

---

Asian Institute of Technology  
School of Engineering and Technology  
AT70.19: Software Development and Quality Improvement



***“AIR QUALITY (PM<sub>2.5</sub>) MONITORING SYSTEM”  
(IOT-BASED PLATFORM)***

***System Documentation***

Submitted by:

Smrity Baral (st121662)

Shubhangini Gontia (st121473)

Suyogya Ratna Tamrakar (st121334)

Younten Tshering (st121775)

Submitted to:

Dr. Apichon Witayangkurn

Submitted Date:

28<sup>th</sup> April, 2021

---

## Table of Contents

1. Spring Boot Overview .....	3
1.1 Application of Spring Boot.....	3
2. Apache Kylin .....	4
2.1 Installation Guide.....	4
2.1.1 Software Requirements.....	4
2.1.2 Hardware Requirements.....	4
2.2 Hadoop Environment .....	5
2.3 Kylin Installation .....	5
2.3.1 Kylin tarball structure .....	5
2.4 Start Kylin.....	5
2.5 Using Kylin.....	6
2.6 Stop Kylin .....	6
3. Run Kylin with Docker.....	6
3.1 Quickly try Kylin .....	6
3.2 Container resource recommendation .....	7
4. Connecting Tableau Desktop and Tableau Server with Apache Kylin .....	8
4.1 Prerequisites.....	8
4.1.1 Apache Kylin JDBC Driver .....	8
4.1.2 Compiling Apache Kylin JDBC Driver.....	8
4.1.3 Tableau Server on Linux.....	8
4.1.4 Tableau Server and Tableau Desktop on Windows .....	8
4.2 Creating report in Tableau Desktop - connecting to Apache Kylin.....	8
4.3 Publishing reports from Tableau Desktop to Tableau Server.....	11
5. Testing tool .....	12
5.1 Setup Robot Framework .....	12
5.1.1 JDK 1.8-1 .....	12
5.1.2 Python .....	12
5.1.3 Robot Framework Library .....	13
5.1.4 Robot Framework Selenium Library .....	14
5.1.5 Develop Test Script.....	14
5.1.6 Chrome Driver .....	14

5.1.7 Try to write test script and run test script. ....	15
5.2 Setup JMeter .....	17
5.2.1 Requirements .....	18
5.2.2 JMeter - overview of how to use.....	18
5.2.3 Testing Demo.....	22
6. Continuous integration.....	26
6.1 Setup Jenkins: .....	26
6.2 Overview of Jenkins .....	27
6.2.1 Software Requirements for full cycle for CI: .....	27
6.2.2 Jenkin prerequisites.....	27
6.2.3 Run Jenkins .....	28
6.2.4 Creating Pipeline.....	28

# 1. Spring Boot Overview

It is a framework that gives us needed resources to build up an application.

## 1.1 Application of Spring Boot

Spring Boot can be applied and used in different ways like; If we need security, we can use security model available, logging integration, connecting to database (mongo dB, MySQL, etc.), is easy to learn, production ready, can build microservices, dependency injection, configuration, Great community and many more.

Link for beginner to start using Spring Boot:

<https://spring.io/quickstart>

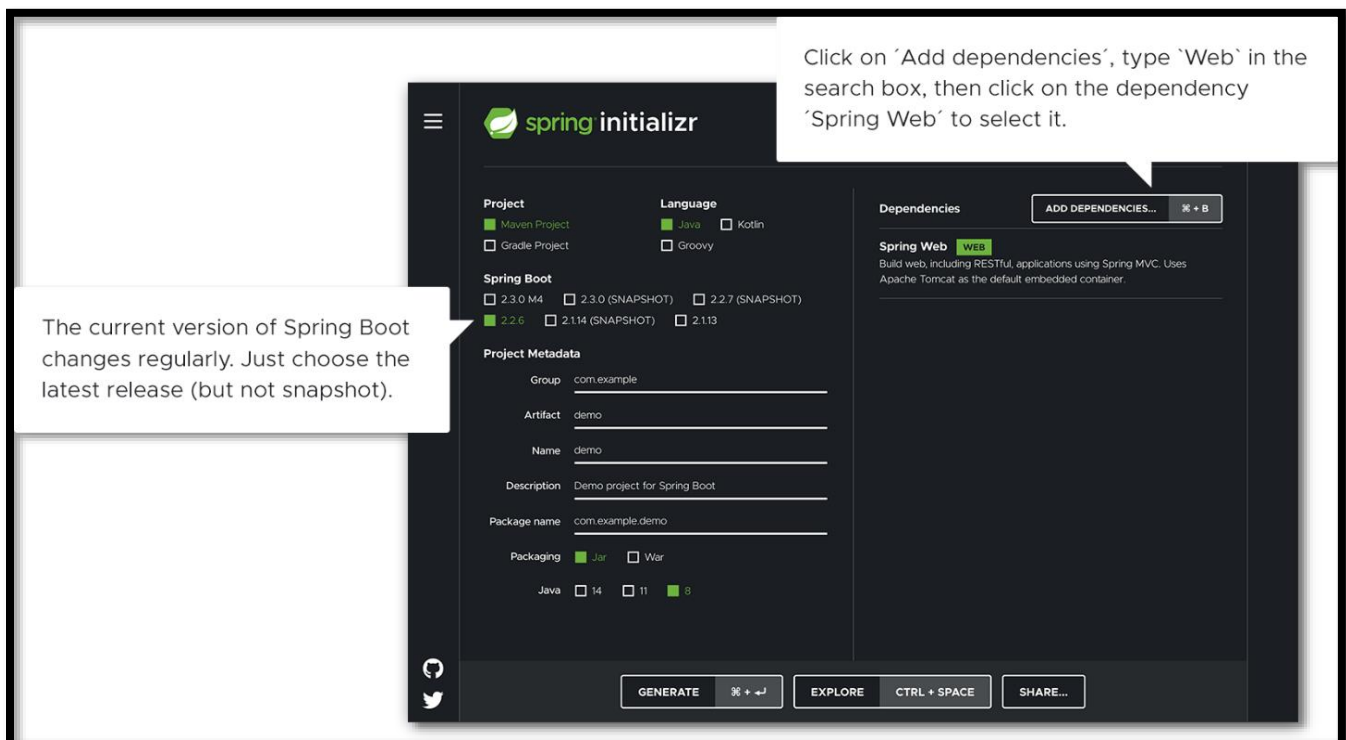
Steps to follow:

1. Choose one Integrated Developer Environment (IDE) and Install it
2. We need JDK (Java Development Kit), Install the correct version.

After the above steps, we are ready to start new Spring Boot project.

Use [start.spring.io](https://start.spring.io) to create a “web” project.

In the “Dependencies” dialog search for and add the dependencies that are needed such as shown in the screenshot. Hit the “Generate” button, download the zip, and unpack it into a folder on your computer.



After this, open this folder from your IDE and add your code or you can copy and paste the code from the mention website. Then we can run the program from IDE terminal giving the following command:

1. MacOS/Linux

```
./mvnw spring-boot:run
```

2. Windows

```
mvnw spring-boot:run
```

If you have followed the steps from mention website, then you will be able to see Hello World! When you open your browser and typed the <http://localhost:8080/hello>.

You can check: <http://localhost:8080/hello?name=SDQI%20Project>

**Output:**

Hello SDQI Project!

## 2. Apache Kylin

Analytical Data Warehouse for Big Data

### 2.1 Installation Guide

#### 2.1.1 Software Requirements

- ✓ Hadoop: 2.7+, 3.1+ (since v2.5)
- ✓ Hive: 0.13 - 1.2.1+
- ✓ HBase: 1.1+, 2.0 (since v2.5)
- ✓ Spark (optional) 2.3.0+
- ✓ Kafka (optional) 1.0.0+ (since v2.5)
- ✓ JDK: 1.8+ (since v2.5)
- ✓ OS: Linux only, CentOS 6.5+ or Ubuntu 16.0.4+

#### 2.1.2 Hardware Requirements

The minimum configuration of a server running Kylin is 4 core CPU, 16 GB RAM and 100 GB disk. For high-load scenarios, a 24-core CPU, 64 GB RAM or higher is recommended.

## 2.2 Hadoop Environment

Kylin relies on Hadoop clusters to handle large data sets. We need to prepare a Hadoop cluster with HDFS, YARN, MapReduce, Hive, HBase, Zookeeper and other services for Kylin to run.

Kylin can be launched on any node in a Hadoop cluster. For convenience, we can run Kylin on the master node.

*Linux accounts running Kylin must have access to the Hadoop cluster, including the permission to create/write HDFS folders, Hive tables, HBase tables, and submit MapReduce tasks.*

## 2.3 Kylin Installation

1. Download a binary package for your Hadoop version from the Apache Kylin Download Site (<https://kylin.apache.org/download/>).
2. Unzip the tarball and configure the environment variable \$KYLIN\_HOME to the Kylin folder.

### 2.3.1 Kylin tarball structure

- ❖ **bin**: shell scripts to start/stop Kylin service, backup/restore metadata, as well as some utility scripts.
- ❖ **conf**: XML configuration files. The function of these xml files can be found in configuration page
- ❖ **lib**: Kylin jar files for external use, like the Hadoop job jar, JDBC driver, HBase coprocessor jar, etc.
- ❖ **meta\_backups**: default backup folder when run “bin/metastore.sh backup”;
- ❖ **sample\_cube**: files to create the sample cube and its tables.
- ❖ **spark**: the default spark binary that built with Kylin.
- ❖ **tomcat** the tomcat web server that run Kylin application.
- ❖ **tool**: the jar file for running utility CLI.

## 2.4 Start Kylin

Run the script, `$KYLIN_HOME/bin/kylin.sh start`, to start Kylin. The interface output is as follows:

```
Retrieving hadoop conf dir...
KYLIN_HOME is set to /usr/local/apache-kylin-2.5.0-bin-hbase1x
.....
A new Kylin instance is started by root. To stop it, run 'kylin.sh stop'
Check the log at /usr/local/apache-kylin-2.5.0-bin-hbase1x/logs/kylin.log
Web UI is at http://<hostname>:7070/kylin
```

## 2.5 Using Kylin

Once Kylin is launched, we can access it via the browser `http://<hostname>:7070/kylin` with specifying `<hostname>` with IP address or domain name, and the default port is 7070.

The initial username and password are `ADMIN/KYLIN`.

After the server is started, we can view the runtime log, `$KYLIN_HOME/logs/kylin.log`.

## 2.6 Stop Kylin

Run the `$KYLIN_HOME/bin/kylin.sh stop` script to stop Kylin. The console output is as follows:

```
Retrieving hadoop conf dir...
KYLIN_HOME is set to /usr/local/apache-kylin-2.5.0-bin-hbase1x
Stopping Kylin: 25964
Stopping in progress. Will check after 2 secs again...
Kylin with pid 25964 has been stopped.
```

## 3. Run Kylin with Docker

To allow users to easily try Kylin, and to facilitate developers to verify and debug after modifying the source code. Kylin provide Kylin's docker image. In this image, each service that Kylin relies on is properly installed and deployed, including:

- ✓ JDK 1.8
- ✓ Hadoop 2.7.0
- ✓ Hive 1.2.1
- ✓ Hbase 1.1.2 (with Zookeeper)
- ✓ Spark 2.3.1
- ✓ Kafka 1.1.1
- ✓ MySQL 5.1.73

### 3.1 Quickly try Kylin

Kylin have pushed the Kylin image for the user to the docker hub. Users do not need to build the image locally, just execute the following command to pull the image from the docker hub:

```
docker pull apachekylin/apache-kylin-standalone:3.1.0
```

After the pull is successful, execute the following command to start the container:

```
docker run -d \  
-m 8G \  
-p 7070:7070 \  
-p 8088:8088 \  
-p 50070:50070 \  
-p 8032:8032 \  
-p 8042:8042 \  
-p 16010:16010 \  
apachekylin/apache-kylin-standalone:3.1.0
```

The following services are automatically started when the container starts:

- ❖ NameNode, DataNode
- ❖ ResourceManager, NodeManager
- ❖ HBase
- ❖ Kafka
- ❖ Kylin

and run automatically `$KYLIN_HOME/bin/sample.sh`, create a “kylin\_streaming\_topic” topic in Kafka and continue to send data to this topic. This is to let the users start the container and then experience the batch and streaming way to build the cube and query.

