Inspection Checklist for Team 23

Code Written By: Shivang Bhut(ayw932)

Code Inspected By: Dakota Main(ykv395)

Code Inspected: FinalizeOrderView.py

https://git.cs.usask.ca/ykv395/team_23/-/blob/ce25479936350998d52f1ac48373210748a1a64b/src/FinalizeOrderView.py

Product: Retail Billing System

Checklist to check for code's correctness, organization, security, performance, documentation and functionality, to meet system requirements and standard

General:

- Does the code function as intended?
  Yes
- Is the code well-organized and easy to understand?
  Yes
- Is there any duplicated code that could be refactored?
  No
- Have all unnecessary or commented-out code blocks been removed?
  No, there are a number of print statements from testing that remain in the code.
- Is the code modular and promotes re-usability?
  Partially, the print and email functions could be separated to improve modualrity and re-usability.
- Are loop conditions and termination conditions well-defined and clear?
  Not applicable, no loops in this code.
- Are all variable names meaningful and clear?
  Yes
- Does the code handle edge cases correctly?
  No, the code does not look at the response from sendgrid.

Python-Specific:

- Does the code pass pylint?
  No, there are a number of pylint warnings
- Is the code properly formatted using black?
  Yes

Security:

- Is user authentication and authorization handled correctly?
  Yes
- If SQL queries are used are proper placeholders used to prevent SQL injection?
  Not applicable as the code is not directly executing sql queries.

Documentation:

- Does each module have a docstring explaining its purpose and functionality?
  No, the module is missing a docstring
- Does each class have a docstring explaining its purpose and functionality?
  No, the FinalizeOrderView class is missing a docstring
- Does each method have a docsting explaining its purpose and functionality?
  No, the Email_Validator method is missing a docstring
- Is any special case behavior documented?
  Yes
- Are inline comments used to provide clarity where necessary?
  Yes

Performance:

- Is the code optimized for performance, with no obvious bottlenecks or inefficiencies?
  Yes
- Can any computational or database operations be further optimized?
  No

Testing:

- Is the code structured to be easily testable, with well-defined functionality?
  Yes
- Are there comprehensive tests covering different features and scenarios?
  Yes
- Do the test cases adequately check the code's functionality and edge cases?
  Yes