**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**

**FACULTY OF ELECTRICAL AND COMPUTER ENGINEERING**

**SOFTWARE ENGINEERING II**

**ACCEPTANCE TEST WORKSHOP - I TERM 2021**

**Objectives:**

* Validate the correct operation of a system through acceptance tests.

**Requirements**

* Python >= 3.7.3
* Behave for Python >= 1.2.6

**Introduction**

BDD is a way for software teams to work that closes the gap between businesspeople and technical people by:

* Encouraging collaboration across roles to build shared understanding of the problem to be solved.
* Working in rapid, small iterations to increase feedback and the flow of value.
* Producing system documentation that is automatically checked against the system’s behavior [1].

ACCEPTANCE TESTING is a level of software testing where a system is tested for acceptability. It is formal testing with respect to user needs, requirements, and business processes conducted to determine whether a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system, and if it is acceptable for delivery [2].

The framework to write the acceptance tests is Cucumber, what use “Gherkin” language. The 10 key words of Gherkin are: [3]:

* Feature
* Scenario
* Given
* When
* Then
* And
* But
* Scenario Outline
* Examples
* Background

Let us review the 10 key words.

* **Given:** This puts the system in a known state. It’s a set of key pre-conditions for a scenario (e.g. user has logged in, user has money in their account etc)
* **When:** This is the key action, a user will take. It’s the action that leads to an outcome
* **Then:** This is the observable outcome. It’s what happens after the user makes that action
* **Scenario:** This is used to describe the scenario & give it a title. The reason we do this is because a feature or user story will likely have multiple scenarios.
* **And:** This is used when a scenario is more complicated. It can be used in association with Given, When, or Then.
* **But:** Can be used in association with Then. It’s used to say something shouldn’t happen as an outcome.
* **Feature:** is used to give a title for the feature/piece of functionality. A feature contains lots of scenarios. For example, “Sign in” might be a feature … or “push alerts” …. it’s the title of a piece of functionality.
* **Scenario Outline / Examples:** These are used together. They are used to combine a set of similar scenarios.
* **Background:** This sets the context for all scenarios below it. If you find that scenarios have common Given/Ands, Background can be used to eliminate the repetition. Background is run before each of your scenarios.

For More information about the key words check [4].

**Workshop activities**

This workshop contains three steps: 1) Prepare the acceptance test project, 2) review examples, and 3) develop acceptance tests.

To develop these activities, we will use a catalog of video games for different consoles. In this catalog, searches must be made with various filters. Each video game has the following information: name, year of publication, manufacturer, and its rating (E: Everyone, T: Teen, M: Mature)

**Part 1: Prepare the project**

|  |  |
| --- | --- |
| Diseño web | pip install behave (o pip3) |

1. Install la Behave library with this command in cmd

|  |  |
| --- | --- |
| Diseño web | git clone [https://github.com/leortyz/AcceptanceTesting\_1.git](https://github.com/leortyz/AcceptanceTesting_2.git)  git clone <https://github.com/leortyz/AcceptanceTesting_2.git> |

1. Clone el repository with the base project.

|  |  |
| --- | --- |
| Diseño web | python main.py |

1. Verify that there are no errors in the code.

**Part 2: Review examples and learn from them.**

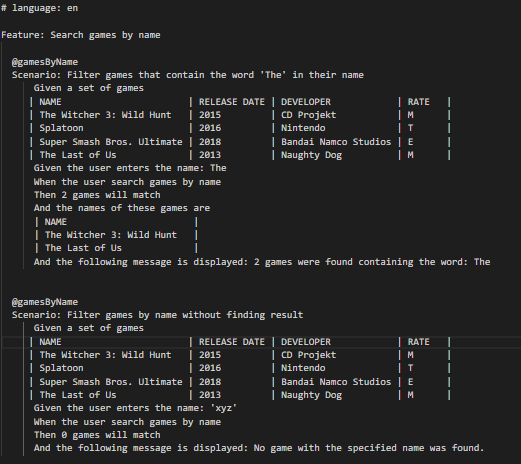
The system requirement is as follows:

As a user I want to query the available video games by entering their name (or a portion of it) to find any matches.

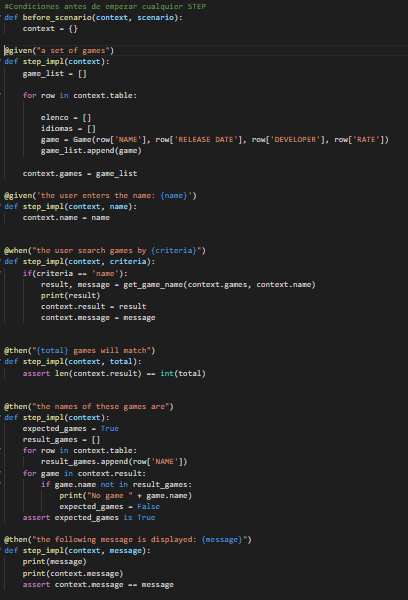
[CA] GIVEN the user enters a video game name (or part of it) WHEN they select the “search by name” option, THEN the name of all matching video games will be displayed.

