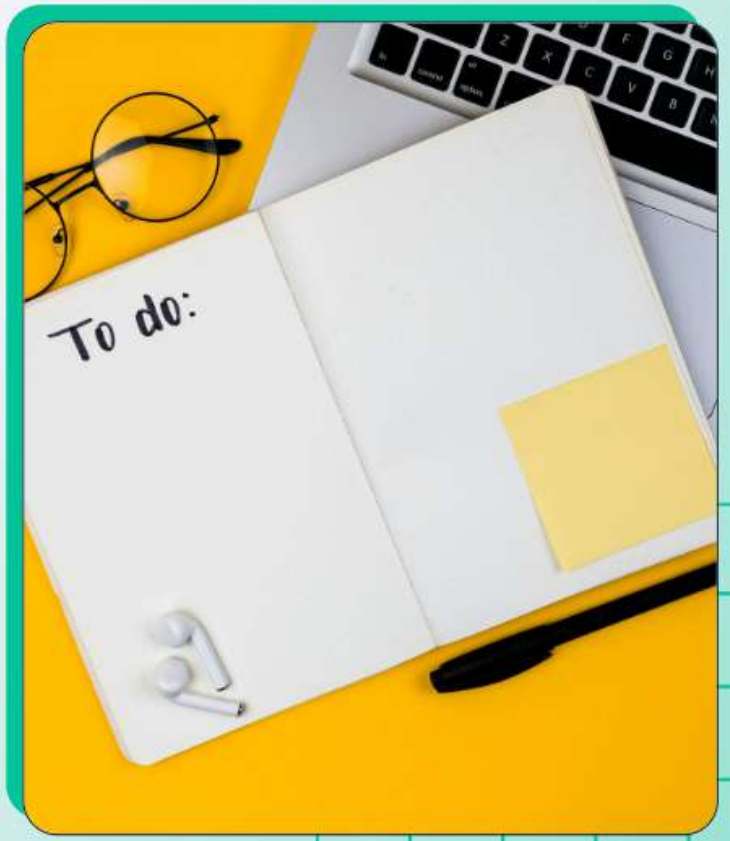
Introduction to To-Do List App

Learn to create a functional To-Do List App using Python features.The To-Do List App is a simple, user-friendly Python application designed to help users manage their tasks efficiently. It provides an intuitive interface for adding, deleting, viewing, and marking tasks as complete. By utilizing basic Python functions and data structures, this app demonstrates the power of programming in solving everyday problems.



Ujjwal Raj

Agenda

Exploring the Development of a Basic To-Do List App in Python

01

Project Overview

Understanding the purpose and scope of developing a basic To-Do List app using functions and data structures in Python.

02

App Functionality

Exploring the essential functions of the To-Do List app: adding tasks, deleting tasks, displaying task lists, and marking tasks as complete.

03

Code Implementation

Discussing the practical implementation of functions and data structures in Python to create the To-Do List app.

04

Code and Output

Reviewing the code snippets and their corresponding outputs to understand the app's functioning.

05

Advantages and Disadvantages

Analyzing the strengths and weaknesses of the To-Do List app in terms of functionality and user experience.

06

Future Scope

Exploring potential enhancements and future development possibilities for the To-Do List app to improve its features and usability.

07

Conclusion

Summarizing key findings, learnings, and outcomes from the development and analysis of the To-Do List app.

Python To-Do List Development

Project Overview

Building a Basic To-Do List App using Python

Understanding Python Functions and Data Structures

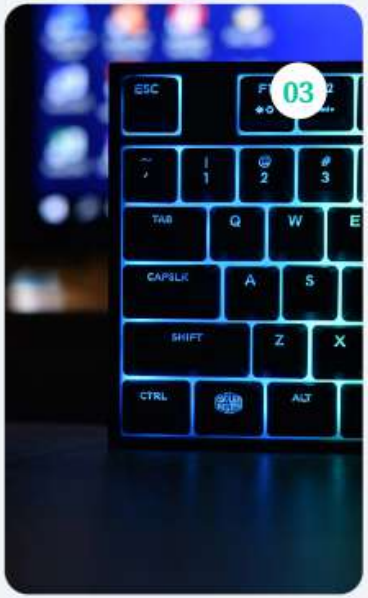
Learning how to implement essential Python functions and data structures for efficient task management.

