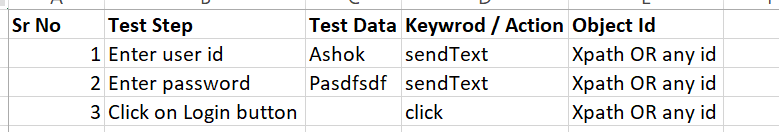
**Page Factory:**

Page Factory is a class provided by Selenium WebDriver to support Page Object Design patterns. In Page Factory, testers use @FindBy annotation. The initElements method is used to initialize web elements. Similarly, one can use @FindBy with different location strategies to find web elements and perform actions on them.

**Frameworks:**

1. Data Driven: here we put test data in Excel / CSV file and read data from this file and put it into your application.

2. Keyword Driven: Here we use Excel and in this file we use keyword and every keyword will have a function. Based on your keyword that particular function will executed



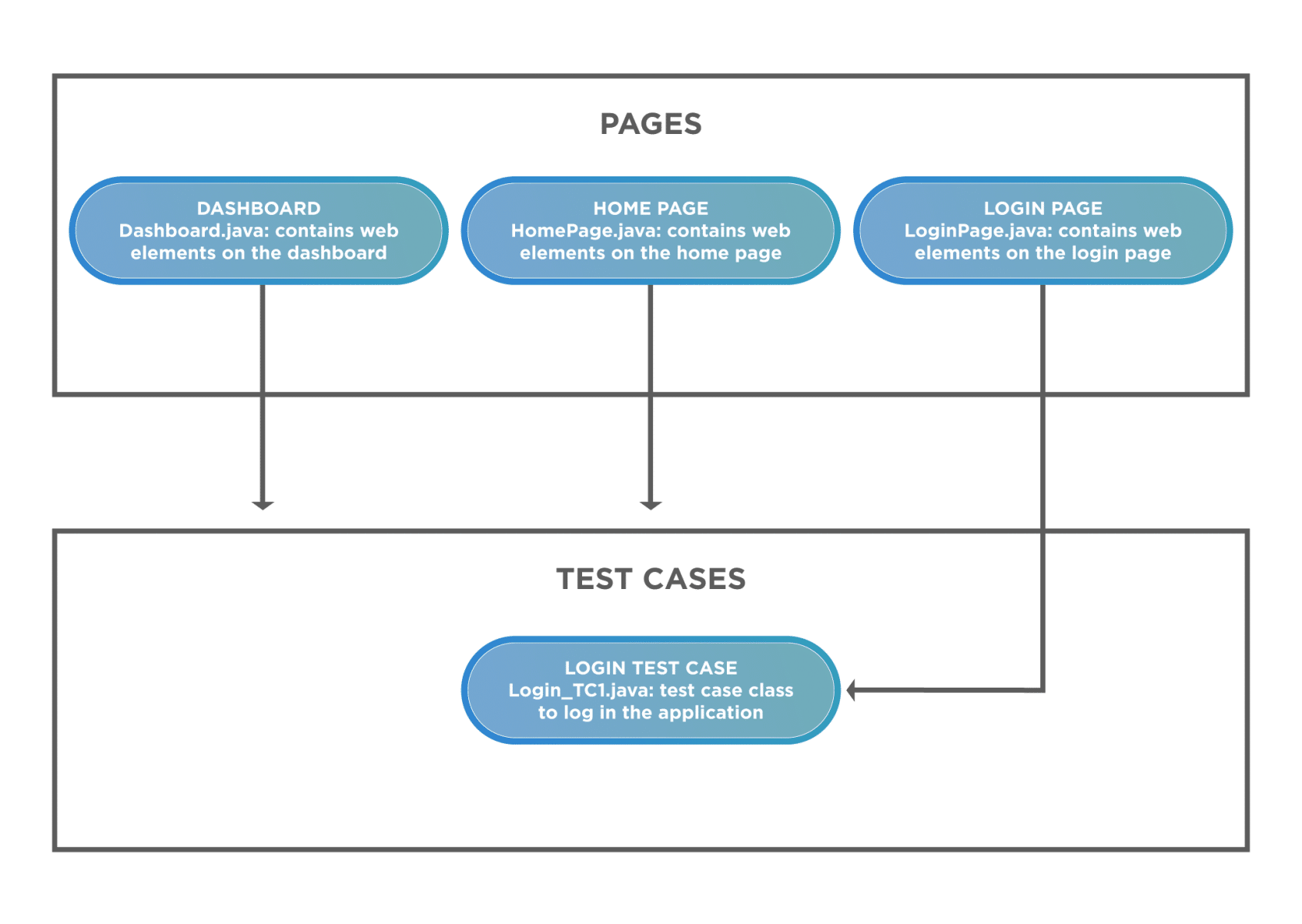
3. Page Object Model(POM)

4. Hybrid (whenever we implement more than one framework then we call it as a Hybrid framework)

**Page Object Model:**

**What is the Page Object Model(POM)?**

***Page Object Model or POM***  is a design pattern or a framework that we use in Selenium using which one can create an object repository of the different web elements across the application. To simplify, in the *Page Object Model framework,* we create a class file for each web page. This class file consists of different web elements present on the web page. Moreover, the test scripts then use these elements to perform different actions. Since each page's web elements are in a separate class file, the code becomes easy to maintain and reduces code duplicity. The below diagram shows a simple project structure implementing *POM-*



As you can see, we create different classes for the multiple pages and then save the web elements on the pages in them. Correspondingly, we save the test cases under a different package, making clear segregation among the different aspects.

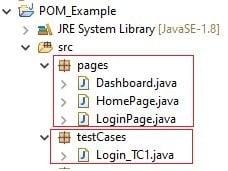
### ***What are the advantages of the Page Object Model?***

In the section above, we understood why a ***non-POM*** project could make our project difficult to maintain, making it a huge advantage of the *POM.* Below are some advantages of using the *Page Object Model* that highlights its efficiency:

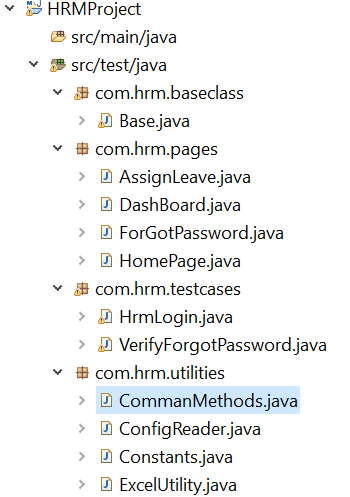
* ***Makes code maintainable-****Since the test classes are separate from the classes containing the web elements and the operation on them, updating the code is very easy if any web element updates or a new one adds.*
* ***Makes code readable-****User can easily read through the project and test scripts due to a fine line between the test classes and the different web pages.*
* ***And, makes code reusable-****If multiple test scripts use the same web elements, then we need not write code to handle the web element in every test script. Placing it in a separate page class makes it reusable by making it accessible by any test script.*

## *How to implement the Page Object Model in Selenium?*

After understanding the ***Page Object Model's importance***, we will now implement the *Page Object Model* for the use case that we considered above. Our *Page Object Model* project structure would look like below:



Page Object Model Architecture(you need to explain below architecture if asked question “what is your framework architecture)





***Property file***

. properties files are mainly used in Java programs to maintain project configuration data, database config or project settings, etc. You can easily read properties from some file using an object of type Properties.