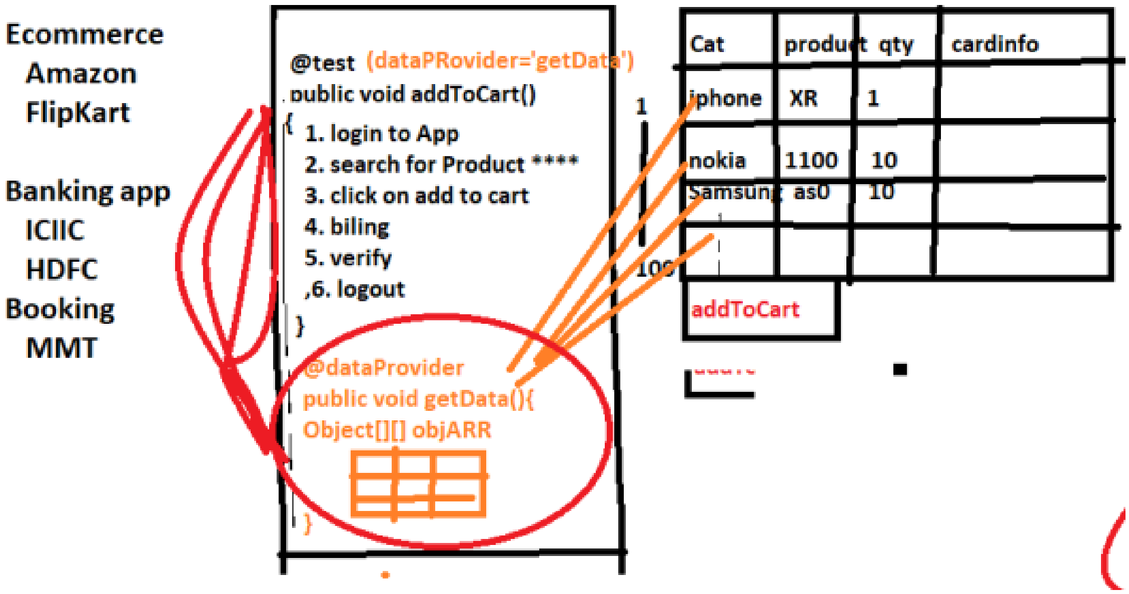
Types of Frameworks

1. Data Driven Framework
2. Modular Driven Framework
3. Method Driven Framework
4. Keyword Driven Framework
5. Test Driven Development [TDD] → TestNG
6. Hybrid Driven Framework
7. Behavior Driven Development [BDD] → Cucumber

Data driven Framework



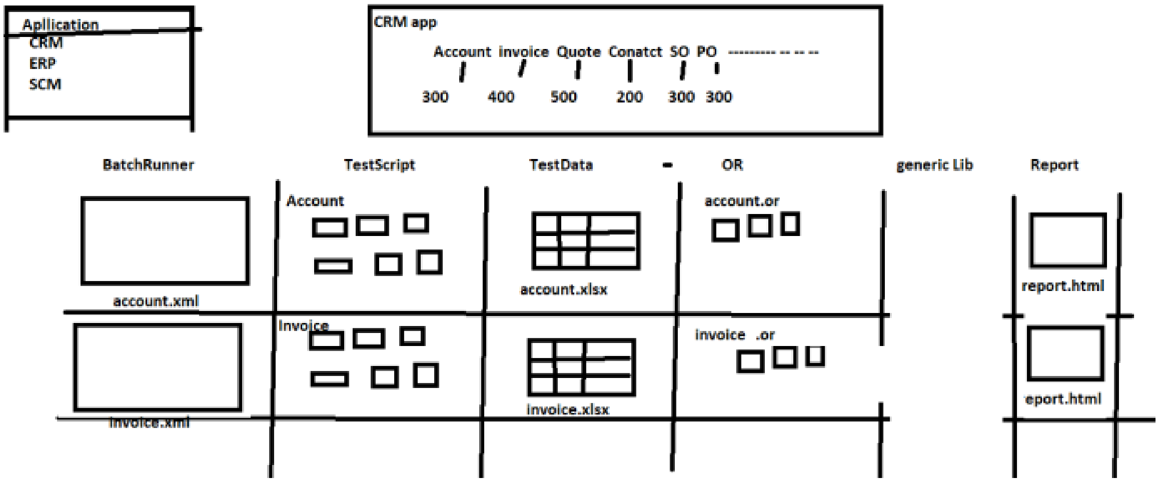
```
@test (dataPRovider='getData')
public void addToCart()
{
  1. login to App
  2. search for Product ****
  3. click on add to cart
  4. biling
  5. verify
 ,6. logout
}
@dataProvider
public void getData(){
Object[][] objARR
}
```

Cat	product	qty	cardinfo
iphone	XR	1	
nokia	1100	10	
Samsung	as0	10	

- ➢ Whenever we need to test the application with huge set of data, we go for data driven framework
- ➢ every test will have dedicated @dataProvider annotation & will have separate excel sheet for every test scenario
- ➢ When ever need to execute same test with different set of data we go for @dataProvider annotation
- ➢ We will be having reusable methods to fetch the Data From Files and Using them in Our Test Scripts Instead of Hard Coding it.

