**COIMBATORE INSTITUTE OF TECHNOLOGY**

**(Government Aided Autonomous Institution)**

**COIMBATORE-641014**

**TAMILNADU**

**LABORATORY PROJECT REPORT**

**DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**

**ACADEMIC YEAR: 2022-2023**

21CSL22-PYTHON PROGRAMMING LABORATORY

**TO DO LIST**

**Batch No. ----**

**Project done by**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Register No.** | **Name of the Student** |
| **1.** | 71762208002 | AAKASH M V |
| **2.** | 71762208003 | ADITHYA B |
| **3.** | 71762208032 | NARESH KUMAR P |
| **4.** | 71762208049 | SHAKTHI NANDHAN P |

**SIGNATURE OF THE FACULTY MEMBER**

**PROJECT TITLE: TO DO LIST**

**AIM OF THE PROJECT:**

To keep track of pending tasks andcategorize them

**SOFTWARE USED :**

* Python
* Tkinter
* Sqlite

**PROJECT DESCRIPTION/THEORY :**

A TO DO LIST, also known as a task list or checklist, is a widely used tool that helps individuals organize and manage their tasks and responsibilities effectively. It serves as a simple yet powerful method to track, prioritize, and accomplish various tasks or goals within a specific timeframe. Whether you are a student, a professional, a homemaker, or anyone with tasks to complete, a TO DO LIST can be an indispensable tool for improving productivity and reducing stress.

**ALGORITHM:**

1. **Start**
2. **Show the main menu window**
3. **When the add tasks button is clicked show the add tasks window**
4. **When the show all tasks button is clicked show all the tasks**
5. **When the checklist is clicked open the checklist window**
6. **Stop**

**FLOWCHART:**

**SOFTWARE IMPLEMENTATION:**

The combined output of tkinter, sqlite and general python logic produces the graphical user interface for the to do list project.

**SOURCE CODE:**

MAIN MENU:

import shower as show

from tkinter import \*

from tkinter import ttk

import Checklist as c

import adder as adder

win= Tk()

win.geometry("950x400")

win.title('tasks')

def open\_add():

show.start()

def open\_pen():

adder.add\_task()

def open\_imp():

c.h()

Label(win, text= "ALL TASKS", font= ('Helvetica 17 bold')).pack(pady=10)

ttk.Button(win, text="Open all task", command=open\_add).pack(pady=5)

Label(win, text= "ADD TASKS", font= ('Helvetica 17 bold')).pack(pady=10)

ttk.Button(win, text="Open task adder", command=open\_pen).pack(pady=5)

Label(win, text= "Checklist", font= ('Helvetica 17 bold')).pack(pady=10)

ttk.Button(win, text="View checklist", command=open\_imp).pack(pady=5)

win.mainloop()

ADDER:

from tkinter import \*

from tkinter import ttk

from tkcalendar import DateEntry

from datetime import datetime

import SqliteHelper as sq

values = []

def add\_to\_list():

global task\_name\_entry

global description\_entry

global urgency\_entry

global due\_date\_entry

global target\_date\_entry

global add\_button

global values

task\_name = task\_name\_entry.get()

task\_name\_entry.delete(0,END)

description = description\_entry.get()

description\_entry.delete(0,END)

urgency = urgency\_entry.get()

urgency\_entry.delete(0,END)

target\_date = target\_date\_entry.get()

target\_date\_entry.delete(0,END)

due\_date = due\_date\_entry.get()

due\_date\_entry.delete(0,END)

values.append([task\_name, description, urgency, target\_date, due\_date])

sq.write\_to\_table(values)

values=[]

def add\_task():

MyFrame= Tk()

MyFrame.geometry("550x350")

global task\_name\_entry

global description\_entry

global urgency\_entry

global due\_date\_entry

global target\_date\_entry

global add\_button

ttk.Label(MyFrame,font=("Arial 20"), text="Introduction", background="#ffffff").grid(row=1, column=2, sticky="w")

