COGNIFYZ TECHNOLOGIES

Software Development Intern Tasks List 3

**Task 3: Create a console application for basic CRUD operations on a list of tasks.**

**Objective: Implement Create, Read, Update, and Delete operations using arrays or lists for data storage.**

**Steps:**

1.Define a Task class with necessary attributes.

2.Implement functionality to create a new task.

3.Develop a method to read and display tasks.

4.Allow users to update task details.

5.Provide an option to delete tasks.

6.Test the application with various scenarios.

CODE

class Task:

def \_init\_(self, task\_id, description, status="Pending"):

"""Initialize a task with an ID, description, and status."""

self.task\_id = task\_id

self.description = description

self.status = status

def \_str\_(self):

"""String representation of a Task object."""

return f"ID: {self.task\_id} | Description: {self.description} | Status: {self.status}"

class TaskManager:

def \_init\_(self):

"""Initialize TaskManager with an empty task list and a counter for IDs."""

self.tasks = []

self.next\_id = 1

def create\_task(self, description):

"""Create and add a new task to the task list."""

new\_task = Task(self.next\_id, description)

self.tasks.append(new\_task)

self.next\_id += 1

print("Task added successfully!")

def read\_tasks(self):

"""Display all tasks in the task list."""

if not self.tasks:

print("No tasks available.")

else:

print("\nTask List:")

for task in self.tasks:

print(task)

def update\_task(self, task\_id, new\_description, new\_status):

"""Update the description and status of a task by its ID."""

for task in self.tasks:

if task.task\_id == task\_id:

task.description = new\_description

task.status = new\_status

print("Task updated successfully!")

return

print("Task ID not found.")

def delete\_task(self, task\_id):

"""Delete a task from the task list by its ID."""

for task in self.tasks:

if task.task\_id == task\_id:

self.tasks.remove(task)

print("Task deleted successfully!")

return

print("Task ID not found.")

def menu(self):

"""Display a menu for task management and handle user input."""

while True:

print("\nTask Manager")

print("1. Create Task")

print("2. View All Tasks")

print("3. Update Task")

print("4. Delete Task")

print("5. Exit")

choice = input("Choose an option (1-5): ")

if choice == "1":

description = input("Enter the task description: ")

self.create\_task(description)

elif choice == "2":

self.read\_tasks()

elif choice == "3":

try:

task\_id = int(input("Enter the Task ID to update: "))

new\_description = input("Enter the new task description: ")

new\_status = input("Enter the new status (e.g., Pending/Completed): ")

self.update\_task(task\_id, new\_description, new\_status)

except ValueError:

print("Invalid input. Please enter a valid Task ID.")

elif choice == "4":

try:

task\_id = int(input("Enter the Task ID to delete: "))

self.delete\_task(task\_id)

except ValueError:

print("Invalid input. Please enter a valid Task ID.")

elif choice == "5":

print("Exiting Task Manager. Goodbye!")

break

else:

print("Invalid choice. Please choose a valid option.")

# Main program to run the Task Manager

if \_name\_ == "\_main\_":

try:

task\_manager = TaskManager() # TaskManager instance

task\_manager.menu() # Call the menu method

except Exception as e:

print(f"An error occurred: {e}")