Modular Driven Framework

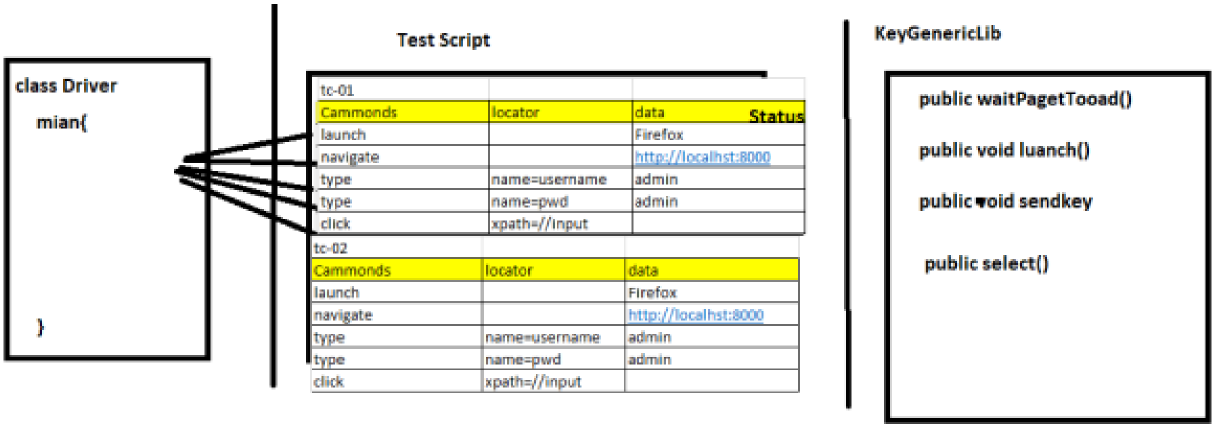
- ➢ Whenever application is huge / contains lots of module we prefer modular driven framework In modular driven framework, all framework components will maintain in module wise



- Even arranging Elements in POM is also example for Modular Driven Testing.

Key Word Driven Framework

- When manual test Engineer want to write Automation test script with less knowledge on automation tool & without knowledge coding we go for keyword Driven framework



Test Script

tc-01

Cammonds	locator	data	Status
launch		Firefox	
navigate		http://localhst:8000	
type	name=username	admin	
type	name=pwd	admin	
click	xpath=//input		

