འབྲུག་མདའ་རྒྱལ་ཁོར་ཡུག་མཐོ་རིམ་སློབ་གྲྭ།
Royal University of Bhutan
Jigme Namgyal Engineering Collage, Dewathang

Draft GUI Report – To Do List Tracker

Submitted by:

Norbu Wangmo (05240141)

DIPLOMA IN COMPUTER SYSTEM AND NETWORK

JIGME NAMGYEL ENGINEERING COLLEGE ROYAL UNIVERSITY OF BHUTAN

DEWATHANG

(20th August, 2025)

Contents

List of Figures

Introduction

The To-Do List Tracker is a comprehensive personal task management system designed to help users organize, track, and manage their daily tasks and activities efficiently. This intuitive application allows users to create tasks, categorize them by status, set priorities, and monitor progress through a structured workflow. This system modernizes productivity by replacing manual lists with digital task management, reducing mental load through organized categorization. It enhances efficiency with visual progress tracking, deadline management, and reliable data persistence for all tasks.

Main Window

This code creates a To-Do List Tracker application with a graphical user interface. It includes user authentication (login/register), task management with priorities and status tracking, and reminder functionality. The application uses SQLite for data storage and features a clean, colorful interface with interactive buttons and filtering options.

```
class TodoApp(tk.Tk):
    def __init__(self):
        super().__init__()
        self.title("To-Do List Tracker")
        self.geometry("600x500")  # Larger welcome interface
        self.resizable(False, False)
        self.configure(bg="#f0f8ff")  # Light blue background

        # Center the window on screen
        self.update_idletasks()
        x = (self.winfo_screenwidth() // 2) - (600 // 2)
        y = (self.winfo_screenheight() // 2) - (500 // 2)
        self.geometry(f"600x500+{x}+{y}")

        self.current_user = None  # (id, username, email, ...)
        self._build_styles()
        self._build_auth_frame()
```

