📄 Calculator Automation Project Documentation

✅ Project Overview

This project automates functional, regression, UI, and cross-browser testing for a calculator web application using **Java, Selenium, TestNG, and Extent Reports.** The project ensures that all key features of the calculator remain functional across different browsers and setups.

✅ Technology Stack

Tool/Framework	Purpose
Java	Programming Language
Selenium 4.21.0	Browser Automation
TestNG	Test Management & Execution
Extent Reports	Reporting
Maven	Dependency Management & Build Tool
WebDriverManager	Auto-Driver Management
Chrome, Edge, Firefox	Supported Browsers

✅ Test Strategy

🔹 Functional Testing

- Verifies basic calculator operations: addition, subtraction, multiplication, division.
- Validates UI component clickability and result display.

🔹 UI Component Testing

- Checks button clickability.
- Validates the display field and clear button functionality.

🔹 Regression Testing

- Ensures previous working features remain unaffected after code changes.

🔹 Integration & End-to-End Testing

- Tests full user workflows from input to result with multi-step operations.
- Verifies correct UI and backend behavior together.

🔹 Cross-Browser Testing

- Runs the same test suite in Chrome, Firefox, and Edge browsers using TestNG's parameterization.

✅ Test Cases Overview

Test Type	Example Test Cases
Functional	2 + 3 = 5, 7 − 4 = 3
UI	All buttons clickable, Clear button resets display
Regression	Validate addition after update, Clear button still functional
Integration	Full flow: 5 × 4 = 20 ➜ Clear ➜ 7 + 3 = 10
Cross-Browser	Runs all above tests in Chrome, Edge, Firefox

✅ Execution Process

🔹 Prerequisites

- Install Java, Maven, Chrome, Firefox, Edge browsers.
- Ensure `chromedriver`, `geckodriver`, and `edgedriver` are auto-managed via WebDriverManager.

🔹 How to Run Locally

1. Open terminal in project root.
2. Run:`mvn test`
3. This will automatically pick the TestNG suite file `Master.xml` and run all tests.
4. Reports are auto-generated in the `/reports` folder.

🔹 Cross-Browser Run

- Execute using:

```
mvn test -DsuiteXmlFile=CrossBrowser.xml
```

✔️ This will run tests in parallel across multiple browsers.

✅ Folder Structure

```
calculator_Automation/
 ├ src/test/java/
 |  ├ testBase/            # Base class for WebDriver setup
 |  ├ testCases/           # Test classes
 |  ├ utility/             # Extent Report and reusable utilities
 ├ reports/               # Generated Extent Reports
 ├ screenshots/           # Captured screenshots on failure
 ├ pom.xml                # Maven project file
 ├ Master.xml             # Main TestNG suite
 ├ CrossBrowser.xml       # Parallel browser suite
 ├ run.bat                # Windows batch file for execution
```

✅ Reporting

- Reports are automatically generated using Extent Reports.
- Screenshots are automatically captured on failure.
- Reports open automatically on test completion.

✅ Key Features

- Thread-safe parallel execution using `ThreadLocal<WebDriver>`.
- Cross-browser support with TestNG parameterization.
- Full screenshot integration on test failure.
- Robust Extent Report setup.
- Clean and maintainable Maven project structure.

✅ Future Enhancements

- API integration testing.
- CI/CD pipeline setup using jenkins.