Unit Testing & JUnit

1. *What is unit testing?*
   It is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software. A unit is the smallest testable part of any software.

2. *What is the difference between manual testing and automated testing?*
   In manual testing, a testing engineer manually executes test cases. This means the testing engineer describes several scenarios and edge cases that they want to verify the functioning of. Manual testing is suited for usability testing, ad-hoc testing, and exploratory testing. Automation testing is the process in which testers utilize tools and scripts to automate testing efforts. Automated testing is good for regression testing, performance testing, load testing, and highly repeatable functional test cases.

3. *Is it necessary to write the test case for every logic? If yes, why*

   Not necessary but it is recommended due to its advantages. Some of the advantages include
   - It makes the coding process more agile
   - Better quality of the code.
   - Early detection of software bugs
   - Helps with maintaining and changing the code.
   - Provides documentation of the system.
   - Simplify the debugging process.
   - Helps in creating better designs.
   - Reduces the cost of bug fixes.

4. *What are the features of JUnit?*

   - JUnit is an open source framework, which is used for writing and running tests.
   - Provides annotations to identify test methods.
   - JUnit tests allow you to write codes faster, which increases quality.
   - Unit is elegantly simple. It is less complex and takes less time.
   - JUnit tests can be run automatically and they check their own results and provide immediate feedback
   - Provides test runners for running tests.

5. *What are the important JUnit annotations? And its usage in coding*
   - @Test – This annotation denotes that a method is a test method.
   - @BeforeClass – Run once before any of the test methods in the class
   - @AfterClass – Run once after all the tests in the class have been run
   - @Before – used on a method containing Java code to run before each test case. i.e it runs before each test execution.

- @After – used on a method containing java code to run after each test case. These methods will run even if any exceptions are thrown in the test case or in the case of assertion failures.

6. *What does Assert class?*

7. *What is Code Coverage?*
   Code coverage is a measurement of how many lines/blocks/arcs of your code are executed while the automated tests are running.

8. *What are the best practices to perform Unit Testing?*

   - Should Be Trustworthy
   - Should Be Maintainable and Readable
   - Should Verify a Single-Use Case
   - Should Be Isolated meaning the test should not be able to influence other tests.

9. *What is Mocking?*
   Mock testing is an approach to unit testing that lets you make assertions about how the code under test is interacting with other system modules.

<u>GIT</u>

1. *What is GIT and its significance in SDLC*
   Git is a distributed version-control system for tracking changes in source code during software development. It is important in software development lifecycle because it facilitates coordination among programmers working on a project.

2. *What is the difference between GIT and SVN?*
   Git is a distributed version control system, whereas SVN is a centralized version control system.

3. *What are the advantages of using GIT?*
   Git is a speedy and efficient distributed VCS tool that can handle projects of any size, from small to very large ones. Git provides cheap local branching, convenient staging areas, and multiple workflows. It is free, open-source software that lowers the cost because developers can use Git without paying money.

4. *What is "Staging Area" or "Index" in GIT?*
   The staging area is the set of files that have been added to the index, but not committed.

5. *What is GIT stash?*
   Git stash temporarily stores changes you've made to your working copy so you can work on something else, and then come back and re-apply them later on.

6. *What is the function of git clone?*

   Git clone is primarily used to point to an existing repo and make a copy of that repo at a new directory, at another location.

7. *How can you create a repository in Git?*
   - Create a directory to contain the project.
   - Go into the new directory.
   - Type git init.
   - Write code.
   - Type git add to add the files
   - Type git commit.

8. *What is the purpose of branching in GIT?*
   Branching divides the code into a separate, completely independent units, where each unit is potentially the entire application, and allows for testing and debugging without affecting the main code.

9. *What is the difference between 'git remote' and 'git clone'?*
   Git remote adds a reference to a remote repository for further tracking, while git clone, clones locally a remote repository.

10. *What is the function of 'git diff ' in git?*
    Git diff command is used in git to track the difference between the changes made on a file.

11. *Explain what the commit message is?*
    The commit command is used to save changes to a local repository after staging in Git.

12. *What is Rebasing?*
    Rebasing is changing the base of your branch from one commit to another making it appear as if you'd created your branch from a different commit.

Maven Fundamentals

1. *Explain what is Maven? How does it work?*
   Maven is a powerful project management tool that is based on POM. Maven helps the developer to create a java-based project more easily. Maven uses a declarative approach, where the project structure and contents are described.

2. *Explain what is POM and its significance*
   A Project Object Model (POM) is the fundamental unit of work in Maven. It is an XML file that contains information about the project and configuration details used by Maven to build the project. When executing a task or goal, Maven looks for the POM in the current directory. It reads the POM, gets the needed configuration information, then executes the goal.

