Test Strategy

SyncStream

Date: 13-02-25

Group Members:

Muhammad Anas (SE-21020)

Bakhtiar Ahmed (SE-21029)

Ammar Ahmed (SE-21033)

Daniyal Lodhi (SE-21041)

Fawad Tariq (SE-21050)

1. Introduction

SyncStream is a powerful Integration Platform as a Service (iPaaS) that enables seamless data integration and synchronization across multiple platforms, including HubSpot, Shopify, and Salesforce. Designed for flexibility, it allows users to define their own APIs and platform details, making it adaptable to various business needs. SyncStream supports real-time, scheduled, and one-time data transfers, ensuring smooth and automated data exchange. With a user-friendly dashboard, businesses can efficiently monitor and manage their integrations, optimizing workflows for both local and global operations.

1.1 Scope

This document defines the testing strategy for the SyncStream, designed to integrate and synchronize data across platforms. The goal is to ensure the platform meets functional, security, and performance requirements, including accurate data transfer, real-time synchronization, and scalability.

The testing process spans unit, integration, and system-level validation. Key focus areas include verifying data accuracy during transfers, testing real-time sync capabilities across heterogeneous platforms, validating security measures (e.g., encryption, authentication), and assessing performance under high-load conditions.

Who will review the document?

- **Reviewers**: The document will be reviewed by the team **Supervisor** (internal reviewer) and **Project Evaluators** (external reviewers).

Who will approve this document?

- **Approvers**: The team **Supervisor** will provide the final approval for the document. The supervisor's approval is essential to ensure that the testing process aligns with the project's goals.

Software Testing Activities:

- The testing activities will be carried out in a structured and phased approach, consisting of the following stages:
    - **Unit Testing (1-2 weeks)**: This phase will focus on testing individual components and modules (e.g: Authentication Module, API Modules). Each part will be tested independently to ensure it functions as expected before integration.
    - **Integration Testing (1 week)**: During this phase, different components will be integrated (e.g: API Modules with frontend), and their interaction will be tested to ensure the platform can handle cross-platform synchronization.

- **System Testing (1 week)**: This phase will validate the entire system ensuring the platform works seamlessly as an Integration Platform, ensuring error-free real-time and scheduled data synchronization between platforms.

- **Security Testing (3-4 days)**: This phase will include security testing to safeguard sensitive business data through penetration testing, vulnerability assessments, and API security audits. Authentication, encryption, and role-based access control mechanisms will be validated to ensure compliance with industry security standards.

- **Performance Testing (3-4 days)**: This phase will conduct performance testing to evaluate system scalability, response times, and reliability under high-load conditions. Simulated data transfers will identify potential bottlenecks and optimize SyncStream's efficiency, ensuring smooth operations even during peak usage.

- **User Acceptance Testing (1 week)**: This phase will involve user acceptance testing to ensure SyncStream meets real-world integration needs. End-users and business stakeholders will validate custom API configurations, data synchronization accuracy, and platform usability, ensuring a seamless experience. Feedback will be incorporated to refine workflows and enhance user satisfaction.

These activities are expected to be carried out in a controlled testing environment, using a combination of manual and automated testing methods and tools. The testing team will execute various test cases to cover functionality, performance, security, and edge cases.

2. Standards

N/A

3. System Test Methodology

3.1 Process of Testing

The testing process follows a well-defined sequence of activities to ensure a thorough evaluation of the SyncStream platform from development to production. The work items performed at each stage of testing are outlined below:

- **Test Planning:** This phase involves the preparation of the overall testing strategy, creating detailed test cases, and setting up the test environment.

- **Test Environment Setup:** Ensuring that the required platforms (Shopify, Salesforce, HubSpot, etc.) are configured properly and that the SyncStream environment is in place for testing.

- **Tool Setup:** Configuring tools like **Postman** for API testing and **pytest-django** for backend testing, etc.

- **Test Execution:** During this phase, predefined test cases are executed, and the outcomes are recorded. Any deviations or unexpected behaviors are logged as defects. Continuous collaboration with developers ensures timely resolution of issues. A comprehensive Test Execution Report is created to monitor progress and highlight potential risks.

- **Reporting & Feedback:** After each phase, results are gathered and analyzed. Any issues, bugs, or discrepancies will be logged in the bug-tracking system for review and resolution. A report detailing the results, bug counts, and potential risks will be compiled.

- **Final Acceptance:** Once all issues are addressed, and testing is complete, a final review will be conducted. This will include validating that all exit criteria are met, and the product is ready for release.

3.2 Testing Levels

The SyncStream platform will undergo the following testing levels to ensure complete coverage:

- **Unit Testing:**
    - **Scope:** This testing level focuses on the individual components to ensure they function correctly in isolation.
    - **Justification:** Unit testing is necessary to catch bugs at the earliest possible stage. It helps ensure that each function and method behaves as expected before integrating them into larger workflows.

- **Integration Testing:**
    - **Scope:** Focuses on testing the interaction between different modules and external platforms.
    - **Justification:** It is crucial to verify that all the integrated components (APIs, data transformations, field mappings, etc.) work together seamlessly. Since the platform deals with data synchronization across different platforms, this level of testing ensures that data flows correctly between systems.

