SOFTWARE ENGINEERING
Lab 5

Name: Rakshith Mahesh
SRN: PES1UG23CS469

Issue Table:

Issue	Type	Line(s)	Description	Fix Approach
B307 (Bandit), W0123 (Pylint) Use of insecure eval()	Security (High)	59	**Use of insecure eval() function:** The eval() function was used, which is a significant security risk as it can execute arbitrary code.	**Remove the eval() call.** For safely evaluating strings as Python literals (like lists or dicts), use ast.literal_eval. In this specific case, the line eval("print('eval used')") appears to be for testing and should be deleted.
B110 (Bandit), E722 (Flake8), W0702 (Pylint) Bare except block	Security / Best Practice (Medium)	19	**Bare except block:** The code uses a try...except...pass block with a bare except, which catches and silences *all* possible errors. This can hide bugs and prevent the program from exiting cleanly.	**Specify the exact exception** you expect to catch. For del stock_data[item], this is likely a KeyError. Change except: to except KeyError: and consider logging the error instead of just passing.

C0103 (Pylint) Invalid function name	Convention / Style (Low)	8, 14, 22, 25, 31, 36, 41	**Invalid function name:** Function names like addItem and removeItem do not conform to the snake_case naming style (e.g., add_item) required by Python's PEP 8 guidelines.	**Rename all functions** to use snake_case (e.g., addItem becomes add_item, loadData becomes load_data, etc.).
C0114, C0116 (Pylint) Missing docstrings	Convention / Style (Medium)	1, 8, 14, 22, 25, 31, 36, 41, 48	**Missing docstrings:** The module is missing a docstring , and all functions are missing docstrings. Docstrings are crucial for documentation.	**Add docstrings.** Add a module-level docstring at the top of the file (e.g., """Inventory management system..."""). Add a docstring inside each function explaining its purpose, arguments, and return value.
W0102 (Pylint) Dangerous default value	Warning / Bug-prone (High)	8	**Dangerous default value:** The addItem function uses an empty list [] as a default argument. Mutable defaults are shared across all function calls, leading to unexpected behavior.	**Use None as the default** and create a new list inside the function. def add_item(..., details=None): if details is None: details = []

R1732, W1514 (Pylint) Improper file handling	Best Practice / Refactor (High)	26, 32	**Improper file handling:** Files are opened without using a with statement and without specifying a text encoding. This can lead to resource leaks and encoding errors on different operating systems.	**Use a with statement** to automatically manage closing the file, and **specify encoding="utf-8"** in the open() call. with open("data.json", "r", encoding="utf-8") as f:
W0603 (Pylint) Ineffective use of global statement	Best Practice / Refactor (Low)	27	**Use of global statement:** The global keyword is used to modify stock_data. This makes the code harder to test and debug, as functions have hidden side effects.	**Refactor to avoid global.** Pass stock_data as a parameter to the functions that need it (like loadData), and have those functions return the modified data.
F401 (Flake8), W0611 (Pylint) Unused import	Lint / Code Quality (Low)	2	**Unused import:** The logging module is imported but is never used in the code.	**Remove the line** import logging to clean up the code and avoid confusion.

E302, E261, E262, E305 (Flake8) Spacing violations	Style (Low)	8, 11, 14, 22, 25, 31, 36, 41, 48, 61	PEP 8 spacing violations: The code has incorrect whitespace, including: 1. Not enough blank lines between functions (E302, E305). 14 2. Incorrect spacing for inline comments (E261, E262). 15	Fix the formatting: 1. Ensure there are two blank lines separating each top-level function. 2. Ensure inline comments start with # (a space after the hash) 16and are preceded by at least two spaces. 17
C0209 (Pylint) Use of regular string	Convention / Style (Low)	12	**Consider using f-string:** A regular string is being formatted where a modern f-string would be more readable.	**Convert the string formatting to an f-string.** For example, print("Added: %s" % item) would become print(f"Added: {item}").