# Automation Framework

↳ Framework is an organized way of maintaining automation files. all the files will communicate each other to perform certain task in framework.

↳ objective : Automate Retesting & Regression tcs

① Reusability
② Maintainability

↳ Types :

① Build-in frameworks : pytest, robot, unittest
② Customized / user defined framework : Datadriven framework, keyword driven framework and hybrid driven framework

↳ Phases :

① Analyze application, technology & skill set of team, choose test cases (100% automation is not possible)
② Design and implementation of framework
③ Execution
④ Maintenance (Version Control system)

↳ Technologies used in framework :

→ Python, Selenium, Pytest, Page Object Model, HTML Reports / Allure-pytest / Pytest-html, pytest-xdist (run tests in parallel)

## * Steps to create a framework !

① Create a new project and install required packages/ plugins

② Create folder structure !

Project Name

- → PageObjects (Package) class with xpath & method
- → testCases (Package) test_login.py, XLUtils.py
- → Utilities (Package) logger.py, readproperties.py
- → TestData (Folder/Directory) external data
- → Configurations (Folder) config.ini (URL, Username & Password)
- → Logs (Folder) Automation.log
- → Screenshots (Folder) TC01 - loginpage.png
- → Reports (Folder) TC03.html
- → Run.bat (to run scripts in one shot).

③ Automate Login test case !

① Create LoginPage Object Class under "pageObjects" (python file)

② Create LoginTest under "testCases" (file)

③ Create conftest.py under "testCases" (fixture)

eg: launching browser

④ Capture Screenshot :

① update Login Test with screenshot under "testCases"

⑤ Read common values from ini file :

① Add "config.ini" file in Configurations folder
② Create "readProperties.py" utility file under utilities package to read common data
③ Replace hard coded values in login test case.

⑥ Adding logs to test case :

① Add "customLogger.py" under utilities package
② Add logs to Login test case

⑦ Run tests on desired browser / cross browser / parallel :

① update conftest.py with required fixtures which will accept command line argument (browser name)
② Pass browser name as argument in command line.

↳ To run tests on desired browsers :

pytest -s -v testCases/test_login.py --browser chrome
pytest -s -v testCases/test_login.py --browser firefox

↳ To run tests in parallel:

```
pytest -s -v -n=3 testCases/test_login.py --browser chrome
pytest -s -v -n=3 testCases/test_login.py --browser firefox
```

## ⑧ Generate pytest HTML reports:

① update conftest.py with pytest hooks

② To generate HTML report, Run below command:

```
pytest -s -v -n=3 --html=Reports\report.html
--self-contained-html
testCases/test_login.py --browser chrome
```

## ⑨ Automate Data-driven test case:

① Prepare test data in excel sheet, place the excel file inside the testdata folder

② Create "ExcelUtils.py" utility class under utilities package

③ Create LoginDataDrivenTest under testCases

④ Run the test case.

## ⑩ Adding new testcases

## (11) Grouping Tests:

(1) Grouping markers (Add markers to every test method)

```
@pytest.mark.sanity
@pytest.mark.regression
```

(2) Add marker entries in pytest.ini file

```
[pytest]
markers = sanity
          regression
```

(3) select groups at run time

```
-m "sanity"
-m "regression"
-m "sanity and regression"
-m "sanity or regression"
```

↳ Run Command !

```
pytest -s -v -m "sanity or regression" --html=./
Reports/report.html testCases/ --browser chrome
```

