**PROJECT OVERVIEW**

Our project is a Pathology Lab Management web application that helps manage various aspects of a pathology lab. The application features the following key components:

1. Login Page:

URL: https://gor-pathology.web.app/

Credentials:

- Username: test@kennect.io

- Password: Qwerty@1234

2. Home Page:

After successful login, the user will land on the home page, where they can view a list of todos and access the Cost Calculator for blood tests.

3. Cost Calculator for Blood Test:

On the home page, there is a cost calculator feature that calculates the cost of various blood tests. The user can select the desired tests and apply discounts if applicable.

4. Adding Patients and Creating Tests:

The application allows the user to add patient details and fill up a form to create test requests for patients. After adding a test, it will be reflected in the list of todos on the home page.

Preferred Testing Tool:

While we prefer the usage of Cypress for this assignment, we understand that you may have experience with other testing tools as well. You are free to choose a testing tool that you are most comfortable with, as long as it effectively tests the functionalities of the application.

**HYBRID AUTOMATION FRAMEWORK**

The project is developed using Hybrid Automation Framework, which includes **Java, Selenium, TestNG, Maven & Page Object Model.**

**I have created the project in Eclipse Maven Project and implemented TestNG features with Selenium.**

I initialized the Chrome Browser using Selenium Webdriver. It means creating a reference variable (driver) of the interface (WebDriver) and creating an Object. Here WebDriver is an Interface and ChromeDriver is a class **Data Abstraction** Java OOPS concept.

I have used **Page Object Model** classes, we declare the data members using @FindBy and initialization of data members will be done using Constructor to utilize those in methods which utilized the **Encapsulation** and **Data Abstraction** Java OOPS concept.

I have also applied **Inheritance** Java OOPs concept by creating a Base Class and Base Page to initialize the WebDriver interface, browsers, waits, reports, etc.

I have **used Polymorphism** Java OOPs concept as well in which I used Method Overriding by using different WebDriver methods and Method Overloading by using Assertion and Waits.

I have used **Properties** (congif.properites) file to provides data like username, password etc.

I have implemented **Reports** using ExtendReportUtility using **ITestListener** interface.

All the **Dependencies** and **Plugins** are included in **pom.xml**

**TestNG.xml** file includes all the test cases and setup of the execution.

**Test Case Details:**

I have developed four Test cases below are the details

|  |  |  |
| --- | --- | --- |
| **Sr No** | **Scenario** | **File name** |
| 1 | Verify user is able to login with valid credentials | **TC\_01\_LoginValidCreds** |
| 2 | Verify user is not able to login with invalid credentials | **TC\_02\_LoginInvalidCreds** |
| 3 | Verify user is able to create Test | **TC\_03\_AddTest** |
| 4 | Verify user is able to viewTest | **TC\_04\_ViewTest** |

**How to run project**

**Using Eclipse**

Open Project -> TestNG.xml -> Run as TestNG Suite

**Using Maven**

Open Project -> pom.xml -> Run as Maven Test

If Maven is setup in local system

Open command prompt where is project is located -> mvn test

**Using Jenkins**

Create new Item as Maven Project, mentioned the Java Path, Git Path and Maven

Github link:

<https://github.com/Premdevi9091/GOR-Pathology-Lab-Management-web-application>

**Note: Unable to complete project because getting ‘Quota Exceeded’ while logging in.**