**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL FACULTY OF ELECTRICAL AND COMPUTER ENGINEERING SOFTWARE ENGINEERING II**

**CODING STANDARDS - 2024**

**Xavier Cobos**

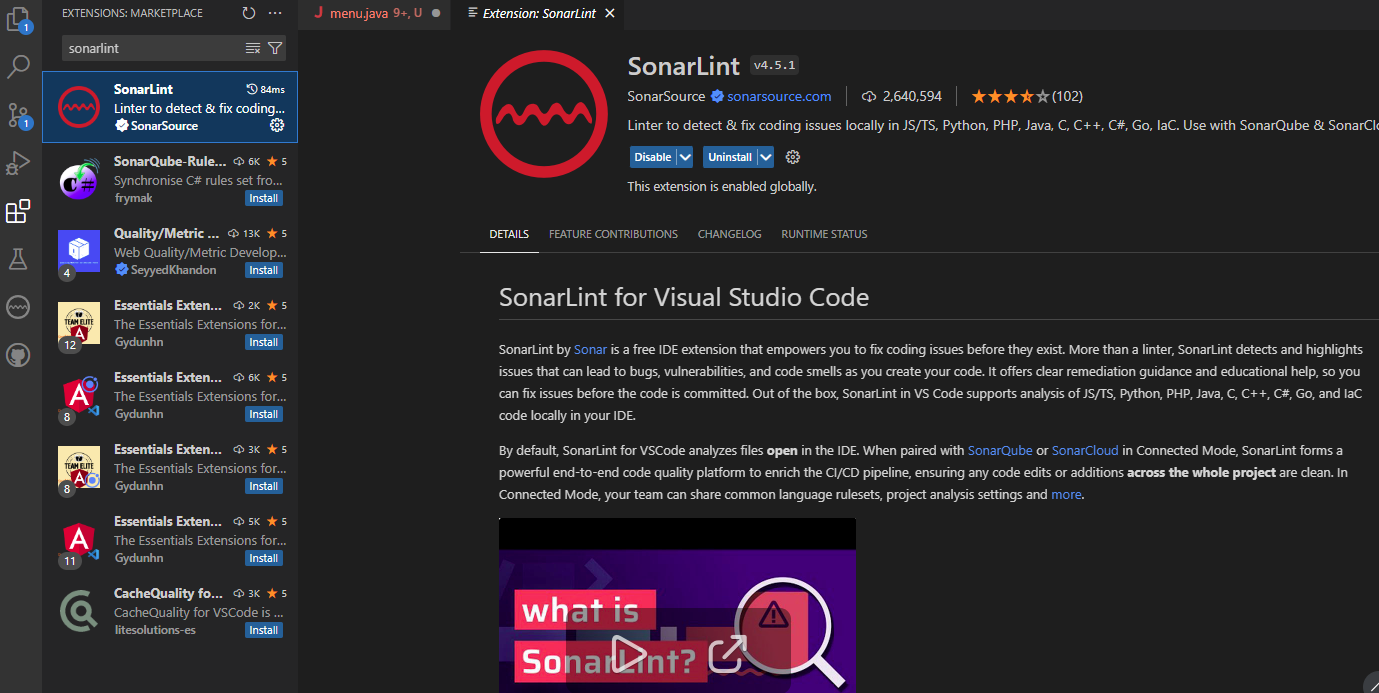
*Python*

# Activity

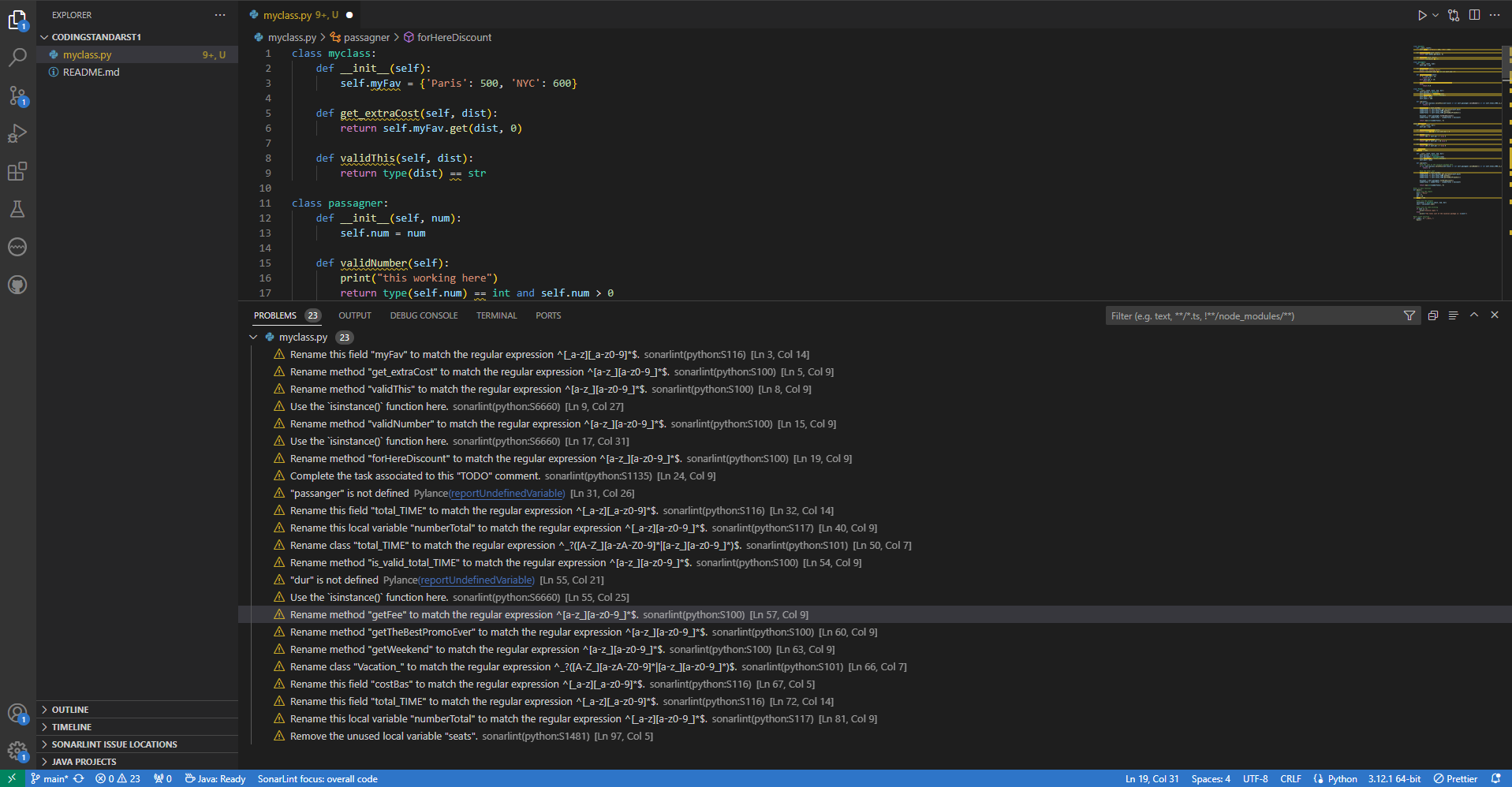
**Step 1 - Choose the Programming Language:**