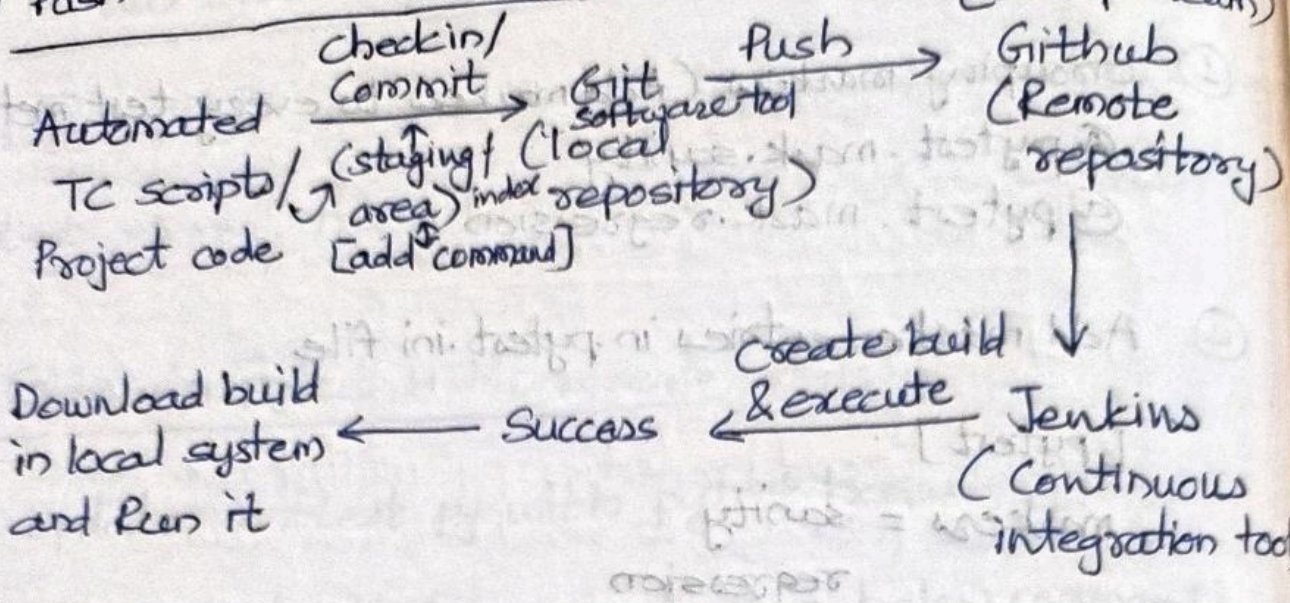
## (12) Run tests in command prompt and run.bat file:

(1) create run.bat file in project directory path:

```
pytest -s -v -m "sanity" --html=./Reports/report
.html testCases/ --browser chrome (rem to ignore)
```

(2) Open command prompt as Administrator and then run run.bat file

## (13) Push the code to Git & Github repository:




## (1) Create local git repository for your project:

(a) git init → create an empty git repository (local repository)

(b) git remote add origin https://github.com/venkatesh-Kalyane/MLfirst.git
↓
(Remote repository)
↳ Connecting local repository with Remote repository

(c) Before doing commit for first time, we need to execute below commands:

```
git config --global user.name "Venkatesh"
git config --global user.email "vsk6895s@gmail.com"
```

(2) git status → Checking status of files (commited/ Not Commited)

③ git add -A → Add all files to staging area

④ git commit -m "My first commit" → commit code into git repository (local repository)
git branch -M master/main

⑤ git push -u origin master/main → Push code to github repository (remote) from git (local)

⑥ git pull → to pull changes from github to local

⑭ Run Tests using Jenkins: Go to http://localhost:8080

① New Item → Enter item name → Freestyle project → OK

Source Code Management → Git → Enter repository URL

Build → Execute windows batch commands → run.bat → Apply & Save

② Home → Manage Jenkins → Global Tool Config

Git → Enter Git path → Apply & Save

③ Build Now

① Element isn't loaded completely: all elements you see aren't completely ready

solution: wait until element is loaded completely

② Element is not clickable at point:

solution: make sure that overlapping element is closed before you try to click on another element.

or

switch to layer that contains the element you want to click

③ Null pointer exception: it is thrown when a reference variable is accessed (or de-referenced) and is not pointing to any object. this error can be resolved by using a try-catch or an if-else condition to check if a reference variable is null before dereferencing it.

④ Timeout exception: when a command performing an operation doesn't complete in stipulated time.

⑤ No Such ~~Element~~ window/frame Exception: when an element with given attribute is not found on web page.

⑥ Element Not Visible Exception: when element is present in DOM (Document Object Model) but not visible on web page.

⑦ Stale Element Exception: when element is either deleted or no longer attached to DOM.

⑧ Webdriver exception ⑨ No Alert Present Exception