**🧑‍💻 User Manual**

**📦 Prerequisites**

Ensure the following tools are installed before setting up and running the test automation framework:

| **Tool** | **Version** | **Notes** |
| --- | --- | --- |
| Java | 8 or higher | Required for running Selenium + TestNG |
| Maven | 3.6+ | Manages dependencies and project build lifecycle |
| Git | Latest | For cloning the repository |
| IDE | Eclipse (062025) | Recommended for working with Java projects |

💡 **Tip:** WebDriverManager automatically downloads the correct version of the WebDriver for the browser version installed on the system.

**🚀 Installation Steps**

1. **Clone the Repository**
2. git clone https://github.com/zzinin/ShaddyM.git
3. **Open in Eclipse IDE**
   * Choose **"Import as Maven Project"** when importing into Eclipse.
   * Let Maven resolve dependencies automatically from pom.xml.
4. **Update Configuration**
   * Modify resources/config.properties as needed:
   * baseUrl=[Restful-booker-platform demo](https://automationintesting.online/)
   * excelPath=./testData/Contact.xlsx
5. **Update Excel Test Data (if required)**
   * Modify testData/Contact.xlsx for data-driven testing.

**🧪 Test Execution (Eclipse IDE Only)**

All test executions must be initiated from the TestNG XML files directly within Eclipse.

**✅ Option 1: Run All Tests Sequentially**

* Right-click on testng.xml in the Project Explorer.
* Select:  
  **Run As > TestNG Suite**

This runs all test cases sequentially using default configuration.

**🔁 Option 2: Run Tests in Parallel Across Browsers**

* Right-click on parallelcrossbrowser.xml
* Select:  
  **Run As > TestNG Suite**

This executes tests in **parallel threads** using multiple browsers, based on the setup defined in the XML.

❗ **Important:** Command-line test execution is not supported or recommended. Always use Eclipse’s TestNG runner for proper environment and suite loading.

📄 **Test Reports**

* After execution, view the report at:
* /test-output/index.html

Open this file in your browser to see a detailed breakdown of test results.

**🏗️ Framework Structure Overview**

.

├── base/

│ └── BasePage.java # Shared WebDriver logic

├── testCases/

│ ├── TC001\_BookNowTest.java

│ ├── TC002\_SubmitMessageTest.java

│ ├── TC003\_NegativeSubmitMessageTest.java

│ ├── TC004\_NegativeBookNowTest.java

│ ├── TC005\_SubmitDDT.java

│ └── TC006\_HappyDaysTest.java

├── utilities/

│ ├── ExcelUtil.java

│ └── DataProviders.java

├── resources/

│ └── config.properties

├── testData/

│ └── Contact.xlsx

├── testng.xml # Main suite for sequential tests

├── parallelcrossbrowser.xml # Suite for parallel browser execution

└── pom.xml # Maven configuration

**✅ Key Benefits**

* **🔧 POM Architecture**: Easy to maintain and scale
* **♻️ Reusability**: Centralized logic for pages and test steps
* **📊 Excel Integration**: Data-driven testing with Apache POI
* **⚙️ Configuration-Driven**: Change URL, browser, and timeouts easily
* **🧪 Parallel Execution**: Supports multi-browser testing with TestNG
* **📈 Auto-Generated Reports**: HTML reports generated after each run
* **🔁 IDE-First Execution**: TestNG execution tightly integrated with Eclipse

**Disclaimer: I Referred AI tools, to present my test automation design concept in**

**user friendly manner**