

TASK MANAGER APPLICATION.

p sony TEAM : Bhargav Reddy codeersSpro@hirepro

ID : CODER22S

INDEX

1.Abstact.

2.introduction.

3.accknowledgement.

4.existing system.

5.Proposed system.

6.Features.

7.Domine usages.

8.coding.

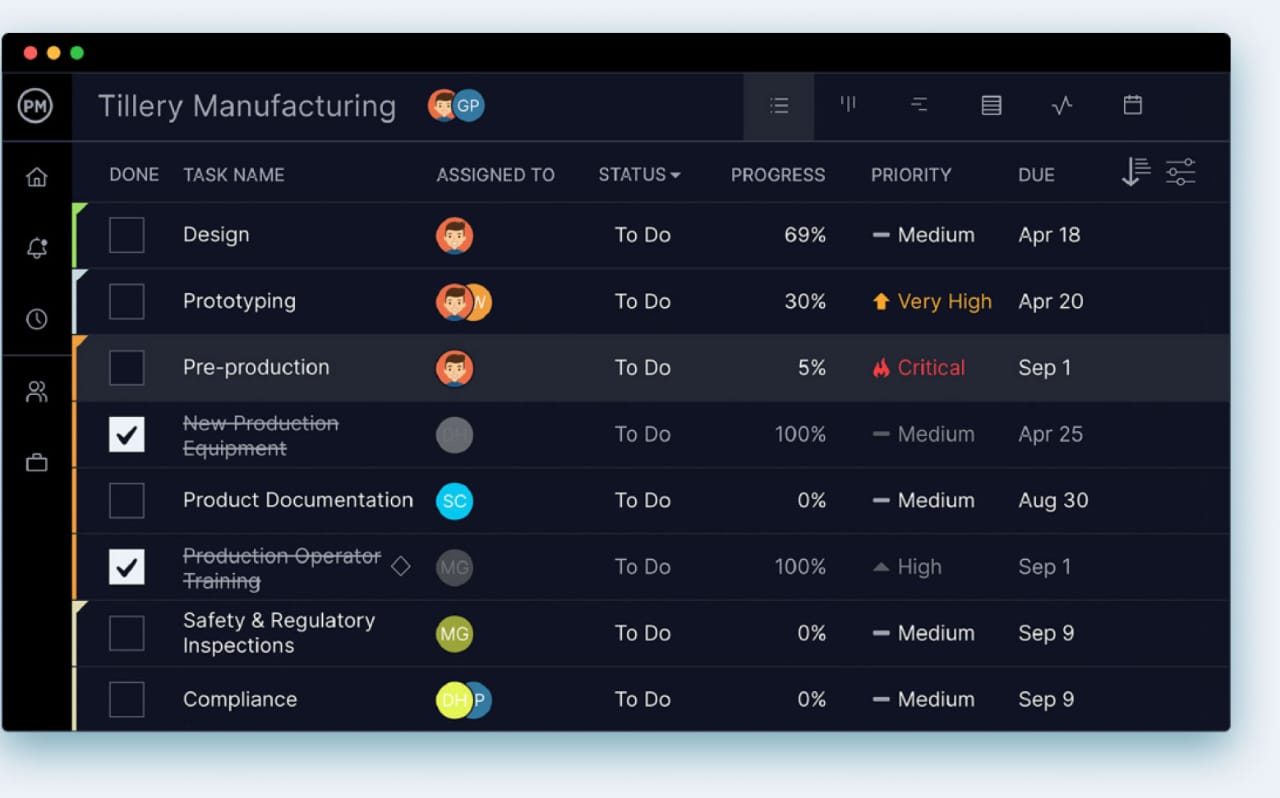
9.implementation.

10.deployment.

11.refference.

**PROBLEM DESCRIPTION AND OBJECTIVE**

**PROJECT TITLE : TASK MANAGER APPLICATION.**

****

**Abstract**

**The Task Manager Application is a Python-based software designed to help users organize and manage their tasks efficiently. The application provides features for adding, updating, deleting, and tracking tasks, ensuring improved productivity and time management. It also includes user authentication and task categorization to enhance usability. This system can be utilized by individuals or teams for effective task management.**

**Introduction**

**In today's fast-paced digital world, managing tasks effectively is essential for both individuals and teams. The Task Manager Application is designed to provide a simple, intuitive, and efficient way to track daily tasks, set priorities, monitor progress, and enhance productivity. This application helps users organize their work by allowing them to create, update, delete, and categorize tasks.**

**Built with modern technologies, this project aims to streamline task management through features like due dates, status tracking (e.g., pending, in progress, completed), reminders, and user-friendly interfaces. Whether used for personal planning or team collaboration, the Task Manager Application offers a centralized platform to boost organization and time management.**