Developing Core Functionalities

Creating features like adding tasks, deleting tasks, displaying task lists, and marking tasks as complete to enhance task management.

Ensuring Code Clarity

Emphasizing on clear and concise code structure to facilitate readability and maintainability of the To-Do List App.



App Functionality Insights

App Functionality

Key Functionalities of the To-Do List App

Add a Task

Users can easily input and add new tasks to their to-do list, enhancing organization and productivity.

Delete a Task

Allows users to efficiently remove any unnecessary or completed tasks, keeping the list up to date and clutter-free.

Display List of Tasks

Enables users to have a clear overview of all pending tasks, aiding in prioritization and time management.

Mark Task as Complete

Provides users with the satisfaction of ticking off completed tasks, giving a sense of accomplishment and progress.

Core Constructs

How the App Works

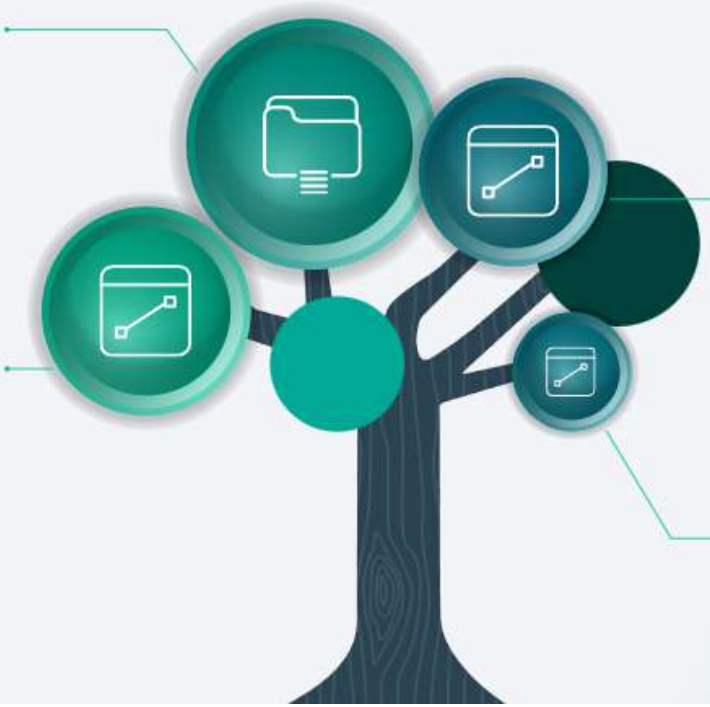
Understanding the Core Python Constructs Used in the App

Lists

Dynamically store tasks, allowing easy addition and removal of items.

Functions

Encapsulate operations for tasks, making the code modular and reusable.



Conditionals

Check task statuses to determine actions, enhancing interactivity.

Loops

Continuously handle user input and display tasks, ensuring smooth user experience.

```
Options:
1. Add a Task
2. Delete a Task
3. Display Tasks
4. Mark Task as Complete
5. Exit
Enter your choice (1-5): 1
Enter the task: TO brush my teeth
Task 'TO brush my teeth' added successfully.

Options:
1. Add a Task
2. Delete a Task
3. Display Tasks
4. Mark Task as Complete
5. Exit
Enter your choice (1-5): 3
To-Do List:
1. TO brush my teeth - ❌

Options:
1. Add a Task
2. Delete a Task
3. Display Tasks
4. Mark Task as Complete
5. Exit
Enter your choice (1-5): 4
To-Do List:
1. TO brush my teeth - ❌
Enter the task number to mark as complete: 1
Task 'TO brush my teeth' marked as complete.

Options:
1. Add a Task
2. Delete a Task
3. Display Tasks
4. Mark Task as Complete
5. Exit
Enter your choice (1-5): 5
Exiting the To-Do List App. Goodbye!
```

