ING SOFTWARE

Workshop: Acceptance Testing

Juan Carlos Gallo Munoz

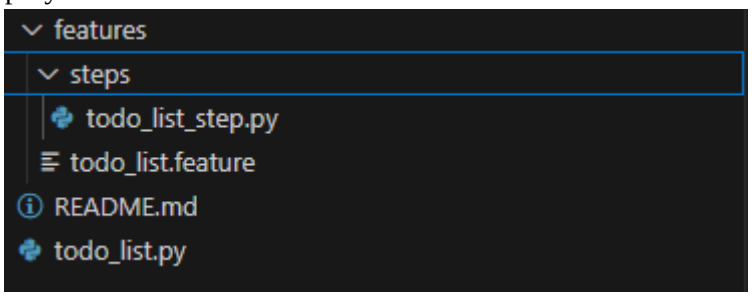
Introducción

Este taller tuvo como objetivo aplicar pruebas de aceptación usando la librería Behave con la metodología BDD, validando el comportamiento del sistema To-Do List Manager.

Parte 1: Configuración y desarrollo del código

Se configuró el entorno Python, se instaló la librería `behave`, se implementó la lógica de una lista de tareas y se añadieron funcionalidades básicas y adicionales.

- Estructura del proyecto



- todo_list.py

```python
class TodoList:
    def __init__(self):
        self.tasks = []

    def add_task(self, description):
        self.tasks.append(Task(description))

    def list_tasks(self):
        return [task.description for task in self.tasks]

    def list_all(self):
        return self.tasks

    def mark_completed_by_index(self, index):
        if 0 <= index < len(self.tasks):
            self.tasks[index].completed = True
        else:
            raise IndexError("Índice inválido.")

    def clear_tasks(self):
        self.tasks = []

    def remove_task_by_index(self, index):
        if 0 <= index < len(self.tasks):
            del self.tasks[index]
        else:
            raise IndexError("Índice inválido.")

    def count_tasks(self):
        return len(self.tasks)
```

Parte 2: Pruebas y ejecución

Se crearon 6 escenarios de prueba usando Gherkin en un archivo `.feature` y se implementaron los pasos correspondientes en Python. Todos los tests fueron ejecutados exitosamente.

- .feature

```
Feature: To-Do List Manager Functionality

  Scenario: Add a task to the to-do list
    Given the to-do list is empty
    When the user adds a task "Buy groceries"
    Then the to-do list should contain "Buy groceries"

  Scenario: List all tasks in the to-do list
    Given the to-do list contains tasks:
      | Task        |
      | Buy milk    |
      | Pay bills   |
    When the user lists all tasks
    Then the output should include:
      | Task        |
      | Buy milk    |
      | Pay bills   |

  Scenario: Mark a task as completed
    Given the to-do list contains a task "Study" with status "Pending"
    When the user marks task "Study" as completed
    Then the task "Study" should be marked as completed

  Scenario: Clear the entire to-do list
    Given the to-do list contains tasks:
      | Task        |
      | Exercise    |
      | Sleep early |
    When the user clears the to-do list
    Then the to-do list should be empty

  Scenario: Remove a task from the to-do list
```

- Fragmento de todo_list_steps.py

```python
from behave import given, when, then
from todo_list import TodoList
import traceback

# === Setup y estado inicial ===

@given('the to-do list is empty')
def step_given_list_empty(context):
    context.todo = TodoList()

@given('the to-do list contains tasks')
def step_given_list_has_tasks(context):
    context.todo = TodoList()
    for row in context.table:
        context.todo.add_task(row['Task'])

@given('the to-do list contains a task "{task}" with status "{status}"')
def step_given_task_with_status(context, task, status):
    context.todo = TodoList()
    context.todo.add_task(task)
    if status.lower() == "completed":
        context.todo.mark_completed(task)

# === Acciones del usuario ===

@when('the user adds a task "{task}"')
def step_when_add_task(context, task):
    context.todo.add_task(task)

@when('the user lists all tasks')
def step_when_list_tasks(context):
    context.result = context.todo.list_tasks()
```

- Salida de la consola con 6 escenarios y 18 pasos exitosos

```
1 feature passed, 0 failed, 0 skipped
6 scenarios passed, 0 failed, 0 skipped
18 steps passed, 0 failed, 0 skipped, 0 undefined
```

Funcionamiento del to do list

```
--- GESTOR DE TAREAS ---
1. Agregar tarea
2. Mostrar tareas
3. Marcar tarea como terminada
4. Eliminar tarea
5. Borrar todas las tareas
6. Salir
Seleccione una opción: 2
Tareas actuales:
[0] terminar exam [Pendiente]
[1] Juanito [Pendiente]
[2] Gyan [Pendiente]

--- GESTOR DE TAREAS ---
1. Agregar tarea
2. Mostrar tareas
3. Marcar tarea como terminada
4. Eliminar tarea
5. Borrar todas las tareas
6. Salir
Seleccione una opción: 3
Índice de la tarea a marcar como terminada: 2
Tarea marcada como terminada.

--- GESTOR DE TAREAS ---
1. Agregar tarea
2. Mostrar tareas
3. Marcar tarea como terminada
4. Eliminar tarea
5. Borrar todas las tareas
6. Salir
Seleccione una opción: 2
Tareas actuales:
```

Conclusiones y recomendaciones

- La metodología BDD facilita validar funcionalidades desde el punto de vista del usuario.

- Las pruebas de aceptación permiten asegurar que los requisitos se cumplen.

- Se recomienda extender la lógica para interfaz gráfica o almacenamiento persistente.