After the container is started, we can enter the container through the `docker exec -it <container_id> bash` command. Of course, since we have mapped the specified port in the container to the local port, we can open the pages of each service directly in the native browser, such as:

- Kylin Web UI: <http://127.0.0.1:7070/kylin/login>
- Hdfs NameNode Web UI: <http://127.0.0.1:50070>
- Yarn ResourceManager Web UI: <http://127.0.0.1:8088>
- HBase Web UI: <http://127.0.0.1:16010>

## 3.2 Container resource recommendation

To allow Kylin to build the cube smoothly, the memory resource we configured for Yarn Node Manager is 6G, plus the memory occupied by each service, please ensure that the memory of the container is not less than 8G, so as to avoid errors due to insufficient memory.

For more information follow the link: <http://kylin.apache.org/docs/>



## 4. Connecting Tableau Desktop and Tableau Server with Apache Kylin

Connect Tableau to Apache Kylin OLAP server, particularly in live mode to use both reporting and analytics features of Tableau together with Apache Kylin's fast query processing engine. The configuration is platform independent - it works for both Windows and Linux installations of Tableau Server.

For the time of writing this guide we tested that it works with **Kylin 3.1.0 and Tableau Server 2020.4**.

### 4.1 Prerequisites

#### 4.1.1 Apache Kylin JDBC Driver

First, we need to get Apache Kylin JDBC Driver - `kylin-jdbc-X.Y.Z.jar` file. You can either get it from the compiled package available on the download page <http://kylin.apache.org/download/> from `lib` folder.

#### 4.1.2 Compiling Apache Kylin JDBC Driver

```
git clone https://github.com/apache/kylin.git
cd kylin
mvn clean package -DskipTests -am -pl jdbc
```

#### 4.1.3 Tableau Server on Linux

If you have installed Tableau Server in a Linux box, copy the driver's jar file to the following location: `/opt/tableau/tableau_driver/jdbc/` and restart Tableau Server.

The server is now ready to create and refresh data from Apache Kylin.

#### 4.1.4 Tableau Server and Tableau Desktop on Windows

For either Tableau Server or Tableau Desktop that is installed on a Windows machine, copy the driver's jar file to the following location `C:\Program Files\Tableau\Drivers` and restart Tableau Server or reopen Tableau Desktop.

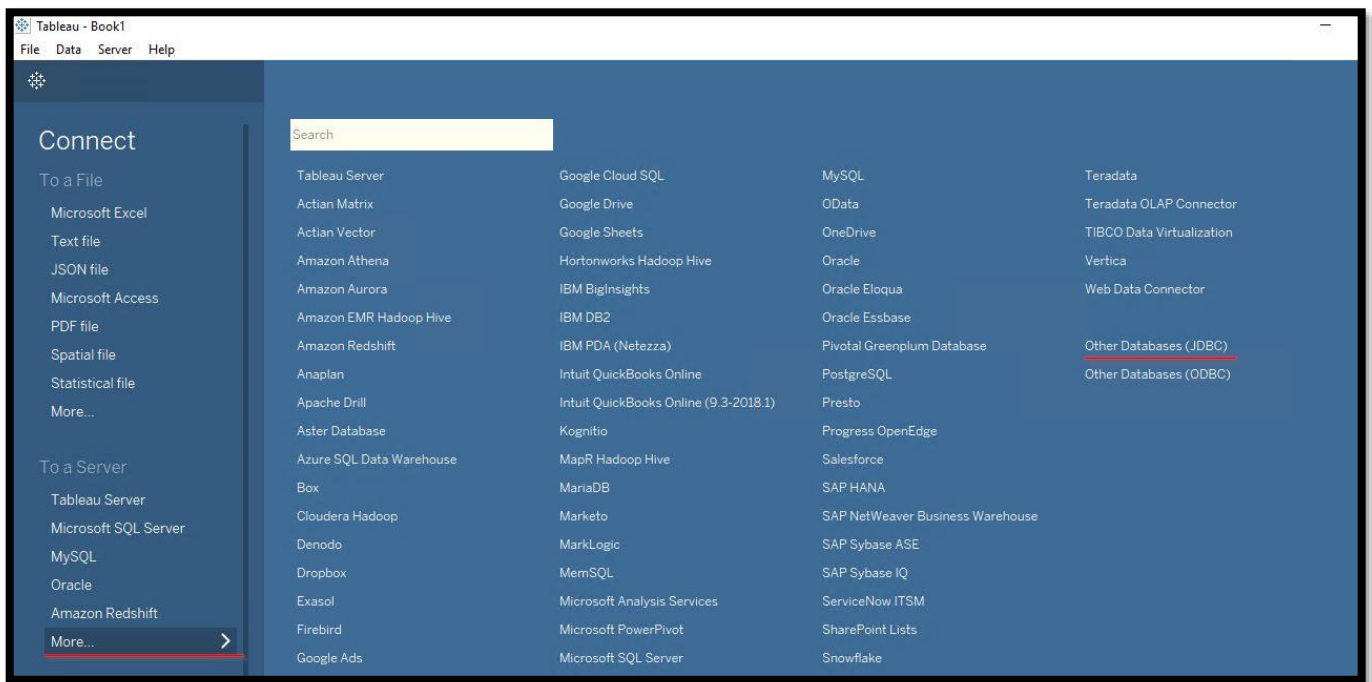
Some more details regarding jdbc connection from Tableau are well described in Tableau's documentation:

[https://onlinehelp.tableau.com/current/pro/desktop/en-us/examples\\_otherdatabases\\_jdbc.htm](https://onlinehelp.tableau.com/current/pro/desktop/en-us/examples_otherdatabases_jdbc.htm)

## 4.2 Creating report in Tableau Desktop - connecting to Apache Kylin

To create report, follow the following steps:

1. Open Tableau Desktop
2. Use "Other Databases (JDBC)" to create connection for the data source



3. Configure the connection in the following way:

URL: jdbc:kylin://<kylin-server-name>:<kylin-port>/<project>

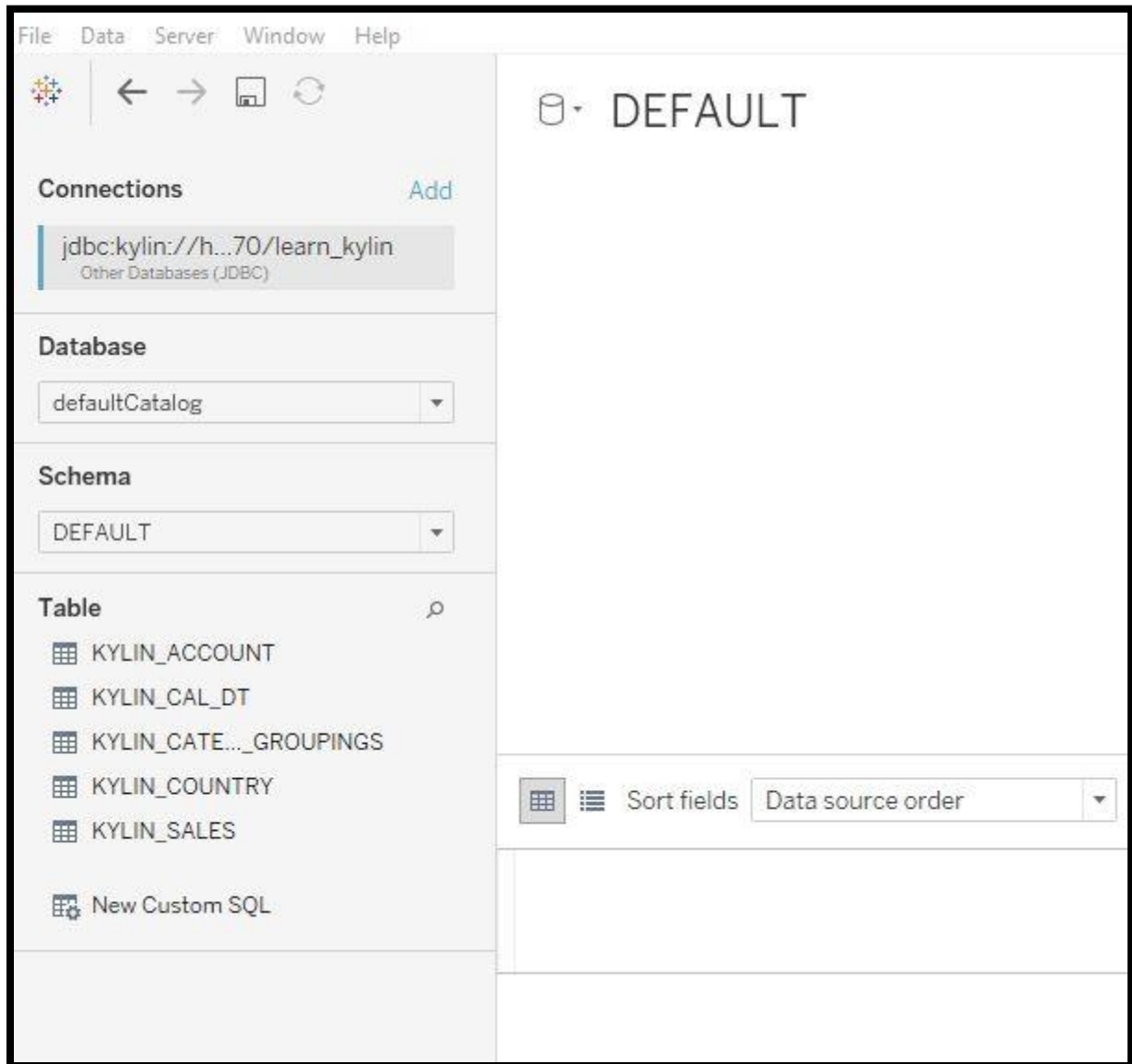
Dialect: SQL92

The screenshot shows the 'Other Databases (JDBC)' configuration dialog. The URL is set to 'jdbc:kylin://localhost:8080/learn\_kylin', the Dialect is 'SQL92', the Username is 'ADMIN', and the Password is masked with dots. The Properties File field is empty, and the Sign In button is highlighted.