- **System Testing:**
    - **Scope:** Ensures the platform functions as a complete product. This includes testing all functionalities, from user-defined API configurations to real-time data sync and dashboard generation.
    - **Justification:** It is important to validate the platform as a whole, ensuring that all integrated features work together and that the system meets user expectations.
- **Acceptance Testing:**
    - **Scope:** Performed after system testing to ensure the platform meets business requirements and user needs.
    - **Justification:** Acceptance testing ensures that the platform is ready for deployment and will meet the needs of the end users and stakeholders.

3.3 Types of Testing

- **Load Testing:**
    - **Scope:** This type of testing evaluates how the platform performs under various levels of load, such as multiple concurrent API calls or large volumes of data transfer.
    - **Justification:** Load testing is necessary to ensure that the platform can handle high traffic and large datasets, especially in real-time synchronization scenarios. This helps identify performance bottlenecks and ensures scalability.
- **Data Integrity Testing:**
    - **Scope:** This testing will ensure that data remains accurate, consistent, and reliable during synchronization across platforms. It includes verifying data mapping, preventing duplication or loss, and ensuring rollback mechanisms function correctly in case of failures.
    - **Justification:** Since SyncStream facilitates real-time and scheduled data transfers, maintaining data integrity is critical to prevent inconsistencies, incomplete records, or synchronization errors that could disrupt business operations.

- **Webhook Validation Testing:**
    - **Scope: T**his testing will focus on validating the reliability of webhook-based synchronization. It includes testing automated webhook registrations, event-driven triggers, delayed and failed webhook scenarios, and retry mechanisms to ensure real-time updates function correctly.
    - **Justification:** Webhooks play a key role in real-time synchronization within SyncStream. Ensuring their accuracy and responsiveness is essential to prevent data loss, ensure timely updates, and maintain seamless communication between integrated platforms.
- **Compatibility Testing:**
    - **Scope:** This testing will ensure that SyncStream functions correctly across different environments, including various operating systems, browsers, and API versions. It includes testing integrations with third-party platforms like Shopify, Salesforce, and HubSpot under different configurations.
    - **Justification:** SyncStream must support a wide range of platforms (such as browsers) and API versions. Compatibility testing ensures seamless integration across diverse environments, preventing unexpected issues due to version mismatches or platform-specific behaviors.
- **Regression Testing:**
    - **Scope:** This testing will verify that new updates or feature enhancements do not introduce defects into existing functionalities. It includes re-running previously passed test cases to ensure that core features like data synchronization, authentication, and API communication remain unaffected by changes.
    - **Justification:** Frequent updates and improvements are necessary for SyncStream. Regression testing ensures that these updates do not break existing functionality, maintaining system stability and reliability over time.

4. Features to be Tested

The following features will undergo testing:

- Connections with HubSpot, Shopify, Salesforce, and user-defined platforms.
- Field mappings between source and destination platforms.
- API definition

- API calls and their responses.
- Data flows for real-time, scheduled, and one-time transfers.
- Data transformation processes.
- Dashboard creation and API integration.
- Real-time synchronization capabilities.

5. Features Not to be Tested

All system features will be covered in the test phases defined above.

6. Configurations to be Tested and Excluded

6.1. Test Environment

- **Testing Device Configuration:**

  Processor: Intel Core i5 (5th Generation)
  RAM: 16 GB
  Storage: 256 GB SSD
  Operating System: Windows 10

- **Platforms Setup:** Platforms such as Shopify, Salesforce and Hubspot must be configured properly to conduct the tests.
- **Backup and Restore Strategy:** Backup of test data and restore strategies will be defined once the test environment is set up.

7. Testing Tools

- **Postman**: For API testing.
- **Pytest-django**: For backend testing.
- **Ngrok (Free Version):** For webhook validation testing
- **Locust:** For load testing

8. System Test Entry and Exit Criteria

8.1 Entry Criteria

Before system testing begins, the following must be satisfied:

- Core functionalities, including data synchronization, should be implemented and ready for testing.
- All required APIs and webhook configurations must be set up and accessible.
- The application must be successfully deployed in a test environment.
- Test cases and scenarios for system testing should be documented and approved.

8.2 Exit Criteria

The following criteria must be met before concluding the system test:

- All system test cases must be executed.
- Performance and security testing benchmarks are met.
- The system successfully meets business requirements and expected functionality in all test scenarios.

9. Test Deliverables

The following outputs will be produced during the testing process:

- Automated test scripts in Pytest-django and Postman for API validation and system testing.
- Test logs, including detailed steps, results, and bugs.
- Test plans and test cases covering unit, integration, system, security, and performance testing scenarios.

10. Risk Analysis

10.1 Potential Risks

- Updates in third-party APIs (e.g., Shopify, Salesforce) may break existing integrations.
- Missed or delayed webhooks may affect real-time synchronization.
- Differences in data formats or structures may cause incorrect synchronization.
- Large-scale data transfers may degrade performance under high load.

10.2 Mitigation Plan

- Ensure regular API updates are monitored, and the system is updated to match the latest API versions.
- Implement automatic retries and fallback mechanisms for failed webhooks.
- Apply standardized data transformation rules to prevent inconsistencies.
- Conduct periodic load testing and optimize database queries to handle peak loads.

10.3 Contingency Plan

- Update APIs and perform regression testing to ensure compatibility.
- Maintain backward compatibility or quick rollbacks for API updates.
- Provide logging and manual sync options for data correction.
- Deploy auto-scaling cloud infrastructure if performance degrades under high traffic.