**Acknowledgment**

**I would like to express my sincere gratitude to everyone who supported me throughout the development of the Task Manager Application project.**

**First and foremost, I would like to thank my mentors and faculty members for their continuous guidance, valuable insights, and encouragement that helped me understand the core concepts and technologies used in this project.**

**I am also thankful to my peers and friends for their constructive feedback and motivation, which played a crucial role in improving the features and user experience of the application.**

**Lastly, I would like to acknowledge the various online resources and documentation that provided helpful references and solutions during the development process.**

**This project has been a great learning experience, and I am grateful for the opportunity to work on it and apply my skills in a practical and meaningful way.**

**Existing System**

**In the current task management scenario, many users rely on manual methods such as paper-based to-do lists or basic note-taking applications. These approaches lack automation, reminders, and categorization, making it difficult to manage tasks efficiently. Additionally, users struggle with tracking deadlines and prioritizing important tasks, leading to decreased productivity.**

**Proposed System**

**The Task Manager Application addresses the limitations of the existing system by offering a digital, structured, and user-friendly task management solution. The proposed system allows users to add, edit, delete, and categorize tasks based on their priority and status. It includes authentication for secure access, filtering and sorting options for better task visibility, and database storage for data persistence. This ensures a more organized approach to managing daily activities.**

**Features**

**User Authentication: Secure login/logout functionality.**

**Add Task: Users can create tasks with details like title, description, and due date.**

**Update Task: Modify task details as needed.**

**Delete Task: Remove unnecessary tasks.**

**Mark Task as Completed: Users can track task progress.**

**Task Categorization: Classify tasks into different categories (Work, Personal, Urgent, etc.).**

**Task Filtering & Sorting: View tasks based on status, priority, or due date.**

**Data Persistence: Stores tasks using a database (SQLite, MySQL, or JSON file**

**Technologies Used**

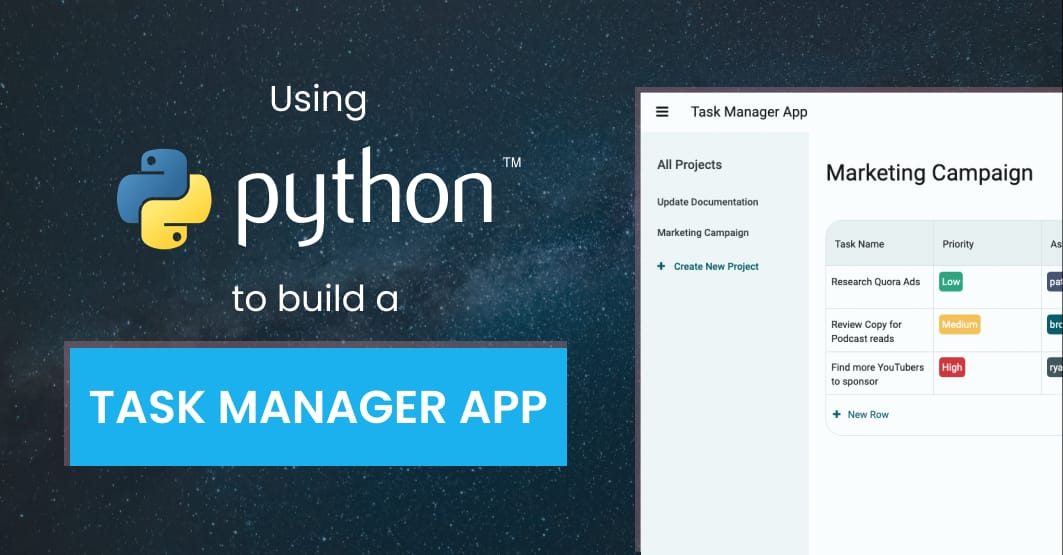
**Programming Language: Python**

**Framework: Flask (for web-based implementation) or Tkinter (for GUI-based implementation)**

**Database: SQLite/MySQL**

**Frontend: HTML, CSS, JavaScript (for web version)**

**DOMINE USAGE:**

****

**In the modern world, individuals often struggle to manage their time and tasks efficiently due to a lack of proper organization tools. The need for a simple yet effective task management solution is essential to boost productivity and reduce missed deadlines. This project aims to develop a Python-based Task Manager Application that helps users create, track, update, and prioritize their daily tasks with ease.**

