EX NO 4(a)	Develop a GUI based to do list application where user can add, delete and manage their task
DATE	

Aim:

To Develop a GUI-based to-do list application where users can add, delete, and manage their tasks.

Algorithm:

Step 1: Start
Step 2: Design a graphical user interface (GUI) with input field to add tasks, a listbox to display tasks, and buttons for adding, deleting, and managing tasks.
Step 3: Use appropriate data structures like lists or dictionaries to store and manage tasks
Step 4: Stop the process

Program:

```
import tkinter as tk

def add_task():
   task = task_entry.get()

   if task:
      tasks.append(task)
      update_task_list()

def delete_task():
   selected_task_index = task_listbox.curselection()
   if selected_task_index:
      tasks.pop(selected_task_index[0])
      update_task_list()

def update_task_list():

   task_listbox.delete(0, tk.END)

   for task in tasks:
      task_listbox.insert(tk.END, task)

root = tk.Tk()
root.title("To-Do List")

tasks = []

task_entry = tk.Entry(root, width=30)
task_entry.grid(row=0, column=0, padx=5, pady=5)
```

Ex. No. 4(b)	**Develop a GUI-based notes application where users can create, edit, and manage notes with features like formatting text, saving notes, and organizing them into categories.**
DATE	

Aim:

To develop a GUI-based notes application that allows users to create, edit, organize, and save notes within categories.

Algorithm:

Step1: Initialize the main window with category, note list, and editor sections.
Step2: Enable adding new categories and display them in the category list.
Step3: When a category is selected, load its notes into the note list.
Step4: Allow creating new notes within the selected category.
Step5: Load selected notes into the editor for editing.
Step6: Save edited content back to the note data structure .

Program:

```
import tkinter as tk
from tkinter import ttk, simpledialog, messagebox

class NotesApp:

  def __init__(self, master):

    self.master = master
    master.title("Simple Notes App")

    self.categories = {}
    self.current_category = None
    self.current_note = None

    # Category frame

    cat_frame = ttk.Frame(master)

    cat_frame.pack(side=tk.LEFT, fill=tk.Y, padx=5, pady=5)

    ttk.Label(cat_frame, text="Categories").pack()
    self.cat_listbox = tk.Listbox(cat_frame)
    self.cat_listbox.pack(fill=tk.Y, expand=True)
    self.cat_listbox.bind('<<ListboxSelect>>', self.load_notes)

    ttk.Button(cat_frame, text="Add Category",
    command=self.add_category).pack(pady=5)
```