Peformance Testing of SAFE application

An Research and Development Report
Submitted in partial fulfillment of
the requirements for the degree of
Master's in Sciences
by

Jatin Lachhwani
(21Q050005)

Supervisor:
Bhaskaran Raman

Department of Computer Science and Engineering
Indian Institute of Technology Bombay
Mumbai 400076 (India)

2 December 2021

Acceptance Certificate

Department of Computer Science and Engineering
Indian Institute of Technology, Bombay

The Research and Development report entitled "Peformance Testing of SAFE application" submitted by Jatin Lachhwani (21Q050005) may be accepted for being evaluated.

Date: 2 December 2021

Bhaskaran Raman

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I declare that I have properly and accurately acknowledged all sources used in the production of this report. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Jatin Lachhwani

Date: 2 December 2021 (21Q050005)

Abstract

SAFE(Smart Authenticated Fast Exams) is a fully fledged application at this stage, serving thousands of users. In future , its use is expected to grow by a few orders of magnitude. As an application scales one of an important area to focus on becomes performance. This projects aims to create scripts that will allow SAFE to measure performance of the Quiz functionality and aims to identify areas where it can improve.

Table of Contents

Abstract **iii**

1 Introduction **1**

1.1 Objective 1

1.1.1 Project Requisites 1

2 Python Scripts for Testing SAFE **2**

2.1 Creating Students to simulate users 2

2.1.1 Create users 2

2.1.2 Add users to the Database 2

2.1.3 Turn added Users to Students 2

2.2 Creating & Adding Quiz and Generating Answers 3

2.2.1 Creating Quiz 3

2.2.2 Adding Quiz 4

2.2.3 Generating the Answers file 4

2.3 Performing Quiz through script 5

2.3.1 Enable an instance of a Quiz and Start it. 5

2.3.2 Providing the Quiz instance number to the script 5

2.3.3 Attempting the quiz through Script 5

2.4 Creating Performance Graph using Downloaded Data 8

2.4.1 Nomenclature of Downloaded Files 8

2.4.2 Create Graph for Response Time vs Number of Users for a particular Number of Questions 8

2.4.3 Create Graph for Response Time vs Number of Questions for a particular Number of Users 9

3 Conclusion **11**

4 Future Work and Repository link **12**

4.1 Future Work 12

4.2 Repository Links 12

Acknowledgements **13**

Chapter 1

Introduction

1.1 Objective

The Objective of the following project is to create Scripts that allow us to measure the performance of the SAFE app by modifying the following Parameters.

1. **Number of students attending the quiz.**
2. **Number of questions / quiz.**

1.1.1 Project Requisites

- How Django works?
- Figuring out APIs triggered while attempting a quiz
- Writing Scripts in Python
    1. A script to populate a given number of dummy users.
    2. A script to create a given number of dummy questions.
    3. To add modifications to the existing code to store the time values
    4. A script to simulate submissions by user
    5. a script to create performance graphs.
- Testing the scripts on a Test Server and using to create Performance Graphs

Chapter 2

Python Scripts for Testing SAFE

2.1 Creating Students to simulate users

2.1.1 Create users

File Name : **CreateCSV.py**
location : /SafeRnD/1.Users Create CSV/
input : numberOfStudents(int) , nameOfFile(string)
output : nameOfFile.csv(This will be used by **AddStudents.py**) , credentials.py(This will be used by **SafeQuizAttempt.py**)
Run : python **CreateCSV.py**

