Hashim Tayyab(19L-1013)

Hassan Hadayat(19L-0940)

**Software Testing Project Phase 3 Documentation**

**Explore the API test automation and Unit Test frameworks.**

The api test automation tool used for this project in nightwatch js tool. We will be using the @nightwatch/apitesting plugin to perform our testing. SuperTest is a framework used for API testing which is integrated into the nightwatch and offer a very interactive API testing tool. It uses expect () function from chai library to perform assertions.

To start the testing process, we followed these steps:

* From a command line run npm init nightwatch@latest
* During the guided setup select end-to-end testing and TypeScript
* Continue through accepting defaults for the remaining prompts.
* Add the API testing plugin by running npm install @nightwatch/apitesting @types/supertest --save-dev
* Change plugins to plugins: ['@nightwatch/apitesting'] to use the API testing plugin of nightwatch

Then simply we had to add the type: npx nightwatch [test\_folder] to run the test.

**JEST framework used for unit testing**

The frontend code of Metabase is purely written in JavaScript so we found that jest would be the most preferred tool to perform unit testing.

Jest is an all-in-one testing framework that includes everything needed to write and run tests, including test runners, assertion libraries, and mock functions. Jest supports various types of testing, including unit testing, integration testing, and snapshot testing. Jest provides built-in support for common web technologies, including React, Vue.js, and TypeScript. Jest has a powerful command-line interface that simplifies running tests and provides detailed feedback on test results. Jest is highly configurable and extendable, with support for plugins and custom test matchers, making it suitable for a wide range of testing scenarios.

**Unit Testing:**

We performed unit testing using jest framework on modules that we got from the Metabase GitHub repository.

We performed tests on several different components such as:

* Login Component
* Password Panel
* Reset Password
* Update User

**Linter/Sast tool:**

For configuring the linter in our project we first downloaded the vscode extension *eslint* and then installed eslint into our projest using the command:

npm init @eslint/config

After adding the tool we could do complete static analysis of our code using linter which provided us with information about errors present in our code. The following video shows the linter tool configured in our project:

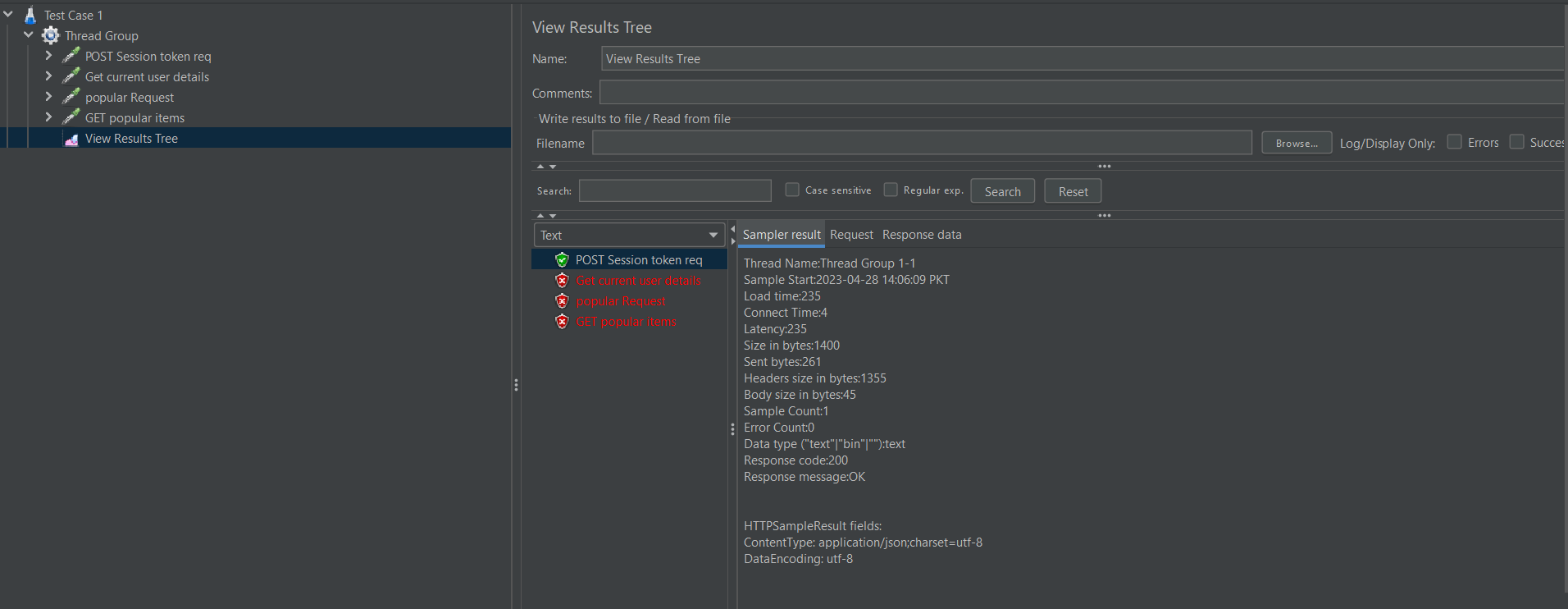
<https://www.loom.com/share/ff46b31b109c46809da66caa7fd84e2c>

**Performance Testing:**

We used JMeter for the process of performance testing. We send API requests using JMeter and observed their performance.

JMeter is a popular open-source software for load testing, functional testing, and performance testing of web applications. It can be used to simulate a heavy load on a server, network or object to test its strength and performance under various real-world scenarios.

Below is a snapshot of one performance test performed by us.



**Dockerfile:**

We created the Dockerfile for running tests smoothly. It runs tests for the unit testing and api testing.

**Problems faced:**

Upon sending first request Metabase blocked the request saying unauthorized. This issue had to be resolved by fist sending the session POST request and obtaining a session token. This token is supposed to be send with each and every API request to be authorized to perform operation.

During performance testing we were not able to send request due to the default content-type being plain text so we had to add a header for configuring body content type as application/json.