P.E.S. UNIVERSITY

Department of Computer Science and Engineering

Session: Jan-May 2020

UE17CS355 – Web Technologies-II Lab

Project Phase – II

Test Report

Project Title: **SCARS - Scalable Charting and Research Service**

Section: 6H

Team Members:

Vishwas Rajashekar PES1201700704

Sarang Ravindra PES1201700972

Amardeep MJ PES1201700149

System Testing

By: Vishwas Rajashekar PES1201700704

Introduction

System Testing is being done using Selenium. Selenium is a popular tool used to take control of and automate the browser. This allows us to perform a set of actions on the browser automatically using a script.

Objective

The objective of System Testing is to ensure that the web interface and the backend APIs are fully functional and work as expected from a user's point of view when they are using the fully functional software.

Test Report (Note: Login cases are only included for completeness - there are more than 10 cases covered since Login is not acceptable according to instructions given in the email)

Case No.	Description	Expected output	Actual Output	Result (Pass/Fail)
1	Signing up for an account - new account	Alerts successful signup	Alerts successful signup	Pass
2	Signing up for an account - account already exists	Alerts Error User exists	Alerts Error User exists	Pass
3	Logging in - Without an account	Alerts Error	Alerts Error	Pass
4	Logging in - With an account	Takes user to Dashboard	Takes user to Dashboard	Pass
5	Trying to go to dashboard without logging in	Shows error and redirects to the login page	Shows error and redirects to the login page	Pass

6	File Upload test - New file (.csv)	File uploaded successfully	File uploaded successfully	Pass
7	File Upload - same file again (.csv)	File already exists alert	File already exists alert	Pass
8	File Upload - new file wrong type (.tar.gz)	Incorrect file type error	Server returns error 500 due to decoding	Fail
9	Plot a file - valid file	Plot page opens up and shows graphs	Plot page opens up and shows graphs	Pass
10	Plot a file - invalid file	New tab should open up with a warning mentioning the file doesn’t exist	Plot page opens up and shows no features or targets	Fail
11	Delete a file	Alert mentioning that file has been deleted	Alert mentioning that file has been deleted	Pass
12	View an uploaded file - valid file	View page opens and shows data	View page opens and shows data	Pass
13	View an uploaded file - invalid file	View page opens and shows warning that file is invalid	View page opens and shows nothing	Fail

14	Create a new model	Model creation page shows up with features and targets. Predict creates a model.	Model creation page shows up with features and targets. Predict creates a model.	Pass
15	Evaluate an existing model	Model evaluation page shows up with the parameters to be entered. Clicking predict gives a result	Model evaluation page shows up with the parameters to be entered. Clicking predict gives a result	Pass
16	Delete a model	Alert saying model has been deleted	Alert saying model has been deleted	Pass
17	Evaluate non-existing model	Evaluate page shows up with warning to user	Evaluate page shows up with no warning, just empty content	Fail

Observation and Conclusion

The application is working as per requirement. Except for the file upload of a different type, the user cannot directly face a bug using the interface. Out of 17 test cases, 4 cases have failed - all of which handle an invalid case. This case does not affect the normal user's experience.

Unit testing

By: Sarang Ravindra PES1201700972

Introduction

The application deals with charting and training neural networks on desired files. This is a unit test done by checking API functioning through postman.

Objective

The objective of this test is to verify if the important APIs of the function work as expected and do not give incorrect results or fail.

Test Report

Case No.	Description	Expected output	Actual Output	Result (Pass/ Fail)
1	Signing up for an account - new account	200 OK	200 OK	Pass
2	Login	200 OK	200 OK	Pass
3	Get file list - valid user	200 OK / 204 No Content	200 OK	Pass
4	Get file list - invalid user	406 Not acceptable	500 Error	Fail
5	New file - valid file	200 OK	200 OK	Pass
6	New file - invalid file	406 Not acceptable	500 Error	Fail
7	Get features - valid file	200 OK	200 OK	Pass
8	Get features - invalid file	406 Not acceptable	500 Error	Fail
9	Get Scatter - valid	200 OK	200 OK	Pass

	file			
11	Get Scatter - invalid	406 Not Acceptable	500 Error	Fail
12	Train - valid file	200 OK	200 OK	Pass
13	Train - invalid file	406 Not Acceptable	500 Error	Fail

Observation and Conclusion

The application passed all normal use case scenarios. However a few finer aspects of the implementation of the APIs were not handled appropriately and caused a buggy experience. However appropriate status codes were received as expected in most cases.

Load (Performance) Testing

By: Amardeep MJ PES1201700149

Introduction

Load testing a type of performance testing that simulates real-world load on any software, application, or website. It examines how the system behaves during normal and high loads and determines if a system, piece of software, or computing device can handle high loads given a high demand of end users. For this project we use locust which is an open source load testing tool written in python. The target of locust is load-testing web sites and checking the number of concurrent users a system can handle. During a locust test, a swarm of locusts will attack the target i.e. website. The behaviour of each locust is configurable and the swarming process is monitored.

Objectives

The objective is to check the performance of our website under different conditions of load i.e., by increasing the number of users accessing the website concurrently. We intend to check the working of our website, trying to swarm users and perform specific tasks and check for any failure conditions. We test 3 APIs i.e, Signup, Login and get_scatter with multiple requests and check for failures/success.

