# Todo List Application - Python



## ▼ ជំហានទី១៖ Project Setup

> 💡 គោលដៅ៖ បង្កើត project structure និង basic file

1. បង្កើត folder ថ្មីមួយឈ្មោះថា todo_list
2. បង្កើត file ថ្មីមួយឈ្មោះ todo_app.py នៅក្នុង folder todo_list
3. បើក folder todo_list នៅក្នុង IDE

## ▼ ជំហានទី២៖ បង្កើត Class

> 💡 គោលដៅ៖ បង្កើតនៅ basic class structure

```
class TodoList:
    def __init__(self):
        self.tasks = []
        self.filename = "tasks.json"
```

តេស្តកូដខាងលើដោយសរសេរដូចខាងក្រោម

```
# Test the class
if __name__ == "__main__":
    todo = TodoList()
    print("TodoList initialized")
```

## ▼ ជំហ៊ានទី៣៖ Adding Tasks

> 💡 គោលដៅ៖ បង្កើត method ដើម្បីបញ្ចូលនូវ tasks

បន្ថែម method ខាងក្រោមទៅក្នុង class TodoList

```
def add_task(self, description):
    """Add a new task"""
    task = {
        'id': len(self.tasks) + 1,
        'description': description,
        'completed': False,
        'created_at': datetime.now().strftime("%Y-%m-%d %H:%M
    }
    self.tasks.append(task)
    print(f"Task '{description}' added successfully!")
```

តេស្តកូដខាងលើដោយសរសេរដូចខាងក្រោម

```
if __name__ == "__main__":
    todo = TodoList()
    todo.add_task("Learn Python")
    todo.add_task("Build a project")
    print(todo.tasks)
```

## ▼ ជំហ៊ានទី៤៖ Viewing Tasks

> 💡 គោលដៅ៖ បង្កើត method ដើម្បីបង្ហាញ tasks ដែលមានស្រាប់

បន្ថែម method ខាងក្រោមទៅក្នុង class TodoList

```
def view_tasks(self):
    """Display all tasks"""
    if not self.tasks:
        print("\nNo tasks found!")
        return

    print("\nYour To-Do List:")
    print("-" * 50)
    for task in self.tasks:
        status = "✓" if task['completed'] else " "
        print(f"{task['id']}. [{status}] {task['description']}")
    print("-" * 50)
```

តេស្តកូដខាងលើ

```
if __name__ == "__main__":
    todo = TodoList()
    todo.add_task("Learn Python")
    todo.add_task("Build a project")
    todo.view_tasks()
```

## ▼ ជំហានទី៥៖ Completing Tasks

💡 គោលដៅ៖ បង្កើត method ដើម្បីអាច កំណត់ថា task ណា complete

បន្ថែម method ខាងក្រោមទៅក្នុង class TodoList

```
def complete_task(self, task_id):
    """Mark a task as completed"""
    for task in self.tasks:
        if task['id'] == task_id:
            task['completed'] = True
            print(f"Task {task_id} marked as completed!")
```

```
                return
        print(f"Task with ID {task_id} not found!")
```

តេស្តកូដខាងលើ

```
if __name__ == "__main__":
    todo = TodoList()
    todo.add_task("Learn python")
    todo.add_task("Build a project")
    todo.view_tasks()
    todo.complete_task(1)
    todo.view_tasks()
```

## ▼ ជំហានទី៦៖ Deleting Tasks

💡 គោលដៅ៖ បង្កើត method ដើម្បីអាចលុប task

បន្ថែម method ខាងក្រោមទៅក្នុង class TodoList

```
def delete_task(self, task_id):
    """Delete a task"""
    for task in self.tasks:
        if task['id'] == task_id:
            self.tasks.remove(task)
            print(f"Task {task_id} deleted!")
            return
    print(f"Task with ID {task_id} not found!")
```

តេស្តកូដខាងលើ

```
if __name__ == "__main__":
    todo = TodoList()
    todo.add_task("Learn python")
    todo.add_task("Build a project")
```

```
    todo.view_tasks()
    todo.delete_task(2) # Delete the second task
    todo.view_tasks() # View updated tasks
```

## ▼ ជំហានទី៧៖ Saving & Loading Tasks

> 💡 គោលដៅ៖ រក្សាទុក tasks ទាំងអស់នៅក្នុង file

បន្ថែម method ខាងក្រោមទៅក្នុង class TodoList

```
def save_tasks(self):
    """Save tasks to file"""
    with open(self.filename, 'w') as file:
        json.dump(self.tasks, file, indent=2)

def load_tasks(self):
    """Load tasks from file it it exists"""
    try:
        with open(self.filename, 'r') as file:
            self.tasks = json.load(file)
    except FileNotFoundError:
        self.tasks = []
```

បន្ថែម method load_tasks នៅពេលដែល program ចាប់ផ្តើមដំណើរការ

```
def __init__(self):
    self.tasks = []
    self.filename = "tasks.json"
    self.load_tasks()
```

Update method ទាំងឡាយនៅក្នុង TodoList class

```
# Add at the end of add_task method:
    self.save_tasks()
```

```
# Add at the end of complete_task method (inside the if block
    self.save_tasks()

# Add at the end of delete_task method (inside he if block)
    self.save_tasks()
```

## ▼ ជំហានទី៨៖ បង្កើតមីនុយ

💡 គោលដៅ៖ បង្កើតមីនុយដើម្បីអោយអ្នកប្រើប្រាស់អាចជ្រើសរើស option ណាមួយតាមចិត្តចង់

```
def main():
    todo = TodoList()

    while True:
        print("\n=== To-Do List Application ===")
        print("1. Add Task")
        print("2. View Tasks")
        print("3. Complete Task")
        print("4. Delete Task")
        print("5. Exit")

        choice = input("\nEnter your choice (1-5): ")

        if choice == '1':
            description = input("Enter task description: ")
            todo.add_task(description)

        elif choice == '2':
            todo.view_tasks()

        elif choice == '3':
```

```
            todo.view_tasks()
            try:
                task_id = int(input("Enter task ID to mark as
                todo.complete_task(task_id)
            except ValueError:
                print("Please enter a valid task ID!")

        elif choice == '4':
            todo_view_tasks()
            try:
                task_id = int(input("Enter task ID to delete:
                todo.delete_task(task_id)
            except ValueError:
                print("Please enter a valid task ID!")

        elif choice == '5':
            print("Thank you for using the To-Do List applica
            break

        else:
            print("Invalid choice! Please try again.")
```

```
if __name__ == "__main__":
    main()
```

## ▼ ជំហានទី៩៖ Testing

> 💡 គោលដៅ៖ ធ្វើការតេស្តទៅលើកម្មវិធីរបស់យើងដើម្បីអោយប្រាកដចិត្តថាគ្មានបញ្ហាណាមួយកើតឡើង

ចាប់ផ្តើមដំណើរការកូដ និងធ្វើការតេស្តទៅលើ feature ដូចខាងក្រោម៖

- Add task ចំនួន ៣
- View the task list

- Complete task មួយចំនួន
- Delete task ណាមួយ
- បិទកម្មវិធី និងបើកសារឡើងវិញ ដើម្បីធានាបានថា data នៅតែមាននៅក្នុង file