4. Configure data source as follows:

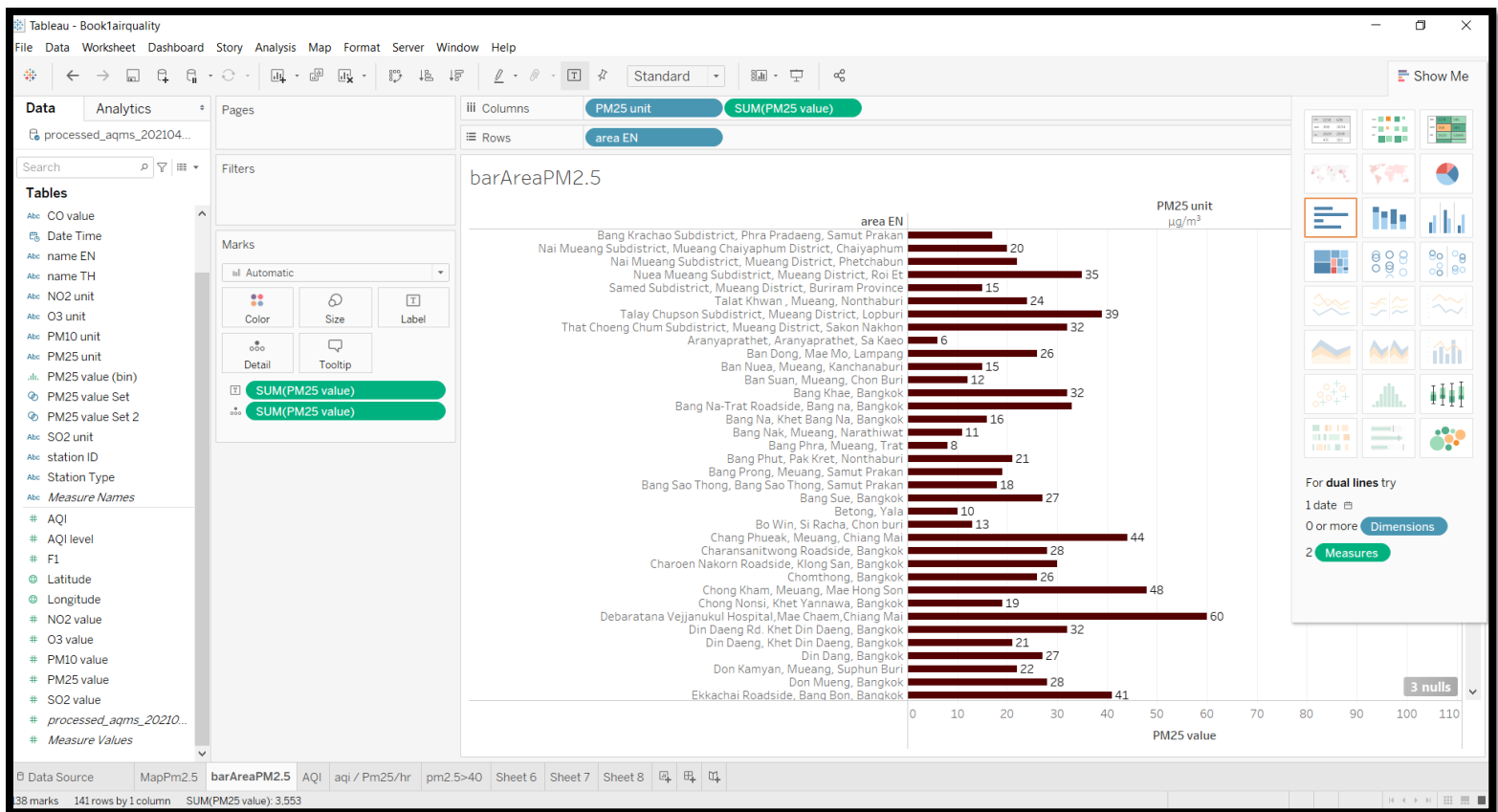
- Database: *defaultCatalog*
- Schema: *DEFAULT*

We should be able to see the tables/cubes in the Apache Kylin's project



**Note:** Decide if we want the data source be in live or extract mode. Some of the functions might not work in live mode as for the other data sources - it's just how Tableau works. Recommendation is to start with live mode to utilize performance of Apache Kylin. If we are forced to switch to extract mode - consider creating a custom query against Apache Kylin's cubes to retrieve as small amount of data as possible as it will help the report to perform well.

## 5. Finish designing our data source and then switch to worksheets, dashboards



## 4.3 Publishing reports from Tableau Desktop to Tableau Server

To publish the data source and the report follow the following steps:

1. In Tableau Desktop from top menu select Server and then Publish
2. Choose the settings for publishing like Project, select sheets
3. Note: For data source Authentication set Embedded - this is very important for data refresh to work, however keep in mind that the credentials will be embedded in the report
4. Publish the report
5. Pop up should be displayed with the preview of the report rendered by the server
6. Verify if the report is displaying properly and can connect to Apache Kylin correctly by opening it directly in Tableau Server web application.

## 5. Testing tool

Robot Framework is a test automation framework that is Python-based. This framework supports writing an object-page model in keyword driven methodology. Robot Framework allows users to write their own test-cases without programming knowledge. In addition to this, it has very descriptive logs including complete debugging capabilities through readable logs.

To use Robot Framework, we must make sure our system has following prerequisites:

1. JDK 1.8-1
2. Python
3. Robot Framework Library
4. Robot Framework Selenium Library
5. Chrome Driver

### 5.1 Setup Robot Framework

#### 5.1.1 JDK 1.8-1

The Java Development Kit (JDK) allows you to code and run Java programs.

Following are the steps to be followed for downloading and installing jdk in our system:

1. Goto <https://www.oracle.com/in/java/technologies/javase-downloads.html> click on *Download* JDK for latest version.
2. Next, Accept the license Agreement.
3. Click on download latest Java JDK depending on version of our PC.
4. Once Download is completed, then run the .exe for installation of JDK. Then click Next.
5. Select Proper PATH for Java installation and click next.
6. Once installation is complete then click close.

#### 5.1.2 Python

Before downloading python, see what version is to be installed, the follow the steps:

1. Open web browser: <https://www.python.org/downloads/release/python-2717/>. Search for desired version and select a link download either Windows x86-64 executable installer or Windows x86 executable installer.
2. Run the executable Installer. Make sure you select the Install launcher for all users and Add Python to PATH checkbox. Then select Install Now.
3. Verify Python was installed or not by typing python -V in command Prompt.

Before downloading, make sure your system has PIP installed.

You can follow this link <https://phoenixnap.com/kb/install-pip-windows> for pip.

```
Administrator: Command Prompt
C:\WINDOWS\system32>python.exe -m pip install --upgrade pip
Collecting pip
  Downloading pip-21.0.1-py3-none-any.whl (1.5 MB)
    | 1.5 MB 1.3 MB/s
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 20.2.3
    Uninstalling pip-20.2.3:
      Successfully uninstalled pip-20.2.3
  Successfully installed pip-21.0.1

C:\WINDOWS\system32>robot --version
Robot Framework 4.0.1 (Python 3.9.4 on win32)

C:\WINDOWS\system32>Type: pip
The filename, directory name, or volume label syntax is incorrect.

C:\WINDOWS\system32>pip install robotframework-selenium2library
Collecting robotframework-selenium2library
  Downloading robotframework_selenium2library-3.0.0-py2.py3-none-any.whl (6.2 kB)
Collecting robotframework-seleniumlibrary>=3.0.0
  Downloading robotframework_seleniumlibrary-5.1.3-py2.py3-none-any.whl (94 kB)
    | 94 kB 401 kB/s
Requirement already satisfied: robotframework>=3.1.2 in c:\users\younten tshering\appdata\local\programs\python\python39\lib\site-packages (from robotframework-seleniumlibrary>=3.0.0->robotframework-selenium2library) (4.0.1)
Collecting selenium>=3.141.0
  Downloading selenium-3.141.0-py2.py3-none-any.whl (904 kB)
    | 904 kB 1.7 MB/s
Collecting robotframework-pythonlibcore>=2.1.0
  Downloading robotframework_pythonlibcore-2.2.1-py2.py3-none-any.whl (10 kB)
Collecting urllib3
  Downloading urllib3-1.26.4-py2.py3-none-any.whl (153 kB)
    | 153 kB 2.2 MB/s
Installing collected packages: urllib3, selenium, robotframework-pythonlibcore, robotframework-seleniumlibrary, robotframework-selenium2library
Successfully installed robotframework-pythonlibcore-2.2.1 robotframework-selenium2library-3.0.0 robotframework-seleniumlibrary-5.1.3 selenium-3.141.0 urllib3-1.26.4

C:\WINDOWS\system32>pip list
Package            Version
-----
pip                21.0.1
robotframework     4.0.1
robotframework-pythonlibcore 2.2.1
robotframework-selenium2library 3.0.0
robotframework-seleniumlibrary 5.1.3
selenium           3.141.0
setuptools         49.2.1
urllib3            1.26.4
```

### 5.1.3 Robot Framework Library

- Once python is installed, you have to access to the pip installer.
- Open command prompt and type

***pip install robotframework***

#### 5.1.4 Robot Framework Selenium Library

Use the following command in command prompt to install robot framework selenium library.

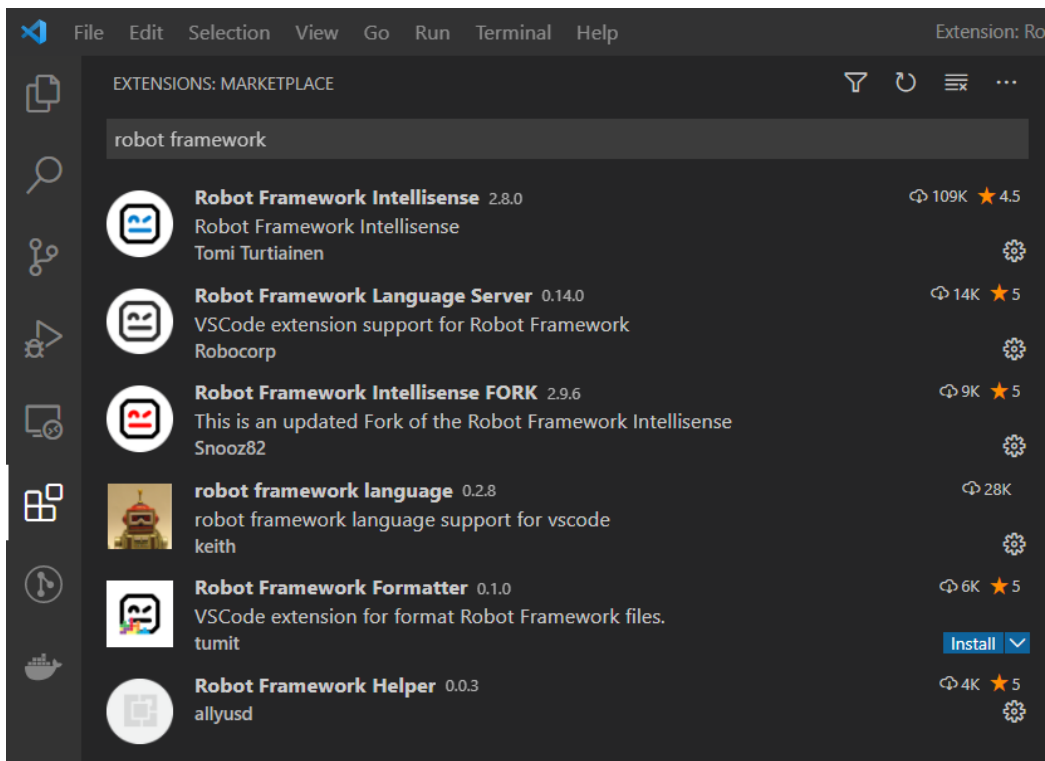
***pip install robotframework-seleniumlibrary***

#### 5.1.5 Develop Test Script

Before having the script, we need to install VS Code for Develop Test Script.

