**OptiCloud Test Plan : Modular Cloud Data Processing System**

**1.0 Introduction:**

The OptiCloud **project** focuses on developing a modular and scalable cloud data processing system for **ShopEase**, a leading e-commerce company. This application will be able to manage and process large datasets to effectively handle product information, customer demands, and market trends. The application will be hosted on **AWS** cloud services, which provide capabilities for scalability, security, and performance optimization. The development environment is **Visual Studio 2022** and the main testing tools are **NUnit** for unit testing and integration and JMeter for performance and stress tolerance tests.

**2.0 Objectives and Tasks:**

**2.1 Objectives:**

1. **Validation of individual modules:** Ensure that each module (input, execution, output) performs its specified functions correctly according to predefined requirements.
2. **Seamless integration:** Ensure that the integration of all modules within the application runs smoothly and that data moves accurately between these modules.
3. **Performance assurance**: Verify application performance in processing large datasets, ensuring low latency and high efficiency.
4. **User acceptance:** Perform user acceptance tests (**UATs**) to ensure that the app meets **ShopEase** requirements and provides an intuitive user experience.

**2.2 Functions:**

* **Develop unit test cases:** Create detailed test cases for each unit, specifying inputs, expected results, and acceptance criteria.
* **Unit testing:** Perform unit testing for individual components using **NUnit** within **Visual Studio 2022**.
* **Integration testing:** Perform tests to verify the interaction between modules, ensuring data integrity and correct functionality.
* **Performance tests:** Use **JMeter** to simulate high load scenarios, measure system response and identify potential bottlenecks.
* **User Acceptance Testing:** Develop and implement **UAT scenarios** in collaboration with **ShopEase stakeholders** to ensure the app meets the needs of the end user.
* **Error Reporting:** Document any problems discovered during testing, classify them by severity, and ensure they are resolved in a timely manner.

**3.0 Test Scope:**

Our test covers verifying all functional aspects of **an OptiCloud application** with a focus on modular design (input modules, implementation, output) and integrating these modules into an integrated system. The scope also includes performance tests in **the AWS**  environment and UAT **user acceptance tests** to ensure that the application meets **ShopEase**'s business requirements.

**4.0 Testing Strategy:**

**4.1 Testing units:**

* **Definition:** Unit testing focuses on verifying the smallest testable parts of an application, such as functions and methods within input, implementation, and output modules.
* **Tools: NUnit** in **Visual Studio 2022**.
* **Test cases:** Each function will be tested individually. For example, the AddProduct(product) function in **the EcommercePlatform class will** be tested to make sure that it updates the product catalog correctly and sends notifications.

**4.2 System Testing and Integration:**

* **Definition:** System testing and integration ensure that individual modules work together as expected.
* **Tools: NUnit** for integration tests.
* **Test cases:** Test cases will be created that simulate real interactions. For example, adding a product in the input module will be tested and processed in the execution module, resulting in a notification and report in the output module.

**4.3 Performance test and pressure tolerance:**

* **Definition:** Performance and stress tolerance test how an application performs under load, including its responsiveness, stability, and resource usage.
* **Tools: JMeter**.
* **Test cases:** Scenarios will simulate high user activities and large inputs. For example, the AnalyzeProductTrends() function in the analytics module will be tested under large data loads.

**4.4 User Acceptance Test (UAT):**

* **Definition:** UAT **ensures** that the application meets business requirements and is easy to use.
* **Tools:** A set of manual and automated tests.
* **Test cases:** Scenarios will be created that simulate real processes such as updating customer contact information and managing product inventory.

**5.0 Physical requirements:**

* **AWS Cloud Environment**: To host the application, equipped to handle production-level workloads.
* **Development and testing devices:** All devices must have **Visual Studio 2022** installed as well as **NUnit** and **JMeter**.