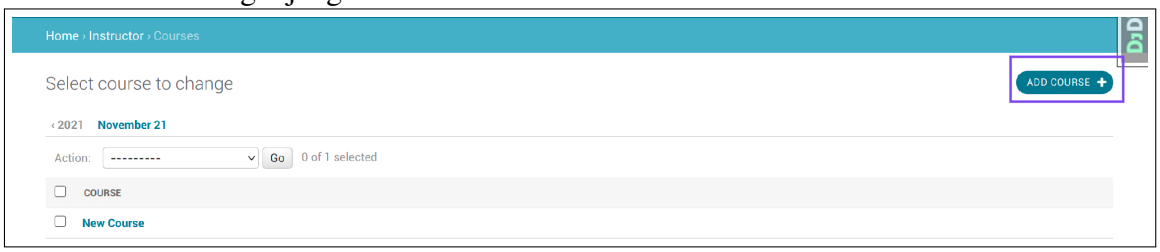
2.1.2 Add users to the Database

Note: Before using this script make sure to create an organization with which the students will be associated.Also create an instructor in the organization to which the user will be associated. Also turn on the Virtual Environment.
caution: Make sure you run this file by placing it in the **/safe_server_v2/backend** folder. In the same folder place the csv file generated in section 2.1.1
File Name : **AddStudents.py**
location : /SafeRnD/2. Script to add CSV to DB/
input : nameOfTheOrganization(string) , nameOfFile.csv(file)(This was generated by CreateCSV.py)
Impact : This will add all the users present in the csv to the database
Run : python **AddStudents.py**

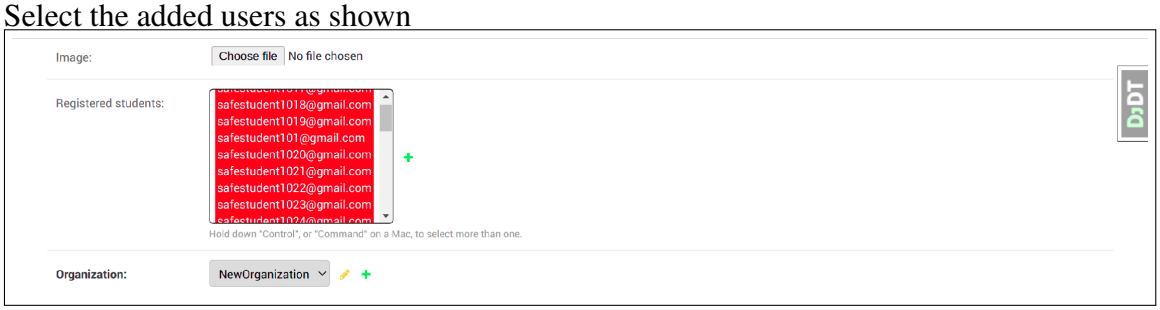
2.1.3 Turn added Users to Students

Note: This will be done through Django-admin interface

1. Add a Course using Django Interface



2. Select the added users as shown



3. Save the Changes

4. The students to perform the test are now available.

2.2 Creating & Adding Quiz and Generating Answers

2.2.1 Creating Quiz

File Name : **Questions.py**

location : /SafeRnD/3.Create and upload quiz and generate answers/

input : singleCorrect(int),setCorrect(int),numericType(int),nameOfQuestionFile(string)

output : nameOfQuestionFile.csv(uploaded using django login interface)

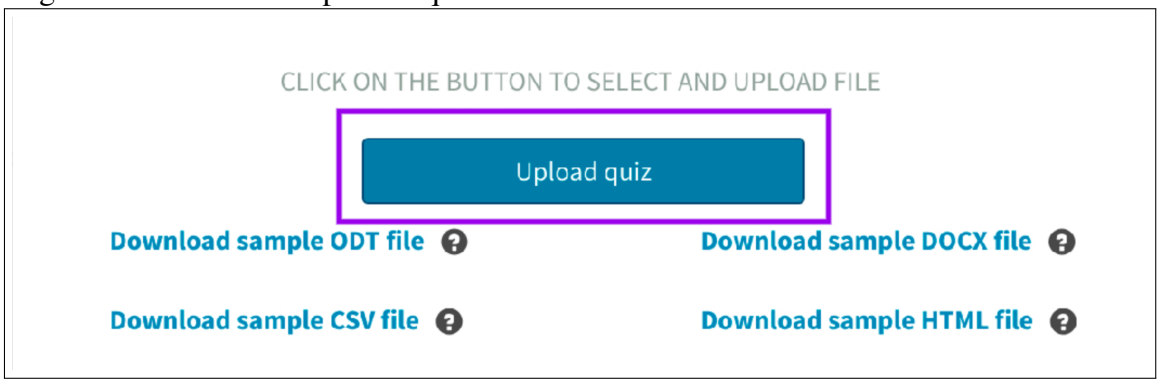
Run : python **Questions.py**

Note: The script is not yet over it will for you to enter the id ,generated after uploading the quiz, for generating answers.(see the image below)