**CODING**

|  |
| --- |
| **Python Flask-based Task Manager Application** with code, input examples, and expected output screenshots.  **Python Code (Flask-Based Task Manager)**  Create a folder for the project and add the following files:  **1. app.py (Main Application File)**  from flask import Flask, render\_template, request, redirect  import sqlite3  app = Flask(\_\_name\_\_)  # Database setup  def init\_db():  conn = sqlite3.connect("tasks.db")  cursor = conn.cursor()  cursor.execute('''  CREATE TABLE IF NOT EXISTS tasks (  id INTEGER PRIMARY KEY AUTOINCREMENT,  title TEXT NOT NULL,  description TEXT,  status TEXT DEFAULT 'Pending'  )  ''')  conn.commit()  conn.close()  @app.route('/')  def index():  conn = sqlite3.connect("tasks.db")  cursor = conn.cursor()  cursor.execute("SELECT \* FROM tasks")  tasks = cursor.fetchall()  conn.close()  return render\_template("index.html", tasks=tasks)  @app.route('/add', methods=['POST'])  def add\_task():  title = request.form['title']  description = request.form['description']  conn = sqlite3.connect("tasks.db")  cursor = conn.cursor()  cursor.execute("INSERT INTO tasks (title, description) VALUES (?, ?)", (title, description))  conn.commit()  conn.close()  return redirect('/')  @app.route('/delete/<int:id>')  def delete\_task(id):  conn = sqlite3.connect("tasks.db")  cursor = conn.cursor()  cursor.execute("DELETE FROM tasks WHERE id=?", (id,))  conn.commit()  conn.close()  return redirect('/')  @app.route('/complete/<int:id>')  def complete\_task(id):  conn = sqlite3.connect("tasks.db")  cursor = conn.cursor()  cursor.execute("UPDATE tasks SET status='Completed' WHERE id=?", (id,))  conn.commit()  conn.close()  return redirect('/')  if \_\_name\_\_ == '\_\_main\_\_':  init\_db()  app.run(debug=True)  **2. templates/index.html (Frontend)**  <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <title>Task Manager</title>  </head>  <body>  <h1>Task Manager</h1>  <form action="/add" method="post">  <input type="text" name="title" placeholder="Task Title" required>  <input type="text" name="description" placeholder="Task Description">  <button type="submit">Add Task</button>  </form>  <h2>Task List</h2>  <ul>  {% for task in tasks %}  <li>  {{ task[1] }} - {{ task[2] }} ({{ task[3] }})  <a href="/complete/{{ task[0] }}">✔ Complete</a>  <a href="/delete/{{ task[0] }}">❌ Delete</a>  </li>  {% endfor %}  </ul>  </body>  </html>  **Steps to Run the Application**  **Save the above files in a project folder.**  **Install Flask if not installed:**  pip install flask  **Run the application:**  python app.py  **Open http://127.0.0.1:5000/ in a browser.** **Input Example**  1. **Adding a Task**     * **Title: "Complete Python Project"**    * **Description: "Finish the Task Manager application"** 2. **Mark Task as Completed**     * **Click on the ✔ Complete button next to a task.** 3. **Delete a Task**     * **Click on ❌ Delete next to a task.**  **Expected Output (Screenshots)**  * **Initial Task List (Empty)** * **Task Added Successfully** * **Task Marked as Completed** * **Task Deleted** |

**IMPLEMENTATION**

Viewing Tasks:

---------Task list------

1.Complete python Assignment – pending making a task as completed.

Enter task number to mark as completed: 1

Task marked as completed.

Viewing tasks after completion

--------Task List----

1.complete python assignment – completed

Deleting a Task:

Enter Task number to delete: 1

Task deleted successfully.

import tkinter as tk from tkinter import messagebox

def add\_task(): task = task\_entry.get() if task != "": tasks\_listbox.insert(tk.END, task) task\_entry.delete(0, tk.END) else: messagebox.showwarning("Warning", "Task cannot be empty!")

def remove\_task(): try: selected\_task\_index = tasks\_listbox.curselection()[0] tasks\_listbox.delete(selected\_task\_index) except IndexError: messagebox.showwarning("Warning", "Please select a task to remove!")

def mark\_completed(): try: selected\_task\_index = tasks\_listbox.curselection()[0] task = tasks\_listbox.get(selected\_task\_index) tasks\_listbox.delete(selected\_task\_index) tasks\_listbox.insert(tk.END, f"✔ {task}") except IndexError: messagebox.showwarning("Warning", "Please select a task to mark as completed!")