Canvas(MyFrame, height=1, background="#a0a0a0", highlightthickness=0, highlightbackground="white").grid(row=1, column=3, columnspan=3, sticky="we")

task\_name\_l=ttk.Label(MyFrame,text="Enter task name:",width=30)

task\_name\_l.grid(row=3,column=2)

task\_name\_entry = ttk.Entry(MyFrame, width=20)

task\_name\_entry.grid(row=3, column=3)

description\_l=ttk.Label(MyFrame,text="Enter task description:",width=30)

description\_l.grid(row=4,column=2)

description\_entry = ttk.Entry(MyFrame, width=20)

description\_entry.grid(row=4, column=3)

urgency\_l=ttk.Label(MyFrame,text="Enter task urgency:",width=30)

urgency\_l.grid(row=5,column=2)

urgency\_entry = ttk.Entry(MyFrame,width=20)

urgency\_entry.grid(row=5,column=3)

target\_l=ttk.Label(MyFrame,text="Enter target date:",width=30)

target\_l.grid(row=6,column=2)

target\_date\_entry = DateEntry(MyFrame,width=20,date\_pattern='yyyy-mm-dd')

target\_date\_entry.grid(row=6,column=3)

due\_l=ttk.Label(MyFrame,text="Enter due date:",width=30)

due\_l.grid(row=7,column=2)

due\_date\_entry = DateEntry(MyFrame,width=20,date\_pattern='yyyy-mm-dd')

due\_date\_entry.grid(row=7,column=3)

add\_button = ttk.Button(MyFrame,text="Add Task",command=add\_to\_list)

add\_button.grid(row=8,columnspan=2,pady=(10))

MyFrame.mainloop()

CHECKLIST:

from tkinter import \*

import tkinter

def newcheck(ba):

global m

global a

global b

global to

global changeval

m+=1

to=Frame(top)

done=IntVar()

txt=ba.get()

ba.delete(0,END)

a=Label(to,text=txt,bg="#fff")

ha=Checkbutton(to,variable=done,onvalue=3,offvalue=1,height=5,width=5)

ha.grid(row=0,column=0)

b=Button(to,text="x",command=lambda: [a.destroy(),b.destroy(),to.destroy(),globals().update(m=m-1)])

a.grid(row=0,column =1)

b.grid(row=0,column=2)

to.grid(row=m,column=columnval)

def ch(to):

global n

col=to.grid\_info().get("column")

col=2 if col==1 else 1

to.grid(column=col)

n+=1

def h():

global top

global ti

global m

global n

global columnval

top = tkinter.Tk()

top.title("Checklist of Tasks")

top.geometry("950x250")

ti=Label(top,text="Checklist of tasks",font=("Arial 7"))

ti.grid(row=0,column=1)

i=Label(top,text=" ")

n=IntVar()

m=2

columnval=1

n=1

task=["task1",n]

entry=Entry(top)

add=Button(top,text="+",font=("Arial",10),command=lambda: newcheck(entry))

i.grid(row=1,column=0)

entry.grid(row=1,column=0)

add.grid(row=1,column=2)

tc=Label(top,text="Completed",font=("Arial 7"))

tc.grid(row=0,column=3)

top.mainloop()

SQLITE HELPER:

import sqlite3 as sq

connection=sq.connect("Task.db")

cursor=connection.cursor()

'''cursor.execute("""CREATE TABLE task (

task\_name VARCHAR(255),

description VARCHAR(255),

urgency INT,

target\_date DATE,

due\_date DATE

)""")'''

def read\_table():

cursor.execute('SELECT \* FROM task')

rows = cursor.fetchall()

list\_of\_lists = []

for row in rows:

list\_of\_lists.append(list(row))

return list\_of\_lists

def write\_to\_table(list\_of\_lists):

for inner\_list in list\_of\_lists:

task\_name = inner\_list[0]

description = inner\_list[1]

urgency = inner\_list[2]

target\_date = inner\_list[3]

due\_date = inner\_list[4]

cursor.execute("INSERT INTO task (task\_name, description, urgency, target\_date, due\_date) VALUES (?, ?, ?, ?, ?)", (task\_name, description, urgency, target\_date, due\_date))

connection.commit()

a=read\_table()

print(a)

SHOW ER:

from tkinter import \*

import tkinter as tk

import SqliteHelper as sq

def start():

data=sq.read\_table()

root=Tk()

root.geometry("950x250")

root.title("EISENHOWER MATRIX")

headers = ['Task Name', 'Description', 'Urgency', 'Target Date', 'Due Date']

for j, header in enumerate(headers):

label = tk.Label(root, text=header)

label.grid(row=0, column=j)

for i, row in enumerate(data):