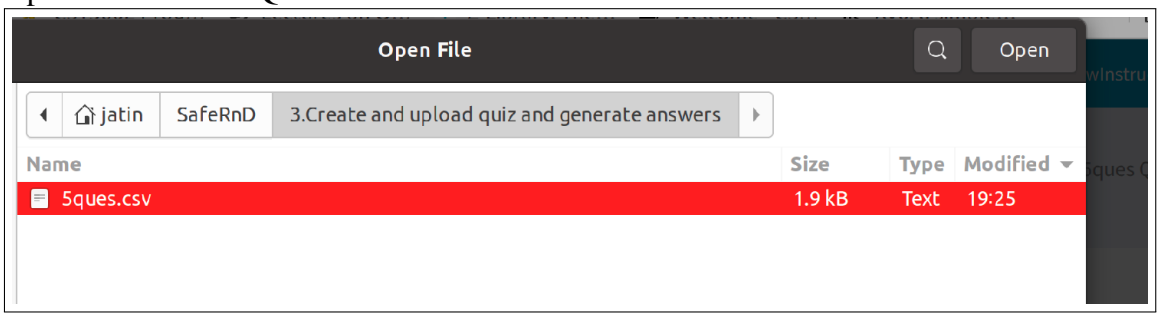
```
Enter the number of questions of single correct : 5
Enter the number of questions of setcorrect : 5
Enter the number of questions of numeric type : 5
Enter the name of the question file( without csv extension) : 5ques
Upload the 5ques.csv to the quiz app and publish the quiz that will generate the name of the quiz .
Note the id of the first question .
Enter the id of the first uploaded question from the quiz :
```

2.2.2 Adding Quiz

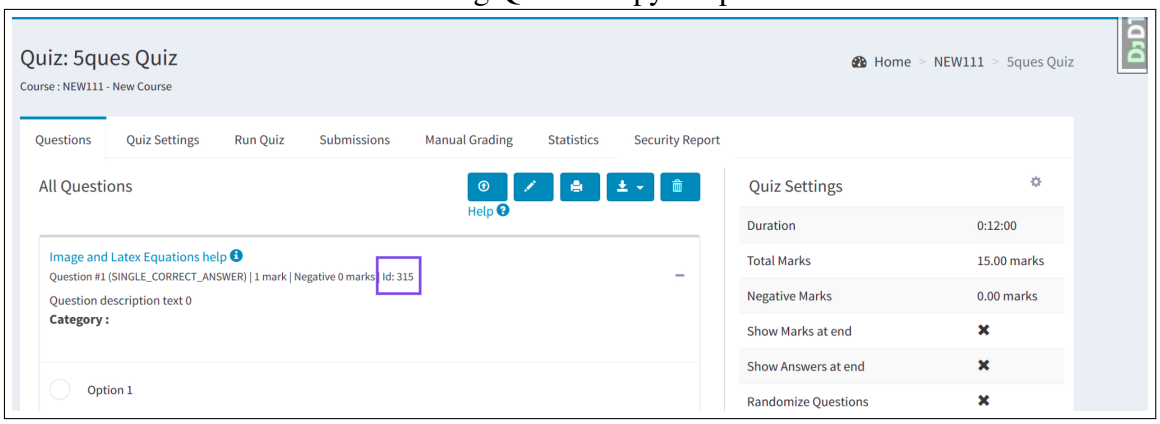
1. Login as instructor and upload a quiz



2. Upload the nameOfQuestionFile.csv



3. **Note the Id** and feed it to the waiting Questions.py script



2.2.3 Generating the Answers file

Note: The earlier used Questions.py was not closed yet it is waiting for the id and now we will provide it with the id to create the Answers file.

File Name : **Questions.py**

location : /SafeRnD/3.Create and upload quiz and generate answers/

input : id (int), nameOfAnswersFile(string)

output : nameOfAnswersFile.py(will be used by SafeQuizAttempt.py as an include file)

Note: The usage for this is shown below. Also refer to 3.2.1 for better understanding.

