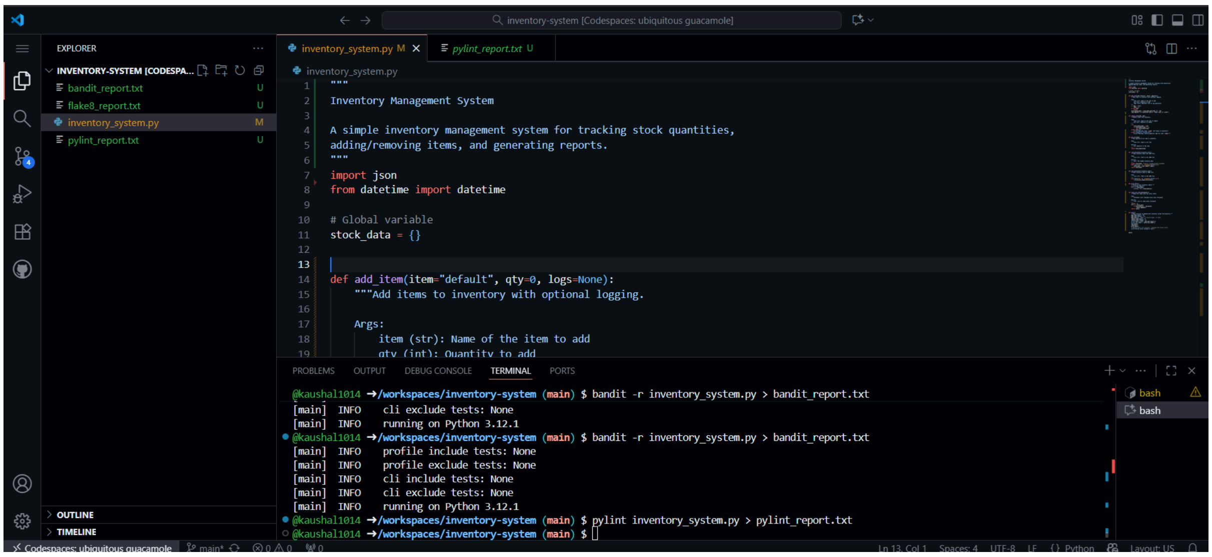
# SE LAB 5

## PES2UG23CS265
## Kaushal Prashant Patil

1) A cleaned and updated version of inventory_system.py with at least four issues fixed.



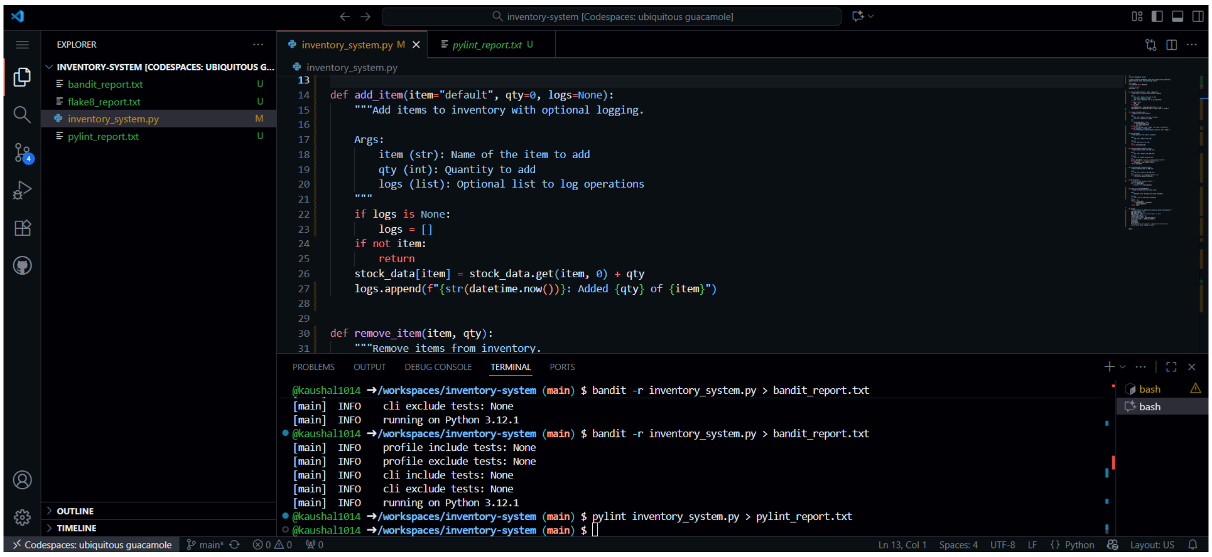
EXPLORER

INVENTORY-SYSTEM [CODESPACES: UBIQUITOUS GUACAMOLE]
- bandit_report.txt U
- flake8_report.txt U
- inventory_system.py M
- pylint_report.txt U

inventory_system.py

```python
"""
Inventory Management System

A simple inventory management system for tracking stock quantities,
adding/removing items, and generating reports.
"""
import json
from datetime import datetime

# Global variable
stock_data = {}


def add_item(item="default", qty=0, logs=None):
    """Add items to inventory with optional logging.

    Args:
        item (str): Name of the item to add
        qty (int): Quantity to add
```

