Test Policy

Video games sales Analysis

# **1. Introduction**

This test policy outlines the approach to testing the video game sales prediction project. The project includes a Machine Learning model for predicting video game sales and a backend server for serving the predictions.

# **2. Test Objectives**

The objectives of testing the project are:

- Ensure the accuracy and reliability of the ML model

- Validate the functionality of the backend server

- Verify the integration between the ML model and the backend server

# **3. Test Levels**

The testing process will include the following levels:

## 3.1 Unit Testing

Unit tests will be created for individual components of the ML model and backend server code.

These tests will focus on ensuring that each component functions as expected in isolation.

## 3.2 Integration Testing

Integration tests will be developed to test the interaction between the ML model and the backend server using the Flask-Testing library.

These tests will focus on ensuring that the ML model can be successfully loaded and used by the backend server to generate predictions.

## 3.3 System Testing

System tests will be performed to validate the complete functionality of the project, including both the ML model and the backend server.

These tests will focus on assessing the overall accuracy and performance of the project.

## 3.4 Acceptance Testing

Acceptance tests will be conducted to verify that the project meets the requirements of the stakeholders.

These tests will focus on ensuring that the project is ready for deployment and use.

**4. Test Techniques and Tools**

4.1 Test Techniques

Black-box testing: This technique will be used to test the functionality of the backend server without considering its internal structure.

White-box testing: This technique will be applied to test the ML model's code by considering its internal structure and design.

4.2 Test Tools

Python unittest and Flask-Testing: The Python unittest and Flask-Testing frameworks will be used to create and run unit tests and integration tests for the ML model and backend server code.

Postman: Postman will be used for testing the API endpoints of the backend server.

**5. Test Deliverables**

The following test deliverables will be provided:

- Test plan

- Test cases and scripts

- Test data

- Test results and reports

**6. Test Schedule**

The test schedule will be aligned with the project development schedule. The testing process will be iterative, with tests being performed at each stage of the project development.

**7. Test Resources**

The test resources will include:

- Test environment (hardware, software, and network configurations)

- Test tools (Python unittest, Python pytest Flask-Testing, Postman)

- Test team members

**8. Test Responsibilities**

The test team will be responsible for:

- Developing and executing test cases and scripts

- Analyzing test results and reporting issues

- Collaborating with the development team to resolve issues

- Ensuring the project meets the defined quality standards

**9. Test Risks and Contingencies**

Possible risks during the testing process include:

- Inadequate test coverage

- Insufficient test data

- Limited availability of test resources

### Contingency plans for these risks include:

- Regularly reviewing and updating test cases and scripts

- Generating additional test data as needed

- Allocating additional resources if required