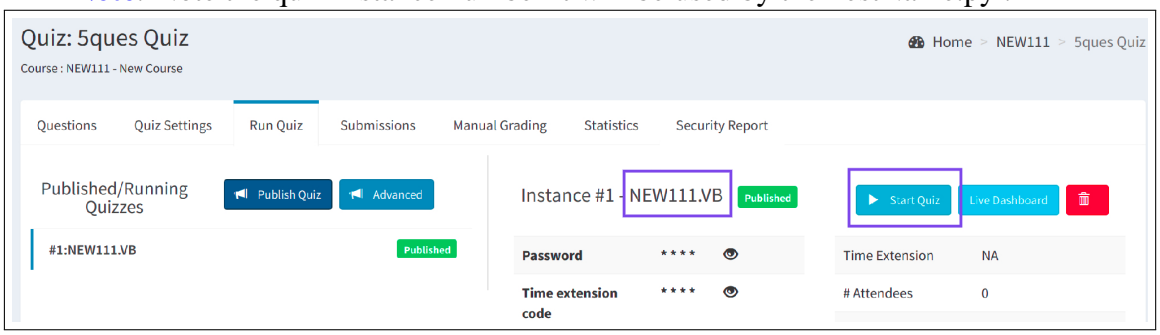
```
Note the id of the first question .
Enter the id of the first uploaded question from the quiz : 315
Enter the name of the answer file without the (.py extension) : Answers
jatin@Hawkeye:~/SafeRnD/3.Create and upload quiz and generate answers$ ls
5ques.csv  Answers.py  Questions.py
jatin@Hawkeye:~/SafeRnD/3.Create and upload quiz and generate answers$
```

2.3 Performing Quiz through script

Note: Turn on the virtual machine and django server before attempting the quiz. Also disable the Safe version check from the back end.

2.3.1 Enable an instance of a Quiz and Start it.

Note: Note the quiz instance number it will be used by the TestName.py .



2.3.2 Providing the Quiz instance number to the script

File Name : **TestName.py**

location : /SafeRnD/4.Attempt quiz/

input : quizInstanceNumber(int)

impact : modify the value in testname file(will be used by SafeQuizAttempt.py)

```
jatin@Hawkeye:~/SafeRnD/4.Attempt quiz$ python TestName.py
Enter the name of the quiz instance : NEW111.VB
```

2.3.3 Attempting the quiz through Script

Note: Before using this Script make sure to add the **credentials.py** generated by **CreateCSV.py** (Refer to Section 3.1.1). Also **include** the Answers **nameOfAnswersFile (Answers.py)** file generated in chapter 3.2.3 in **SafeQuizAttempt.py** .Also This file will run by using locust .

```
jatin@Hawkeye:~/SafeRnD/4.Attempt quiz$ ls
answers.py  credentials.py  SafeQuizAttempt.py  testname  TestName.py
```

File Name : **SafeQuizAttempt.py**

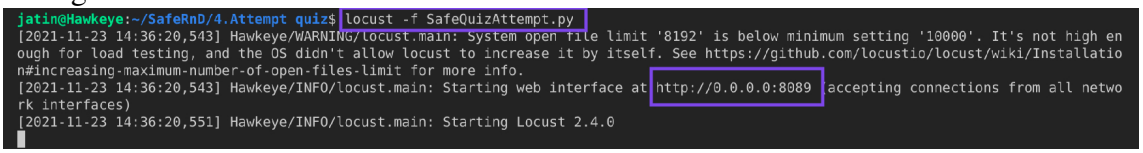
location : SafeRnD/4.Attempt quiz/

output : submit.txt(This contains the Response of the last API call (for debugging))