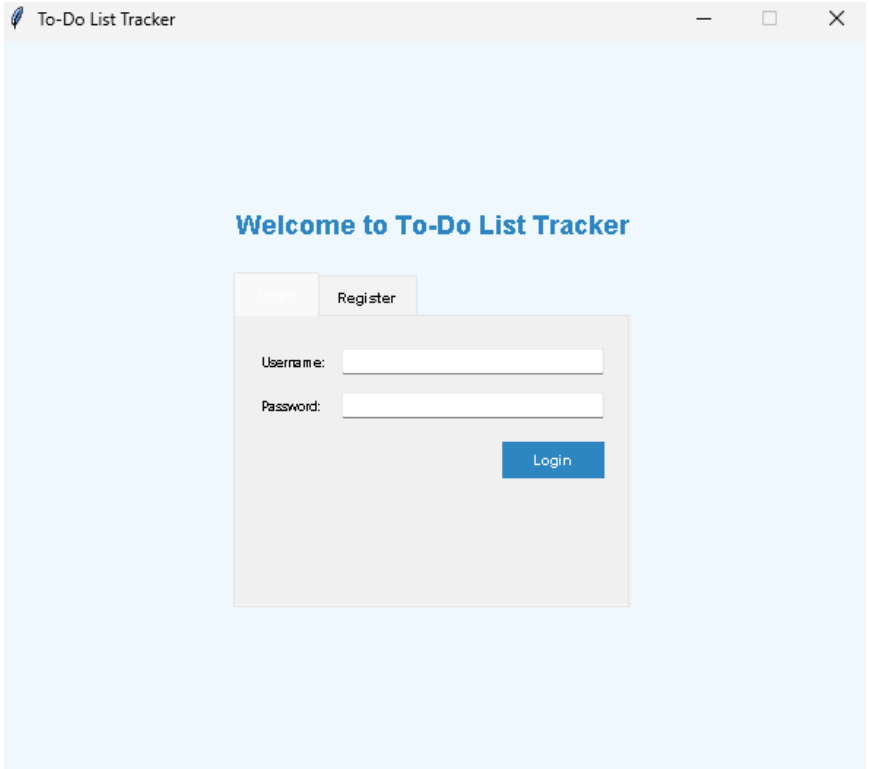


Figure 1 welcome interface

Input Fields

The interface includes text boxes for entering usernames, passwords, email addresses, and task details like titles, descriptions, and due dates, with dropdown menus for selecting status and priority levels.

```
# Login tab
self.login_username = tk.StringVar()
self.login_password = tk.StringVar()

ttk.Label(self.login_tab, text="Username:").grid(row=0, column=0, sticky="w", pady=6)
ttk.Entry(self.login_tab, textvariable=self.login_username).grid(row=0, column=1, sticky="ew", pady=6, padx=(10, 0))
ttk.Label(self.login_tab, text="Password:").grid(row=1, column=0, sticky="w", pady=6)
ttk.Entry(self.login_tab, textvariable=self.login_password, show="*").grid(row=1, column=1, sticky="ew", pady=6, padx=(10, 0))
self.login_tab.grid_columnconfigure(1, weight=1)
```

```
# Register tab
self.reg_username = tk.StringVar()
self.reg_email = tk.StringVar()
self.reg_password = tk.StringVar()
self.reg_password2 = tk.StringVar()

ttk.Label(self.register_tab, text="Username:").grid(row=0, column=0, sticky="w", pady=6)
ttk.Entry(self.register_tab, textvariable=self.reg_username).grid(row=0, column=1, sticky="ew", pady=6, padx=(10, 0))
ttk.Label(self.register_tab, text="Email:").grid(row=1, column=0, sticky="w", pady=6)
ttk.Entry(self.register_tab, textvariable=self.reg_email).grid(row=1, column=1, sticky="ew", pady=6, padx=(10, 0))
ttk.Label(self.register_tab, text="Password:").grid(row=2, column=0, sticky="w", pady=6)
ttk.Entry(self.register_tab, textvariable=self.reg_password, show="*").grid(row=2, column=1, sticky="ew", pady=6, padx=(10, 0))
ttk.Label(self.register_tab, text="Confirm Password:").grid(row=3, column=0, sticky="w", pady=6)
ttk.Entry(self.register_tab, textvariable=self.reg_password2, show="*").grid(row=3, column=1, sticky="ew", pady=6, padx=(10, 0))
self.register_tab.grid_columnconfigure(1, weight=1)
```

1. **Username Entry Fields:**
   The application creates labeled text entry fields for username input in both the login and registration forms, allowing users to enter their credentials.

2. **Password Security Fields:**
   For security, the password fields display asterisks (*) instead of actual characters when users type, protecting sensitive information from being visible.

3. **Registration Form Fields:**
   The registration form includes additional fields specifically for email input and a second password field for confirmation, ensuring accurate account creation.

4. **Task Management Inputs:**
   For task creation and editing, the application provides text entry fields for title, description, and due date, along with dropdown menus for selecting status (Pending, To Do, Done) and priority levels (1, 2, 3).

Figure 2 Text entry fields

```
ttk.Label(frm, text="Status").grid(row=3, column=0, sticky="w", pady=4)
ttk.Combobox(frm, textvariable=self.var_status, values=["Pending","To Do","Done"], state="readonly", width=18)\
    .grid(row=3, column=1, sticky="w", pady=4)

ttk.Label(frm, text="Priority (1-3)").grid(row=4, column=0, sticky="w", pady=4)
ttk.Combobox(frm, textvariable=self.var_prio, values=["1","2","3"], state="readonly", width=8)\
    .grid(row=4, column=1, sticky="w", pady=4)
```

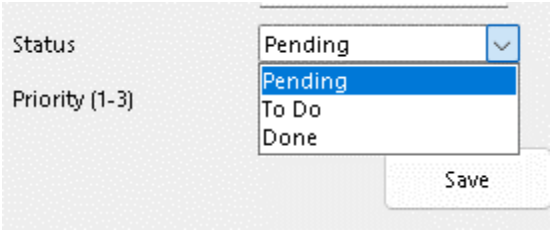


Figure 3 Status level

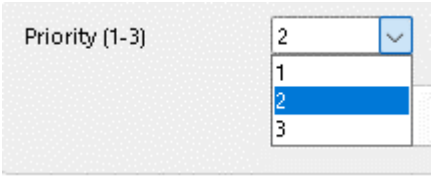


Figure 4 Priority level

Buttons

Colorful, interactive buttons allow users to perform all actions including logging in, registering, adding/editing/deleting tasks, marking tasks as done, applying filters, and logging out.