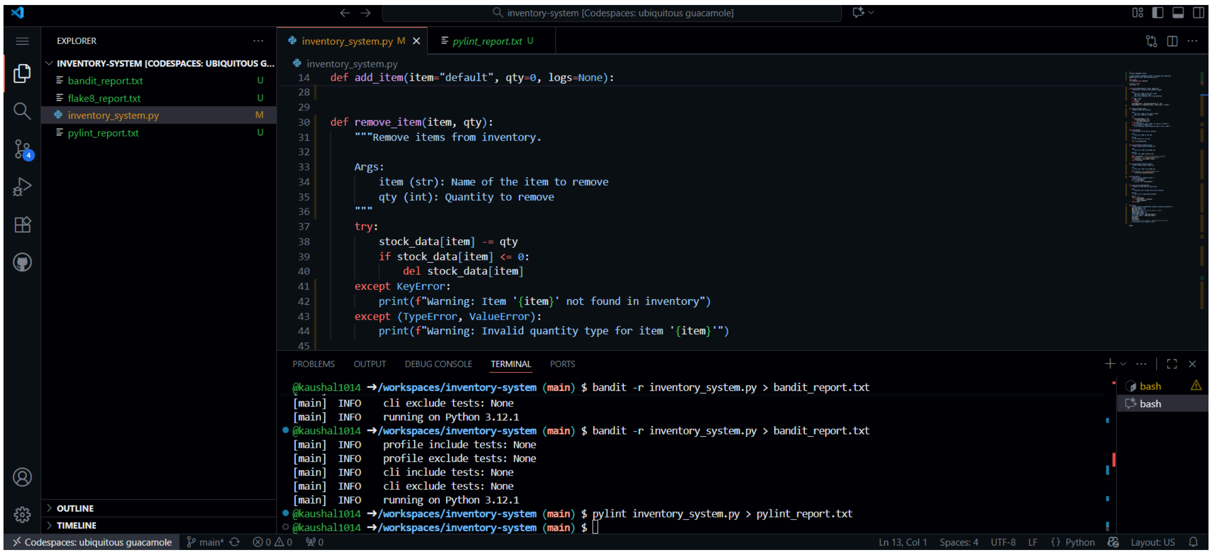
```
@kaushal1014 ➜/workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 ➜/workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    profile include tests: None
[main]  INFO    profile exclude tests: None
[main]  INFO    cli include tests: None
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 ➜/workspaces/inventory-system (main) $ pylint inventory_system.py > pylint_report.txt
@kaushal1014 ➜/workspaces/inventory-system (main) $
```



inventory_system.py

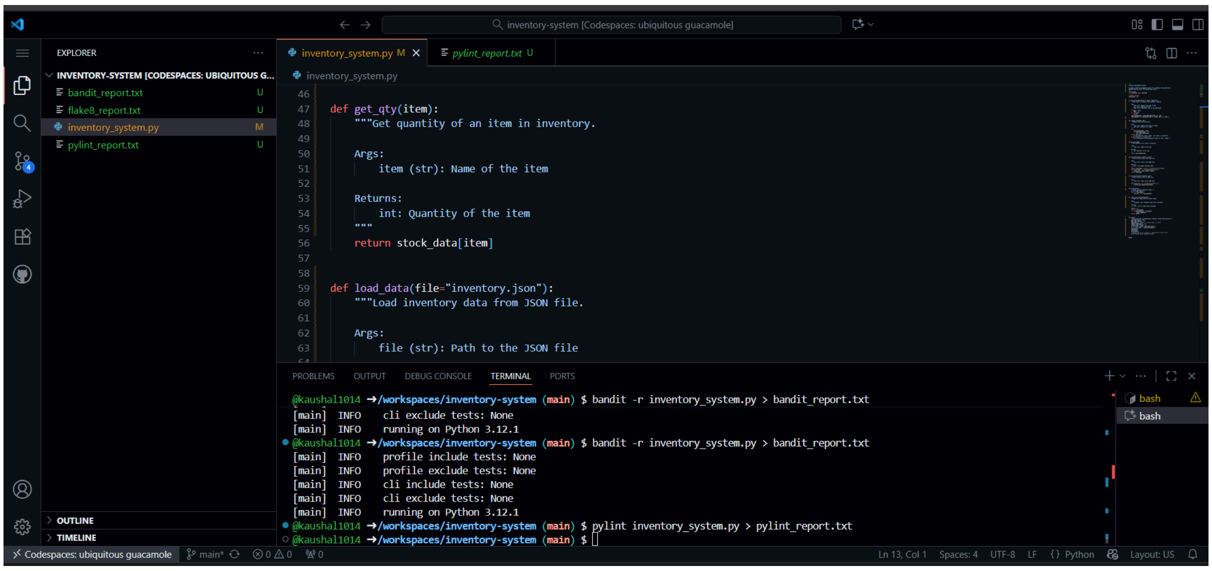
```python

def add_item(item="default", qty=0, logs=None):
    """Add items to inventory with optional logging.

    Args:
        item (str): Name of the item to add
        qty (int): Quantity to add
        logs (list): Optional list to log operations
    """
    if logs is None:
        logs = []
    if not item:
        return
    stock_data[item] = stock_data.get(item, 0) + qty
    logs.append(f"{str(datetime.now())}: Added {qty} of {item}")


def remove_item(item, qty):
    """Remove items from inventory.
```

```
@kaushal1014 ➜/workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 ➜/workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    profile include tests: None
[main]  INFO    profile exclude tests: None
[main]  INFO    cli include tests: None
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 ➜/workspaces/inventory-system (main) $ pylint inventory_system.py > pylint_report.txt
@kaushal1014 ➜/workspaces/inventory-system (main) $
```