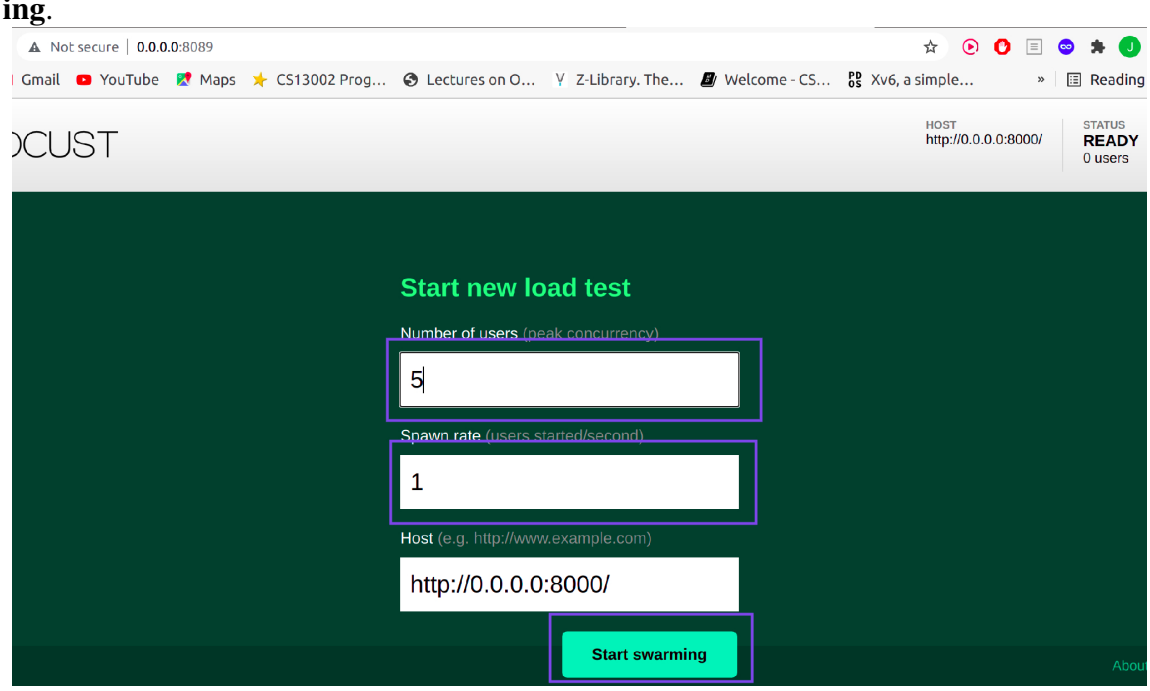
Impact : This will be used for running the Performance Test.

1. Run : locust -f **SafeQuizAttempt.py**.

This will turn on locust with the **SafeQuizAttempt.py** file. Go to the highlighted port through a browser.



2. Fill in the **number of users** to spawn and the **spawn rate** and then click **Start Swarming**.

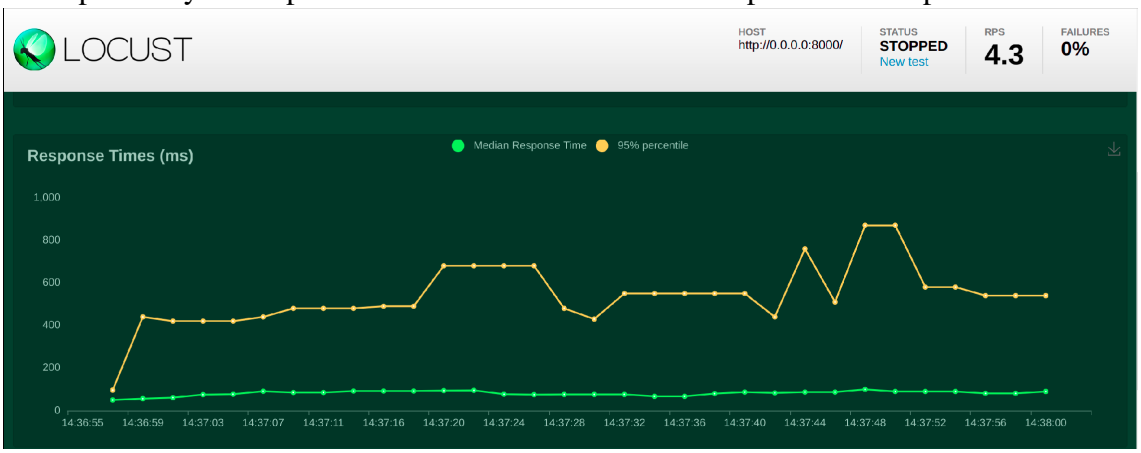


3. Locust will start executing the script , run it for the time period based on the type of analysis and then **stop** it. You can use the **Charts** created by locust or **Download Data** to do your own analysis