Steps to install VS Code and configure:

1. Download Visual Studio Code
2. Run the execution file and make it run.
3. We must add some few extension as shown in below diagram:



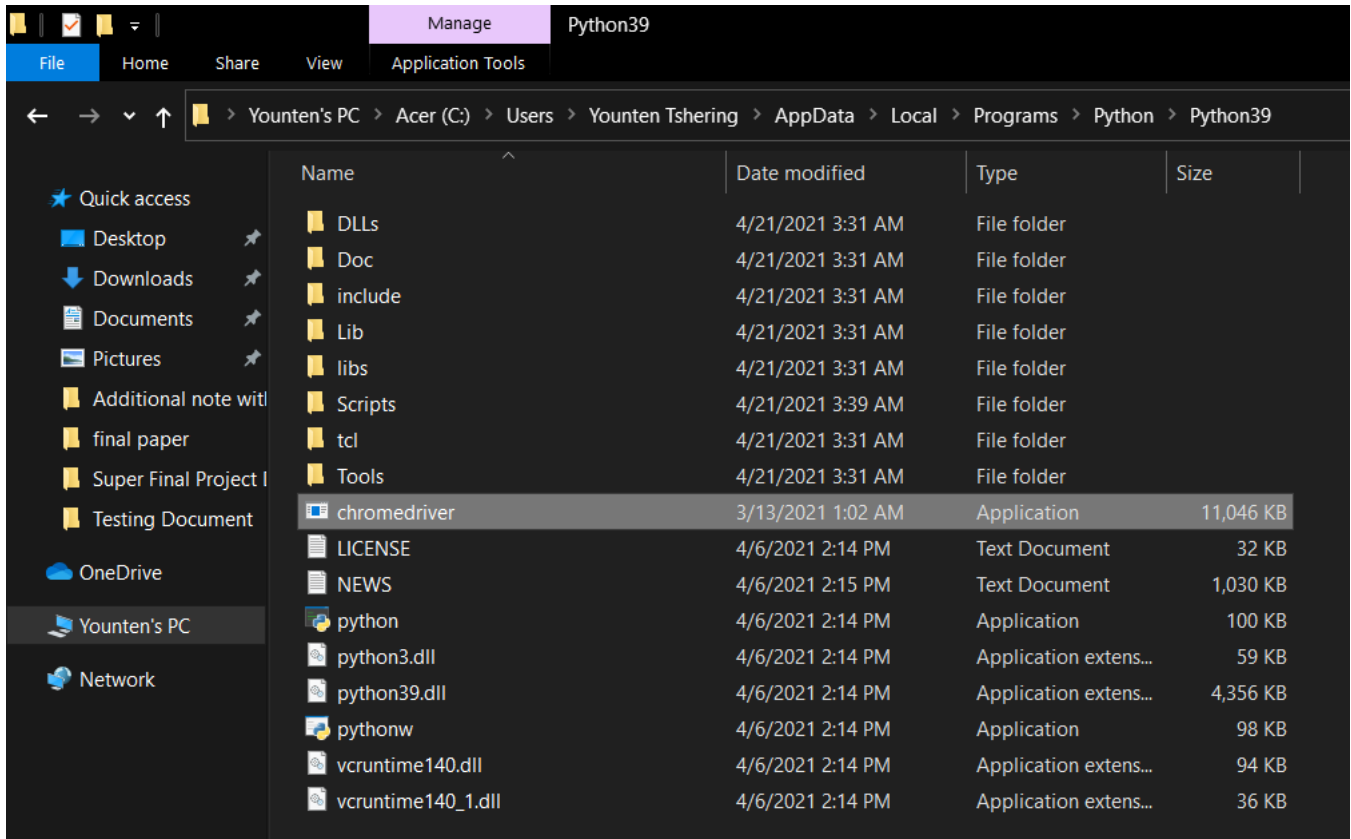
#### 5.1.6 Chrome Driver

Steps for chrome driver installation:

1. Check Version of Google Chrome
2. Download Chrome Driver using this link :

<https://sites.google.com/a/chromium.org/chromedriver/>

3. Click Latest release, then click on the driver zip file as per the version of your system to download chromedriver eg: chromedriver\_win32.zip
4. Unzip/ extract zip file in python folder as shown below:



5. Run executable exe file.
6. Make sure that system environments variables are properly configured.

For more details or complete guide, follow the link: <https://www.youtube.com/watch?v=gleaQkECggo>.

### 5.1.7 Try to write test script and run test script.

Before running the script, we need to complete the install of chrome driver and then:

1. Create sample Robot Script for testing
2. Create variables
3. Create functions
4. Testing Step: Test cases

For more information follow the link: <https://medium.com/edureka/robot-framework-tutorial-f8a75ab23cfd>.



## Test Robot with simple website access:

Write the script in VS Code with robot file extension and run. The example shown is the file uploaded by professor in basecamp, but project testing is included in the project documentation.

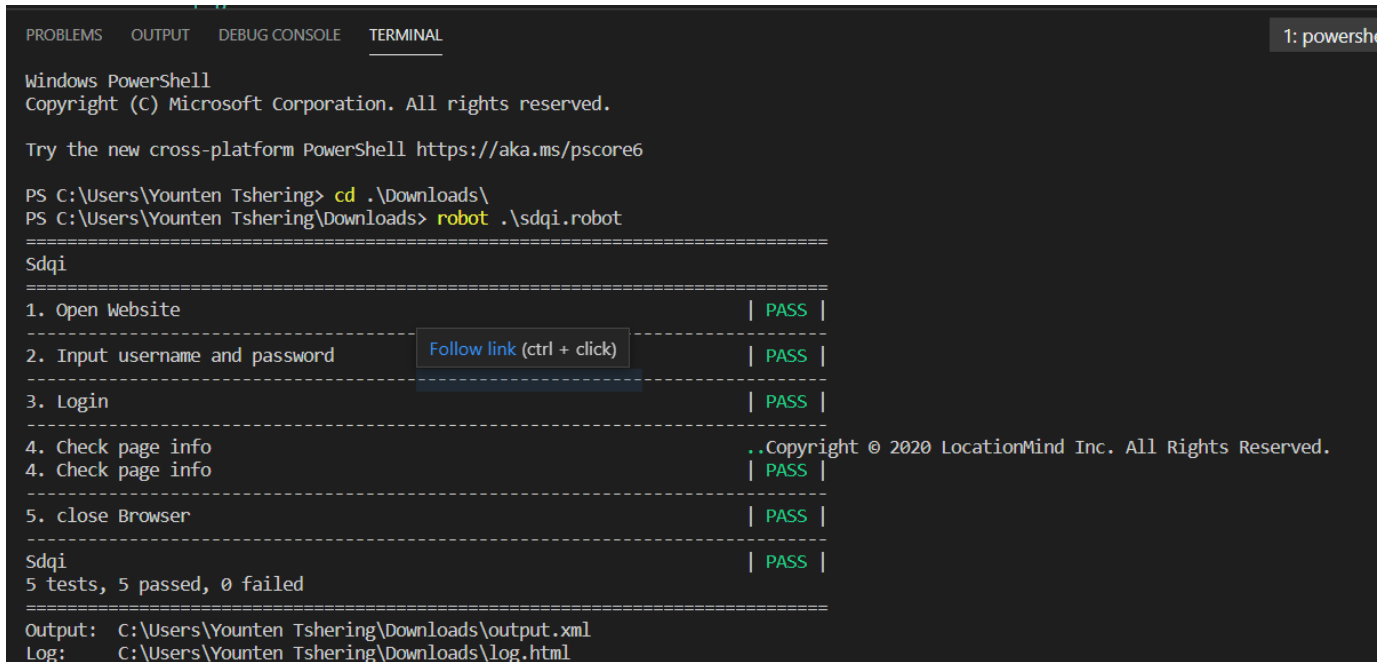
```
sdqi.robot X
C:\Users\Younten Tshering\Downloads> sdqi.robot > ...
1  *** Settings ***
2  Library Selenium2Library
3
4  *** Variables ***
5  ${expect} LocationMind
6  ${url} http://lmwebmap.gisserv.com
7  ${Browser} chrome
8  ${delay} 1
9
10 *** Test Cases ***
11 1. Open Website
12   Open Browser ${url} ${Browser} options=add_experimental_option("excludeSwitches", ["enable-logging"])
13   Maximize Browser Window
14   Set Selenium Speed 0.3
15 2. Input username and password
16   Input Text name=usernameTextBox tester
17   Input Text name=passwordTextBox tester
18 3. Login
19   Click Button name=submitButton
20 4. Check page info
21   Click Link xpath=//a[@href="#"][2]
22   ${result} Get Text xpath=//div[8]
23   Log To Console ${result}
24   Should Contain ${result} ${expect}
25
26 5. close Browser
27   Close Browser
28
29
30
```

Output if the chrome driver is not installed:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: powershell
PS C:\Users\Younten Tshering\Downloads> robot .\sdqi.robot

Sdqi
=====
1. Open Website | FAIL |
WebDriverException: Message: 'chromedriver' executable needs to be in PATH. Please see https://sites.google.com/a/chromium.org/chromedriver/home
-----
2. Input username and password | FAIL |
No browser is open.
-----
3. Login | FAIL |
No browser is open.
-----
4. Check page info | FAIL |
No browser is open.
-----
5. close Browser | PASS |
-----
Sdqi | FAIL |
5 tests, 1 passed, 4 failed
=====
Output: C:\Users\Younten Tshering\Downloads\output.xml
Log: C:\Users\Younten Tshering\Downloads\log.html
Report: C:\Users\Younten Tshering\Downloads\report.html
PS C:\Users\Younten Tshering\Downloads>
```

Final output of automation testing if everything works fine and we can see the test case being passed or failed.



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Younten Tshering> cd .\Downloads\
PS C:\Users\Younten Tshering\Downloads> robot .\sdqi.robot

=====
Sdqi
=====
1. Open Website | PASS |
2. Input username and password | PASS |
3. Login | PASS |
4. Check page info | PASS |
4. Check page info | PASS |
5. close Browser | PASS |
Sdqi | PASS |
5 tests, 5 passed, 0 failed
=====
Output: C:\Users\Younten Tshering\Downloads\output.xml
Log: C:\Users\Younten Tshering\Downloads\log.html
..Copyright © 2020 LocationMind Inc. All Rights Reserved.
```

## 5.2 Setup JMeter

**JMeter** for the purpose of load testing is considered as the one of the best tools. Apache JMeter is a Java open-source GUI-based software that is used as a performance testing tool for analyzing and measuring the performance of our application.

**Performance testing** can be widely categorized as

1. **load testing** - focuses on testing whether the system can handle *concurrent* user accesses without breaking.
2. **stress testing** focuses on how our system copes with high load and limited resources under constrained conditions.