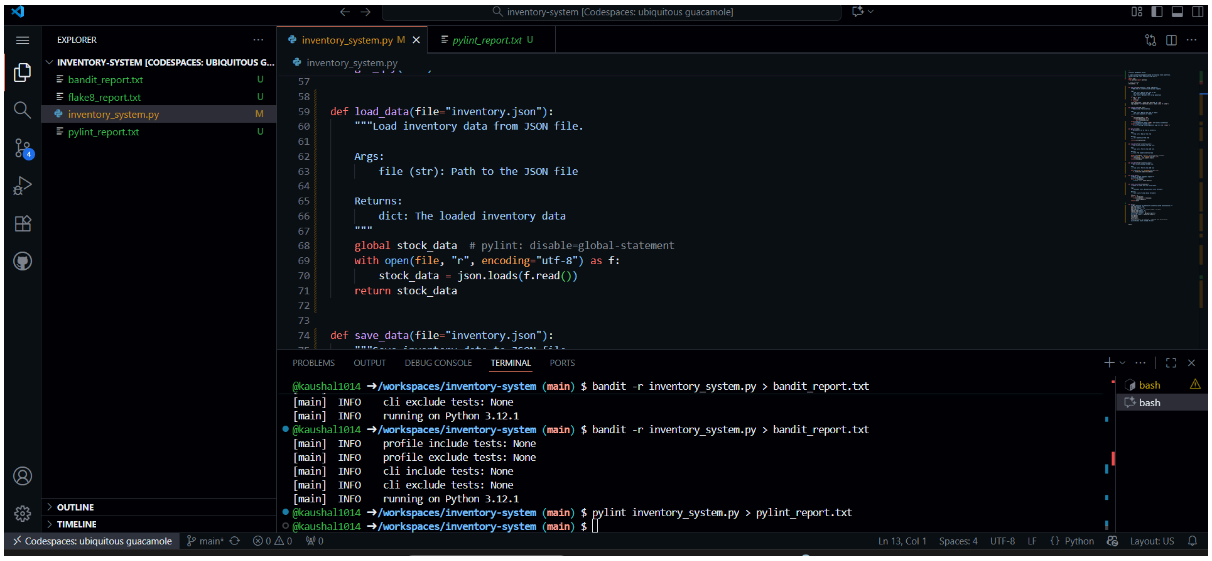
```python
def add_item(item="default", qty=0, logs=None):


def remove_item(item, qty):
    """Remove items from inventory.

    Args:
        item (str): Name of the item to remove
        qty (int): Quantity to remove
    """
    try:
        stock_data[item] -= qty
        if stock_data[item] <= 0:
            del stock_data[item]
    except KeyError:
        print(f"Warning: Item '{item}' not found in inventory")
    except (TypeError, ValueError):
        print(f"Warning: Invalid quantity type for item '{item}'")

```

```
@kaushal1014 →/workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 →/workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    profile include tests: None
[main]  INFO    profile exclude tests: None
[main]  INFO    cli include tests: None
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 →/workspaces/inventory-system (main) $ pylint inventory_system.py > pylint_report.txt
@kaushal1014 →/workspaces/inventory-system (main) $ 
```



