NAME : RAMA VISHWATEJ

HALLTICKET NUMBER : 2503A52L01

BATCH : 16

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
Program Name: B. Tech		**Assignment Type: Lab**	**Academic Year:**2025-2026
Course Coordinator Name		Venkataramana Veeramsetty	
Instructor(s)Name		1. Dr. Mohammed Ali Shaik 2. Dr. T Sampath Kumar 3. Mr. S Naresh Kumar 4. Dr. V. Rajesh 5. Dr. Brij Kishore 6. Dr Pramoda Patro 7. Dr. Venkataramana 8. Dr. Ravi Chander 9. Dr. Jagjeeth Singh	
Course Code	24CS002PC215	**Course Title**	AI Assisted Coding
Year/Sem	II/I	**Regulation**	R24
Date and Day of Assignment	06-08-2025	**Time(s)**	
Duration	2 Hours	**Applicable to Batches**	
AssignmentNumber:6.5(Present assignment number)/**24**(Total number of assignments)			

Q.No.	Question	ExpectedTime to complete
1	**Lab 6: AI-Based Code Completion: Working with suggestions for classes, loops, conditionals** Lab Assignment 1: Intelligent Code Completion for Object-Oriented Programming **Objective:** To explore AI-powered code assistants for writing Python classes, constructors, and methods through intelligent suggestions. Suppose that you are hired as an intern at a tech company that develops inventory management systems. Your manager asks you to create a **Product** class and a **Warehouse** class with some basic methods. You have decided to use AI-powered code suggestions to help speed up development and reduce syntax errors. Tasks to be completed are as below **1. Setup AI Coding Tool:** • Install and configure GitHub Copilot or Kite with VS Code or JetBrains IDE. • Enable real-time code suggestions. **2. Class Design Using AI Assistance:** • Begin defining a Product class with attributes: name, price, quantity.	15.08.2025 EOD

- Use the AI suggestion feature to automatically complete the __init__() method.
- Add a method calculate_value() to return price * quantity.

3. Create Another Class:
- Define a Warehouse class with a list of Product objects.
- Use code completion to help implement:
  - A method to add a product.
  - A method to display the most valuable product.

4. Reflection:
- Identify how much of the code was completed by AI and what manual edits were needed.
- Comment on the relevance and accuracy of AI suggestions.

Requirements:
- VS Code with Github Copilot or Cursor API and/or Google Colab with Gemini

Deliverables:

- Python script with both classes and comments on AI-generated suggestions.
- Short report (1 page) summarizing your experience with AI code completion.
  .

Prompt : generate program that develops inventory management systems. Create a Product class and a Warehouse class with some basic methods. Use AI-powered code suggestions to help speed up development and reduce syntax errors

CODE :

Main.py

```python
from product import Product
from warehouse import Warehouse

warehouse = Warehouse()

# Take product entries from user
num_products = int(input("How many products do you want to add? "))
for i in range(num_products):
    name = input(f"Enter name for product {i+1}: ")
    price = float(input(f"Enter price for product {i+1}: "))
    quantity = int(input(f"Enter quantity for product {i+1}: "))
    product = Product(name, price, quantity)
    warehouse.add_product(product)
```

```python
# Display most valuable product
most_valuable = warehouse.most_valuable_product()
print(f"Most valuable product: {most_valuable.name}
(${most_valuable.calculate_value()})")
```

Product.py

```python
class Product:
    def __init__(self, name, price, quantity):
        self.name = name
        self.price = price
        self.quantity = quantity

    def calculate_value(self):
        return self.price * self.quantity
```

Warehouse.py

```python
class Product:
    def __init__(self, name, price, quantity):
        self.name = name
        self.price = price
        self.quantity = quantity

    def calculate_value(self):
        return self.price * self.quantity


class Warehouse:
    def __init__(self):
        self.products = []

    def add_product(self, product):
        self.products.append(product)

    def most_valuable_product(self):
        if not self.products:
            return None
        return max(self.products, key=lambda product:
product.calculate_value())

    def display_most_valuable_product(self):
        product = self.most_valuable_product()
        if product:
```

```
        return f'Most Valuable Product: {product.name}, Value:
{product.calculate_value()}'
      return 'No products in warehouse.'
```

Output :

```
How many products do you want to add? 1
Enter name for product 1: Smart watch
Enter price for product 1: 600
Enter quantity for product 1: 9
Most valuable product: Smart watch ($5400.0)
```

Observation :

This report reflects on the experience of utilizing AI-powered code completion during the development of the Python project that includes a `Product` class and a `Warehouse` class.

Throughout the project, AI code completion was employed to assist in writing the following components:

The AI suggestions were generally relevant and provided a solid foundation for the project code. However, there were instances where the AI misunderstood the context or provided overly generic code snippets. Continuous interaction and refinement were necessary to achieve the desired outcomes.

Conclusion

Overall, the integration of AI-powered code completion significantly accelerated the development process, allowing for quicker prototyping and implementation of features. While it was not without its challenges, the experience was largely positive, and the AI's contributions were invaluable in shaping the final product. Future projects will benefit from this technology, with an emphasis on iterative refinement and manual oversight to enhance accuracy and relevance.