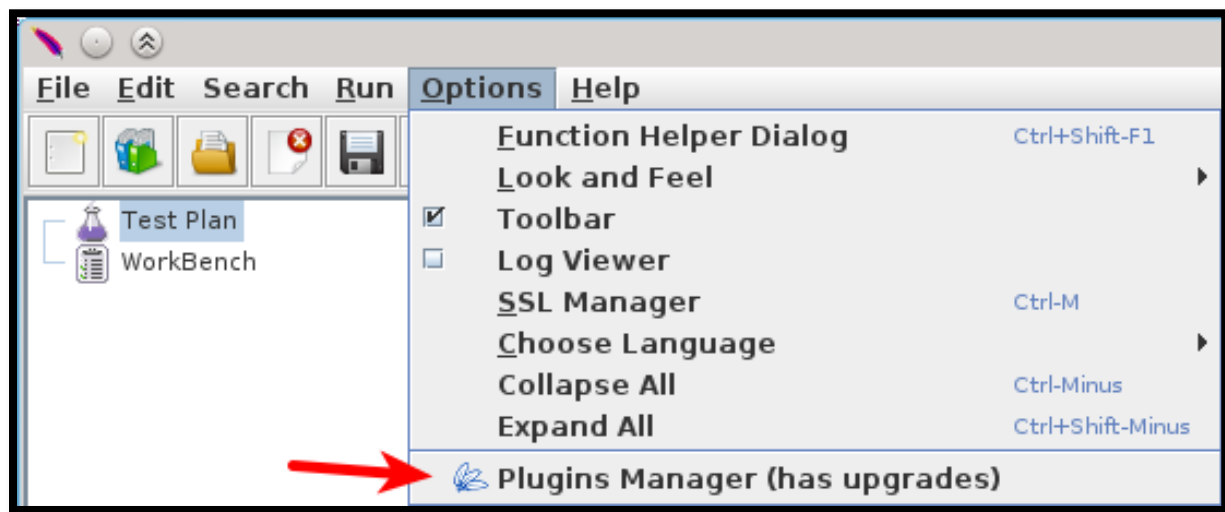
JMeter is extensible, meaning that we can write custom script, using the pre-built interface of JMeter, making it even more powerful. Of course, it already has many powerful features such as

- a) creating reusable test plans (in XML format) so we can reuse these similar test plans across all our products,
- b) supporting various protocols such as HTTP, JDBC, SOAP, JMS, and FTP,

- c) supporting various testing such as stress testing, distributed testing, web service testing, allowing user to record HTTP and HTTPS to create test plan,
- d) having loads of plugins which we can installed via Option > Plugin Manager and support dashboard report generation.

### 5.2.1 Requirements

- Make sure you got java installed (Requires Java 8+)
- Install JMeter via <http://jmeter.apache.org>
- Download the Plugins Manager [JAR file](#) and put it into JMeter's lib/ext directory. Then start JMeter and go to "Options" menu to access the Plugins Manager.



### 5.2.2 JMeter - overview of how to use

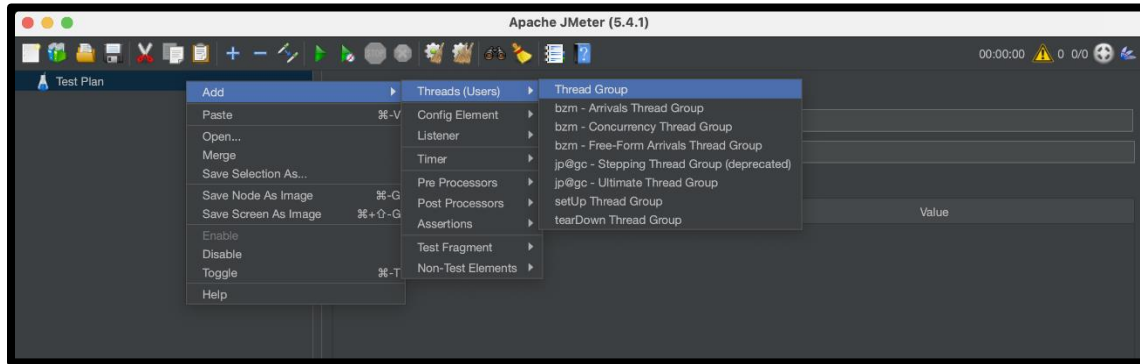
JMeter is conceptualized around a concept called “Test Plan”. To create a test plan which is the central execution unit of a test, we first need to understand the four important elements of a test plan:

1. Thread Group
2. Samplers
3. Listeners
4. Configuration

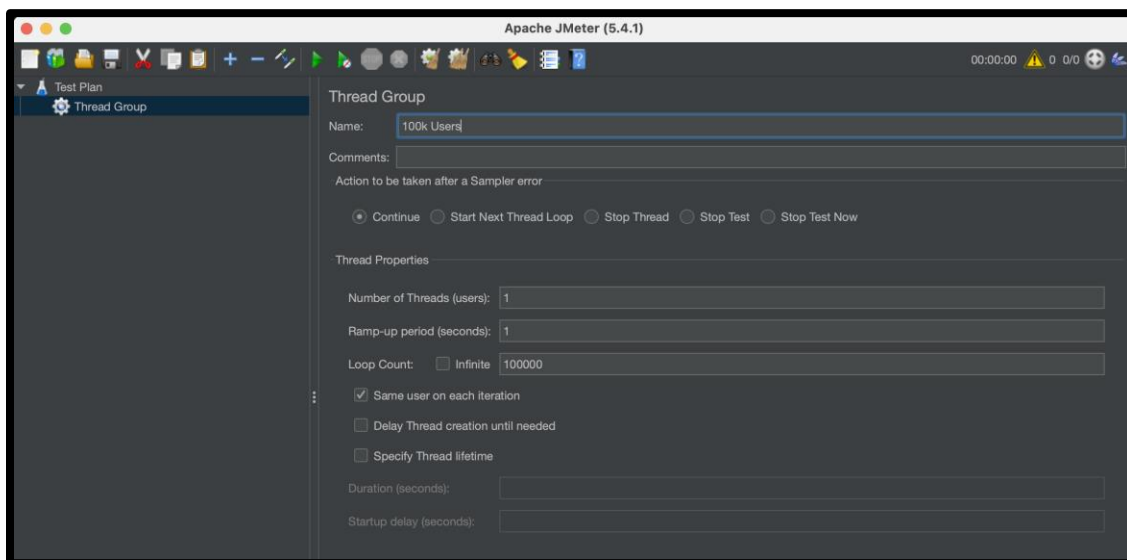
#### Thread group

- Represents a collection of threads. Each thread represents one user. Basically, in a test plan, we must specify how many users you want. If you set 100 threads, it simply means you have 100 users.

- Note that by default, these threads are not concurrent. If you would like to use concurrent threads, go to Option > Plugin Managers, and install Custom Thread Group which allows for testing concurrent threads.
- In the JMeter interface, under Test Plan, the first thing you want to do is create a thread group by right clicking like the following:



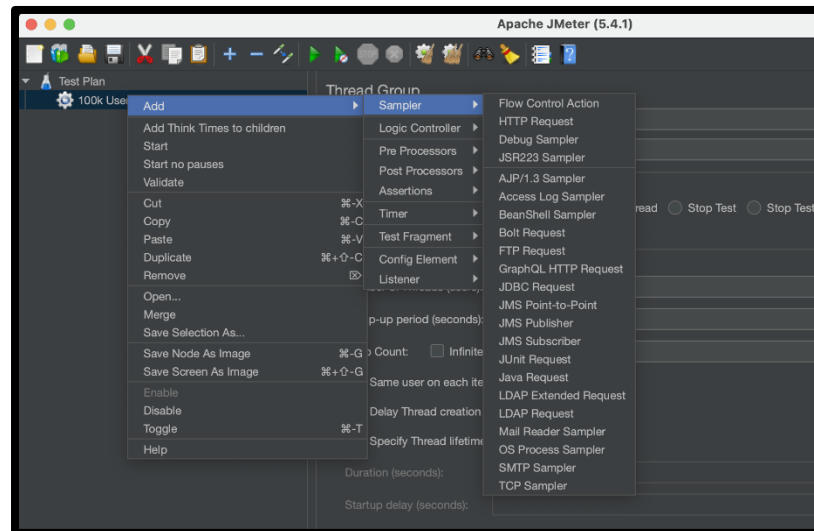
- Under the threads group, we can specify 100k users in the Name and Loop Count (100000) fields like this:



## Samplers

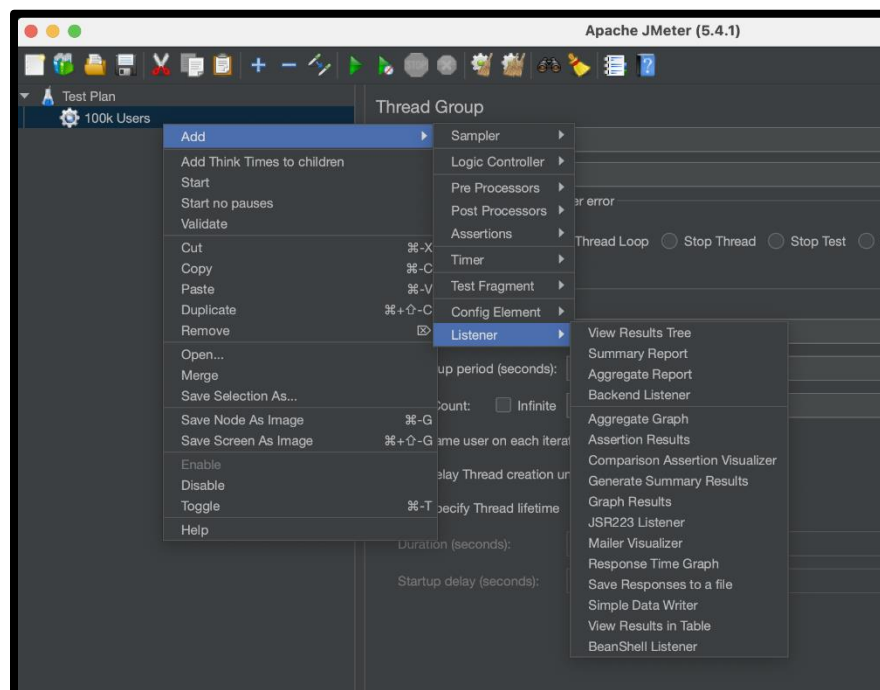
- As we have mentioned already, JMeter supports testing several protocols such as HTTP, FTP, JDBC, and many more. For example, if we want to send a download request to a server, then we can use FTP requests for our thread groups; if we want to check how a certain query will run in our database, use JDBC request for our thread groups (before we can use, we need to set the connection configuration - see Configuration); if we want to check how our web service works, use HTTP request. If we want to check how our email will work, use SMTP Sampler.

- Under the Thread Group we have just created, we can right click, select Sampler, and we will see many possible protocols we can test.
- We shall look at specifically HTTP requests and JDBC requests later in our short demo section.