Type	Name	# Requests	# Fails	Median (ms)	90%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures/s
POST	SignInAttempt	20	0	97	220	120	76	274	192	0.8	0
POST	Submission	15	0	290	400	307	198	586	62	0.7	0
GET	authenticate	17	0	68	77	67	47	92	176	0.5	0
GET	course code	20	0	42	92	47	22	94	169	0.7	0
GET	course id	18	0	440	720	481	359	718	1064	0.5	0
GET	quiz content	18	0	46	74	51	35	101	1286	0.5	0
GET	safe_course	20	0	56	150	77	43	168	669	0.7	0
	Aggregated	**128**	**0**	**77**	**420**	**158**	**22**	**718**	**522**	**4.4**	**0**

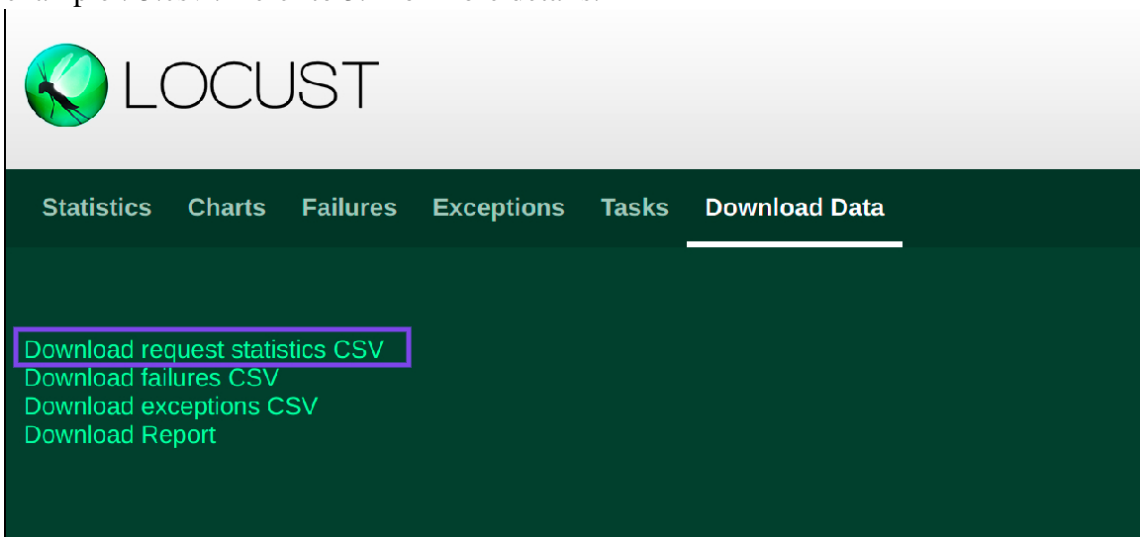
4. Using the existing **Charts** feature of locust.

It will provide you the performance characteristics for that particular script.



5. Downloading Data for further analysis.

Follow the naming convention where the name is the number of users followed by csv. example : 5.csv . Refer to 3.4 for more details.



2.4 Creating Performance Graph using Downloaded Data

2.4.1 Nomenclature of Downloaded Files

Note: This is necessary to create graphs using the Existing Scripts.
There are two Scripts for creating Performance graphs.

1. **QuestionsVsRT.py**
2. **UsersVsRT.py**

Suppose you have question papers with 10,20,30,40 and 50 questions and for each of them you run the test for 3,6,9,12 and 15 users.