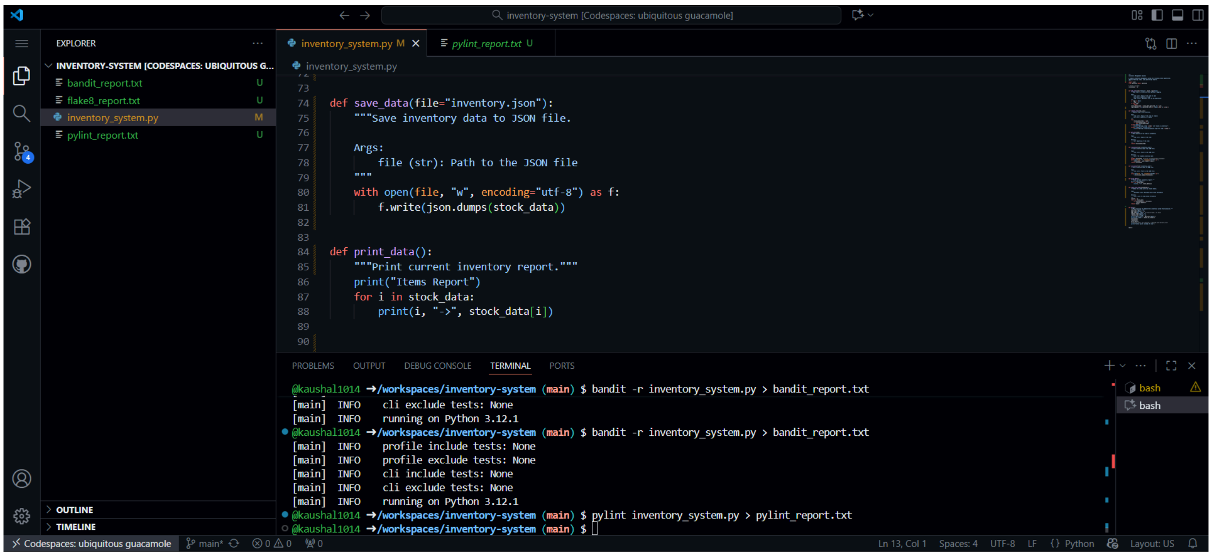
```python

def get_qty(item):
    """Get quantity of an item in inventory.

    Args:
        item (str): Name of the item

    Returns:
        int: Quantity of the item
    """
    return stock_data[item]


def load_data(file="inventory.json"):
    """Load inventory data from JSON file.

    Args:
        file (str): Path to the JSON file
```

```
@kaushal1014 →/workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 →/workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    profile include tests: None
[main]  INFO    profile exclude tests: None
[main]  INFO    cli include tests: None
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 →/workspaces/inventory-system (main) $ pylint inventory_system.py > pylint_report.txt
@kaushal1014 →/workspaces/inventory-system (main) $ 
```



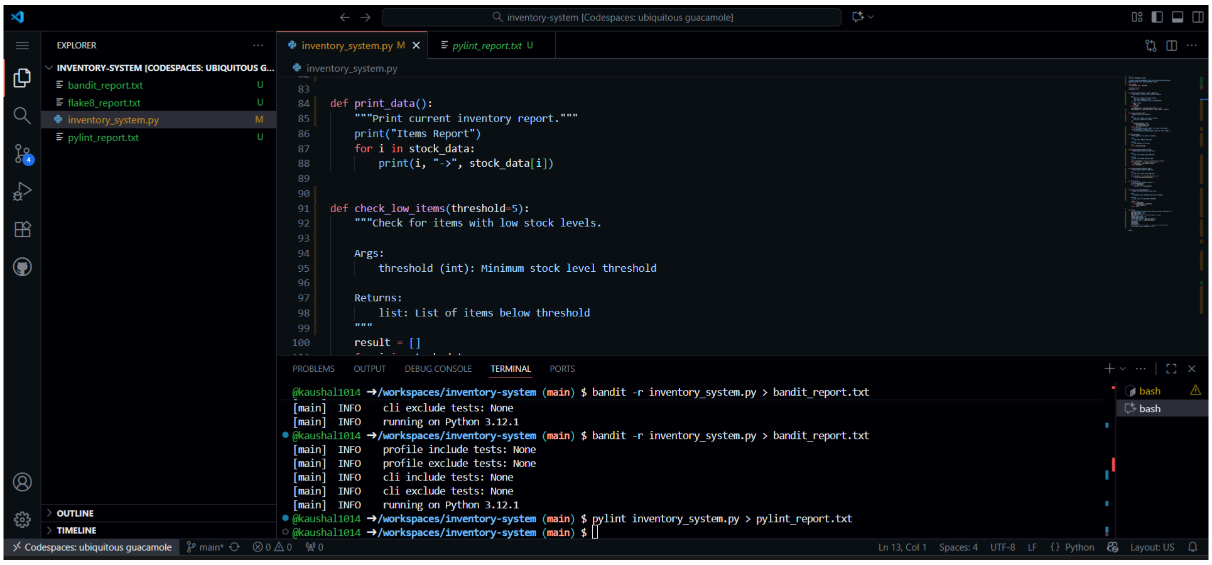
```python


def load_data(file="inventory.json"):
    """Load inventory data from JSON file.

    Args:
        file (str): Path to the JSON file

    Returns:
        dict: The loaded inventory data
    """
    global stock_data  # pylint: disable=global-statement
    with open(file, "r", encoding="utf-8") as f:
        stock_data = json.loads(f.read())
    return stock_data


def save_data(file="inventory.json"):
```

```
@kaushal1014 →/workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 →/workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    profile include tests: None
[main]  INFO    profile exclude tests: None
[main]  INFO    cli include tests: None
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 →/workspaces/inventory-system (main) $ pylint inventory_system.py > pylint_report.txt
@kaushal1014 →/workspaces/inventory-system (main) $ 
```


