Project Report

GUI To-Do List Application

Submitted by: ARYAN BARDIYA

REG.NO. = 25BAI10890

Language: Python 3 (Tkinter)

Date: 24-11-2025

1. Abstract

The GUI To-Do List is a desktop application designed to help users organize tasks efficiently. Built using the tkinter library, it features a CRUD (Create, Read, Update, Delete) architecture managed through a graphical interface. The application emphasizes visual feedback, using color-coded rows and real-time counters to track user progress. This report outlines the system architecture, design choices, and implementation details.

2. Introduction

Task management is a fundamental application of computer science. This project aims to create a robust "To-Do List" that moves beyond simple text storage. By utilizing a Graphical User Interface, we provide users with buttons, input fields, and interactive lists, making the experience intuitive for non-technical users.

3. System Analysis

3.1 Requirements

* Python 3.x: The core interpreter.

* Tkinter: Built-in GUI module for Python.

* OS: Cross-platform compatibility (Windows, Linux, macOS).

3.2 Feasibility

The project is highly feasible as it requires no external database engines or internet connectivity. It runs locally, ensuring speed and privacy.

4. System Design

4.1 Modular Layout

The User Interface (UI) is divided into distinct Frames to ensure a clean layout:

* Header Frame: Displays the application title.
* Input Frame: Contains the Entry widget and "Add" button.
* Button Frame: Houses the action controls (Complete, Delete, Clear).
* List Frame: Contains the Treeview widget for displaying tasks.
* Counter Frame: A footer showing live statistics.

4.2 Data Structure

The application uses a List of Dictionaries to store state:

```
self.tasks = [
    {'text': 'Buy Groceries', 'completed': False},
    {'text': 'Study Python', 'completed': True}
]
```

This structure allows us to separate the data (the list) from the view (the Treeview widget), adhering to basic MVC (Model-View-Controller) principles.

5. Implementation Details

5.1 The Treeview Widget

We utilized the ttk.Treeview widget to create a multi-column list.

* Columns: Status, Task Description.
* Tags: We defined tags ('pending', 'completed') to dynamically change the background color of rows based on the task status.

5.2 Event Binding

The application utilizes event listeners to improve UX:

* <Return>: Pressing Enter in the input box adds the task.
* <Double-1>: Double-clicking a row toggles its status between Pending and Complete.

5.3 Style Configuration

We used ttk.Style() to customize the look and feel. Standard Tkinter widgets often look outdated; by configuring styles, we achieved a modern, flat design with a specific color palette (Teal, Blue, Orange).

6. Limitations & Future Scope

6.1 Limitations

* Persistence: Tasks are stored in RAM. Closing the application clears the list.

* Single List: The app currently supports only one main list, not multiple categories (e.g., Work vs. Personal).

6.2 Future Scope

* Database Integration: Connecting SQLite to save tasks permanently.

* Due Dates: Adding a date picker to set deadlines for tasks.

* Dark Mode: Implementing a toggle for Dark/Light themes.

7. Conclusion

The To-Do List application successfully demonstrates the power of Python's tkinter for creating desktop GUI applications. It fulfills all functional requirements—adding, updating, and deleting tasks—while providing a polished user experience through careful styling and layout management.