Test Report

Case No	Description	Expected Output	Actual Output	Result (Pass/Fail)
1	Stimulating 100 users at 5/s in 10s	Perform scatter tasks without failure and login/signup with possible failures	No failures in all tasks	Pass

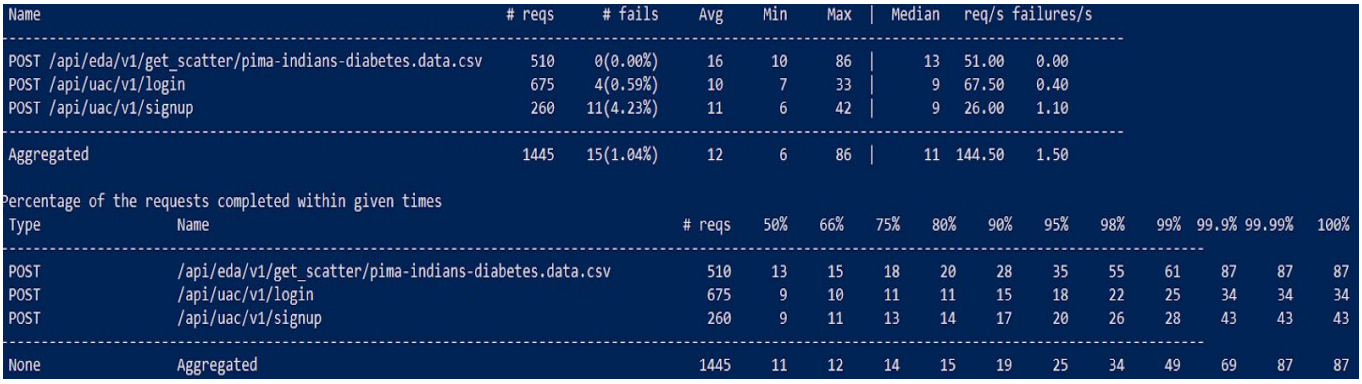
2	Stimulating 200 users at 50/s in 10s	Perform scatter tasks without failure and login/signup with possible failures	No failure in scatter tasks and login. 1 request failure in signup	Pass
3	Stimulating 400 users at 10/s in 20s	Perform scatter tasks without failure and login/signup with possible failures	No failure in scatter tasks. 5 failure in signups and 1 in login	Pass
4	Stimulating 400 users at 10/s in 10s	Perform scatter tasks without failure and login/signup with possible failures	No failure in scatter tasks and login. 2 request failure in signup	Pass
5	Stimulating 300 users at 10/s in 20s	Perform scatter tasks without failure and login/signup with possible failures	No failure in scatter tasks. 4 failure in signups and 2 in login	Pass
6	Stimulating 200 users at 10/s in 20s	Perform scatter tasks without failure and login/signup with possible failures	No failure in scatter tasks and login. 6 request failure in signup	Pass
7	Stimulating 400 users at 5/s in 20s	Perform scatter tasks without	No failure in scatter tasks and login. 2	Pass

		failure and login/signup with possible failures	request failure in signup	
8	Stimulating 600 users at 10/s in 10s	Perform scatter tasks without failure and login/signup with possible failures	All task successful	Pass
9	Stimulating 1000 users at 10/s in 10s	Perform scatter tasks without failure and login/signup with possible failures	No failure in scatter tasks and login. 2 request failure in signup	Pass
10	Stimulating 1000 users at 90/s in 10s	Perform scatter tasks without failure and login/signup with possible failures	No failure in scatter tasks. 11 failure in signups and 4 in login	Pass
11	Stimulating 1000 users at 150/s in 10s	Perform scatter tasks without failure and login/signup with possible failures	No failure in scatter tasks and login. 17 request failure in signup	Pass
12	Stimulating 1500 users at 150/s in 10s	Perform scatter tasks without failure and login/signup with possible	No failure in scatter tasks. 23 failure in signups and 1 in login	Pass

		failures		

An image of the data we get when the testing is done:

Case 10:



Name	# reqs	# fails	Avg	Min	Max	Median	req/s	failures/s
POST /api/eda/v1/get_scatter/pima-indians-diabetes.data.csv	510	0(0.00%)	16	10	86	13	51.00	0.00
POST /api/uac/v1/login	675	4(0.59%)	10	7	33	9	67.50	0.40
POST /api/uac/v1/signup	260	11(4.23%)	11	6	42	9	26.00	1.10
Aggregated	1445	15(1.04%)	12	6	86	11	144.50	1.50

Percentage of the requests completed within given times

Type	Name	# reqs	50%	66%	75%	80%	90%	95%	98%	99%	99.9%	99.99%	100%
POST	/api/eda/v1/get_scatter/pima-indians-diabetes.data.csv	510	13	15	18	20	28	35	55	61	87	87	87
POST	/api/uac/v1/login	675	9	10	11	11	15	18	22	25	34	34	34
POST	/api/uac/v1/signup	260	9	11	13	14	17	20	26	28	43	43	43
None	Aggregated	1445	11	12	14	15	19	25	34	49	69	87	87

Observation and Conclusion

From the load testing, we observe that 100% of test cases passed for get_scatter with a negligible failure rate of signups/login due to the disability of allowing usernames with the same credentials. We also observe that average time and max time for response increases with increasing numbers of users.