SUNVERA QA OFFLINE TASK:

1. Describe your experience in automating functional tests for web applications using Selenium and Page Object Model.

   As per my experience, POM (Page Object Model) framework is using in the software testing. POM will helps to Maintain the code reusability and readability.

   In test automation the web pages / components are separated by classes.

   * Install JDK and set up the IDE (Eclipse)

   * Download the Webdriver executable file for the browser to automate.

   * Create a new Project in java.

   * Add all the needed Selenium webdriver and java dependencies with the help of Maven

   * Create a separate class for each webpage /components which represent the same.

   *In UI, the Web elements and related methods are Encapsulated within the classes.

   * Writing test scripts for different scenarios and functionalities, initiate page objects to perform actions and assertions.

   * TestNG is used to run and execute the test classes.

   * Generating detailed test execution reports which includes passed and failed test cases, execution time and more.

   * Jenkins, tools are used to integrate the automated testcase into CI/CD pipe line.

a. How would you structure your test scripts?

   *Page class – Create a separate class for each webpage /components which represent the same.

   Eg: Pages/

   - LoginPage.java
   - DashboardPage.java
   - CensusPage.java

*Test class: - Writing test scripts for different scenarios and functionalities, initiate page objects to perform actions and assertions.

   Eg: - tests/

   - LoginTest.java

- DashboardTest.java
- CensusTest.java

*Resources: - It includes the config properties files like drivers executable files.

* Baseclass: - Reusable method are common functions that is being used in pageclass.

*Utils class: - All the common methods which has being used by entire frame work, like reading configuration properties, handling waits, or reading test data.

* All input data which has being used in the framework in testdata class.

b. What tools and libraries would you use to manage test data and generate reports?

Test management data:

We can manage the test data with built – in data provider mechanism in TestNG and also we can use the @DataProvider annotation to manage data in separate class

We can manage through external data source like Excel sheets, CSV files,JSON files etc..

We can use Configuration Property files – class to read and manage the data.

Report Generation:
Report generation can be one with TestNG basic HTML Reports by default, Extent reports are widely used interactive HTML reports and Allure is another popular reporting framework that provides user friendly interface.

4. Describe your experience in using TestNG and JUnit to execute automated tests for web applications.

As per my experience, TestNG is a powerful testing framework for Java that simplifies the process of writing and executing automated tests, including tests for web applications.

Why TestNG is used:

Numerous test cases can be organized with ease by translating them into the testng.xml file
Sets priorities on test case execution
A test case can be executed at different times using the keyword invocation Count by assigning an iteration number to invocation Count variable
Supports cross-browser testing and supports tools like Maven, Jenkins etc.
Unlike WebDriver, TestNG has a built-in mechanism to generate reports in a readable format

Setup:
 Create a Java project and add TestNG dependency.
Write test classes with @Test methods.

Annotations:
Use annotations (@BeforeMethod, @AfterMethod, etc.) for setup and teardown.
Annotate test methods with @Test.

Dependency Management:
Use dependsOnMethods to specify test method dependencies.

Parallel Execution:
Enable parallel execution in testng.xml for faster tests.

Grouping:
Group related tests using @Test(groups = "groupName").
Execute specific groups via testng.xml.

Parameterized Tests:
Employ @DataProvider to supply data to test methods.

Test Configuration:
Define test configuration in testng.xml.

Reporting:
TestNG provides default HTML reports.
Enhance reports using listeners or third-party frameworks like ExtentReports.

a. **How do you structure your test classes and methods**

   Structure of Test Classes:

   *Application modules /features will reflecting by organizing the test classes within the packages.

   *Naming the test classes with the naming convention, which indicates the functionality they are testing.

   *Beginning of the class, Import the needed packages and libraries.

   *We can use the Annotations in TestNG like @Test, @BeforeSuite,@AfterSuite,@BeforeTest,@AfterTest,@BeforeClass,@AfterClass,@BeforeMethod,@AfterMethod.

   *Variables like Webdriver instances or Page Object instances will declare in class level. If needed we can create constructor to initialize resources.

Structure of Test Method:

*Naming the test Methods with the naming convention, which indicates the scenario and actions being tested.
*Naming convention eg: loginTest
*@Test annotation should mark in each Test methods if we working in TestNG framework.

* Each Test Method focused on a specific scenario and Functionality

* Do necessary cleanup process at the end of each test method.

b. What annotations do you use to manage test dependencies, data providers, and reporting?

*@Test annotation should mark in each Test methods as a test case. “dependsOnMethods” attribute used to manage the test Dependencies.

* We can use the Annotations in TestNG like @Test, @BeforeSuite,@AfterSuite,@BeforeTest,@AfterTest,@BeforeClass,@AfterClass,@BeforeMethod,@AfterMethod.

* @DataProvider annotates a method providing the test data to the test methods.

* It returns a two-dimensional array of objects, where each inner array represents a set of test data.

* For the test execution behavior and reporting in testNG we will use “@Listeners” annotations with “ITestListener” interface.