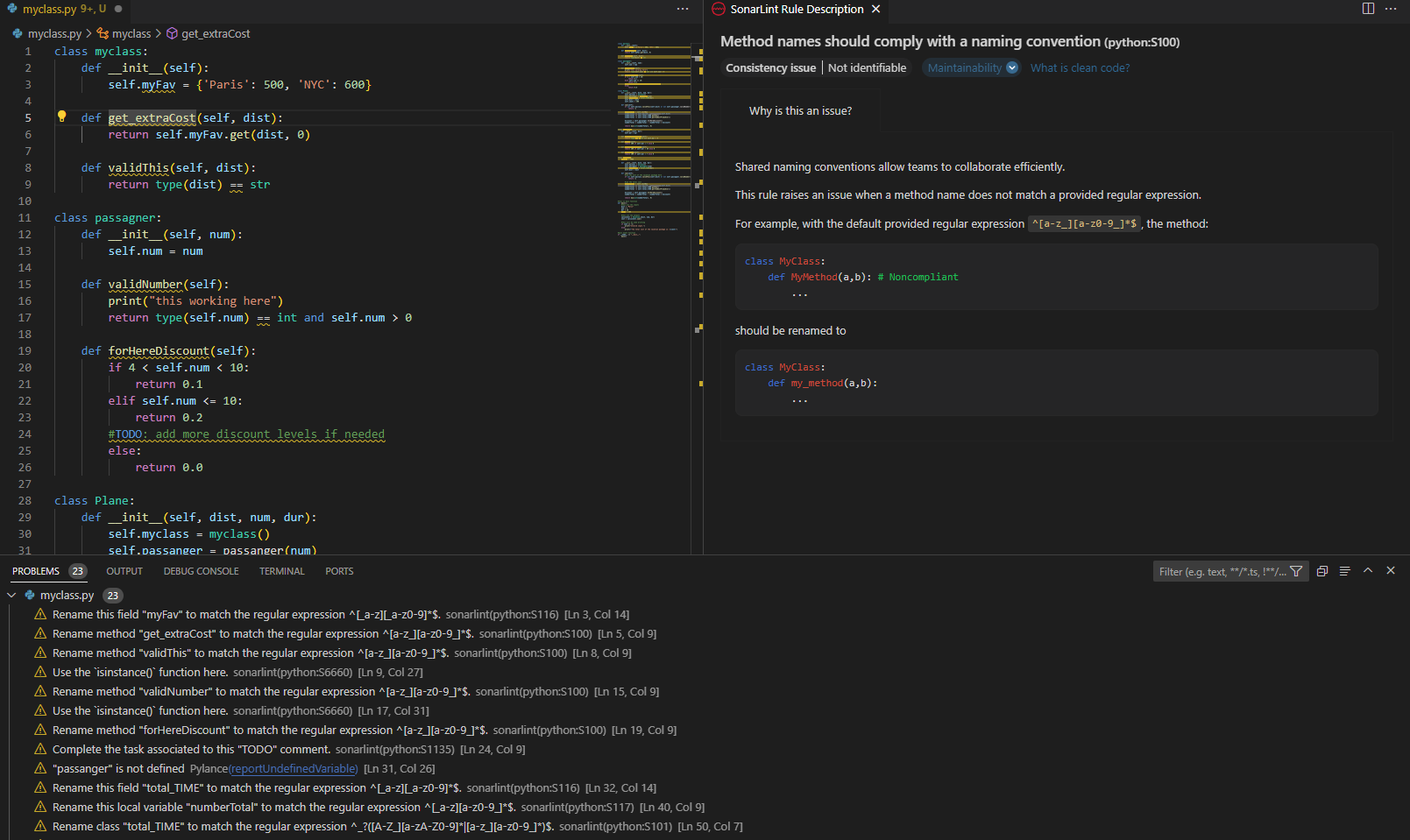
* I chose Python because, for me, it is an easier programming language to learn and use. Python has a simple and clear syntax, which closely resembles natural language, making it easy to read and write code.