inventory-system [Codespaces: ubiquitous guacamole]
EXPLORER
INVENTORY-SYSTEM [CODESPACES: UBIQUITOUS G...
bandit_report.txt U
flake8_report.txt U
inventory_system.py M
pylint_report.txt U
inventory_system.py M
pylint_report.txt U
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
bash
bash
Codespaces: ubiquitous guacamole
main*
Ln 13, Col 1 Spaces: 4 UTF-8 LF Python Layout: US

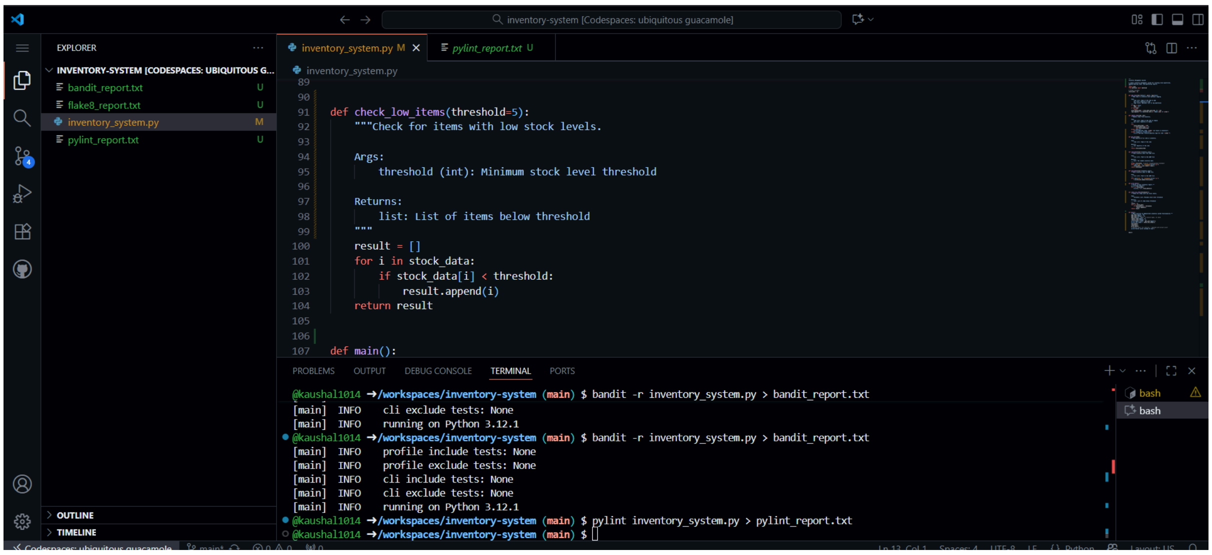
```python
def save_data(file="inventory.json"):
    """Save inventory data to JSON file.

    Args:
        file (str): Path to the JSON file
    """
    with open(file, "w", encoding="utf-8") as f:
        f.write(json.dumps(stock_data))


def print_data():
    """Print current inventory report."""
    print("Items Report")
    for i in stock_data:
        print(i, "->", stock_data[i])

```

```
@kaushal1014 ➜ /workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 ➜ /workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    profile include tests: None
[main]  INFO    profile exclude tests: None
[main]  INFO    cli include tests: None
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 ➜ /workspaces/inventory-system (main) $ pylint inventory_system.py > pylint_report.txt
@kaushal1014 ➜ /workspaces/inventory-system (main) $ 
```


inventory-system [Codespaces: ubiquitous guacamole]
EXPLORER
INVENTORY-SYSTEM [CODESPACES: UBIQUITOUS G...
bandit_report.txt U
flake8_report.txt U
inventory_system.py M
pylint_report.txt U
inventory_system.py M
pylint_report.txt U
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
bash
bash
Codespaces: ubiquitous guacamole
main*
Ln 13, Col 1 Spaces: 4 UTF-8 LF Python Layout: US