[CA] GIVEN the user enters a video game name (or part of it) WHEN they select the “search by name” option, but there is no match with the video game list, THEN the following message will be displayed: No search found video game that matches the name entered.

1. Review the file *features/filter\_game\_name.feature.*

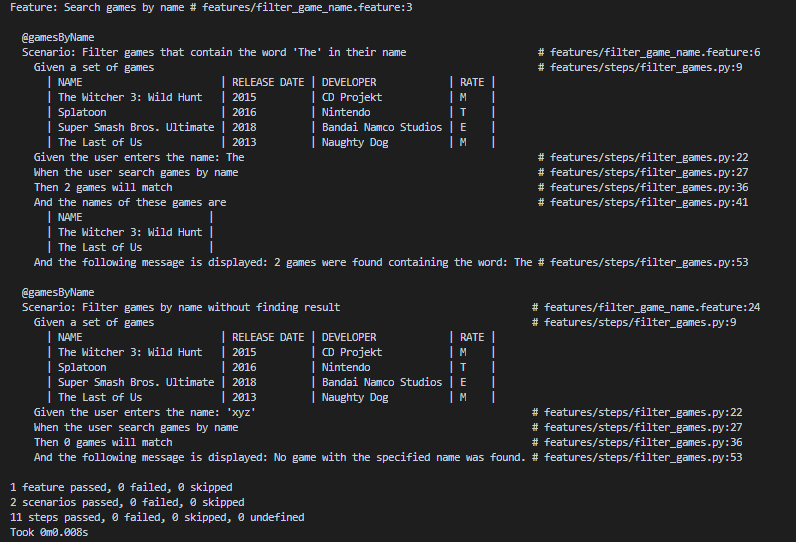


1. Review the file *features/steps/filter\_games.py.*



1. Run the acceptance test and analyze the result.

|  |  |
| --- | --- |
| Diseño web | behave |



**Part 3. Develop acceptance tests.**

For this activity, you must perform acceptance tests for two system requirements described below. The deliverables of this workshop are detailed on the next page.

1. As a user I want to view a list of the games by rating, selecting one or more categories [E, T, M] to find games by type of audience.
   * 1. [CA] GIVEN the user selects one or more ratings WHEN choosing the "search by ratings" option, THEN the name of all the games that correspond to these categories will be displayed.
     2. ii) [CA] GIVEN the user selects one or more ratings WHEN choosing the "search by ratings" option, but there is no match with the list of games, THEN the following message will be displayed: No game with the specified rating was found.
2. As a user I want to get the list of games by manufacturer to find all the games that have been developed by a certain video game studio
   * 1. [CA] GIVEN the user enters a study WHEN they select the "search by study" option, THEN the name of all the games that have been developed by said study will be displayed.
     2. [CA] GIVEN the user enters a manufacturer WHEN he selects the "search by study" option, but there is no match with the list of games, THEN the following message will be displayed: No game developed by {study} was found

**Deliverables**

1. Develop the acceptance tests around the 4 requirements. Upload the tests on a branch of the repositories.

<https://github.com/leortyz/AcceptanceTesting_1.git>

<https://github.com/leortyz/AcceptanceTesting_2.git>

1. Use Cucumber and document the tests. Name the report with the name of each student.

**Students per group**: 2

**Rubric**

|  |  |
| --- | --- |
| **Descripción** | **Valor** |
| Test 1 | 15 |
| Test 2 | 15 |
| Test 3 | 15 |
| Test 4 | 15 |
| Report | 40 |
| Penalty per hour or fraction of delay. | -30 |
| Penalty for not uploading required deliverables as specified | -30 |

# **References**

|  |  |
| --- | --- |
| [1] | https://cucumber.io/docs/bdd/. |
| [2] | Software Testing Fundamentals, «Acceptance Testing,» [En línea]. Available: http://softwaretestingfundamentals.com/acceptance-testing/. |
| [3] | R. Hewitt, «Gherkin for Business Analysts,» Modern analyst, 2017. [En línea]. Available: https://www.modernanalyst.com/Resources/Articles/tabid/115/ID/3810/Gherkin-for-Business-Analysts.aspx. |
| [4] | «Gherkin Reference,» [En línea]. Available: https://cucumber.io/docs/gherkin/reference/..  S. Vergara, «¿Qué es BDD (Behavior Driven Development)?,» ITDO, 18 Julio 2019. [En línea]. Available: https://www.itdo.com/blog/que-es-bdd-behavior-driven-development/. |