Customised Framework for

        YOUFACTORS

***Introduction:***

*This framework takes into consideration testing in the Add Near Miss entry.*

* *Test Cases with frequently changing requirements*
* *Test cases executed on an ad-hoc basis.*
* *Test Cases for individual fields in the entry form*
* *Test cases for collaborating selenium and windows popup*
* *Test cases for corrective actions checkboxes which pop up after clicking a particular checkbox.*

Automation testing not only finds the errors, it also delivers valuable data and insights about failed test cases which help in improving the end product and solves many technical issues. Many of the tests here can be used for smoke, sanity and regression tests. The list below gives you an exact picture of the various tools used in bringing about this automation into reality.

***Tools used for structuring automation:***

* **Automation framework -** is built using JAVA programming language
* **Testing framework        -** TestNG and Extent reports.
* **Dependency manager -** is MAVEN this is chosen so we can download and manage the versions as dependencies.
* **Graphics GUI automation tool -** SIKULI
* **Version control repository system -** GITHUB
* **Build tool -** CI CD - JENKINS

***Workflow:***

A class is created for each page. Every page class has the following components

\*Creation of object repository using locators

\*Constructor

\*Methods for each operation in that page

Every Test class has the following components.

\*Instantiates the corresponding page class

\*Initialises the page objects

\*Invokes the methods created in the page class

As you can see, we create a page class for each page, locate the web elements present in that page using xpath or id whichever is best suited, store them under meaningful names. The framework focuses on the Add Near Miss Incidents Module. We can include all the other modules in a similar fashion present in our web application.

Various types of web elements are present in *Add Near Miss Page* including Input text, choose from date and time, opting from dropdown, multiple choice options, choosing image from the local system. As the number of elements increase, finding each element using a locator and initializing it becomes a cumbersome process, to avoid this *Page Factory* is used. Usage of the *Page Object Model* approach plus Page Factory simplifies code management and makes it more maintainable and reusable.

The focus is on automating the Add Near Miss page, where each web element is unique. Unlike regular input forms, this near miss register form is customised for client’s use - so this automation approach is based on the idea of checking each element separately, validating the form for mandatory fields and finally validating the entire form. The custom made validations ensures every web element is individually verified and also verified in unison with other elements as a whole near miss incident submission page. Functionality test for the Add Near Miss incidents is carried out successfully in this custom framework.

***Project Exploration:***

Drivers         - folder to hold all the different drivers like Chrome, Edge, Firefox

Reports        - reports generated by extent reports are saved here in this folder

Screenshots - All the failed test screenshots are saved here.

Main folder   -  pages  ,BasePackage,Utilities

 Test folder   - has a subfolder FreshNearMissTests which has test program for each page file that was created in the pages folder

Testng.xml  - is a test suite that runs all the test file

                      Variations  using listener class ,groups are also available

Testrunner - runs all the tests without using separate test Manager

Pom.xml    - has all the dependencies as jar files in it

Testdata    - stores screenshot of windows open text input box and Open button images which are used in open Image button verification using Sikuli

Utilities      - Configuration properties file holds reusable objects like URL, username, password

 BaseClass - for all the before class and After class activities

 Extent Manager - manages extent report version 5 Spark reporter

***Screenshot of project:***

Text

Description automatically generated with medium confidence

This project is a simple framework which has room for lot of changes. As our application slowly and steadily increases the use of POM and Page factory will help us leverage many of its benefits. Since a repository of all web elements that  are present in a page are clubbed together anywhere  we can use them.

Few options for report production are :

Using TestNG.xml which runs tests in parallel and produces testng reports

 TestNGUsingListeners.xml runs tests based on the listener class and interprets individual tests based on pass, fail or skipped.

TestNGUsingGroups.xml runs tests which are grouped as smoke tests,sanity tests and regression tests .

**Suggestions:**

1.Mark the necessary tests for sanity, smoke and regression tests and group them.

2.Maintain a schedule for these tests. Present focus is on Near miss incident report whose report appears in a few milliseconds.

3.Add new page classes and corresponding test classes for the remaining options in the Main menu.

4.Plug in the test cases into the extent report and produce colourful and meaningful reports for quick access to managers.

5.Extent reports when run will give the file path for the screenshots saved for failed tests so you can look up for the file quickly.

6. For handling windows alerts, if manual screenshots taken for coding purposes does not work, Sikuli itself can be used to take screenshots of windows popups and alerts.

7.Toast messages need to be validated.

Capture toast message Xpath using debugger.

8.Framework can be expanded for other modules as well.