While naming make sure that folders are named like "number_of_questions" followed by "q" e.g for 30 questions the folder should be named 30q

```
jatin@Hawkeye:~/SafeRnD/5.Analysis$ ls
10q  20q  30q  40q  50q  QuestionsVsRT.py  UsersVsRT.py
```

and each folder should contain all the user count for which it was run. e.g 3.csv for test case with 3 users.

```
jatin@Hawkeye:~/SafeRnD/5.Analysis/30q$ ls
12.csv  15.csv  3.csv  6.csv  9.csv
```

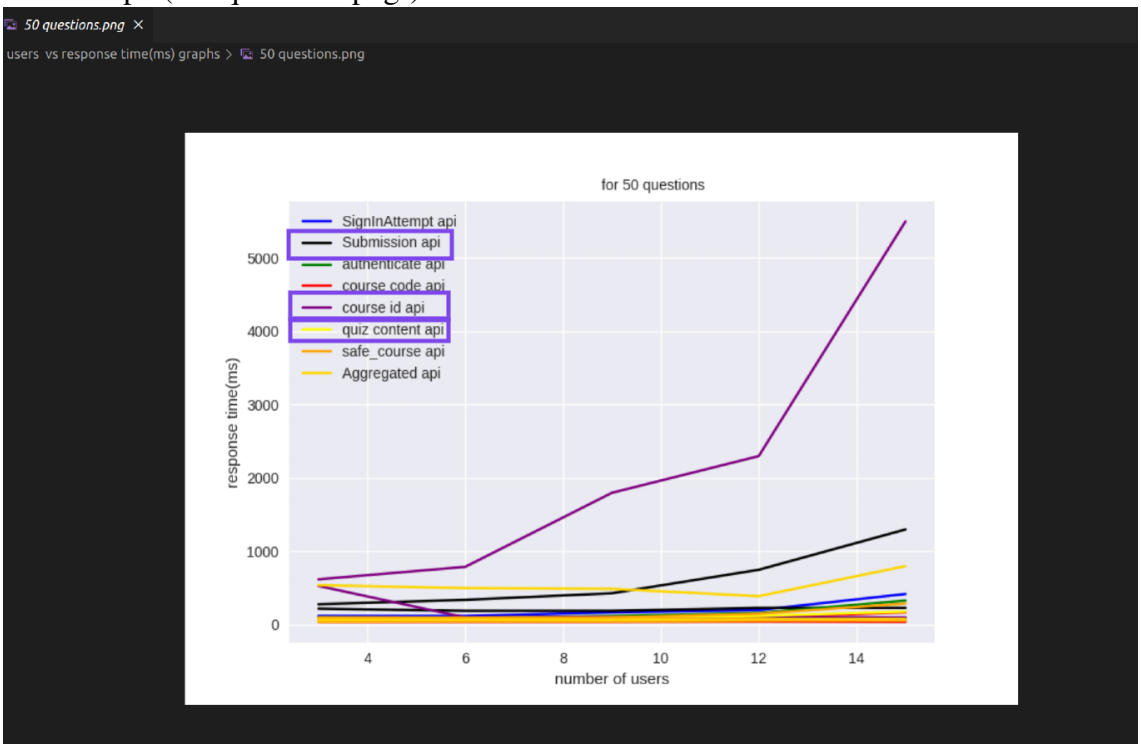
2.4.2 Create Graph for Response Time vs Number of Users for a particular Number of Questions

Note: Make sure that the script is just outside all the question folders like this.

```
jatin@Hawkeye:~/SafeRnD/5.Analysis$ ls
10q  20q  30q  40q  50q  QuestionsVsRT.py  UsersVsRT.py
```

File Name : **UsersVsRT.py**
location : /SafeRnD/5.Analysis/
output : 'users vs response time graphs'/{'10 questions.png', '30 questions.png','50 questions.png', '20 questions.png','40 questions.png}(files)
,log of folders processed (string) Run : python **UsersVsRT.py**

One example('50 questions.png') shown below.