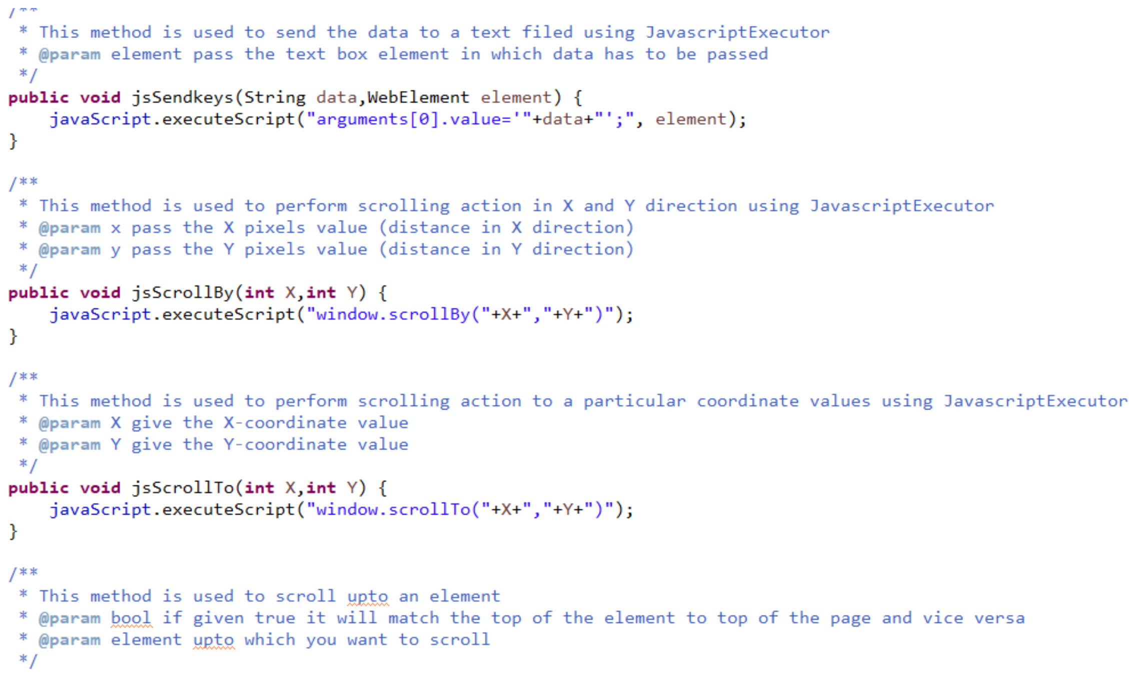
tc-02

Cammonds	locator	data
launch		Firefox
navigate		http://localhst:8000
type	name=username	admin
type	name=pwd	admin
click	xpath=//input	

- It can be feasible only for small applications

Method Driven Framework

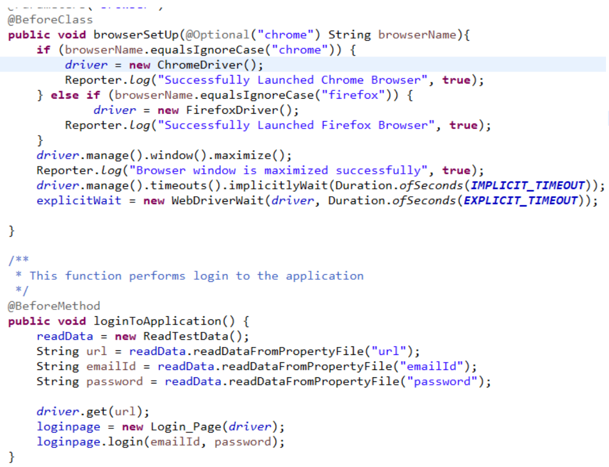
- Whenever Application is bit complex, Writing the Instructions becomes a tidies job, so we will be using method Driven Framework concept that is creating reusable components [generic and business utils] inside the project and using them efficiently



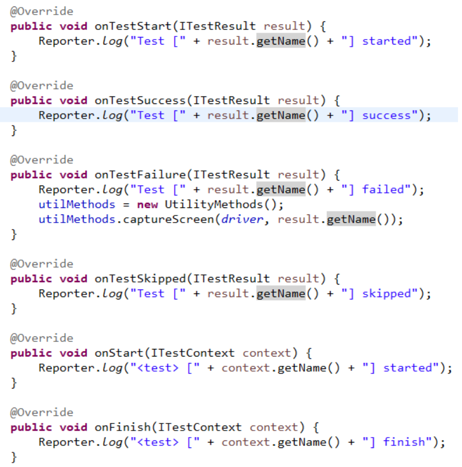
```
/**
 * This method is used to send the data to a text filed using JavascriptExecutor
 * @param element pass the text box element in which data has to be passed
 */
public void jsSendkeys(String data,WebElement element) {
    javaScript.executeScript("arguments[0].value='"+data+"';", element);
}

/**
 * This method is used to perform scrolling action in X and Y direction using JavascriptExecutor
 * @param x pass the X pixels value (distance in X direction)
 * @param y pass the Y pixels value (distance in Y direction)
 */
public void jsScrollBy(int X,int Y) {
    javaScript.executeScript("window.scrollBy("+X+","+Y+")");
}

/**
 * This method is used to perform scrolling action to a particular coordinate values using JavascriptExecutor
 * @param X give the X-coordinate value
 * @param Y give the Y-coordinate value
 */
public void jsScrollTo(int X,int Y) {
    javaScript.executeScript("window.scrollTo("+X+","+Y+")");
}

/**
 * This method is used to scroll upto an element
 * @param bool if given true it will match the top of the element to top of the page and vice versa
 * @param element upto which you want to scroll
 */
```

