PROJECT OF AUTOMATION

Introduction: This is sample Automating Test project with Standalone Web application
I developed a testing framework based on this application to test the major functionalities.
In bellow I describing each and every single step that will help you to find the available need so far.

FRAMEWORK:
1. It has designed under a Data Driven Framework following Page Object Model (POM) pattern
   1. Data Driven test
      1. TestNG Data Provider
      2. Difference sets of data by XLS Sheet
   2. Page Factory
   3. Environmental Variables
   4. Behavior Drivern Development (**BDD)** FrameWork.
   5. Report Generating
      1. Log4j
      2. Extent Report
      3. TestNG Report
      4. HTML Report
      5. XML report
      6. JSON Report
   6. Using SelfDepending Functionality
      1. ReTryAnalyzer
      2. Transformation
      3. Web Even tListener

TESTING WERE PERFORMED:
2. Difference Type of testing I performed
   1. Sanity Test
   2. Complete Regression Test
   3. UAT Test including:
      1. Features file
      2. Test Runner
      3. Step Definition

4. Scenario Outlines

PROGRAMMING CONCEPT:

3. Concept I using JAVA OOP as much as possible including
    1. Encapsulation
    2. Polymorphism
    3. Abstractions
    4. Inheritance. Including :
        1. OverLoad
        2. OverRide
    5. Singleton

WEB BROWSER:

1. Difference type of Browser Driver including
    1. Chrome
    2. Firefox
    3. Safari

CI/CD:

1. Using continuous Process VIA CI pipeline
    1. Jenkins

MAJOR TECHNOLOGIES:

1. Overall Technology I using Including:
    1. JAVA
    2. SELENIUM WEBDRIVER
    3. TestNG
    4. Junit
    5. MAVEN
    6. Gherkins
    7. Cucumber
    8. Jenkins
    9. GIT
    10. MS Excel

DEPENDENCIES:

1. Dependency I using :
    1. Selenium
    2. Testing
    3. Junit
    4. Apache POI
    5. POI-OOXML
    6. POI-OOXML SCHEMAS
    7. POI OPENXML4J
    8. OOXML Schemas
    9. Cucumber-Core

10. Cucumber-java
11. Cucumber-JVM-Deps
12. Cucumber Junit
13. Cucumber Extent Rreports
14. Common Io
15. Log4J

ENVIRONMENT:
1. Environments :

1. MAC (I using)
2. Windows compatible 100%.

Above mentioned technologies and processes I used and follow .
I was told to using **Gradle, windows, and Java V-11,** Unfortunately I couldn't made it because I did not have available resources ,
Basically In personally I am using MAC and I did not have any widows machine in may hand at this Time, But I am comfortable
To working both OS.

To execute rather then Mac, Please follow the instructions:
Only 4 things need to be change:
1. **Properties file location**
2. **Excel file location**
3. **Driver file location**
4. **Sample application file location**

If you are using Windows: after cloning from the git —>>
1. Create a folder in c:/ as "MyTestSuit"
2. Past from git location to "MyTestSuit" Location
3. Open any IDE and import this workspace
4. Go to project hierarchy click on "com.base"
5. Open "TestBase.Java" keep open
6. Go to "com.configuration" package —> config.properties —> get the file path (root)
7. Go to TestBase Class —> FileInputStream fis = **new** FileInputStream(" config file path); >>> **Properties file location**
8. Get your chromeDriver path and past it to —>> System.*setProperty*("webdriver.chrome.driver", "chromePath); >> **Driver File Location**
9. Same as geckoDriver, safari driver path as well
10. Got to Package —>>"com.utill">>TestUtill>>*TEST_DATA_SHEET_PATH* = "**Excel File path**" which will be available in all the way down of project

hierarchy and get the file path past.
11. Now your **simple Application file location** will be available middle of the project hierarchy>> open it >>> get path of index.html file path and >> open "config.properties">> past on "url" Parameter .

Thats it. Bingo>>>>>>>>>>>>>