**Task 1 – Test Automation Strategy & ROI Calculation**

Chosen Project: Project B

1. Is test automation necessary? Why?

Yes, test automation is necessary for Project B due to its long duration (3 years), daily ETL processing, and complex logic in dashboards. With regular data extraction and transformation, along with dashboards that contain complex calculations, the need for regression testing and repeatability is critical. Automation ensures faster feedback cycles and consistency in validation across daily data pipelines.

1. What should/could be automated? Why? How?

* ETL Pipeline Validation

Automate validation checks to confirm data consistency post-extraction and transformation. This includes row counts, null checks, key joins, and data accuracy validations. Python + pytest with libraries like pandas and sqlalchemy can be used.

* Dashboard/Report Verifications

Automate UI checks for the dashboards: field presence, chart rendering, logic validation (e.g., totals, averages). Selenium-based tests can validate these elements against expected data or snapshots.

* Smoke and Regression Tests

Identify and automate key flows to catch issues early during daily or release deployments.

* API or Data Access Layer Checks

If the project exposes endpoints or uses services for data loading, API testing with requests and pytest can be effective.

1. Processes and Practices to Set Up

* Build a PyTest-based automation framework with modular design.
* Integrate Selenium for dashboard UI testing.
* Set up configurations, conftest.py, and data fixtures.
* Add CI/CD integration for scheduled test runs (e.g., daily ETL validation).
* Store results in Allure for transparent reporting.
* Design reusable test utilities for data comparison and snapshot testing.

1. Project-specific Context

* Programming Language: Python (aligned with automation toolset).
* Existing Automation: None available yet
* Participants: 2 QA engineers – one technical, one manual.
* Ownership: The technical QA will lead automation efforts; manual QA can contribute using BDD (optional, Gherkin syntax).
* Project Stack: Python-based, ETL tools, cloud data storage, and dashboards.

1. ROI Calculation

Assumptions:

Manual testing: ~16 hours/week (8 hrs/week each QA), for 3 years.

TAF Implementation: 80 hours (basic PyTest + Selenium framework).

Test scenario creation: 30 hrs/week \* 16 weeks = 480 hours.

Test execution + result review: 4 hrs/week over 3 years.

Cost of Manual Testing (CM):

= 16 hours/week \* 52 weeks/year \* 3 years

= 2,496 hours

Investment (I) = FW + S + E + R

FW (Framework Implementation) = 80 hours

S (Scenario Creation) = 30 hrs/week \* 16 weeks = 480 hours

E (Execution) = 2 hrs/week \* 52 weeks \* 3 years = 312 hours

R (Results Analysis) = 2 hrs/week \* 52 weeks \* 3 years = 312 hours

I = 80 + 480 + 312 + 312 = 1,184 hours

ROI = (CM – I) / I = (2496 – 1184) / 1184 = 1.11 or 111%

1. Conclusion

Test automation for Project B provides a clear ROI of 111%, showing that the time and effort invested will be paid back and more over the course of 3 years. Given the daily ETL, data complexity, and long-term nature of the project, setting up a Python-based automation framework is both a practical and strategic choice. It will help catch data issues early, reduce regression effort, and improve overall testing efficiency.