**Installation of tools:**

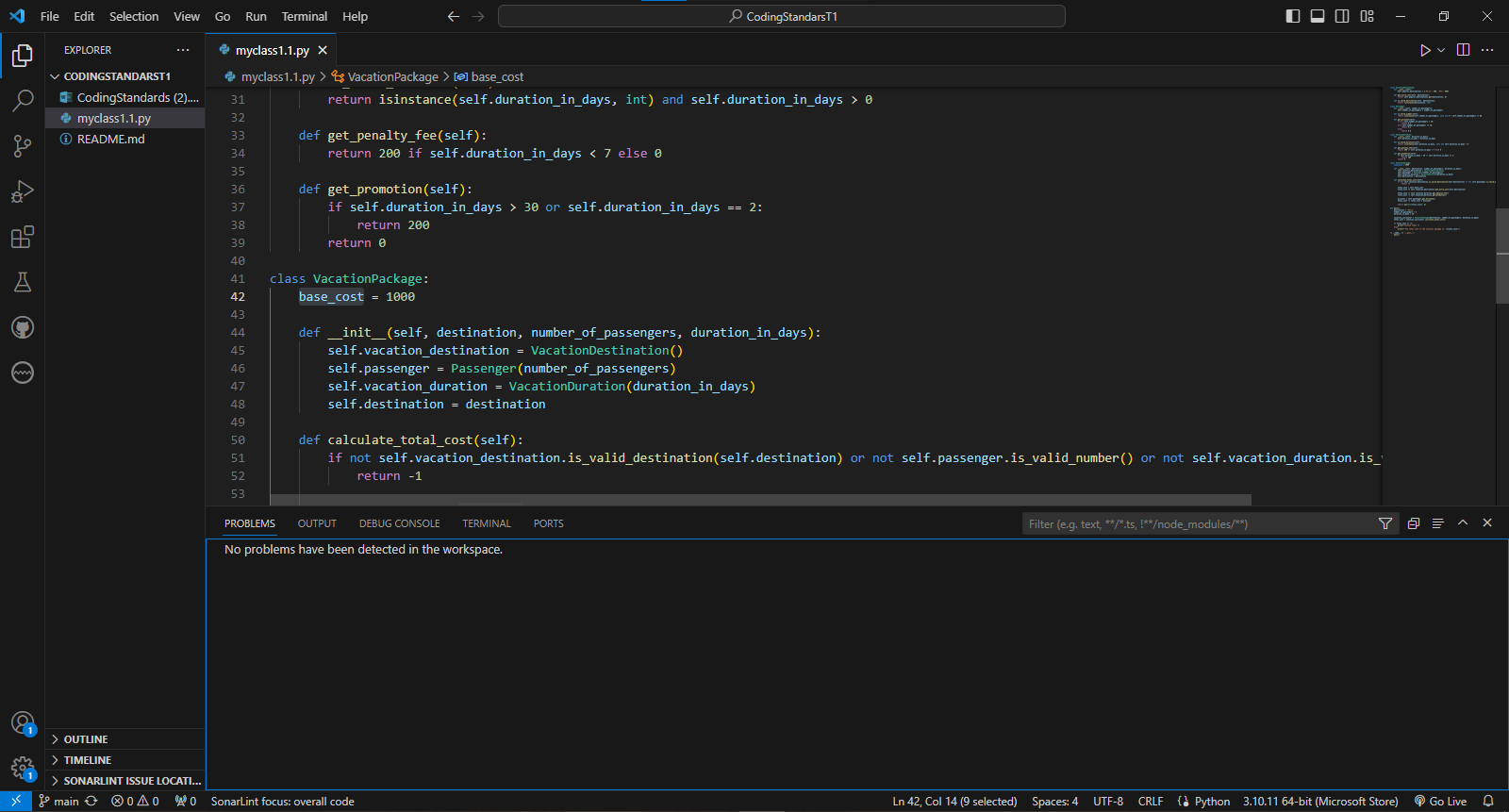


Errors:





Resolved and without errors or warnings:



**Annotated editions:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Original Element | New Element | Change Type | Description | Purpose |
| myclass | VacationDestination | Rename | Rename the class to be more descriptive. | Improves readability and clarity of the class purpose. |
| passagner | Passenger | Rename | Correct the spelling and make the name clearer. | Improves readability and clarity of the class purpose. |
| total\_TIME | VacationDuration | Rename | Rename the class to be more descriptive. | Improves readability and clarity of the class purpose. |
| Vacation\_ | VacationPackage | Rename | Rename the class to be more descriptive. | Improves readability and clarity of the class purpose. |
| myFav | popular\_destinations | Rename | Rename the variable to be more descriptive. | Improves readability. |
| get\_extraCost | get\_extra\_cost | Rename | Rename the method to be more descriptive and use snake\_case convention. | Consistency and readability of the code. |
| validThis | is\_valid\_destination | Rename | Rename the method to be more descriptive and use snake\_case convention. | Consistency and readability of the code. |
| num | number\_of\_passengers | Rename | Rename the variable to be more descriptive. | Improves readability. |
| validNumber | is\_valid\_number | Rename | Rename the method to be more descriptive and use snake\_case convention. | Consistency and readability of the code. |
| forHereDiscount | get\_discount | Rename | Rename the method to be more descriptive and use snake\_case convention. | Consistency and readability of the code. |
| dur | duration\_in\_days | Rename | Rename the variable to be more descriptive. | Improves readability. |
| is\_valid\_total\_TIME | is\_valid\_duration | Rename | Rename the method to be more descriptive and use snake\_case convention. | Consistency and readability of the code. |
| getFee | get\_penalty\_fee | Rename | Rename the method to be more descriptive and use snake\_case convention. | Consistency and readability of the code. |
| getTheBestPromoEver | get\_promotion | Rename | Rename the method to be more descriptive and use snake\_case convention. | Consistency and readability of the code. |
| costBas | base\_cost | Rename | Rename the variable to be more descriptive. | Improves readability. |
| sum (en Vacation\_) | calculate\_total\_cost | Rename | Rename the method to be more descriptive and use snake\_case convention. | Consistency and readability of the code. |
| Plane | Deleted | Deletion | The elimination of Plane and the consolidation of logic in VacationPackage simplify code design, removing an unnecessary layer of abstraction. | Code simplification. |
| getWeekend | Deleted | Deletion | Method deleted because it was not used in the vacation package cost calculation logic. | Code simplification. |
| self.seats | Deleted | Deletion | Attribute deleted from the Plane class since the Plane class was deleted. | Code simplification. |
| seats | Deleted | Deletion | Input variable deleted from the main method since it was not used in the calculation logic. | Code simplification. |
| print("this working here") | Deleted | Deletion | Debugging line deleted to clean up the code. | Code cleanup. |

