**Manual and Automation Testing Summary and Execution Guide**

**Project Overview**

The project is based on a **Pathology Lab Management** web application designed to manage lab operations such as logging in, calculating blood test costs, and managing patient data and test requests.  
**URL**: <https://gor-pathology.web.app>  
**Test Credentials**:

* **Username**: test@kennect.io
* **Password**: Qwerty@1234

**Test Automation Approach**

**Tool Used**

I used **Selenium WebDriver with Java**, integrated with **TestNG** and **Maven** for test execution and dependency management. Although Cypress was recommended, I chose Selenium due to prior experience and suitability for cross-browser UI testing.

**Framework Design**

* **Maven**: Project management and build tool.
* **TestNG**: Test framework for organizing and running tests.
* **Page Object Model (POM)**: To enhance reusability and maintainability.
* **ExtentReports**: (Optional) For better HTML-based test reporting.
* **WebDriverManager**: To manage browser drivers automatically.

**Test Coverage**

1. **Login Functionality**
   * Valid credentials login verification
   * Navigation to the home page post-login
2. **Dashboard Page Validation**
   * Verification of todos list
   * Navigation to cost calculator
3. **Blood Test Cost Calculator**
   * Selection of different test types
   * Application of discounts
   * Validation of final cost
4. **Patient Management**
   * Filling patient details
   * Creating new test requests
   * Verifying added tests in the home page list

**Running the Tests**

**Pre-requisites**

* Java JDK 17+
* Maven 3.6+
* Chrome Browser

**Steps to Run**

1. **Clone the Repository** (if applicable):

bash

CopyEdit

git clone <repo-url-https://github.com/Palde22/Framework.git>

1. **Install Dependencies**:

bash

CopyEdit

mvn clean install

1. **Execute Tests**:

bash

CopyEdit

mvn test

1. **View Reports** (Extent Reports):
   * Navigate to the Reports folder and open the generated HTML report.

**Challenges Faced**

1. **Dynamic UI Elements**:
   * Some UI web elements like the blood test lists were dynamically generated, requiring usage of advanced XPath and wait strategies to ensure stability.
2. **Authentication Waits**:
   * Post-login transitions took time; using implicit waits helped ensure the page was fully loaded before assertions.

**Conclusion**

This automation suite validates the core workflows of the Pathology Lab Management application (Login to Dashboard). It is structured for scalability, allowing future enhancements like data-driven testing, integration with CI/CD, or switching browsers for compatibility testing.