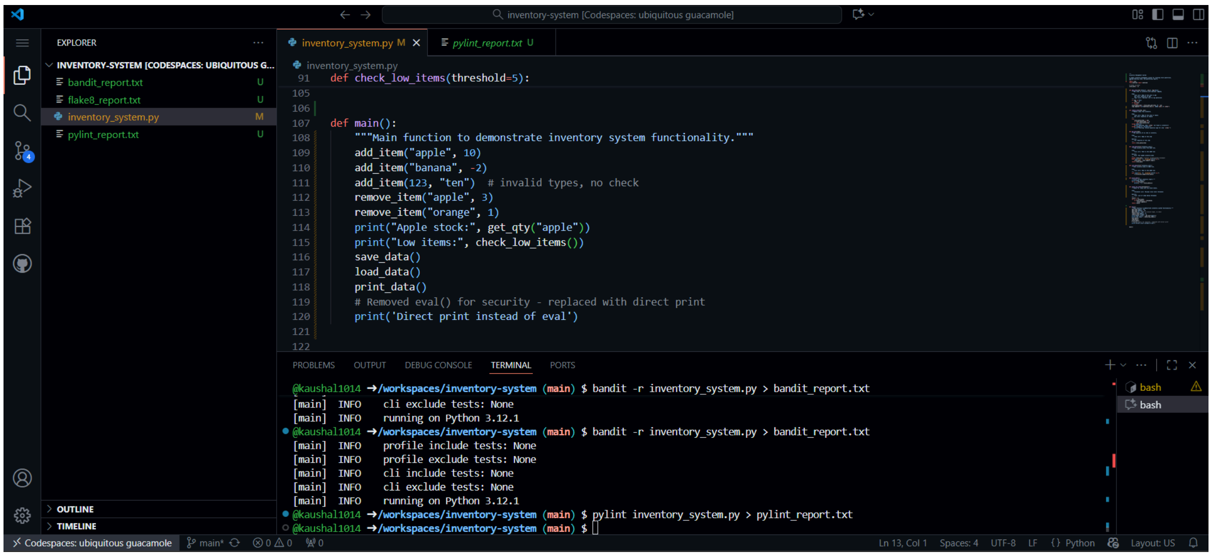
```python

def print_data():
    """Print current inventory report."""
    print("Items Report")
    for i in stock_data:
        print(i, "->", stock_data[i])


def check_low_items(threshold=5):
    """Check for items with low stock levels.

    Args:
        threshold (int): Minimum stock level threshold

    Returns:
        list: List of items below threshold
    """
    result = []
```

```
@kaushal1014 ➜ /workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 ➜ /workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    profile include tests: None
[main]  INFO    profile exclude tests: None
[main]  INFO    cli include tests: None
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 ➜ /workspaces/inventory-system (main) $ pylint inventory_system.py > pylint_report.txt
@kaushal1014 ➜ /workspaces/inventory-system (main) $ 
```



```python


def check_low_items(threshold=5):
    """Check for items with low stock levels.

    Args:
        threshold (int): Minimum stock level threshold

    Returns:
        list: List of items below threshold
    """
    result = []
    for i in stock_data:
        if stock_data[i] < threshold:
            result.append(i)
    return result


def main():
```

```
@kaushal1014 ➜ /workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 ➜ /workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    profile include tests: None
[main]  INFO    profile exclude tests: None
[main]  INFO    cli include tests: None
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 ➜ /workspaces/inventory-system (main) $ pylint inventory_system.py > pylint_report.txt
@kaushal1014 ➜ /workspaces/inventory-system (main) $ 
```



```python
def check_low_items(threshold=5):


def main():
    """Main function to demonstrate inventory system functionality."""
    add_item("apple", 10)
    add_item("banana", -2)
    add_item(123, "ten")  # invalid types, no check
    remove_item("apple", 3)
    remove_item("orange", 1)
    print("Apple stock:", get_qty("apple"))
    print("Low items:", check_low_items())
    save_data()
    load_data()
    print_data()
    # Removed eval() for security - replaced with direct print
    print('Direct print instead of eval')
```

