GE2 - Part 1
Statement

Collective Ownership of Code and Code Regulations

Software Development

February 2024

Degree in Computer Engineering - Double Degree in Computer Science and ADE

TABLA DE CONTENIDO

OBJECTIVES

The purpose of this guided exercise is to:

- Learn the best practices of collective code ownership that will be necessary to apply during the development of a project following the principles of agile methodologies.
- Establish a coding standard, accepted and implemented by the entire team, to ensure that all programmers in the project are working in a coordinated and similar manner.
- Apply the defined code standard using a tool on top of the integrated development environment (IDE).

DESCRIPTION OF THE COMPONENTS TO BE DEVELOPED

UC3MTravel is a company that specializes in the development of computer technologies for travel management business. It is currently developing a system for managing hotel reservations and stays. One of the requirements is to handle credit card numbers with which reservations are made.

To facilitate the management of reservations, it has been decided to implement a component that is capable of validating a credit card number that has been received. Usually VISA, Maestro and Mastercard credit cards have 16 digits, with the sixteenth being a control digit. This control digit is checked using the Luhn algorithm.

https://en.wikipedia.org/wiki/Luhn_algorithm

To facilitate this management of reservations, it has been decided to implement a software to generate a reservation number as soon as the travelers are registered. Once the traveler arrives at the hotel, the component will generate a stay code that will be used during the stay and to finally check the traveler out of the hotel.

The software engineers at UC3MTravel started the task of implementing this component, but given the difficulties of the component, they decided to launch a development contract to a group of independent software engineers (you). The base code is available in Aula Global.

This component will be developed in Python, creating a virtual environment for the project with the PyCharm IDE and the code will be delivered in a shared software management system with version control that will be GitHub.

TASKS TO BE CARRIED OUT IN THE EXERCISE

The steps to follow in the exercise are:

1. Configure the PyCharm environment with the necessary components to connect to GitHub, and to review the code with the PyLint plugin.
2. Create a private repository on GitHub for this exercise, following the steps outlined in the configuration addendum available in Aula Global. The repository must be created by one team member and must be shared with the other two team members and the teacher.
3. Create a project in PyCharm, set up the virtual environment, and include the code provided in Aula Global.
4. Include a new functionality to the software that, given a 16-digit card number, checks if it is correct taking into account that position 16 must comply with Luhn's Algorithm. You'll need to include two examples, one with valid code and one with invalid code so you can verify that the implementation is correct.
   More help on validating 16-digit credit card numbers on:

   https://en.wikipedia.org/wiki/Payment_card_number

5. The code must follow the PEP-8 standard, but the group will need to modify 10-15 rules and document them. These rules will be described in a PDF document. A good layout and presentation of this document will be taken into account. This adapted regulation will be used in the second part of the exercise (GE2.2).
6. Configure the PyLint plugin to recognize defined rules. In the PyLint configuration file, the rules that have been modified will be identified by comments.
7. Remove all warnings and errors that PyLint indicates for the code.
8. Share the project via GitHub with team members. All three team members need to have downloaded the repository, set it up, made edits, and pushed to the repository in a balanced way. Although this step is put at the end, the interaction with GitHub should be continuous, that is, as steps are taken in practice, the project should be pushed to the repository. If the history shows that one of the team members has not made any code submissions, or that the submission consists of a minor change (e.g., deleting a line or adding a comment), they can be penalized.
9. Share the project via Github with the teacher. It is important to follow the exact naming of the project name indicated in the Annex so that the teacher can find your exercise when evaluating it.

The Configuration Annex describes in detail how to perform the necessary configurations to go through the guided practice. In many cases the configuration we indicate is very similar to the one we asked for, but in some cases names of projects have been used as examples. Keep in mind that you must adjust the steps for the needs of your guided practice.

RULES FOR THE DELIVERY OF GUIDED EXERCISES

This Guided Exercise will be conducted in **groups of 3 people**. Students who cannot find a partner or who cannot attend the sessions should contact the teachers of the subject to inform them of their situation.

The delivery will consist of the following parts:

1) Collective ownership of code.

   To successfully deliver, the private GitHub repository created by each group must include the **UC3MTravel** solution with the code that correctly implements the restrictions of the regulations defined by the students. This repository will be shared with the corresponding teacher.

2) Code Regulations.

   The code rules will be provided in a PDF document called "**Coding_Regulation.pdf**". In each section of the regulations, the PyLint rules that have been modified or added shall be included in order to verify the restrictions that are specified. This document must be included in the root directory of the software project corresponding to the requested module, for each group of practices.

   Likewise, this regulation must be implemented in Pylint for PyCharm by leaving the .pylintrc file with this regulation in the root directory**.** In this file, the following comments will be marked with comments:

   #MODIFIED RULE

   #< ORIGINAL RULE>

<MODIFIED RULE>

You must collect a screenshot with the result of passing the PyLint scan to the code before modifying it, where the errors are displayed, and another screenshot after, where the errors have disappeared after making the proper changes to the code to adjust it to the new regulations. These screenshots will also be included in the root of the project.

The source code found in the GitHub repository must comply with the code regulations established in those files.

ASSESSMENT

This exercise is part of Guided Exercise 2 that will be delivered in week 9 of class according to the course schedule. In this sense, the following criteria will be taken into account to evaluate the use of the GitHub environment and the code regulations:

- The project is well-created and well-configured on Github. It clones smoothly.
- The documents are uploaded to the project as indicated by the defined rules.
- All participants have made submissions to the Github repository and these make sense.
- The implemented PyLint regulations counts with a formal document, with a professional, well-structured format, that explains the regulations to people who have to work with them (describe each rule verbatim, include examples...). The document must include the PyLint rules configured in the corresponding sections. It will be appreciated to include additional code to be able to apply all the rules that have been defined.
- The number and rules chosen are appropriate and varied, showing the power of a code standard and its possibilities.
- In the .pylintrc file, the rules that have been modified or added are appropriately

marked using comments.

- The code has been implemented and works properly. Remember to include two examples, one with a valid code and one with an invalid code to prove that your program is working correctly.
- All the code complies with the regulations defined by the group and there are no errors or warnings.