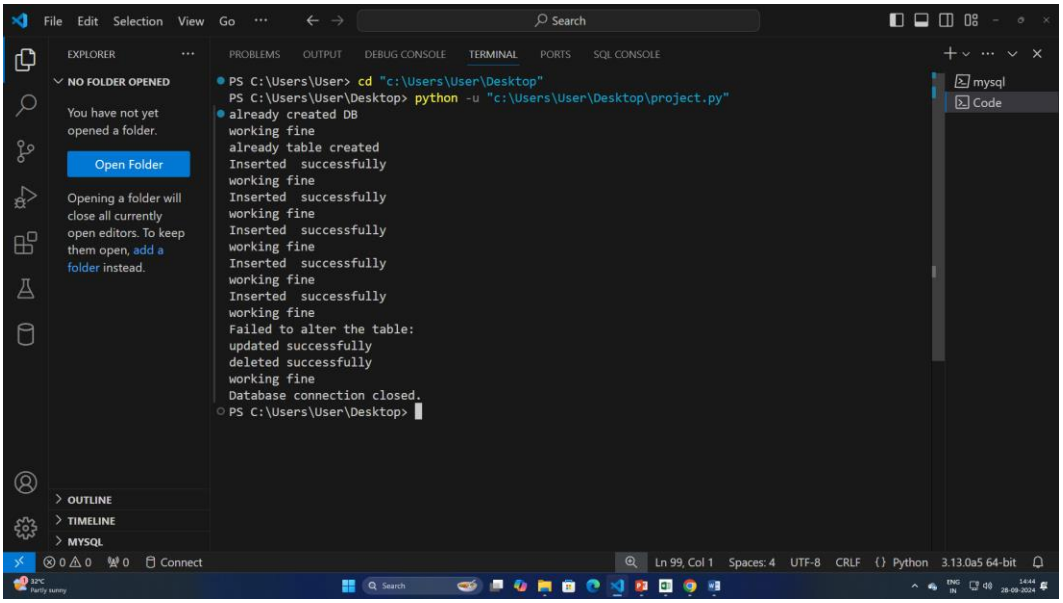
Code Implementation:

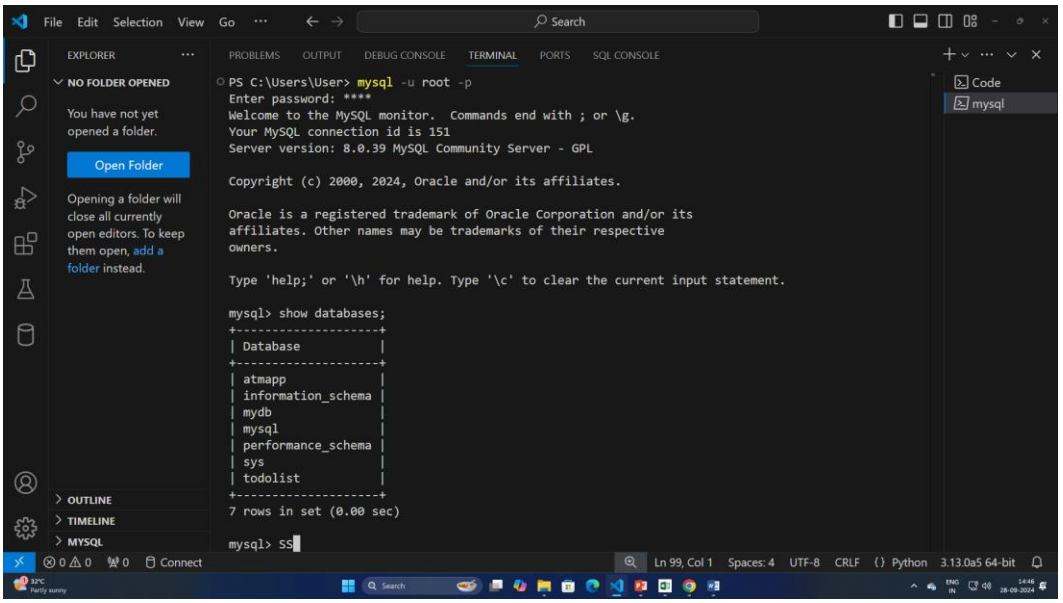
```
import mysql.connector
mydb=mysql.connector.connect(
            host="localhost",
            user="root",
            password="root"

         )
mycursor=mydb.cursor()
class TaskList:
    def __init__(self,pid,name,status,skip):
        self.pid=pid
        self.name=name
        self.status=status
        self.skip=skip
    def __str__(self):
        return f'{self.pid},{self.name},{self.status},{self.skip},'
class TaskManager:
    def createDB():
        try:
            mycursor.execute("CREATE DATABASE ToDoList")
            print("DataBase Create Successfully")
        except Exception :
            print("already created DB")
        else:
            print("working fine")
    def useDB():
        try:
            mycursor.execute("use ToDoList")
        except Exception:
            print("already used DB")
        else:
            print("working fine")
    def createProgram():
        try:
            mycursor.execute("create table Person(pid int(10),name
varchar(30),status varchar(30))")
            print("table created")
        except Exception:
            print("already table created ")
        else:
            ("working fine")
    def insertProgram( pid,name, status,skip):
        try:
```