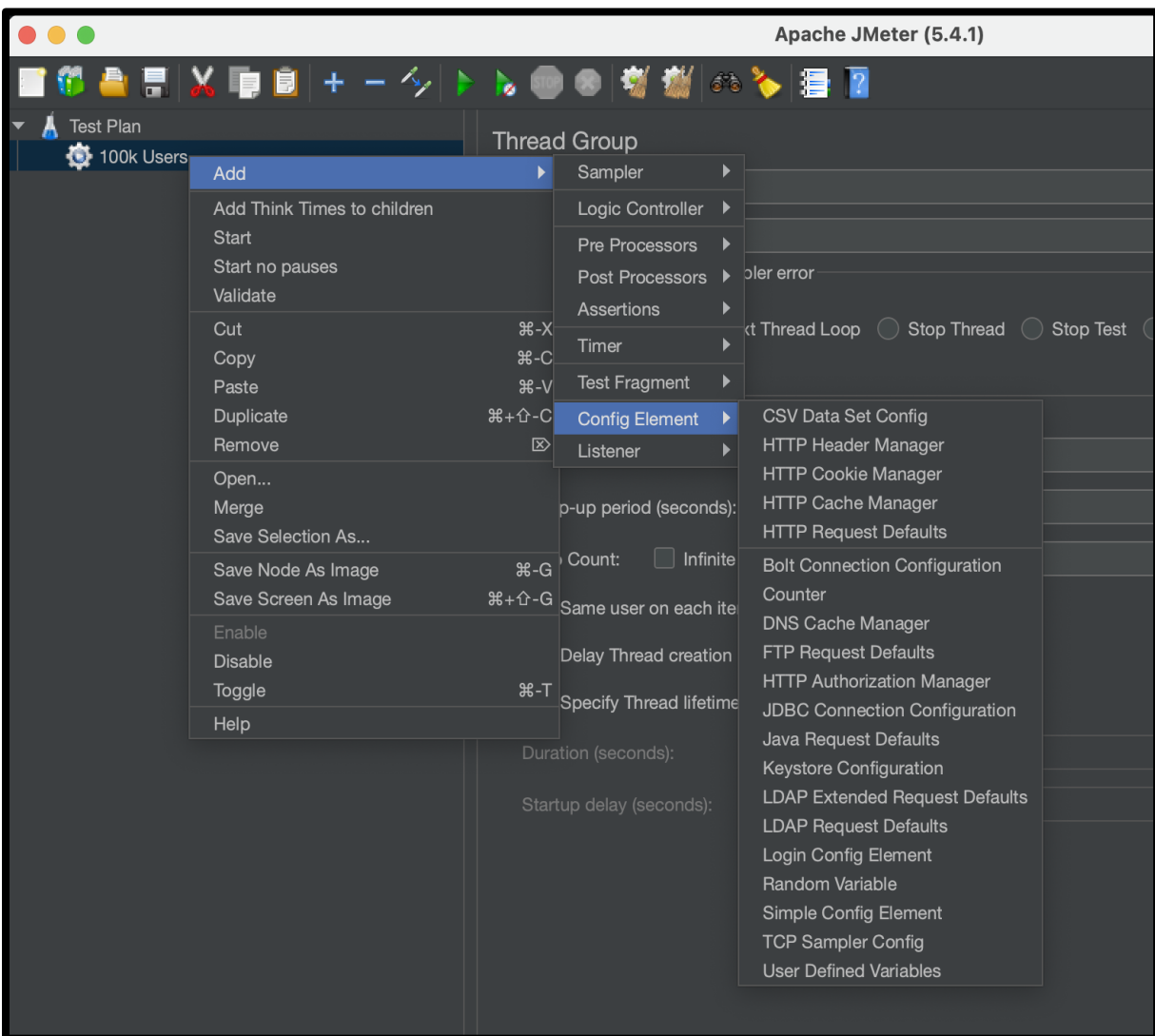
## Listeners

- Once we set the Thread Group and the associated Samplers, we want to see the results and that's the Listeners.
- JMeter supports many nice visualizations in (1) graph, (2) table, (3) tree, and (4) text format.



## Configuration

The last element to set for Thread Group is Configuration, which is about setting up some variables such as database connection. For example, for CSV Data set Config, we can put in any variable we want in a CSV format which will be automatically parsed and can be used when we script. HTTP Cookie will store all cookies of a particular website for future requests. HTTP Request Defaults set some default variables such as domain; so instead of typing <http://google.com> as our target 100 times, we can just config one time here. Login Config Element allows us to set up the username and password in a user request.

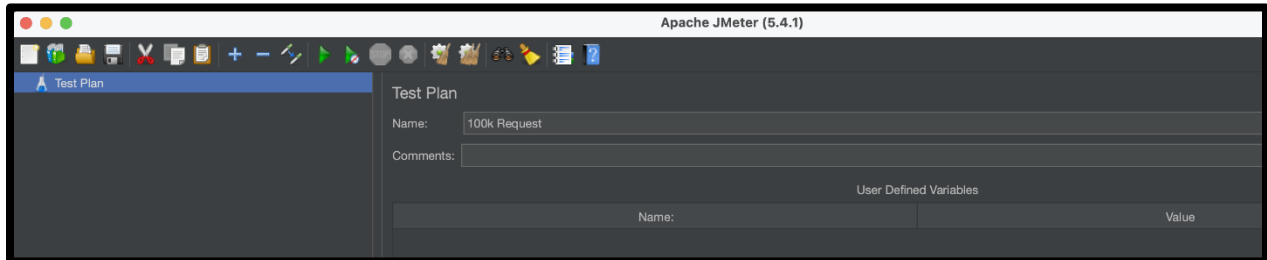


### 5.2.3 Testing Demo

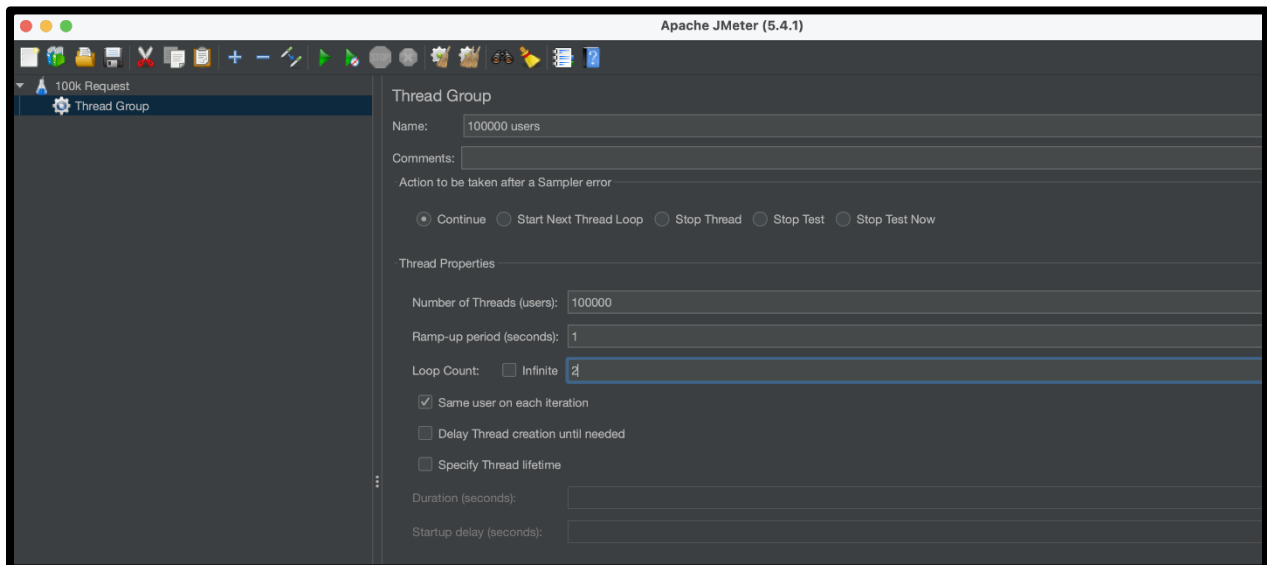
#### Demo 1: Simple HTTP Request

Very basic HTTP Request load testing

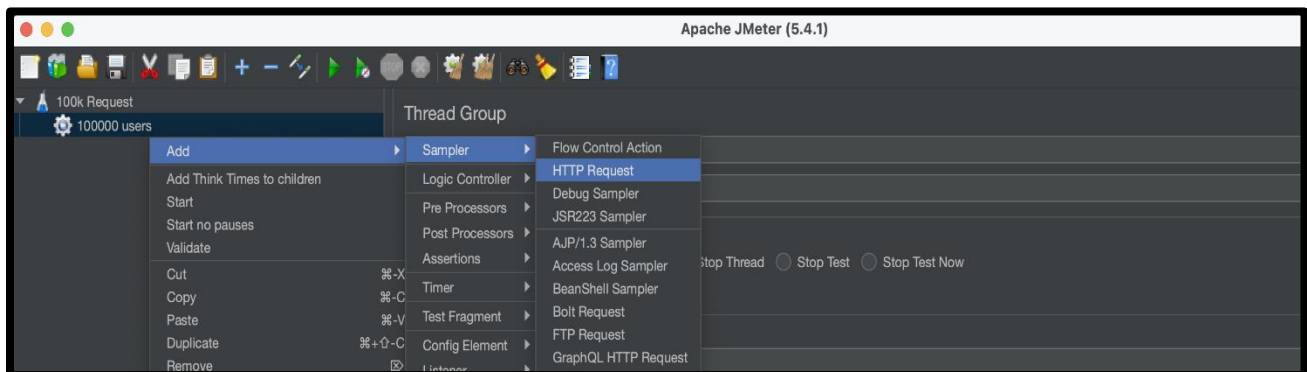
1. First let's name the Test Plan as 100k Request



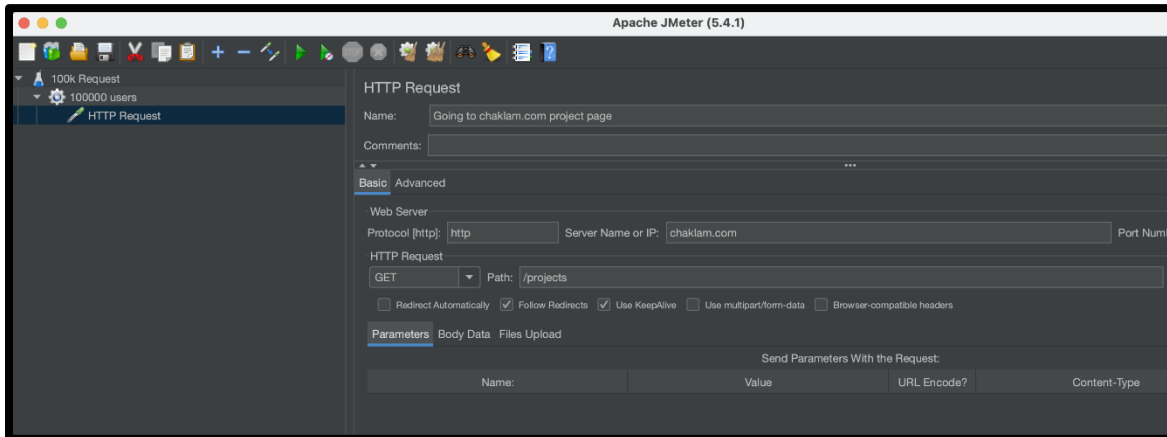
2. Right click the Test Plan and add Thread Group. In the Thread Group, put any Name, and specify the Number of threads and Loops.