Task Management

Code

```
tasks = []
def add_task(task):
  task_item = {
    "task": task,
    "completed": False
  }
  tasks.append(task_item)
  print(f"Task '{task}' added successfully.")

def delete_task(index):
  if 0 = index len(tasks):
    removed_task = tasks.pop(index)
    print(f"Task '{removed_task['task']}' deleted successfully.")
  else:
    print("Invalid index. Please try again.")

def display_tasks():
  if not tasks:
    print("No tasks to display.")
  else:
    print("To-Do List:")
    for i, task in enumerate(tasks):
      status = "✔" if task['completed'] else "❌"
      print(f"{i + 1}. {task['task']} - {status}")

def mark_task_complete(index):
  if 0 = index len(tasks):
    tasks[index]['completed'] = True
    print(f"Task '{tasks[index]['task']}' marked as complete.")
  else:
    print("Invalid index. Please try again.")
```

```
# Main loop to interact with the user
def main():
  while True:
    print("\nOptions:")
    print("1. Add a Task")
    print("2. Delete a Task")
    print("3. Display Tasks")
    print("4. Mark Task as Complete")
    print("5. Exit")
    choice = input("Enter your choice (1-5): ")
    if choice == '1':
      task = input("Enter the task: ")
      add_task(task)
    elif choice == '2':
      display_tasks()
      index = int(input("Enter the task number to delete: ")) - 1
      delete_task(index)
    elif choice == '3':
      display_tasks()
    elif choice == '4':
      display_tasks()
      index = int(input("Enter the task number to mark as complete: ")) - 1
      mark_task_complete(index)
    elif choice == '5':
      print("Exiting the To-Do List App. Goodbye!")
      break
    else:
      print("Invalid choice. Please try again.")
# Run the main loop
if __name__ == "__main__":
  main()
```

Pros and Cons

Advantages and Disadvantages

Evaluating the Pros and Cons of a Basic To-Do List App

Limited functionality, may not meet advanced needs.

Not suitable for large-scale task management projects.
Basic UI, which is not user-friendly for all.

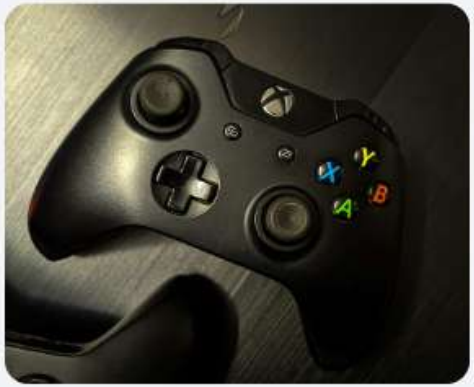
Simple and easy to understand, perfect for beginners.

Helps manage tasks efficiently with clear functions.
Great practice tool for beginner Python developers.

App Enhancements

Future Scope

Enhance the To-Do List App for Improved Functionality

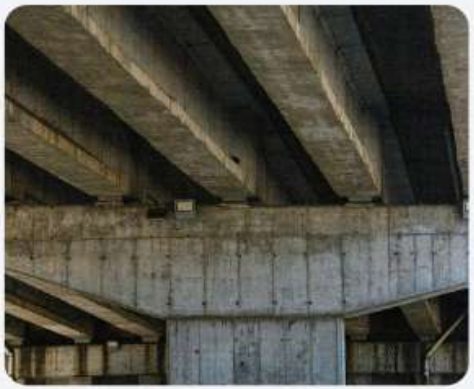


Add a Graphical User Interface (GUI)

Incorporating a GUI will enhance user experience by providing a visual way to interact with the app, making task management more intuitive.

Integrate with Databases for Persistent Storage

Connecting the app to databases will allow users to store their tasks securely and access them across different sessions, ensuring data persistence.



Implement Additional Features for Task Prioritization and Deadlines

Adding features like task prioritization and setting deadlines will help users organize their tasks more effectively and improve time

Enable User Authentication for Personalized Task Lists

Introducing user authentication will enable personalized task lists, ensuring that each user has a secure and customized experience based on their preferences.

Conclusion

The To-Do List App is a straightforward yet powerful example of how basic Python programming can be used to solve real-world problems. By integrating fundamental concepts such as lists, functions, and user input, this app provides an interactive way to manage tasks efficiently.

This project highlights the importance of programming in enhancing productivity and demonstrates the potential of even the simplest applications to make a significant impact on daily life. It serves as an excellent learning tool for beginners, illustrating how coding can translate ideas into functional, user-friendly applications.

Overall, the To-Do List App not only helps users stay organized but also encourages them to explore further possibilities in software development, making it a valuable stepping stone in the journey of learning to code.

Thank you