Task 13.

**Project ‘A’.**  Duration of 1 year. The scope of the project is the migration of the Data Warehouse from the on-premises system into the cloud. The testing team is planned to be 1 QA. The Dev team consists of 3 persons. The main activities are migration and setup of the Dashboards with the same logic as it was on the on-premises Database. No automation testing framework is available and the tech stack on the project is mostly Java-based + with some services which are new to you.

**Is It Necessary to Set Up Test Automation for This SUT?**

In the context of Project A, the implementation of test automation is crucial due to the limited resources available. With only one QA and a small development team of three persons, manually verifying each phase of the data warehouse migration could quickly become time-consuming, repetitive, and prone to human error. Automation will help alleviate the burden on the QA and provide a scalable approach to ensure that the dashboards and the data migration maintain accuracy and functionality.

Additionally, this project involves working with Java-based services, some of which are new to me. Automated testing frameworks can help streamline the learning process by enabling consistent validation across the unfamiliar services and platforms. Given the timeline of one year, automation is also key to ensuring long-term efficiency as the project evolves.

**What Should Be Automated and Why?**

1. The core of this project is migrating the data warehouse. Automating tests for the ETL (Extract, Transform, Load) processes will ensure that data is correctly transferred, transformed, and loaded into the cloud environment. This is crucial to ensure data integrity and consistency across environments.
2. Automating tests to verify the integrity and completeness of the data before and after migration ensures that no data is lost or corrupted during the transition from on-premises to the cloud.
3. Given that the dashboards must retain the same logic as in the previous system, it’s essential to automate the testing of key functionalities. Automating UI tests using tools like Selenium will allow the QA to verify that data is displayed correctly and the dashboard behaves as expected after each change.
4. Automation is critical for running regression tests regularly. This ensures that new changes do not negatively affect the existing system, helping maintain stability throughout the project lifecycle.

**How to Implement Automation?**

* Since the project is primarily Java-based, the test automation framework should be developed using Java, leveraging widely-used libraries such as Selenium for UI testing and TestNG or JUnit for test execution and reporting.
* The framework should be designed with modularity in mind, allowing the addition of automated tests for different services and components as the project evolves.
* Setting up a CI pipeline) is necessary to automatically trigger tests after each build. This will ensure that issues are identified as early as possible.
* Data migration projects often involve large volumes of data. Developing scripts to generate and validate test data automatically will reduce the time spent on manual preparation and analysis.

**Required Processes and Practices**

1. : Although there is only one QA, the development team needs to be involved in the automation process, especially in areas where the tech stack is new. Collaboration between developers and QA will ensure the tests are accurate and aligned with the implementation.
2. As the project progresses, automated tests will need to be maintained. Establishing a process for updating tests when there are changes in the dashboard logic or ETL processes will be critical to avoiding outdated or broken tests.
3. If the tech stack is new , it will be important to dedicate time to learning the specific tools and libraries involved in building automation frameworks for Java and cloud-based data warehouses.

**Additional Questions to Consider**

* Project Requirements for Testing Tools
* Programming Language Used
* Existing Tests
* Team Involvement
* Test Team Size

Task 1

Assuming that the QA manually tests the system for an average of 10 hours per week throughout the project's 1-year duration.

CM = 10 hours/week \* 52 weeks = 520 man-hours.

**Framework Implementation**

Developing a new automation framework is expected to take approximately 40 man-hours, considering that the framework is relatively simple and intended for Java-based services.

**Scenario Creation**

For 6 months, the QA will spend around 20 hours per month creating and refining automated test scenarios for the ETL, dashboards, and data integrity.

S = 20 hours/month \* 6 months = 120 man-hours.

**Test Execution**

Executing the automated tests is estimated to take 4 hours per week for the entire year, as the tests will run after each build or major release.

E = 4 hours/week \* 52 weeks = 208 man-hours.

**Results Analysis**

Analyzing the test results, fixing potential issues, and maintaining the test environment will take around 2 hours per week for the entire year.

R = 2 hours/week \* 52 weeks = 104 man-hours.

**Total Investment into Automation**

I=FW+S+E+R=40+120+208+104=472 man-hours

**ROI Calculation**

Now, we calculate the ROI using the formula:

ROI=(CM−I)/I=(520−472)/472=0.10 so 10 %

Conclusion

The ROI for Project A is 10%, indicating that the investment in automation will begin to pay off relatively early in the project's lifecycle.

Task 2.

I used the DQE\_LAB\_2024 framework code as a foundation for the PyTest + Selenium tests.

Integrated unit and integration tests into the framework using fixtures for setup and teardown.

Added two checks for the Microsoft Income Statements Power BI report

