Section 1:

Different Flavours of Selenium

1.Theory Task:

- Write a short note (150–200 words) explaining the four main flavours of Selenium and give one real-life usage example for each.
- Compare Selenium IDE vs Selenium WebDriver in a table.

2. Research Task:

- Find two cloud platforms that use RemoteWebDriver (e.g., BrowserStack, Sauce Labs).
- Write how they help in cross-browser testing

Answers:

1. Theory Task

Four Main Flavours of Selenium

Selenium is a powerful suite of tools for automating web browsers. It comes in four main flavours, each serving a distinct purpose.

1. Selenium IDE

A browser extension (Chrome/Firefox) for recording and playing back tests.
Ideal for beginners and quick prototyping.
Example: A QA tester records login functionality for a banking site without writing code.

2. Selenium RC (Remote Control)

- Legacy tool that allowed test execution via a server using multiple languages.
- Now deprecated in favor of WebDriver.
- **Example**: Automating tests for a legacy e-commerce site using Java and Selenium RC.

3.Selenium WebDriver

- The most widely used flavour.
- Offers direct browser control via programming languages like Java, Python, and C#.
- **Example**: Automating checkout flow in an online store using Java WebDriver.

4.Selenium Grid

- Enables parallel test execution across multiple machines and browsers.
- Ideal for large-scale cross-browser testing.
- **Example**: Running tests on Chrome, Firefox, and Safari simultaneously for a travel booking site.

Criteria	**Selenium IDE**	**Selenium WebDriver**
Usage	Record & Playback tool	API for browser automation

Coding Required	No programming needed	Requires programming knowledge
Browser	Chrome, Firefox	All major browsers
Test Complexity	For simple, quick tests	Handles complex scenarios
Fiexibility	Limited	Highly fexible and extensible
Integration	Standalone tool	Integrates with test frameworks
Export Capabilities	Can export to code	Native code based scripting

2.Research Task

Cloud Platforms Using RemoteWebDriver

1. BrowserStack

- Offers real device cloud testing across 3000+ browser-device combinations.
- Uses RemoteWebDriver to run automated Selenium tests on remote machines.
- Cross-Browser Testing Benefits:
- Real-time testing on actual devices.
- Supports parallel execution.
- Provides screenshots, logs, and video recordings for debugging.
- Seamless integration with CI/CD tools.

2. Sauce Labs

- Provides access to 700+ browser/OS combinations and 300+ real devices.
- Uses RemoteWebDriver for distributed test execution.

- Cross-Browser Testing Benefits:
- Live and automated testing across browsers.
- Supports Sauce Connect Proxy for testing behind firewalls.
- Offers visual testing, error reporting, and performance analytics.

Section 2:

WebDriver Introduction & Architecture

3.Flow Diagram Task:

o Draw the WebDriver Architecture showing interaction between Test Script → Client Library → Browser Driver → Browser.

o Label each step.

4.Explaining Task:

o In your own words, explain what happens internally when driver.get("https://example.com") is executed.

- **Task-3**

- ## Interaction Flow

[Test Script]

|

v

[Client Library (Selenium WebDriver)]

|

v

[Browser Driver (e.g., chromedriver.exe)]

|

v

[Browser (e.g., Chrome)]

- ## Step-By-Step

1.Test Script

- Automation code written using Selenium APIs in Java, Python, etc**.**

2. **Client Library**
    - Converts commands into JSON over HTTP using WebDriver protocol.
3. **Browser Driver**
    - Receives HTTP requests and translates them into native browser commands.
    - **Examples:** chromedriver, geckodriver, msedgedriver.
4. **Browser**
    - Executes actions like opening URLs, clicking elements, etc., and sends responses back**.**

- **Task-4**

- **InternalExecutionofdriver.get("https://example.com")**
  - **Step-by-Step Explanation**

1. **Test Script Execution**
   - driver.get("https://example.com") is called in the automation script.
2. **Client Library Translation**
   - Selenium WebDriver converts the command into a JSON-formatted HTTP request**.**
3. **Request Sent to Browser Driver**
   - The request is sent to the browser driver via a local server (e.g., localhost:9515).
4. **Browser Driver Parses Request**
   - The driver interprets the command and uses the browser's native automation protocol to communicate**.**
5. **Browser Opens the URL**
   - The browser launches and navigates to https://example.com.
6. **Response Propagation**
   - relays it to the client library and finally to the test script. The browser sends a response back to the driver, which

---

Section 3:

Installation & Configuration

5.Setup Task:

- Install JDK, Eclipse, and Selenium JARs.
- Download ChromeDriver and configure it in a Java project.
- Take a screenshot of Eclipse with the Selenium JARs added in the Build Path.