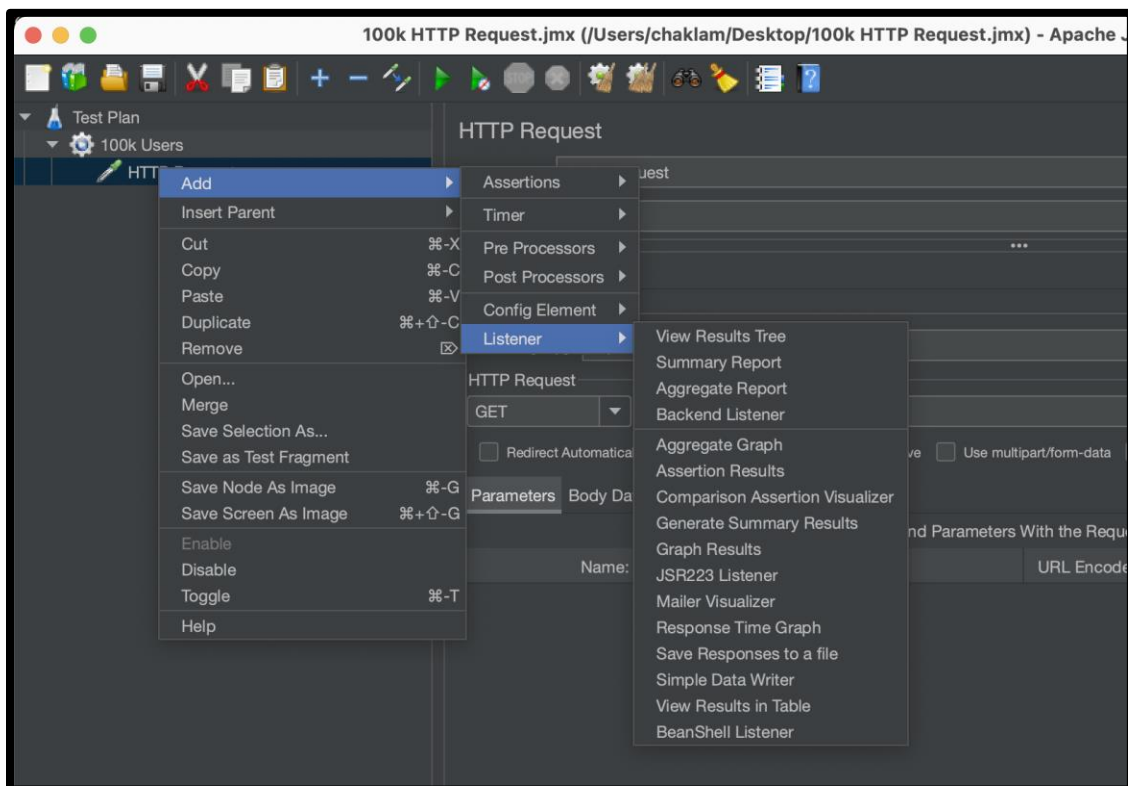
3. Once we have the Thread Group, we specify what are the testing protocols we will be using. Right click on the Thread Group and add Sampler > HTTP Request.



4. Specify the HTTP Request parameters as
  - a. Protocol: HTTP
  - b. Server name: chaklam.com
  - c. Path: /projects

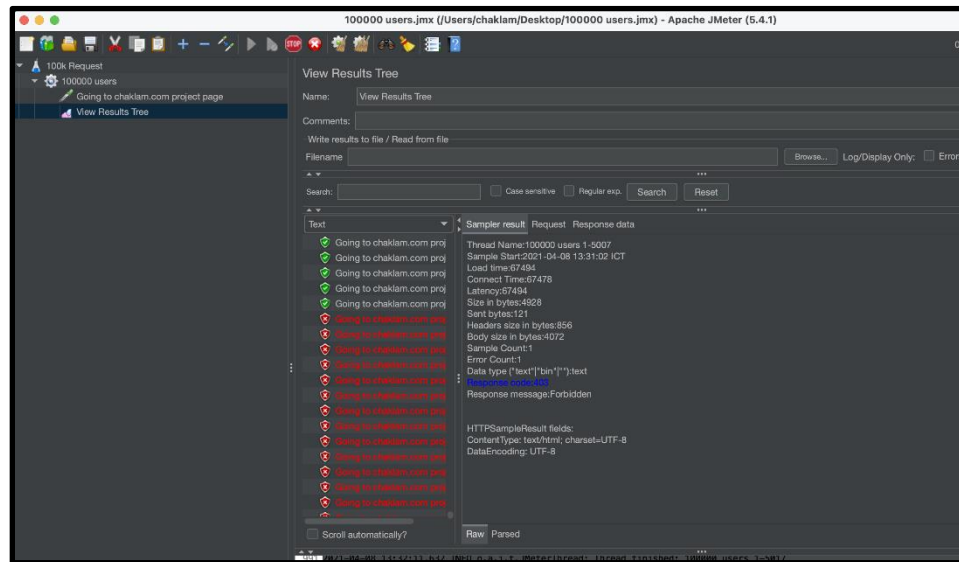


5. Next, before we actually run the test, let's add some visualization so we can better understand our result. Right click on the Thread Group and add Listener > View Result Tree





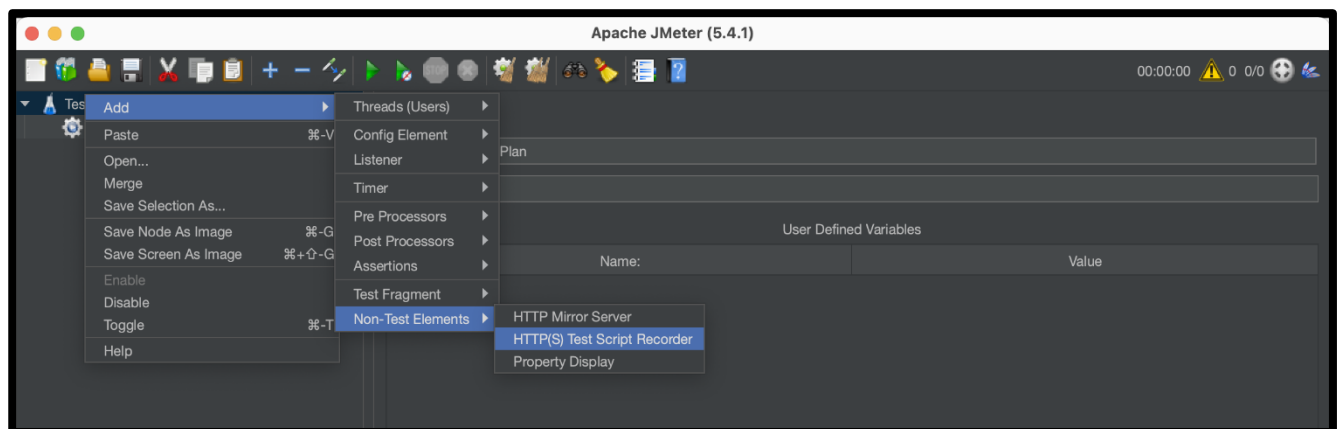
- Before you click the Play (Green Triangle) button, the system will ask us to save. Save it so that we can reuse this whole Test Plan for our other project. Here is the result of my test result. As we can see, website can hardly handle 100k users as there are way too many reds.



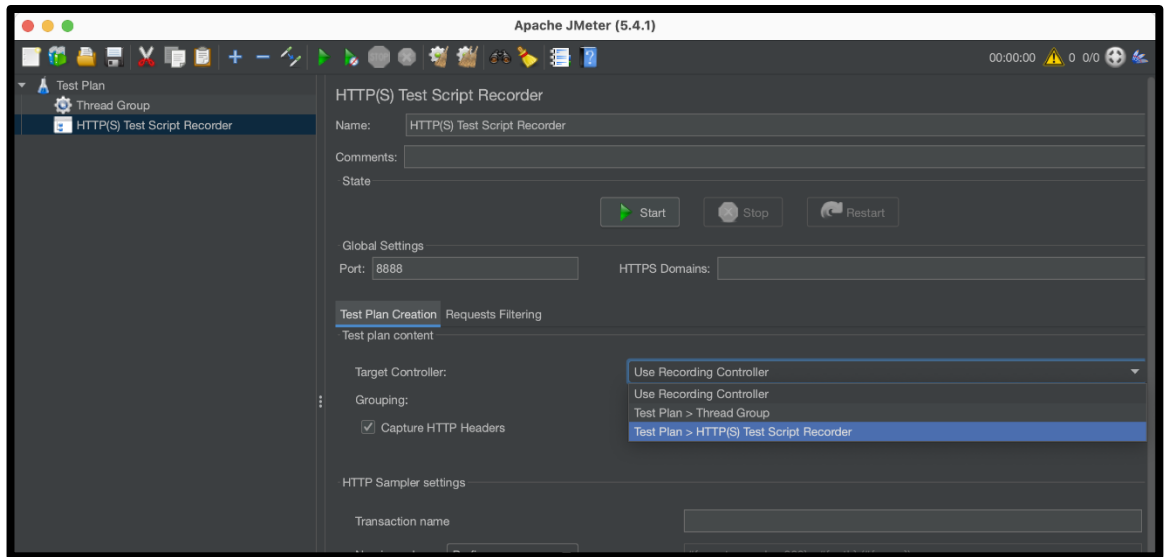
## Demo 2: Recording

Sometimes we are lazy to configure the HTTP Request, we can use the Record feature to record all our actions so we can replay all our action as a test procedure.

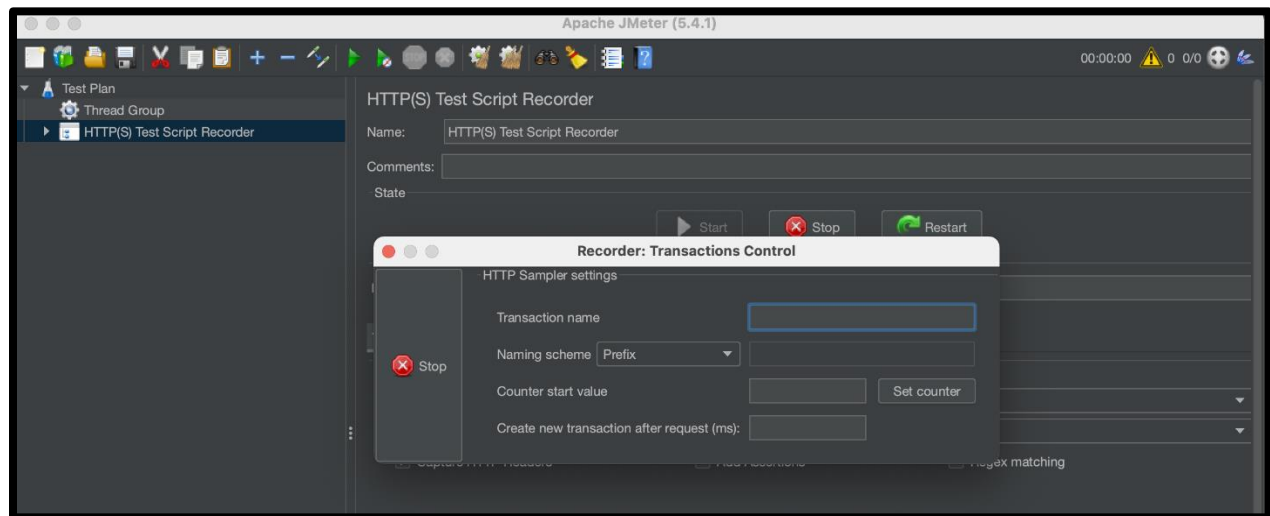
- First, create a Thread Group. Then, under the Test Plan, add Non-Test Elements > HTTP(s) Test Script Recorder



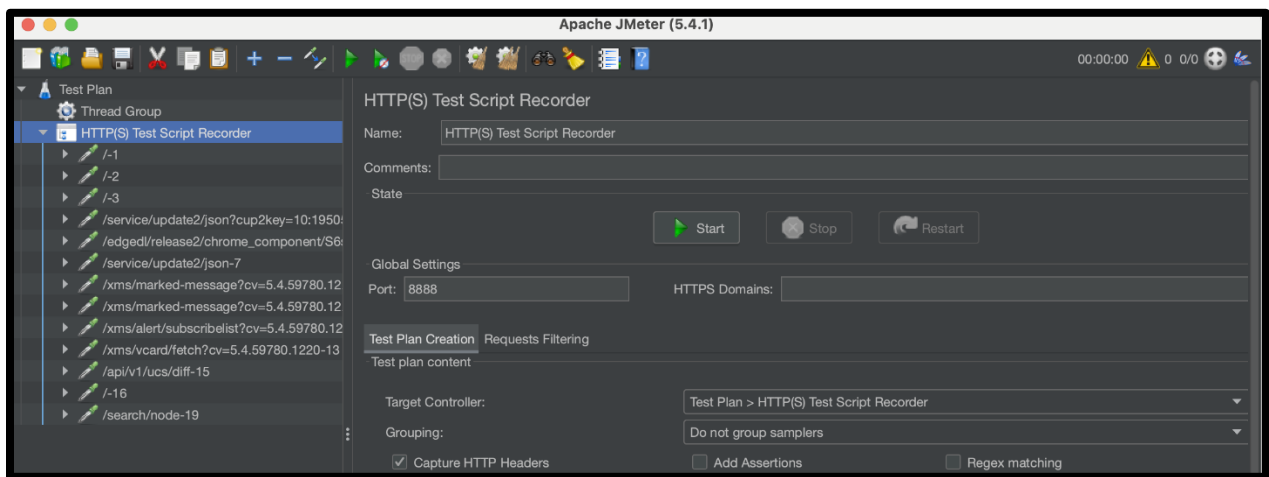
- Set the Port to whatever we want. I will leave the default 8888. For the Target Controller, we select the last option. You can actually choose Use Recording Controller which will give you more options, but we will keep it simple for demonstration.