**Conclusion:**

The vacation package cost estimation project has been successfully redesigned to address bugs and improve code clarity and efficiency. The original version of the code, although functional, had data organization and validation problems. In particular, the Plane class and other elements were not well structured, resulting in calculation errors and improper handling of user input.

In the final version, significant changes have been made to improve the readability and robustness of the code. The calculation logic was distributed into three clearly defined classes: VacationDestination, Passenger and VacationDuration, each responsible for a specific part of the data and business rules. This not only makes the code more modular and easier to maintain, but also ensures that input validations are performed effectively.

The VacationPackage class centralizes the logic for calculating the total cost of the vacation package, integrating business rules related to popular destinations, group discounts and penalties or promotions based on the length of the trip. Thanks to these enhancements, the system now accurately calculates the total cost of a vacation package and handles invalid entries appropriately, returning a value of -1 when an invalid entry is detected.

In conclusion, the implemented changes have corrected previous errors and optimized the system to meet the specified requirements, providing a solid foundation for future extensions and enhancements.

**Link GITHUB:**

<https://github.com/xxcobos/CodingStandarsT1.git>

**Rubric**

|  |  |
| --- | --- |
| **Description** | **Value** |
| Section A  Criteria: lab report content (10pts)   * All pieces of evidence: sections, screenshots of the process, initial and fixed reports, challenge and URL are included: 9-10pts * Just both reports are included: 2-8pts * Empty, partial, or undelivered report: 0 pts Criteria: fixed issues (20 pts) * All the errors/warnings were fixed: 20 pts * All the errors/varnings fixed, no use of tool: 10 pts * Some errors/warnings were fixed: 5 pts * Empty report or neither error/warning fixed: 0 pts Criteria: code (20 pts) * The code fulfills all user requirements given: 20 pts * The code misses some of the requirements given: 10 pts * The code is not related to the user requirements at all: 0 pts | 50 |
| Section B  Criteria: lab report content (10pts)   * All pieces of evidence: sections, screenshots of the process, initial and fixed reports, challenge and URL are included: 9-10pts * Just both reports are included: 2-8pts * Empty, partial, or undelivered report: 0 pts Criteria: fixed issues (20 pts) * All the errors/warnings were fixed: 20 pts * All the errors/varnings fixed, no use of tool: 10 pts * Some errors/warnings were fixed: 5 pts * Empty report or neither error/warning fixed: 0 pts Criteria: code (20 pts) * The code fulfills all user requirements given: 20 pts * The code misses some of the requirements given: -10 pts * The code is not related to the user requirements at all: 0 pts | 50 |
| Penalty for not having **url of a public repository** | -100 |

**Late Submission Policy**

|  |  |
| --- | --- |
| **Delay (§)** | **Penalty (Ω)** |
| 1 hour or less | loss of 10% |
| 1 to 6 hours | loss of 20% |
| 6 to 24 hours | loss of 30% |
| Over 24 hours: | loss of 100% |

**(§) every clock hour counts including weekends or holidays.**

**(Ω) automatic and non-negotiable penalty**