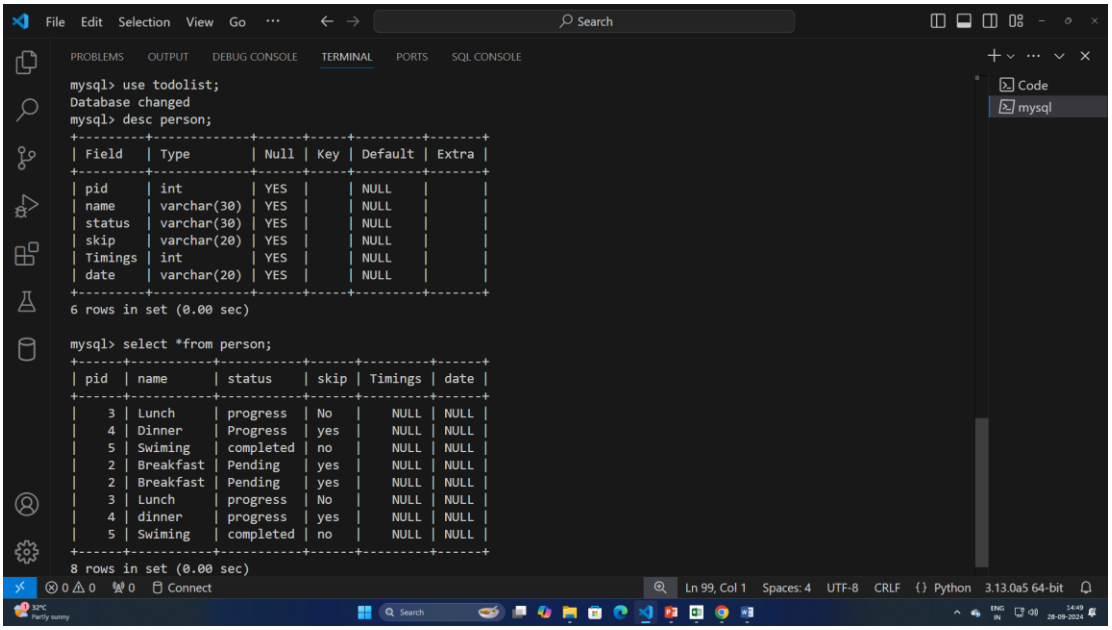
```
            sql = "INSERT INTO Person (pid,name, status,skip) VALUES (%s, %s,
%s,%s)"
            mycursor.execute(sql, (pid,name,status,skip))
            mydb.commit()
            print("Inserted  successfully")
        except Exception :
            print("Failed to insert data into table:")
        else:
            print("working fine")

    def alterTable():
        try:
            sql = "ALTER TABLE Person drop column(date)"
            mycursor.execute(sql)
            mydb.commit()
            print("Altered table successfully")
        except Exception :
            print("Failed to alter the table:")
        else:
            print("Working fine")

    def updateProgram():
        try:
            mycursor.execute("update Person set status ='pending' where pid=1")
            mydb.commit()
            print("updated successfully")
        except Exception:
            print("isuess while creating DB")
        else:
            ("working fine")
    def deleteProgram():
        try:
            mycursor.execute("delete from Person  where pid=1")
            mydb.commit()
            print("deleted successfully")
        except Exception:
            print("issue while Deleteing")
        else:
            print("working fine")
tm=TaskManager
tm.createDB()
tm.useDB()
tm.createProgram()
tm.insertProgram("1", "Exercise","Completed","NO")
tm.insertProgram("2", "Breakfast","Pending","yes")
tm.insertProgram("3", "Lunch","progress","No")
tm.insertProgram("4", "dinner","progress","yes")
```

```
tm.insertProgram("5", "Swiming","completed","no")
tm.alterTable()

tm.updateProgram()
tm.deleteProgram()
mycursor.close()  # Optional: Close the cursor
mydb.close()      #Close the database connection
print("Database connection closed.")
```

output:







Conclusion:

This program demonstrates a simple Task Management System using MySQL and python. With enhancements and extensions, it become a efficient task management system