Mindset Challenge

1. **Identified a problem or inefficiency in a process related to quality or development, and implemented a change to improve it**

   Project Background- While I was working for UBank as a Senior Quality Analyst for a project called "Ledger, Strategy and Performance, Origination and Credit decisioning". It was a "BAU" project for the core-banking platform which is a shared asset between NAB and UBank. UBank participate with NAB in monthly and quarterly enterprise releases and continuously work on keeping the core-banking stack up-to date as per the business and compliance requirements in Australia.

   Problem Statement - Since core-banking platform was a shared asset and multiple teams were participating in the releases, I observed that we were encountering bugs in production for the functionalities that were successfully tested in lower environments.

   After investigation, I found that the functionalities that were tested successfully in the lower environments were failing due to the changes made by other teams participating in the releases and making changes to the assets which in-directly impacted UBank.

   How did I improve it?

   As there were no streamlined process/portal where all the underlying changes directly or in-directly impacting UBank were recorded, it was very difficult to track what could have caused an error in production. I came to know that NAB was running a Change meeting and UBank was not part of it.

   - The first and foremost thing was to have UBank representation in these change meetings to get a fair idea of what all changes are going to be made in the release.
   - Record the details of the changes being discussed in the change meeting for further analysis
   - Perform impact analysis on the changes that were impacting the shared assets and update UBank Regression suite based on the analysis

2. **Built or implemented new testing tools or frameworks**

   Project Details- When I had joined UBank Team, the regression suite for the release was not automated and was executed manually due to which –

   - Time taken to execute the regression suite was tremendous and hence it slowed down the overall delivery.
   - Team has to re-run the suite manually every time even a small change was made.
   - Team was not able to add additional scenarios in suite to enhance the BAU coverage and hence there was always a risk of defect leakage.

   What's my role –

   - It was my initial days in the project, I ran 1-2 releases manually along with my product owner to understand the functionality and business criticality for each scenario.
   - Since most of my team members are business and have a little understanding of technology, I also made sure the framework is simple and easy to use.

Automation and Tooling Challenges –

- Implementation Cost – As the regression was in BAU mode so we haven't had enough fund to automate all the regression scenarios (150 Test cases). So had challenge to complete all tasks in approved budget (25% of overall ask) like – Proof of Concept, Tool Implementation, Automation and Maintenance
- Tooling – The tool must be simple so that everyone in the team can use it for regression as and when needed.
- Less maintenance – there shouldn't be much future maintenance cost to modify scenarios, test data management and addition of new scenario.

Solution –

- Considering the challenges and limitations, I had implemented the Cypress – E2E Test Automation tool with BDD.
- Cypress was chosen because of its simplicity and light weight implementation.
- I have implemented Cypress framework for Automation challenge where I have added enough details about why cypress and its benefits.
- Overall, we were able to achieve the expected outcome, and the automation was successfully implemented with given budget. The automation is still working like a charm, which helped business to reduce Release regression execution time by 50%.

3. **Made changes to prevent defects from occurring or improve the quality before reaching the testing stage**

- Defect reoccurring / Defects in late stages of test lifecycle are the big problems in Project Lifecyle, which has impacts on Project timelines and cost.
- In one of my assignment in UBANK, I observed that every release the integration defect counts were high. When I did deep dive into the defect data then observed that most of the issues (70-80%) are related to B to C (API to Downstream) data mapping, data transfer and data conversion.
- With modern tooling with shift left implementation, these B to C defects can be easily identified early in System Test. To get more idea why its not happening in my project, I have interviewed few of the developers and came across below pain points –
    - Tooling challenge
    - Missing quality mindset
    - Test data
- To address these issues, I did my analysis and came up with below mitigation plan which helped us to enable integration quality checks at System test level.
    - Tooling challenge – POSTMAN (one of the most simple and powerful tool) was available for ST, its just dev team was not comfortable with the usage. Mostly dev was doing manual testing which is kind of less interest for them. To address this, I have standardized the POSTMAN collection with – adequate test coverage scenarios, standard naming convention (env file) and mature reusable JavaScript assertion.
    - Missing quality mindset – The dev's had an impression that Quality is someone else accountability and it doesn't have any implication on them. But when I conducted Quality Guild and presented the defects stats with 70-80 integration defects leading to ($ XXXXX) amount along with delays in delivery then

developers were bit amazed but still thought its usual thing in all project. But when I explained them the benefits of –

- How easily we can identify these defects in ST using simple tools like postman
- How it basically reduces their significant amount of effort in Integration Cycle (mostly defect fix)

  - Test data - I reached out to test data management team and we came to an agreement with them to update the database with the data required by the new functionality implementation during the development phase which is well before the integration test phase starts.