2.4.3 Create Graph for Response Time vs Number of Questions for a particular Number of Users

Note: Make sure that the script is just outside all the question folders like this.

```
jatin@Hawkeye:~/SafeRnD/5.Analysis$ ls
10q  20q  30q  40q  50q  QuestionsVsRT.py  UsersVsRT.py
```

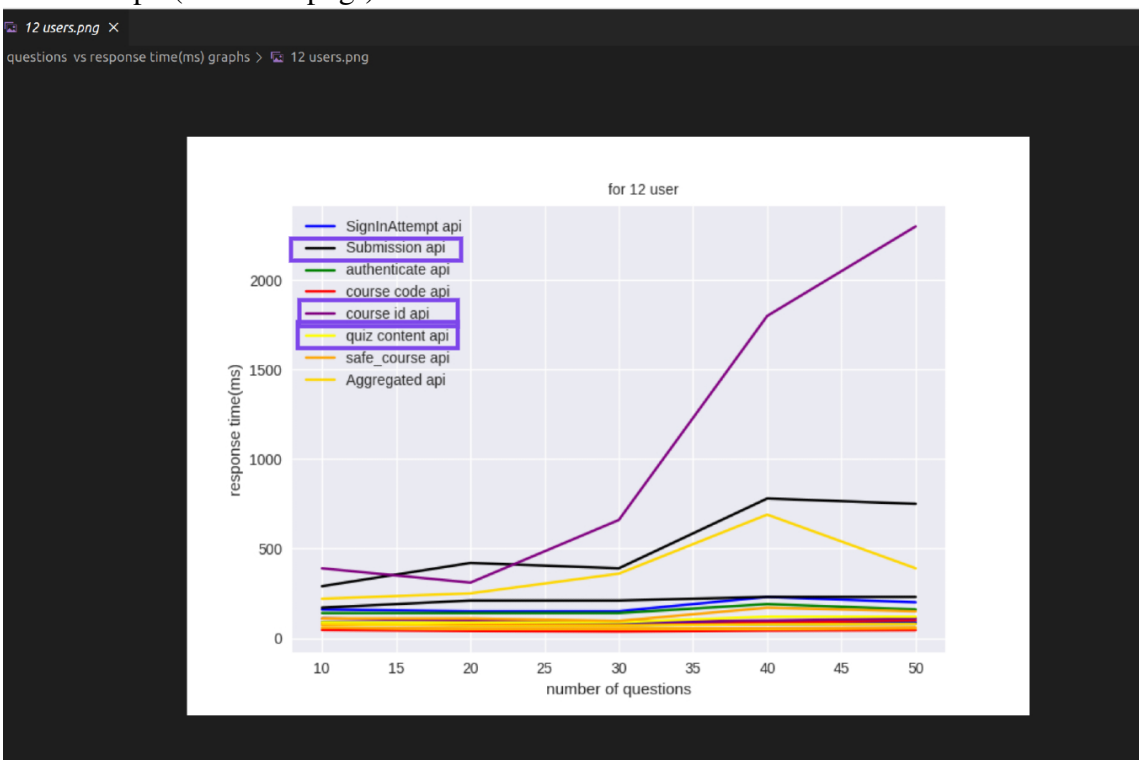
File Name : **QuestionsVsRT.py**

location : /SafeRnD/5.Analysis/

output : 'questions vs response time graphs'/{'12 users.png', '15 users.png','3 users.png','6 users.png','9 users.png'}(files),log of files processed (string)

Run : python **QuestionsVsRT.py**

One example('12 users.png') shown below.



Chapter 3

Conclusion

The Scripts that are created can be used to figure out the potential areas of improvement. For example in the above graphs in section 3.4, it is visible that we can get significant gain in performance by reducing the Response Time for the 3 APIs listed below.

1. **course id API**
2. **Submission API**
3. **quiz content API**

Chapter 4

Future Work and Repository link

4.1 Future Work

- To Try these scripts on the actual server.
- Downloading the data generated by performance Testing, for further analysis.
- Using the Analysis Scripts to identify potential areas where it is possible to improve performance.
- Optimizing the potential regions identified above , leading to better overall performance.

4.2 Repository Links

Branch Link: **Project Scripts**

Acknowledgements

I am thankful to Prof. Bhaskaran Raman for his constant support, invaluable guidance and meaningful feedback throughtout the creation of this project. I would also like to thank my seniors (Khyati Oswal and Chirag Jain) for helping with chosing and setting up tools for the project. Without their support this project would not come to fruition.

Jatin Lachhwani
IIT Bombay
2 December 2021