6.Driver Manager Task:

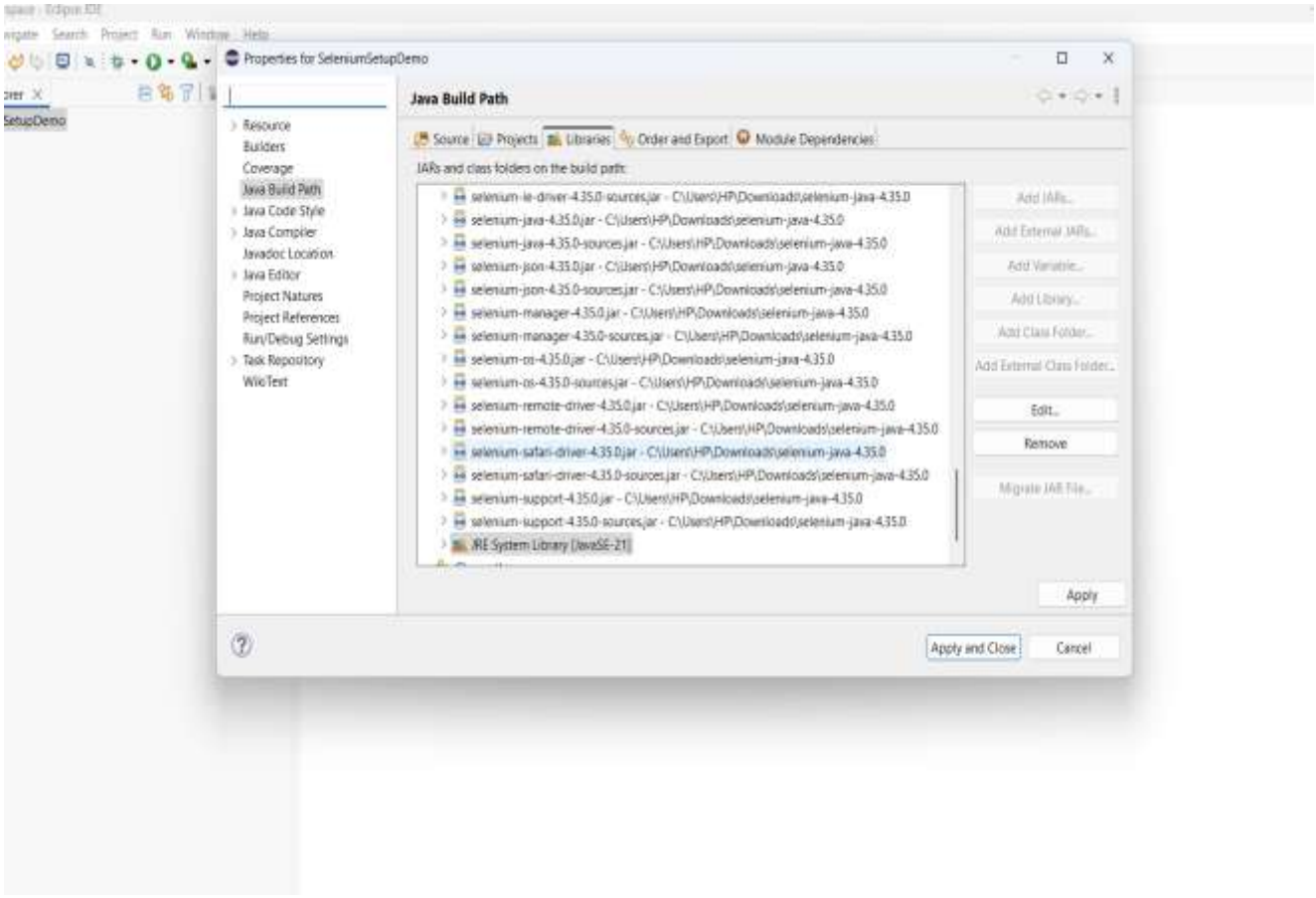
- Use WebDriverManager (instead of manually downloading drivers) in a Maven project to launch Chrome.

5. Setup Task

Setup Steps:

1. Install the latest Java Development Kit (JDK) and set JAVA_HOME.
2. Install Eclipse IDE for Java development.
3. Download Selenium Java Client JARs from the official site.
4. In Eclipse, create a new Java project and add the Selenium JARs to the project's Build Path.

5. Download ChromeDriver and set its location using System.setProperty or put it in your system's PATH.



6.Driver Manager Task

- **Using WebDriverManager in a Mave Project:**
- **pom.xml:**

<dependency>

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenium-java</artifactId>

```
 <version>4.25.0</version>
</dependency>
<dependency>
 <groupId>io.github.bonigarcia</groupId>
 <artifactId>webdrivermanager</artifactId>
 <version>5.9.2</version>
</dependency>
```

JAVA CODE:

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import io.github.bonigarcia.wdm.WebDriverManager;

public class LaunchChrome
 {
   public static void main(String[] args)
{
      WebDriverManager.chromedriver().setup();
      WebDriver driver = new ChromeDriver();
      driver.get("https://www.google.com");
      System.out.println("Title: " + driver.getTitle());
      driver.quit();
```

```
    }
}
```

Section 4:

Creating First Test Script

7. **Hello Selenium Task:**
   - **Write a script that opens https://google.com, searches for "Selenium WebDriver", and prints the number of search results.**
8. **Title Validation Task:**
   - **Write a script that opens https://opensource-demo.orangehrmlive.com/, logs in with admin credentials, and verifies if the page title after login is "OrangeHRM"**

7.Hello Selenium Task:

```
import org.openqa.selenium.*;
import org.openqa.selenium.chrome.ChromeDriver;
import io.github.bonigarcia.wdm.WebDriverManager;
import java.util.List;
```

```
public class GoogleSearch
 {
   public static void main(String[] args)
{
      WebDriverManager.chromedriver().setup();
      WebDriver driver = new ChromeDriver();
      driver.get("https://www.google.com");
      WebElement searchBox =
driver.findElement(By.name("q"));
      searchBox.sendKeys("Selenium WebDriver");
      searchBox.submit();
      List<WebElement> results =
driver.findElements(By.cssSelector("div.g"));
      System.out.println("Results on first page: " +
results.size());
      driver.quit();
   }
}
```

8.Title Validation Task:

```
import org.openqa.selenium.*;
import org.openqa.selenium.chrome.ChromeDriver;
import io.github.bonigarcia.wdm.WebDriverManager;

public class LoginTest
{
   public static void main(String[] args)
{
      WebDriverManager.chromedriver().setup();
      WebDriver driver = new ChromeDriver();
      driver.get("https://opensource-
demo.orangehrmlive.com/");

driver.findElement(By.id("txtUsername")).sendKeys("Admin");

driver.findElement(By.id("txtPassword")).sendKeys("admin123
");
      driver.findElement(By.id("btnLogin")).click();
      String title = driver.getTitle();
      if(title.equals("OrangeHRM"))
{
         System.out.println("Login successful! Title verified.");
  }
```

```
Else
{
        System.out.println("Title verification failed!");
 }
     driver.quit();
   }
}
```

Section 5: Locators Practice

Using the OrangeHRM Demo site:

URL: https://opensource-demo.orangehrmlive.com/

9. By ID

- Locate the username and password fields by id and log in.
- JAVA
- driver.findElement(By.id("txtUsername")).sendKeys("Admin");
  driver.findElement(By.id("txtPassword")).sendKeys("admin123");
  driver.findElement(By.id("btnLogin")).click();

10. By Name

- Locate the Login button by its class name and click it.
- JAVA
- **driver.findElement(By.name("txtPassword")).sendKeys("admin123");**
  driver.findElement(By.name("txtUsername")).sendKeys("Admin");

11. By Class Name

- Locate the Login button by its class name and click it.
- JAVA
- driver.findElement(By.className("button")).click();

12. By Tag Name

- Find and print all hyperlinks on the dashboard.
- JAVA
- List<WebElement>links=driver.findElements(By.tagName("a"));
  for(WebElement link : links)
  {
  System.out.println(link.getText());
  }

13. By XPath

- Locate the Admin menu by XPath and click it.
- Use both absolute and relative XPath.

- JAVA
- driver.findElement(By.xpath("/html/body/div[1]/div/ul/li[1]/a")).click();

14. By CSS Selector

- Locate the search box in the Admin tab using CSS selector and type “Admin”.
- JAVA
- driver.findElement(By.cssSelector("input#searchSystemUser_userName")).sendKeys("Admin");

  driver.findElement(By.cssSelector("input#searchSystemUser_userName")).sendKeys("Admin");

Section 6: Combined Task

15. End-to-End Automation Task:

- Open https://opensource-demo.orangehrmlive.com/
- Log in using valid credentials.
- Navigate to the PIM section.
- Add a new employee (first name, last name, upload photo).
- Verify the employee appears in the list.

- Log out.

JAVA

```
import org.openqa.selenium.*;
import org.openqa.selenium.chrome.ChromeDriver;
import io.github.bonigarcia.wdm.WebDriverManager;

public class OrangeHRMEndToEnd
{
   public static void main(String[] args) throws
InterruptedException
{
      WebDriverManager.chromedriver().setup();
      WebDriver driver = new ChromeDriver();
      driver.get("https://opensource-demo.orangehrmlive.com/");
      driver.findElement(By.id("txtUsername")).sendKeys("Admin");

driver.findElement(By.id("txtPassword")).sendKeys("admin123");
      driver.findElement(By.id("btnLogin")).click();
 driver.findElement(By.id("menu_pim_viewPimModule")).click();
      driver.findElement(By.id("btnAdd")).click();
       driver.findElement(By.id("firstName")).sendKeys("John");
      driver.findElement(By.id("lastName")).sendKeys("Smith");
```

```
driver.findElement(By.id("photofile")).sendKeys("C:\\path\\to\\photo.jpg");
      driver.findElement(By.id("btnSave")).click();
driver.findElement(By.id("menu_pim_viewEmployeeList")).click();

driver.findElement(By.id("empsearch_employee_name_empName")).sendKeys("John Smith");
      driver.findElement(By.id("searchBtn")).click();
      String result = driver.findElement(By.xpath("//a[contains(text(),'John')]")).getText();
      System.out.println("Employee Added: " + result);
      driver.findElement(By.id("welcome")).click();
      Thread.sleep(2000);
      driver.findElement(By.linkText("Logout")).click();
      driver.quit();
   }
}
```