- The project team liked the idea and decided to implement a pilot in 2 scrum team, and at the end of release we measured the outcome w.r.t number of defects. The outcome was phenomenal as the B to C defects got reduced by 70%.

4. **Built or implemented CLI processes and/or test management reporting integration into the test workflow.**
   - External execution results from either automated or manual Tests reporting is important in Test Workflow. If this feature is not available, then it's a manual effort to report test execution summary from CI pipeline / Automation / manual back to test tool which is significant effort.
   - I had faced this issue in JIRA tool where all the Agile and Test Management activities were tracked. I observed that QA team was using manual approach to update the test execution result. I did some research and found out a simple JIRA Plugin – XRAY.
   - XRAY provides –
     - Manual result upload, using the Import Execution Results action
     - the REST API to integrate with Continuous Integration (CI) platforms or other external execution processes
   - When I proposed the XRAY plugin along with benefits, the team has received it positively. And we (as team) implemented this capability using XRAY RestApi – Junit XML Output format.

5. **Attempted to convince someone to change the way they were working, even though they didn't want to.**
   - I personally believe that quality is not a responsibility of one person in a team. However, each member of the team is responsible to drive the overall quality of the project/application.
   - However, building this quality mindset among the team is a challenge as each member of the team carries a different opinion about the quality and think of it as an additional accountability.
   - This misconception about the quality and fear of additional accountability can be resolved by knowledge sharing and creating a transparent environment by taking few actions as below –
     - Conduct knowledge sharing sessions/ quality guilds for the project team to educate them about how quality can be embedded at each stage just by introducing few checks.

    - Conduct few 1:1 interview with different team members to understand their pain-points and challenges/fears considering quality.
    - Identify major 3-4 problem statements that is common across the team.
    - Perform analysis on how those problems can be addressed with simple techniques.
    - Explain the big picture of how this is going to help each member of the team in terms of saving efforts with better quality deliverables with the facts and figures.
- Considering the example in the 3rd question, I was able to convince the developers and project team by following above steps and improved the overall quality of the system.

6. **When you are testing a new feature, under what circumstances would you deviate from a test script while performing manual testing?**
    - Deviation in Design vs Actual Implementation

      Usually Test Script / Cases are written in parallel with development by referring the design document or UX design. And if there is any deviation in Design vs Actual implementation then Test Scripts won't help to drive the end to end scenarios. In this case, I have to conduct exploratory test, and get the design changes approved from Designer / BAs, accordingly, will modify the test script. But most of the test will already get completed before the test script updating.

    - Unknown Defects in Regression Components after New feature implementation
        - There are instances when we come across unknown regression defects which might come up while conducting progression test for new feature. In this case, we have to conduct the exploratory test across the impacted functionality to identify any unknown bugs. Mostly, the exploratory test happens first then some test script gets added to track the defects or as a reference for future release regression.

7. **If you joined our team and could implement and change any process(es) you'd like, how would you ensure that the team delivers quality software?**
    - Change in Work Model / Mindset –
        - First thing is to change the Agile working Model where everyone has federated responsibility rather than having common role. I will talk to Product Owner and propose the new work model, where – No Developer or Tester role instead everyone would be Team Member. This will clear the expectations in every team member mind.
        - Quality Engineering Session –
        - It is important to explain the importance of Quality to team members rather than teaching how to test the system. This is because if team member understands why we require quality? And how to achieve quality easily then they will take more interest in Quality.
        - For example, I can showcase –
            - how easily a test can be conducted on day to day by using different tools?
            - how it helps to reduce defect resolution time?
            - how it improves efficiency?
    - Introduction of Quality Check at early stages in Software Delivery –

- Test Driven Development can be implemented to make developers identify test and get it reviewed from QE before development starts.

OR

- Introduction of BDD where BA, QE and Developer works on common agenda to define the requirement in Gherkin language. This helps to avoid any understanding gaps or ambiguity in requirement while building the system.

- Introduce Quality Gates at Feature / User Story Level Acceptance Criteria –
  - Once the developer completes build and test then the review will happen against acceptance criteria where QE can play a role of gatekeeper / governance role. This means no sperate test is require pushing the build in prod.