3. *Explain what a Maven artifact is?*
   Maven artifact is the resulting output of the maven build, generally a jar or war or other executable file. Artifacts in maven are identified by a coordinate system of groupId, artifactId, and version. Maven uses the groupId, artifactId, and version to identify dependencies needed to build and run your code

4. *List out the dependency scope in Maven?*
   Maven dependency scope attribute is used to specify the visibility of a dependency. The 6 dependency scopes in Maven are Compile, Provided, Runtime, Test, System and Import.

5. *List out what are the build phases in Maven?*
   The build phases in Maven are validate, compile, test, package, verify, install and deploy

6. *Mention the three build lifecycle of Maven?*
   - default: handles project build and deployment
   - clean: handles project cleaning
   - site: handles the creation of project site documentation

7. *List out what are the aspects does Maven Manages?*

   - Build.
   - Documentation.
   - Reporting.
   - Dependencies.
   - SCMs.
   - Releases.
   - Distribution.
   - Mailing list.

8. *Explain what a Maven Repository is? What are their types?*

   Maven repository is a directory where all the project jars, library jar, plugins or any other project specific artifacts are stored and can be used by Maven easily. Maven repository are of three types: local, central and remote

9. *Explain how you can exclude dependency?*
   By adding an <exclusions> element in the <dependency> element.

10. *For POM what are the minimum required elements?*
    The minimum required elements for POM are project root, modelVersion, groupID, artifactID and version.

CI/CD

1. *What are the fundamental differences between DevOps & Agile?*
   DevOps is a practice of bringing development and operations teams together whereas Agile is an iterative approach that focuses on collaboration, customer feedback and small rapid releases. DevOps focuses on constant testing and delivery while the Agile process focuses on constant changes.

2. *What is the need for DevOps?*
   DevOps' allows enterprises to create and improve products at a faster pace than traditional software development methods.

3. *What are the advantages of DevOps?*

   - Shorter Development Cycles, Faster Innovation
   - Reduce Implementation Failure, Reflections and Recovery Time
   - Better Communication and Cooperation
   - Greater Competencies
   - Reduce Costs and IT Staff

4. *Explain with a use case where DevOps can be used in industry/ real-life.*
   Using the DevOps, employees helped car manufacturers to catch the error while scaling the production, which was not possible before. Key Bank used DevOps to reduce the time taken for the integration of security and compliance into the process from 3 months to 1 week. With the benefit of DevOps, United Airlines saved $500,000 by changing to continuous testing standards.

5. *What are the differences between continuous integration, continuous delivery, and continuous deployment?*

   Continuous integration merges their changes back to the main branch as often as possible. Continuous delivery on the other hand is an extension of continuous integration to make sure that you can release new changes to your customers quickly in a sustainable way. Continuous deployment goes one step further than continuous delivery and makes sure every change that passes all stages of your production pipeline is released to your customers

6. *What role does the Quality Assurance (QA) team play in DevOps?*

   - QA aims to detect a bug at the earliest point in the cycle and also prevent potential bugs from reappearing in the production environment.
   - QA takes ownership of the improvement process in DevOps.
   - QA make sure that all environments required for testing are standardized and the deployments are automated.
   - They look for any opportunity to improve the quality of the product.

7. *Describe an efficient workflow for continuous integration*

- Developer checks out code
- Developer commits changes into the source repository.
- The CI server monitors the source repository and when it detects a change it triggers a build of the relevant sources.
- After a successful build, a build label will be been to the source code that was used for the build, meaning that the build could be recreated if necessary.
- In the event of a build failure, the CI server sends notifications to the relevant developers who restart the process from step 1 to make the changes necessary to resolve the build errors.
- After the unit and integration testing has taken place, the relevant team members are notified of the test results.

8. *What are the best practices for DevOps implementation?*

- Test Automation
- Integrated Configuration Management
- Integrated Change Management
- Continuous Integration
- Continuous Delivery
- Continuous Deployment
- Application Monitoring
- Automated Dashboards

9. *How will you approach when a project needs to implement DevOps?*

- Identify the need to implement the DevOps practice.
- Once decided, break the culture of organizational silos, and encourage collaboration among the teams.
- Bring the focus back to the customer and look to improve end-user satisfaction.
- Take small steps and then scale up.
- Automate whatever can be automated.
- Select compatible tools.
- Ensure real-time visibility into the projects.
- Integrate and deliver continuously.
- Implement robust monitoring and feedback systems.