1.Testing in an Agile Framework

i) General Testing Concepts and Process

Agile Testing: Continuous testing throughout the development lifecycle, with a focus on collaboration and flexibility.

QA Team Process:

1. Planning: Participate in sprint planning to understand requirements and define testing scope.
2. Design: Create test plans and test cases based on user stories and acceptance criteria.
3. Execution: Perform various types of testing (functional, integration, regression, etc.) during and after development.
4. Reporting: Document and report defects, providing clear reproduction steps and screenshots.
5. Review: Continuously review and update test cases based on feedback and changes in requirements.

Detailed Testing Steps

Preliminary Design Phase

Testing the Initial Design Status:

- Review initial design documents and prototypes.
- Conduct exploratory testing to identify potential issues early.
- Document findings and share with the design team.

Remarks for the designer:

- Provide detailed feedback on how the design addresses or fails to address current issues.
- Highlight any gaps or inconsistencies.
- Suggest improvements based on user experience and technical feasibility.

Begin Considering E2E Test Cases:

- Begin drafting end-to-end test cases that cover the entire user journey.
- Ensure test cases are comprehensive and cover all critical paths.

Prior to or During Development

- Add Test Scenarios to Tickets:
- Create detailed test scenarios and add them to the relevant development tickets.
- Include clear acceptance criteria.

Record Test Outcomes in Tickets:

- Document pass/fail results for each test scenario.
- Ensure fails have clear screenshots, reproduction steps, and expected outcomes.

Following Development Work

- Test Selected Scenarios in Integration Environment:
- Ensure all components work together as expected.

Pre-Production Regression Testing:

- Perform regression testing to ensure new changes do not affect existing functionality.
- Check previous app builds with this release to ensure backward compatibility.
- Verify that new features are working on web, iOS, and Android platforms.
- Run automated regression tests to confirm existing features are unaffected.

Post-Production

- Verify Deployment of New Features:
    - Verify that the new functionality is live and working as expected.
- Verify Production Sanity Checks:
    - Conduct basic sanity checks to ensure the production environment is stable.

Best Practices be followed by the QA team

- Collaboration: Foster close communication between developers, testers, and stakeholders.
- Early Testing: Start testing early in the development process to catch issues sooner.
- Test Automation: Invest in automation for repetitive and regression testing to speed up feedback.
- Frequent Feedback: Gather and act on feedback from stakeholders and end-users regularly.
- Emphasize the iterative nature of Agile testing and the need for flexibility and continuous improvement.
- Signoff Checklist: Use a signoff checklist to ensure all necessary steps are completed before deployment.

ii)When should test cases be written, and who should see them?

- When: Test cases should be written during the planning phase, before development begins.
- Who: Test cases should be visible to the entire development team, including developers, designers, and product owners, to ensure alignment and transparency.

2. How do you perform regression testing ?

- Identify Scope: Determine which features or functionalities might be affected by recent changes or new features.
- Select Test Cases: Choose relevant test cases from the test suite that cover the impacted areas.
- Automate When Possible: Use automated test scripts to run regression tests quickly and frequently.
- Manual Testing: Perform manual regression tests for complex or critical functionalities not covered by automation.
- Run and Verify: Execute the tests and compare results with expected outcomes. Document any deviations or issues.
- Manual Regression Suite Maintenance : Periodically review and update test cases to ensure they align with the current application features and business requirements. Identify and remove test cases that are no longer relevant due to deprecated features or changes in application functionality.Incorporate new test

cases to cover newly introduced features or bug fixes.Group related test cases into logical test suites to improve manageability and execution efficiency

i)Execution of Test Cases

Overview :

1.Preparation

- Test Environment Setup: Ensure the test environment is configured correctly with the necessary hardware, software, and network settings.
- Test Data Preparation: Gather and prepare the required test data.

2.Execution

- Follow Test Cases: Execute each test case step-by-step as documented.
- Record Results: Document the outcome of each test case, noting whether it passed or failed.
- Capture Evidence: Take screenshots or videos to capture the results, especially for failed test cases.

3.Reporting

- Log Defects: For any failed test cases, log defects in the defect tracking system with detailed reproduction steps, expected results, and actual results.
- Communicate Findings: Share test results and defects with the development team and stakeholders.

4.Review and Retest

- Review Defects: Work with the development team to review and prioritize defects.
- Retest: Once defects are fixed, retest the affected areas to ensure the issues are resolved.

5.Regression Testing

- Impact Analysis: Identify areas that might be affected by recent changes and include them in regression testing.
- Execute Regression Tests: Perform regression tests to ensure that new changes have not adversely affected existing functionalities.

6.Final Reporting

- Summary Report: Provide a summary report of the testing activities, including the number of test cases executed, passed, failed, and any critical defects found.
- Sign-Off: Obtain sign-off from stakeholders to confirm that the testing phase is complete and the product is ready for the next stage

Platforms: Conducted on windows, iOS, and Android platforms.
Devices: Use real devices with approved OS versions to ensure accuracy and reliability.
Data Syncing: Confirm syncing of data between all types of devices to ensure consistency and integrity.

ii)Reporting Test Results

1. Documentation: Document pass/fail results for each test case.
2. Detailed Reports: Provide a summary report of the testing activities, including the number of test cases executed, passed, failed, and any critical defects found.
3. Communication: Share test results with the development team and stakeholders for review and action.

iii)Prioritizing Test Cases

- Critical Functionality: Prioritize test cases that cover the most critical functionalities of the application.
- High-Risk Areas: Focus on areas that have undergone recent changes or have a history of issues.
- User Impact: Consider the impact on the end-user experience when prioritizing test cases.
- Frequency of Use: Give priority to features and functionalities that are frequently used by users.

3.How do you decide what test cases to write?

Phase-Based Approach

This approach helps ensure that testing is thorough and aligned with the development phase, providing both immediate and long-term quality assurance

1.Functional Testing

Focus: Edge cases and different combinations of scenarios.

Attributes:

- High volume of test cases.
- Mostly one-off test cases tailored to specific user stories or features.

Objective: Ensure comprehensive coverage of new functionalities and identify potential issues early.

2.Regression Testing

Identify areas that might be affected by recent changes and include them in regression testing

Focus: End-to-End (E2E) test cases.

Attributes:

- Aim to automate these test cases as early as possible.
- Designed to be run frequently to ensure stability of the application.

Objective: Verify that existing functionalities remain unaffected by new changes and ensure overall system integrity.

i) What Should Be Included in Test Cases

- **Test Case ID:** A unique identifier for each test case.
- **Title:** A clear and concise title that summarizes the test case.

Example: “Login Functionality with Valid Credentials”

- **Description:** Clear and concise description of what the test case is intended to validate.
- **Pre-conditions:** Any setup or conditions required before executing the test case.
- **Test Steps:** Detailed steps to perform the test.
- **Expected Results:** The expected outcome for each step or the overall test case.
- **Actual Results:** The observed outcome after executing the test case.
- **Status:** The pass/fail status of the test case.
- **Comments:** Any additional notes or observations relevant to the test case execution.
- **Attachments**: Screenshots or other evidence supporting the test results.