Test Documentation: E-Commerce Flask Application

This document outlines the purpose and implementation of the three tests included in the **E-Commerce Flask Application** and their integration into the CI/CD pipeline.

Test 1: Route Test

Purpose

To verify the application's route behavior when invalid HTTP methods are used. This ensures the server responds with the correct HTTP status codes (e.g., 405 for Method Not Allowed).

Implementation

- Test the /products endpoint, which accepts GET requests.
- Use a POST request to simulate an invalid method.
- Validate the server returns a 405 Method Not Allowed response.

Test 2: Database Read Operation

Purpose

To verify the application connects to MongoDB successfully and performs read operations without errors. This test uses the MongoDB ping command to ensure connectivity.

Implementation

- Establish a connection with the MongoDB database using credentials from the environment variables.
- Use the admin.command("ping") method to confirm the database is reachable.

Test 3: Database Write Operation

Purpose

To verify the database accepts write operations (e.g., inserting a document) and that the data can be retrieved accurately.

Implementation

- Insert a sample document into the products collection.

- Use assertions to confirm the document was inserted successfully by querying it.

Integration with CI/CD Pipeline

Step 1: Install Dependencies

Ensure the pipeline installs necessary dependencies:

```
pip install -r requirements.txt
```

Step 2: Set Environment Variables

Securely load environment variables into the pipeline. For GitHub Actions:

Step 3: Run Tests

Execute all tests:

Conclusion

These tests validate critical functionalities:

1. Route responses to invalid methods.
2. MongoDB connectivity and read operations.
3. MongoDB write operations and data integrity.

By integrating these tests into the CI/CD pipeline, the application ensures high reliability and stability during development and deployment.