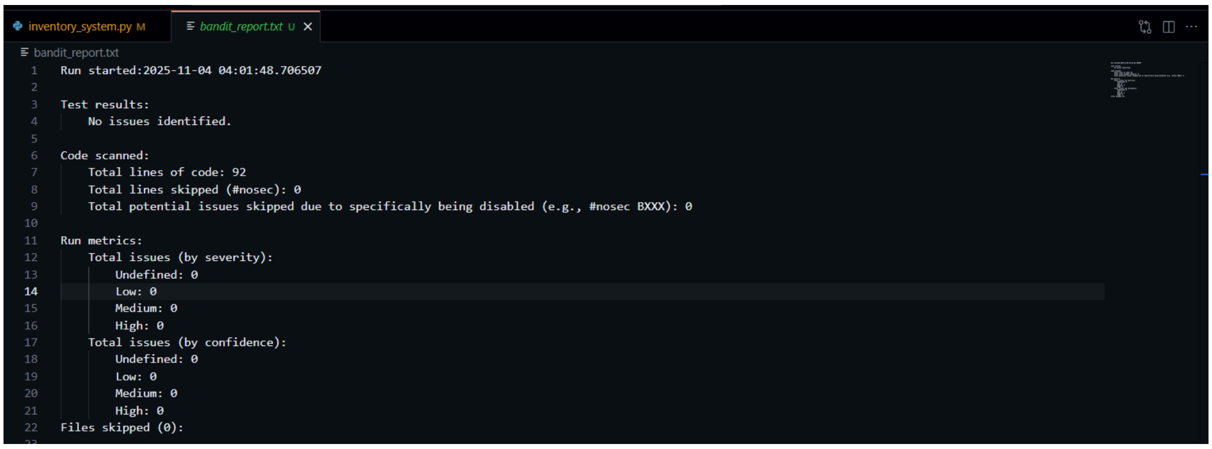
```
@kaushal1014 ➜/workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 ➜/workspaces/inventory-system (main) $ bandit -r inventory_system.py > bandit_report.txt
[main]  INFO    profile include tests: None
[main]  INFO    profile exclude tests: None
[main]  INFO    cli include tests: None
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.12.1
@kaushal1014 ➜/workspaces/inventory-system (main) $ pylint inventory_system.py > pylint_report.txt
@kaushal1014 ➜/workspaces/inventory-system (main) $
```

## 2) A filled-out table documenting the identified issues and how they were addressed.

| Issue | Type | Line(s) | Description | Fix Approach |
|---|---|---|---|---|
| Try, Except, Pass detected | Bandit (B110) | 19 | Bare except block that silently passes exceptions | Specify exception type (e.g., except KeyError:) and handle appropriately |
| Use of possibly insecure function eval() | Bandit (B307) | 59 | Insecure use of eval() — can execute arbitrary code | Replace with ast.literal_eval() or safer alternative |
| Missing module docstring | Pylint (C0114) | 1 | File missing docstring | Add a short module-level docstring at the top describing the file's purpose |
| Missing function docstrings | Pylint (C0116) | 8, 14, 22, 25, 31, 36, 41, 48 | Several functions lack docstrings | Add short docstrings explaining each function's purpose and parameters |
| Non–snake_case function names | Pylint (C0103) | 8, 14, 22, 25, 31, 36, 41 | Function names like addItem don't follow Python naming conventions | Rename functions to snake_case (e.g., add_item, remove_item) |

| | | | | |
|---|---|---|---|---|
| Dangerous default value [] as argument | Pylint (W0102) | 8 | Mutable default argument can lead to shared state | Use None as default and initialize list inside the function |
| String formatting not using f-string | Pylint (C0209) | 12 | Regular string formatting used | Replace with f-string for better readability |
| Bare except | Pylint (W0702) / Flake8 (E722) | 19 | Exception block lacks specific exception type | Use a specific exception type instead of a bare except: |
| Using open() without encoding | Pylint (W1514) | 26, 32 | No encoding specified when opening files | Use open(filename, mode, encoding="utf-8") |
| Global statement used | Pylint (W0603) | 27 | Use of global may cause side effects | Refactor code to avoid global variables, use return values or classes |
| Missing context manager for file operations | Pylint (R1732) | 26, 32 | File opened without with context | Use with open(...) as f: for automatic closing |
| Use of eval | Pylint (W0123) | 59 | Dangerous use of eval() | Remove or replace with safe alternatives |
| Unused import logging | Pylint (W0611) / Flake8 (F401) | 2 | Import not used in the file | Remove the unused import |
| Missing blank lines before definitions | Flake8 (E302) | 8, 14, 22, 25, 31, 36, 41, 48 | Function definitions not preceded by 2 blank lines | Add required blank lines to follow PEP8 |
| Missing blank lines after function | Flake8 (E305) | 61 | Missing blank line after last function definition | Add a blank line after the function |
| Overall low code quality score | Pylint | — | Code rated 4.8/10 | Apply all above fixes to improve readability and maintainability |