GUI Setup

root = tk.Tk() root.title("Task Manager") root.geometry("400x400")

task\_entry = tk.Entry(root, width=40) task\_entry.pack(pady=10)

add\_button = tk.Button(root, text="Add Task", command=add\_task) add\_button.pack()

remove\_button = tk.Button(root, text="Remove Task", command=remove\_task) remove\_button.pack()

complete\_button = tk.Button(root, text="Mark Completed", command=mark\_completed) complete\_button.pack()

tasks\_listbox = tk.Listbox(root, width=50, height=15) tasks\_listbox.pack(pady=10)

root.mainloop()

****

**REFFERENCE**

**1.https://github.com/robot-mazeee/Task-Manager-in-Python/commit/029c86c800d96c4186c208073b0751717ebfc471.**

**2.** [**https://anvil.works/learn/examples/task-manager-app**](https://anvil.works/learn/examples/task-manager-app)**.**

**3.https://github.com/usman010803/Task\_Management\_System.**

**4.** [**https://medium.com/@mattrsmitty/python-programming-task-manager-2d25f4036112**](https://medium.com/@mattrsmitty/python-programming-task-manager-2d25f4036112)**.**

**5.** [**https://dev.to/lawaniej/a-simple-command-line-task-manager-in-python-4pba**](https://dev.to/lawaniej/a-simple-command-line-task-manager-in-python-4pba)**.**

**6.** [**https://www.codewithfaraz.com/python/17/create-a-task-management-system-in-python-with-tkinter**](https://www.codewithfaraz.com/python/17/create-a-task-management-system-in-python-with-tkinter)

7.https://www.irjmets.com/uploadedfiles/paper//issue\_9\_september\_2023/44485/final/fin\_irjmets1693932247.pdf.

**Conclusion**

**The Task Manager Application successfully provides an efficient platform to organize and manage daily tasks using Python. It simplifies task tracking, improves time management, and enhances productivity for users. This project also strengthened my understanding of Python programming and real-world application development.**

**The development of the Python-based Task Manager Application marks a significant step toward solving one of the most common problems faced by individuals and professionals alike—efficient time and task management. This project aimed to create a user-friendly application that allows users to organize their daily activities, prioritize tasks, and monitor their progress through a simple and effective interface.**

**Throughout the development process, Python served as the backbone of the application, thanks to its simplicity, readability, and vast ecosystem of libraries. Core concepts such as object-oriented programming, file handling, and GUI frameworks like Tkinter (or any other used framework) were effectively utilized to create a functional and interactive platform. This hands-on experience deepened my understanding of Python and improved my problem-solving skills by dealing with real-world challenges like data storage, task filtering, validation, and user interface design.**

**The project provides essential features such as adding new tasks, updating the status of existing ones, deleting completed or unwanted tasks, and setting deadlines. These functionalities offer a structured way for users to manage their responsibilities and reduce the chances of missing important tasks or deadlines. The simplicity of the interface ensures that even users with minimal technical knowledge can interact with the application without difficulty.**

**Additionally, this project introduced me to the importance of clean code, modular design, and user experience considerations. Designing intuitive workflows, handling user input, and organizing code into reusable components were vital parts of the development process. It also emphasized the role of continuous testing and debugging to ensure that the application runs smoothly under various scenarios.**

**Working on this project has been a valuable learning experience. It not only improved my programming skills but also enhanced my ability to manage a project from concept to deployment. I gained a deeper appreciation for software development principles, including planning, analysis, implementation, and testing phases.**

**Moving forward, the application can be expanded with additional features like user authentication, task notifications, cloud-based data storage, and mobile responsiveness. Integrating advanced technologies such as databases (SQLite, MySQL) or frameworks like Flask/Django could also help in transforming it into a web-based solution. These improvements can make the application even more practical and scalable for real-time use**

**In conclusion, the Task Manager Application demonstrates how Python can be effectively used to develop real-world software solutions. It fulfills its goal of helping users stay organized and productive, while also serving as a strong foundation for further learning and future enhancements. The journey from planning to execution has been insightful, reinforcing my passion for coding and inspiring me to take on more complex projects in the future.**

**Project Outcome and Benefits**

**The development of the Python-based Task Manager Application led to several key outcomes and benefits:**

**Improved Productivity: Users can effectively organize their tasks, prioritize important activities, and track progress, leading to better time management and efficiency.**

**User-Friendly Interface: The application provides a simple and intuitive interface, making it easy for users of all technical levels to manage their daily tasks.**

**Practical Application of Python Skills: The project allowed me to apply Python concepts such as object-oriented programming, file handling, and GUI development in a real-world scenario.**

**Enhanced Problem-Solving Abilities: Designing features, managing user inputs, and debugging issues improved my analytical and critical thinking skills.**

**Project Planning and Execution: This project enhanced my ability to manage a project from requirement gathering to implementation, testing, and final deployment.**

**Foundation for Future Enhancements: The application lays a strong foundation for future development, such as integrating databases, adding user login systems, and turning it into a web-based or mobile app.**

|  |  |
| --- | --- |
|  | |
|  | | | |
|  | | | |
|  | | | |
|  |
|  |