****6.0** Test table with forecasts:**

|  |  |  |  |
| --- | --- | --- | --- |
| Expected result | Input | Case Description | Unit Name |
| Successfully add product, and update DataGridView to show the new product | **Enter product type, size, color, price and inventory** | **Add a new product** | **Product Management** |
| |  | | --- | | **Prevent product addition and display error message** |  |  | | --- | |  | | |  | | --- | | **Leave the price field blank** |  |  | | --- | |  | | |  | | --- | | **Add a new product (edge case)** |  |  | | --- | |  | | |  | | --- | | **Product Management** |  |  | | --- | |  | |
| The product has been deleted from the system, and it no longer appears in DataGridView | **Provide a valid product ID for deletion** | **Delete a product** | **Product Management** |
| Display an error message indicating that the product does not exist with the specified ID | **Try to delete a non-existent product** | **Delete a product (edge status)** | **Product Management** |
| |  | | --- | | **Successfully add client, and update DataGridView to show the new client** |  |  | | --- | |  | | |  | | --- | | **Enter customer name, address, and contact details** |  |  | | --- | |  | | |  | | --- | | **Add a new customer** |  |  | | --- | |  | | **Customer Management** |
| |  | | --- | | **Display an error message and prevent client from being added** |  |  | | --- | |  | | |  | | --- | | **Leave the Name field blank** |  |  | | --- | |  | | |  | | --- | | **Add a new client (edge status)** |  |  | | --- | |  | | **Customer Management** |
| |  | | --- | | **Successfully update customer information** |  |  | | --- | |  | | |  | | --- | | **Edit existing customer details** |  |  | | --- | |  | | |  | | --- | | **Update customer information** |  |  | | --- | |  | | |  | | --- | | **Customer Management** |  |  | | --- | |  | |
| |  | | --- | | **Display an error message and prevent updating** |  |  | | --- | |  | | |  | | --- | | **Trying to update without providing an ID** |  |  | | --- | |  | | |  | | --- | | **Update customer information (edge status)** |  |  | | --- | |  | | |  | | --- | | **Customer Management** |  |  | | --- | |  | |
| |  | | --- | | **Delete and remove the client from the DataGridView** |  |  | | --- | |  | | |  | | --- | | **Provide a valid customer ID for deletion** |  |  | | --- | |  | | |  | | --- | | **Delete a client** |  |  | | --- | |  | | |  | | --- | | **Customer Management** |  |  | | --- | |  | |
| |  | | --- | | **Display an error message indicating that the customer does not exist with the specified identification number** |  |  | | --- | |  | | |  | | --- | | **Try to delete a client that doesn't exist** |  |  | | --- | |  | | |  | | --- | | **Delete a client (edge status)** |  |  | | --- | |  | | **Customer Management** |
| |  | | --- | | **Send a notification to all subscribers** |  |  | | --- | |  | | |  | | --- | | **Add a new product** |  |  | | --- | |  | | |  | | --- | | **Send notifications when a new product is added** |  |  | | --- | |  | | |  | | --- | | **Notification System** |  |  | | --- | |  | |
| |  | | --- | | **No errors, no notifications** |  |  | | --- | |  | | |  | | --- | | **Add a new product without subscribers** |  |  | | --- | |  | | |  | | --- | | **Send notifications when a new product is added (edge status)** |  |  | | --- | |  | | |  | | --- | | **Notification System** |  |  | | --- | |  | |
| |  | | --- | | **Send notifications to all subscribers after the update** |  |  | | --- | |  | | |  | | --- | | **Update an existing product** |  |  | | --- | |  | | |  | | --- | | **Product update and send notifications** |  |  | | --- | |  | | |  | | --- | | **Notification System** |  |  | | --- | |  | |

**7.0 Control Procedures:**

* **Reporting Issues:** All issues discovered during testing will be recorded with severity details, reproduction steps, and responsible team members. Each issue will be followed up until it is resolved.
* **Change requests:** A formal approval process will be followed for impact assessment and approval by project leaders.

**8.0 Features to be tested:**

* **Input module:** Data validation, including product information and customer orders.
* **Implementation unit:** Processing efficiency, including inventory updates and sales reporting generation.
* **Output unit:** accuracy of notifications and analytical reports.

**9.0 Features that will not be tested:**

* **Non-critical features:** Features that are not critical to key functions or are scheduled to be updated in the future will not be tested at this stage.

**10.0 Tools:**

* **NUnit:** For unit tests and integrations within **Visual Studio 2022**.
* **JMeter:** For performance and pressure tolerance tests, especially under high load conditions.

**Final Evaluation:**

This document clearly and elaborates the test plan for the application, including all types of tests required, tools used, and timeline. The translation is identical to the original, and is suitable for answering the section on tests.