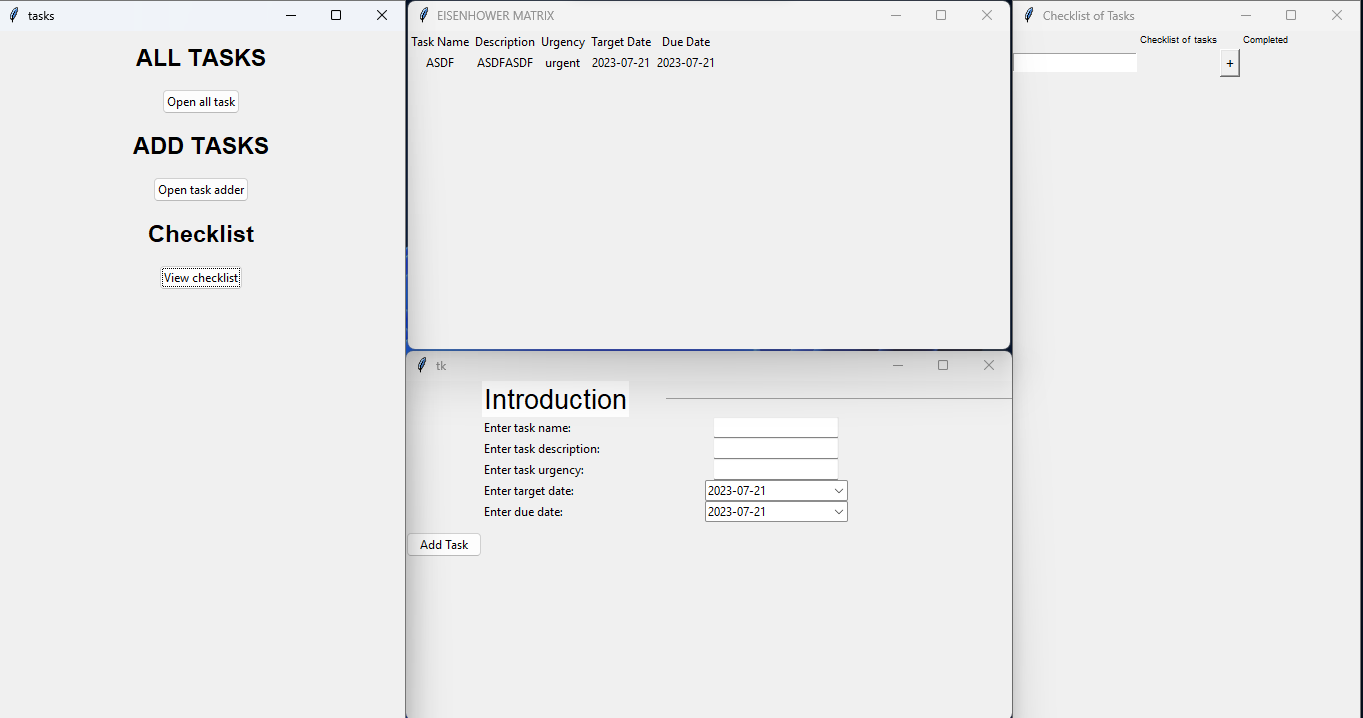
for j, value in enumerate(row):

label = tk.Label(root, text=value)

label.grid(row=i+1, column=j)

root.mainloop()

**SIMULATION RESULTS – SCREENSHOT OF OUTPUT**



**RESULTS & DISCUSSION**:

From this project we learnt what the python modules Tkinter and Sqlite 3 can do and we have successfully implemented a to do list where we can keep our work organised and we make sure that we don’t forget any of our work and finish them within the deadline.

**CONCLUSION:**

Thus the Python task management program provides an efficient and user-friendly solution for organizing tasks and priorities. Its robust functionality enhances productivity and reduces task oversight risks. This project highlights Python's effectiveness in addressing real-world challenges.

**REFERENCES:**

**PYTHON TKINTER DOCUMENTATION:**

[**https://docs.python.org/3/library/tkinter.html**](https://docs.python.org/3/library/tkinter.html)

**PYTHON SQLITE 3 DOCUMENTATION:**

**https://www.sqlite.org/docs.html**