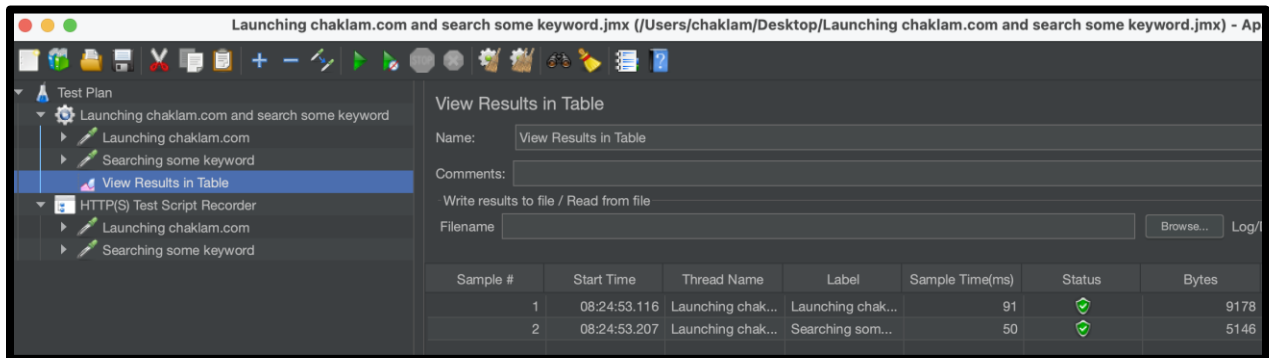
3. If we are Window users, you have to install JMeter certificate in our browser, refer to this manual → [Apache JMeter - User's Manual: Component Reference](#) (Search *Installing the certificate in Chrome or Internet Explorer*)
4. Now, we can press Start. You will see that the Recorder is now recording any activities running at port 8888



5. What we can do is visit some website and search for some keywords, and then press Stop. If everything is right, you will see there are many samplers recorded as the picture below shown.



6. Add some Listeners and we are good to go!



## 6. Continuous integration

Continuous integration (CI) is the practice of automating the integration of code changes from multiple contributors into a single software project. It is a primary software development (Dev) and IT operations (Ops) best practice, allowing developers to frequently merge code changes into a central repository where builds and tests then run.

### 6.1 Setup Jenkins:

1. Download software from link given <https://www.jenkins.io/download/thank-you-downloading-windows-installer-stable/> for Windows
2. Install Jenkins (we can follow the process from class note)

## 6.2 Overview of Jenkins

### 6.2.1 Software Requirements for full cycle for CI:

#### **Define Repository**

- SVN : TortoiseSVN
- Git : GitHub

#### **Define Build Software**

- MS Build : version VS 2017 or later
- DOT NET CORE SDK 2.2
- Java: version
- IONIC

#### **Deploy**

- IIS
- Tomcat
- Android

#### **Testing Software**

- Robot framework
- Selenium
- Selenium 2 library
- Chrome driver
- POSTMAN
- JMeter ( <https://www.jenkins.io/doc/book/using/using-jmeter-with-jenkins/> )

#### **Notification**

- Microsoft Team
- Slack

### 6.2.2 Jenkin prerequisites

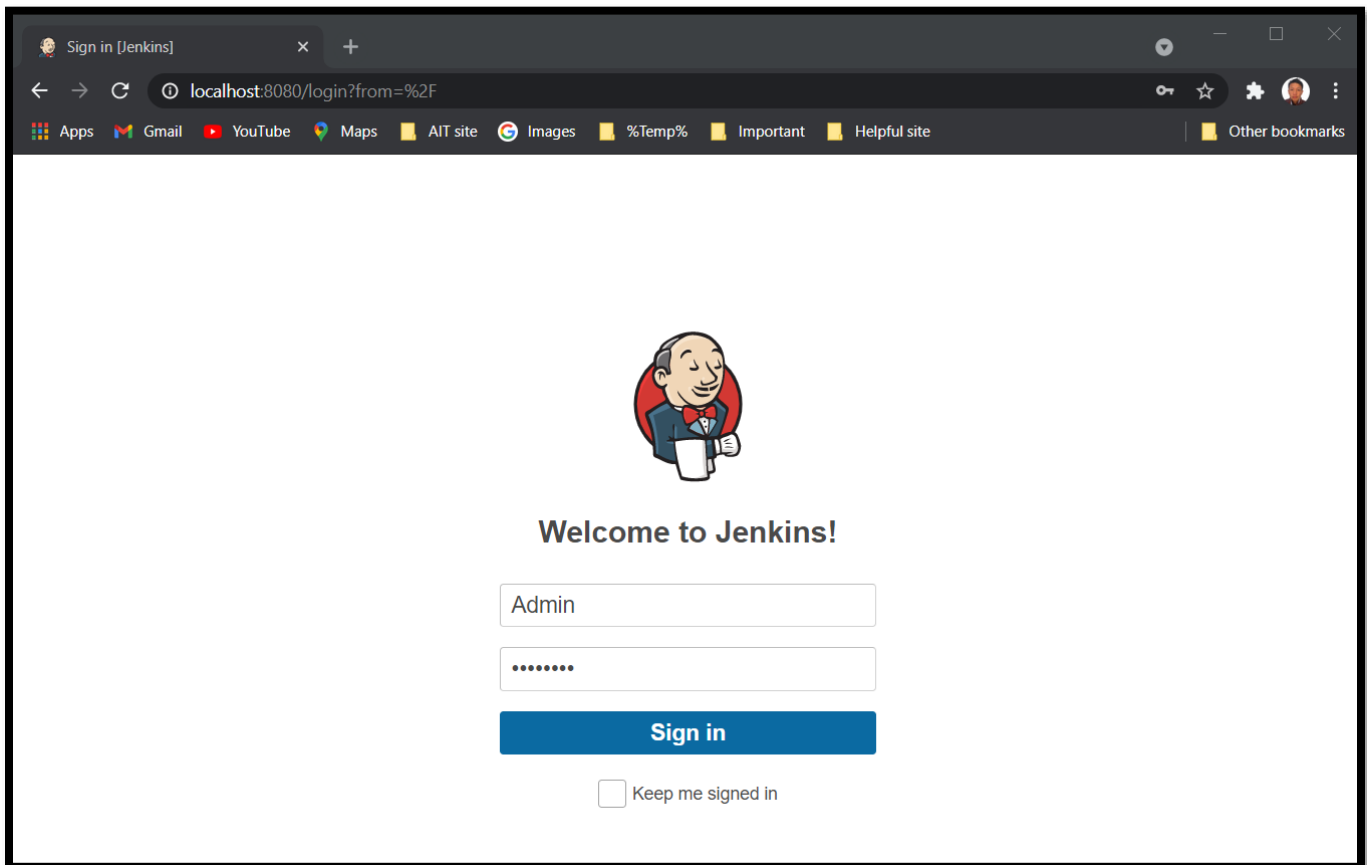
For this tour, you will require:

- A machine with:
  - ✓ 256 MB of RAM, although more than 2 GB is recommended.
  - ✓ 10 GB of drive space

- The following software installed:
  - Java 8 or 11 (either a JRE or Java Development Kit (JDK) is fine)

### 6.2.3 Run Jenkins

1. Run `java -jar jenkins.war --httpPort=8080`.
2. Browse to `http://localhost:8080`.
3. Follow the setup instructions to complete the installation.



### 6.2.4 Creating Pipeline

Jenkins Pipeline (or simply "Pipeline") is a suite of plugins which supports implementing and integrating continuous delivery pipelines into Jenkins.

A continuous delivery pipeline is an automated expression of our process for getting software from version control right through to our users and customers.

Jenkins Pipeline provides an extensible set of tools for modeling simple-to-complex delivery pipelines "as code". The definition of a Jenkins Pipeline is typically written into a text file (called a Jenkinsfile) which in turn is checked into a project's source control repository.

Dashboard [Jenkins]

localhost:8080

Apps

Gmail

YouTube

Maps

AIT site

Images

%Temp%

Important

Helpful site

Other bookmarks

Jenkins

search

Younten

log out

Dashboard

New Item

People

Build History

Manage Jenkins

My Views

Lockable Resources

New View

Build Queue

No builds in the queue.

add description

All

+

S	W	Name ↓	Last Success	Last Failure	Last Duration
		Jenkins_TRY	3 min 0 sec - #1	N/A	2.6 sec

Icon:

S M L

Legend

Atom feed for all

Atom feed for failures

Atom feed for just latest builds

All [Jenkins]

/all/builds

localhost:8080

Maps

AIT site

Images

%Temp%

Important

Helpful site

Other bookmarks

Jenkins

search

Younten

log out

Dashboard > All

New Item

People

Build History

Manage Jenkins

My Views

Lockable Resources

New View

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

Build History of Jenkins

This history is not guaranteed to include all subtasks executed on the node, e.g. Jenkins Pipeline subtasks will not be displayed.

Apr 20

Apr 21

Apr 22

Jenkins\_TRY #1

2hr

3hr

4hr

5hr

Build	Time Since ↑	Status
Jenkins_TRY #1	3 min 44 sec	stable

After adding item and build:

We can see the output for the jobs that we have added or created and check the build history.

The screenshot shows a web browser window displaying the Jenkins interface. The address bar shows the URL `localhost:8080/job/Jenkins_TRY/1/console`. The Jenkins logo is visible in the top left, and the user 'Younten' is logged in. The main content area is titled 'Console Output' and shows the following text:

```
Started by user Younten
Running as SYSTEM
Building in workspace
C:\WINDOWS\system32\config\systemprofile\AppData\Local\Jenkins\.jenkins\workspace\Jenkins_TRY
[Jenkins_TRY] $ cmd /c call C:\WINDOWS\TEMP\jenkins527334089427550948.bat

C:\WINDOWS\system32\config\systemprofile\AppData\Local\Jenkins\.jenkins\workspace\Jenkins_TRY>echo
"Hello.. this is my first Jenkins Demo : Wed 04/21/2021 : 3:14:02.84 "
"Hello.. this is my first Jenkins Demo : Wed 04/21/2021 : 3:14:02.84 "

C:\WINDOWS\system32\config\systemprofile\AppData\Local\Jenkins\.jenkins\workspace\Jenkins_TRY>exit 0
Finished: SUCCESS
```

On the left sidebar, there are links for 'Back to Project', 'Status', 'Changes', 'Console Output' (which is selected), 'View as plain text', 'Edit Build Information', and 'Delete build '#1''. The bottom right corner of the interface shows 'REST API' and 'Jenkins 2.277.3'.