1. "Login" button to authenticate users

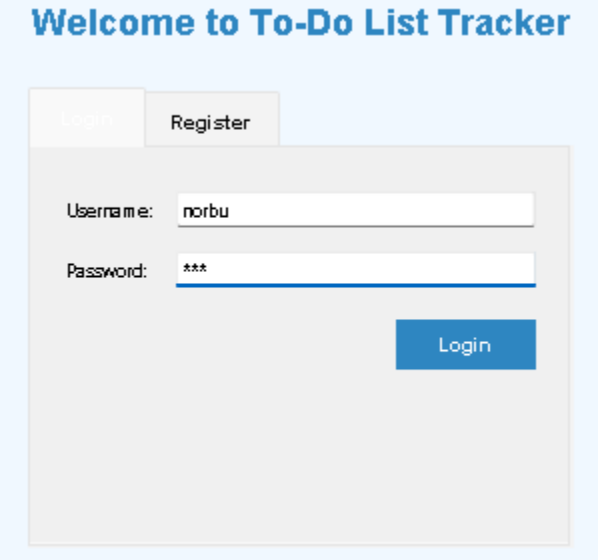


Figure 5 login button

2. "Register" button to create new accounts

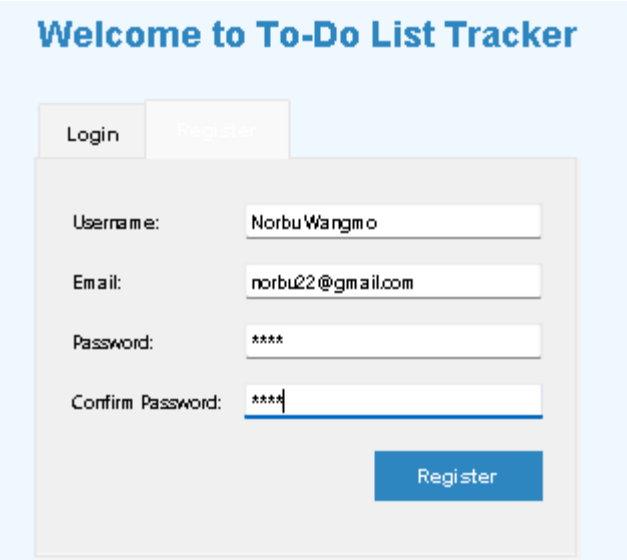


Figure 7 Register button

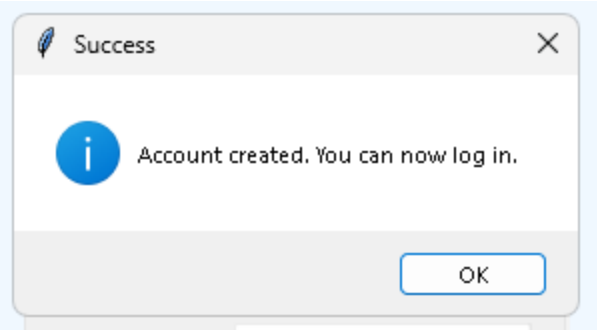


Figure 6 Account created

3. "Add Task" button to insert new tasks

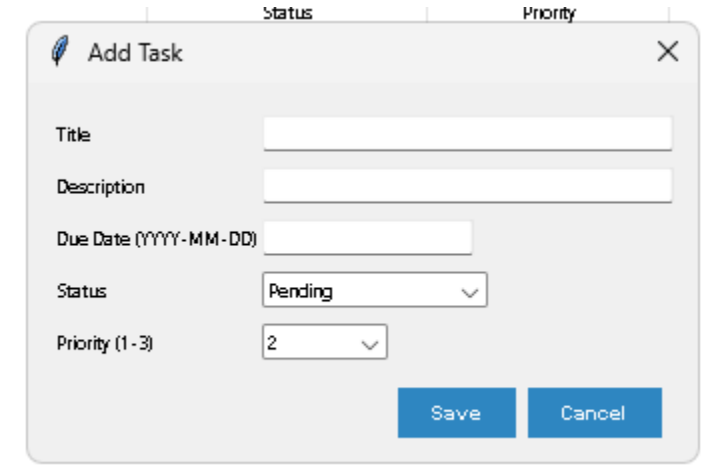


Figure 8 Add task button

4. "Edit Task" button to modify existing tasks

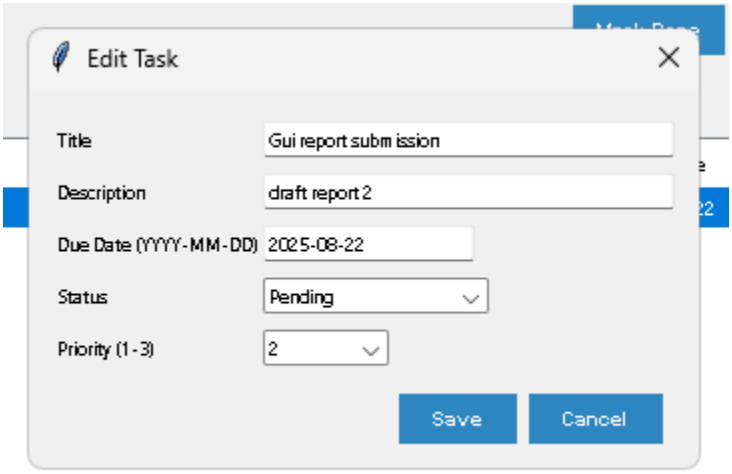


Figure 9Edit task button

5. "Delete Task" button to remove tasks

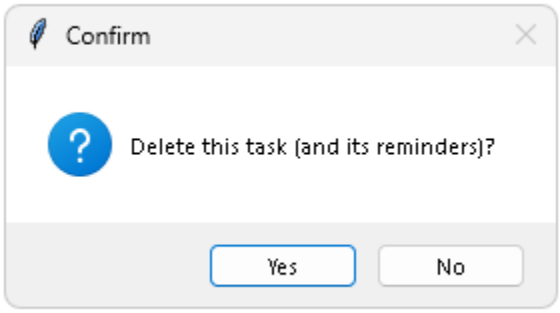


Figure 10 Delete task button

6. "Mark Done" button to update task status

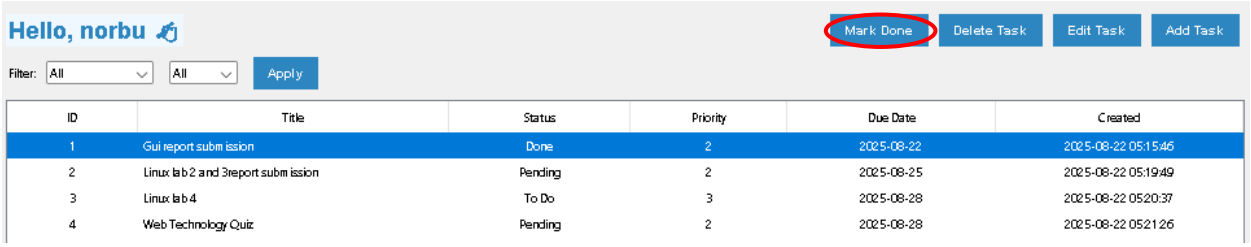


Figure 11 "Mark Done" button

7. "Logout" button to return to login screen

Figure 12Logout button

8. "Apply" button for filtering tasks

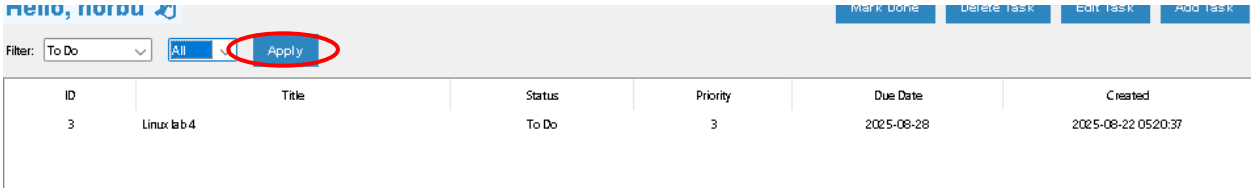
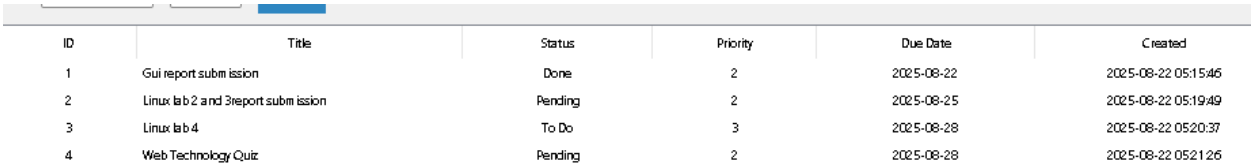


Figure 13Apply button

Display Area

Tasks are shown in a neat table format with columns showing all important information, and filter options help users organize and find their tasks easily.



ID	Title	Status	Priority	Due Date	Created
1	Gui report submission	Done	2	2025-08-22	2025-08-22 05:15:46
2	Linux lab 2 and 3report submission	Pending	2	2025-08-25	2025-08-22 05:19:49
3	Linux lab 4	To Do	3	2025-08-28	2025-08-22 05:20:37
4	Web Technology Quiz	Pending	2	2025-08-28	2025-08-22 05:21:26

Figure 14Display area

Treeview widget shows task records in tabular form with columns for ID, Title, Status, Priority, Due Date, and Created date. The display includes filtering options for status and priority.

Current Status

GUI layout is complete for both authentication and main application

Database connection is implemented

Basic CRUD operations are functional (Create, Read, Update, Delete)

User authentication system is working

Task management features are implemented

Reminder system is partially implemented

Next step: Enhance reminder functionality and add data validation