Gherkin and Behave

Read the Behaviour Driven Development (2020) pages and then use the Gherkin language to create a Gherkin sequence that addresses ONE of the following examples:

- Using a new coffee making machine.
- Interfacing with a new SatNav system.
- Using a computer running the Linux operating system.
- Getting familiar with a new vehicle.
- Creating a batch or shell script.

Your response should consist of at least three scenarios describing different roles such as administrator, user, driver and so on.

To understand how Gherkin and Behave support Behaviour Driven Development (BDD), I reviewed key documentation including the Behave documentation and Gherkin syntax, as well as beginner video tutorials and walkthroughs.

I found that there was a set layout

```
Scenario:
        Given
        When
        Then
```

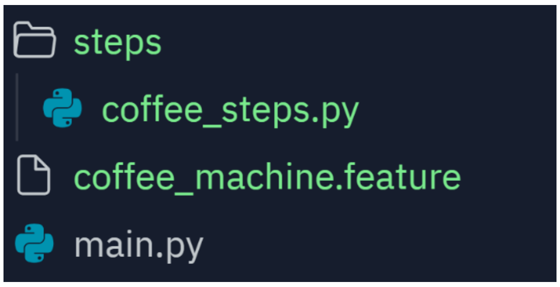
Gherkin

Coffee Machine

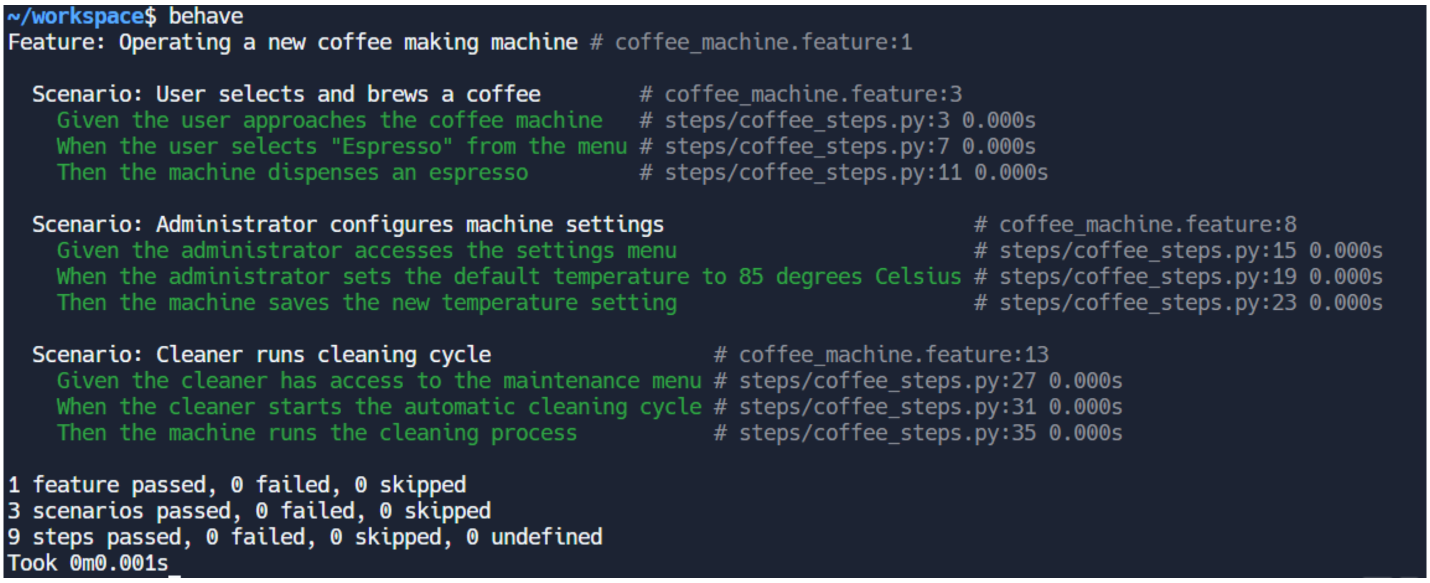
```
Feature: Operating a new coffee making machine

  Scenario: User selects and brews a coffee
    Given the user approaches the coffee machine
    When the user selects "Espresso" from the menu
    Then the machine dispenses an espresso

  Scenario: Administrator configures machine settings
    Given the administrator accesses the settings menu
    When the administrator sets the default temperature to 85 degrees Celsius
    Then the machine saves the new temperature setting

  Scenario: Cleaner runs cleaning cycle
    Given the cleaner has access to the maintenance menu
    When the cleaner starts the automatic cleaning cycle
    Then the machine runs the cleaning process
```

Behave

Feature file and Steps file. When using behave with replit the steps file needs to be in the steps folder.



```python
from behave import given, when, then

@given('the user approaches the coffee machine')
def step_user_approaches(context):
               context.actor = "user"

@when('the user selects "Espresso" from the menu')
def step_user_selects_espresso(context):
               context.selection = "Espresso"

@then('the machine dispenses an espresso')
def step_machine_dispenses(context):
               assert context.selection == "Espresso"

@given('the administrator accesses the settings menu')
def step_admin_accesses_settings(context):
               context.actor = "administrator"

@when('the administrator sets the default temperature to 85 degrees Celsius')
def step_set_temperature(context):
               context.temperature = 85

@then('the machine saves the new temperature setting')
def step_confirm_temperature(context):
               assert context.temperature == 85

@given('the cleaner has access to the maintenance menu')
def step_cleaner_accesses_maintenance(context):
               context.actor = "cleaner"

@when('the cleaner starts the automatic cleaning cycle')
def step_start_cleaning(context):
               context.cleaning_started = True

@then('the machine runs the cleaning process')
def step_confirm_cleaning(context):
               assert context.cleaning_started is True
```



```
~/workspace$ behave
Feature: Operating a new coffee making machine # coffee_machine.feature:1

  Scenario: User selects and brews a coffee        # coffee_machine.feature:3
    Given the user approaches the coffee machine   # steps/coffee_steps.py:3 0.000s
    When the user selects "Espresso" from the menu # steps/coffee_steps.py:7 0.000s
    Then the machine dispenses an espresso         # steps/coffee_steps.py:11 0.000s

  Scenario: Administrator configures machine settings                            # coffee_machine.feature:8
    Given the administrator accesses the settings menu                           # steps/coffee_steps.py:15 0.000s
    When the administrator sets the default temperature to 85 degrees Celsius # steps/coffee_steps.py:19 0.000s
    Then the machine saves the new temperature setting                           # steps/coffee_steps.py:23 0.000s

  Scenario: Cleaner runs cleaning cycle                   # coffee_machine.feature:13
    Given the cleaner has access to the maintenance menu # steps/coffee_steps.py:27 0.000s
    When the cleaner starts the automatic cleaning cycle # steps/coffee_steps.py:31 0.000s
    Then the machine runs the cleaning process           # steps/coffee_steps.py:35 0.000s

1 feature passed, 0 failed, 0 skipped
3 scenarios passed, 0 failed, 0 skipped
9 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.001s
```

Behave (2020) *The Gherkin language – Behave documentation*. Available at: https://behave.readthedocs.io/en/stable/philosophy.html#the-gherkin-language (Accessed: 7 May 2025).

TutorialsPoint (2020) *Behave - Introduction*. Available at: https://www.tutorialspoint.com/behave/behave_introduction.htm (Accessed: 7 May 2025).

YouTube (2020) *Introduction to Behave - BDD in Python*. Available at: https://www.youtube.com/watch?v=zYXUefMfTAM (Accessed: 7 May 2025).