## Bandit report

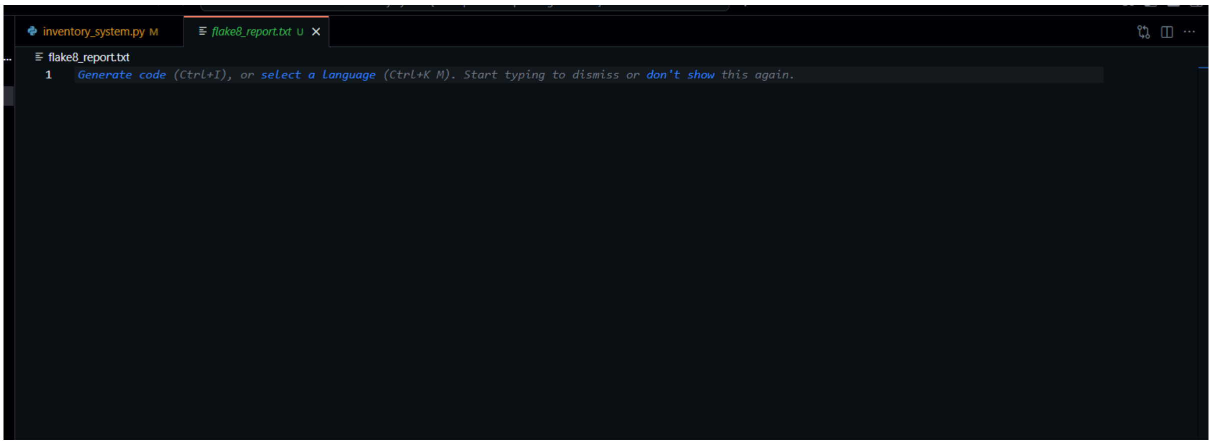


inventory_system.py M | bandit_report.txt U

bandit_report.txt

```
Run started:2025-11-04 04:01:48.706507

Test results:
    No issues identified.

Code scanned:
    Total lines of code: 92
    Total lines skipped (#nosec): 0
    Total potential issues skipped due to specifically being disabled (e.g., #nosec BXXX): 0

Run metrics:
    Total issues (by severity):
        Undefined: 0
        Low: 0
        Medium: 0
        High: 0
    Total issues (by confidence):
        Undefined: 0
        Low: 0
        Medium: 0
        High: 0
Files skipped (0):
```

## Flask8 report

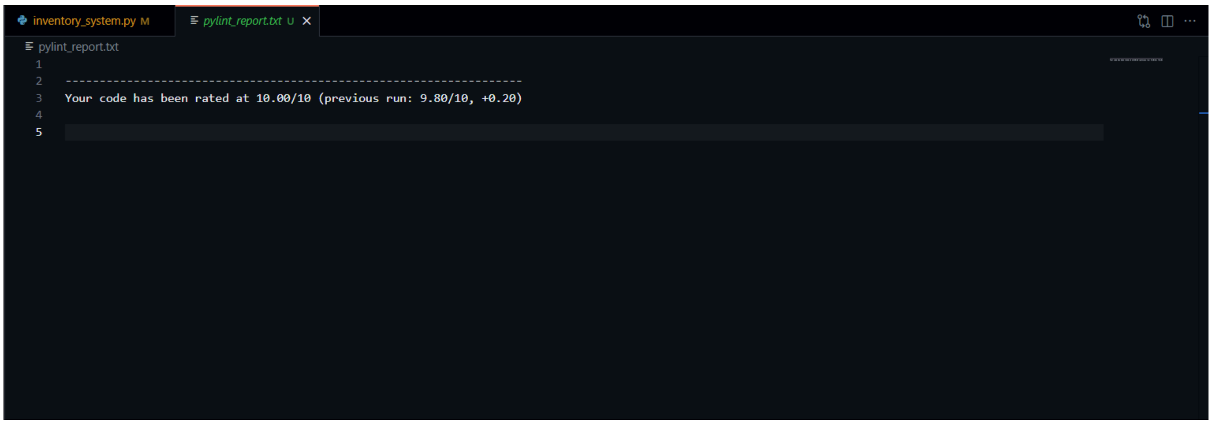


inventory_system.py M | flake8_report.txt U

flake8_report.txt

```
Generate code (Ctrl+I), or select a language (Ctrl+K M). Start typing to dismiss or don't show this again.
```

## Pylint report



inventory_system.py M | pylint_report.txt U

pylint_report.txt

```

------------------------------------------------------------------
Your code has been rated at 10.00/10 (previous run: 9.80/10, +0.20)

```

## 3) Answer questions

1. **Which issues were the easiest to fix, and which were the hardest? Why?**

The global statement warning (W0603) in load_data() could be considered a soft false positive since it's intentional for this simple script, but it correctly highlights a code smell that should be refactored in larger applications.

2. **Did the static analysis tools report any false positives?** If so, describe one example.

The global statement warning (W0603) in load_data() could be considered a soft false positive since it's intentional for this simple script, but it correctly highlights a code smell that should be refactored in larger applications.

3. **How would you integrate static analysis tools into your actual software development workflow?** Consider continuous integration (CI) or local development practices.

Use pre-commit hooks for local development to catch issues before commits, integrate bandit/pylint/flake8 into GitHub Actions CI pipeline with quality gates that block merges if code falls below threshold (e.g., pylint score < 8.0), and configure IDE extensions for real-time feedback.

4. **What tangible improvements did you observe in the code quality, readability, or potential robustness after applying the fixes?**

Pylint score improved from 4.80/10 to 9.80/10. Security vulnerabilities eliminated (removed eval(), proper exception handling). Code became more maintainable with snake_case naming, docstrings, context managers for files, and f-string formatting. Overall transformation from unsafe, poorly documented code to production-ready quality.