Test Driven Development [TDD] using TestNG

- Developing the Test Cases and using them efficiently by using some advance mechanism of Any Unit Testing framework tool is considered as TDD
- We are using TestNG to achieve TDD
- We are implementing config annotations, Listeners, Using DataProvider, creating TestNG suite achieving Batch, Parallel, Group execution of Test cases etc.



```
@BeforeClass
public void browserSetUp(@Optional("chrome") String browserName){
    if (browserName.equalsIgnoreCase("chrome")) {
        driver = new ChromeDriver();
        Reporter.log("Successfully Launched Chrome Browser", true);
    } else if (browserName.equalsIgnoreCase("firefox")) {
            driver = new FirefoxDriver();
        Reporter.log("Successfully Launched Firefox Browser", true);
    }
    driver.manage().window().maximize();
    Reporter.log("Browser window is maximized successfully", true);
    driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(IMPLICIT_TIMEOUT));
    explicitWait = new WebDriverWait(driver, Duration.ofSeconds(EXPLICIT_TIMEOUT));

}

/**
 * This function performs login to the application
 */
@BeforeMethod
public void loginToApplication() {
    readData = new ReadTestData();
    String url = readData.readDataFromPropertyFile("url");
    String emailId = readData.readDataFromPropertyFile("emailId");
    String password = readData.readDataFromPropertyFile("password");

    driver.get(url);
    loginpage = new Login_Page(driver);
    loginpage.login(emailId, password);
}
```



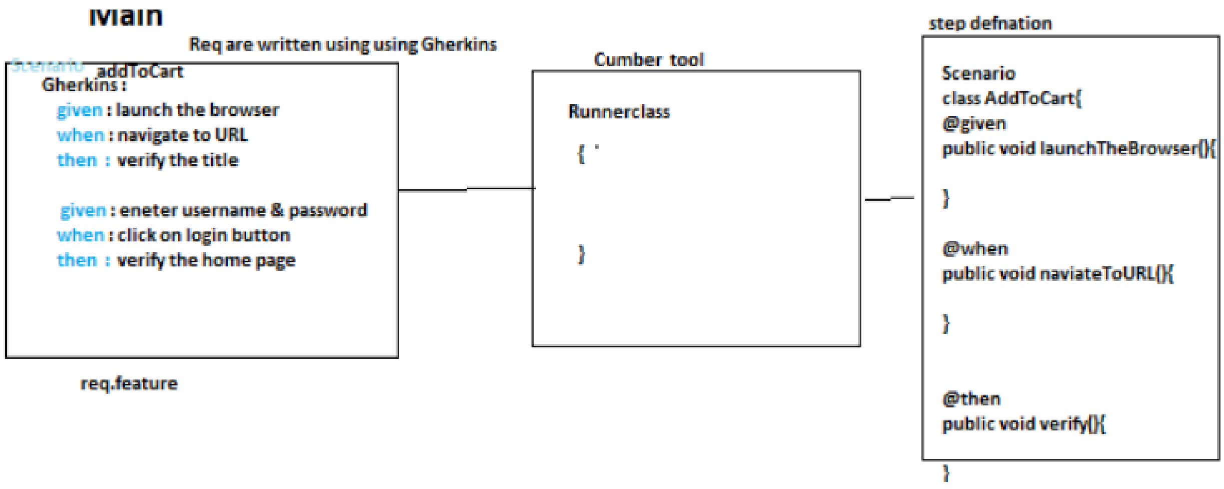
```
@Override
public void onTestStart(ITestResult result) {
    Reporter.log("Test [" + result.getName() + "] started");
}

@Override
public void onTestSuccess(ITestResult result) {
    Reporter.log("Test [" + result.getName() + "] success");
}

@Override
public void onTestFailure(ITestResult result) {
    Reporter.log("Test [" + result.getName() + "] failed");
    utilMethods = new UtilityMethods();
    utilMethods.captureScreen(driver, result.getName());
}

@Override
public void onTestSkipped(ITestResult result) {
    Reporter.log("Test [" + result.getName() + "] skipped");
}

@Override
public void onStart(ITestContext context) {
    Reporter.log("<test> [" + context.getName() + "] started");
}

@Override
public void onFinish(ITestContext context) {
    Reporter.log("<test> [" + context.getName() + "] finish");
}
```

Hybrid Driven Framework

- Combination of more than one framework is called Hybrid driven Framework
- Example: The Framework which we Designed is Hybrid Because its Combination of Data Driven + Modular Driven + Method Driven + TestNG

Behavior Driven Development [BDD] using Cucumber

- BDD framework that is Behavior Driven Development is a software development approach that allows the tester/business analyst to create test cases in simple text language (English).
- The simple language used in the scenarios helps even non-technical team members to understand what is going on in the software project
- Here we will use certain keywords of Gerkin Language like Given, When, Then, etc.
- It needs a tool Called as Cucumber, feature file



```
Gherkins :
  given : launch the browser
  when : navigate to URL
  then  :  verify the title

  given : eneter username & password
  when : click on login button
  then  :  verify the home page
```

```
Runnerclass
 { '


 }
```

```
Scenario
class AddToCart{
@given
public void launchTheBrowser(){

}

@when
public void naviateToURL(){

}


@then
public void verify(){

}
```