Benchmarking Service Test Reports

ProCoders

Bongani Tshela - 14134790

Harris Leshaba - 15312144

Joseph Letsoalo - 15043844

Minal Pramlall - 13288157

Contents

1 Purpose.

This document outlines the various activities performed as part of testing of the algorithm bench-marking system application.

2 System Overview.

The Benchmarking service will be to provide users with the capability of testing viability between algorithms with graphical data backed by performed tests. Programmers are taught multiple ways to complete a task, and with a benchmarking tool at their disposal, they will be able to make informed decisions with which is the best algorithm to use among the ones that they have considered. The system runs on a bare-metal server (Linux-Ubuntu) and uses remote ubuntu servers as seperate benchmarking nodes. The system uses ngrok for purposes of making the site public (tunneling). The user will communicate with the system through a website interface, they can submit multiple algorithms to benchmark and multiple datasets for them to be tested against. The system will benchmark all of these concurrently and provide the user with graphs and other visual output on the results. The system will be released into the open-source domain upon completion (Per request of the client).

3 Types of testing.

1.**Unit Testing :**
We are using unit testing in Java [Junit] to test and verify each and every function implemented.
2.**Integration Testing :**
We are using Sahi Pro to test both the functions in web scripting and the intergrated application or system.

4 Functional Requirements or Modules Testing.

4.1 User Management Module.

User registration and log-in :
The functions for user registration and verification were written in PHP and are being tested with sahi pro. The functions work properly with all test conditions we put it in. The module queries the database successfully and returns and displays relevant errors when results are not found.

4.1.1 User Registration

Adds user to the system if the provided results are verified to be true or valid.

Function	Input	Expected Output	Output	Passing
RegisterUser()	Email Address, FName, LName Password	Boolean or Error message	Add user to DB	Yes

4.1.2 User Login

Connects to the database, request to verify if user exists in the database, if the user exists it is a successful log-in and unsuccessful i the user does not exist in the database.

Function	Input	Expected Output	Output	Passing
LoginUser()	Email Address, Password	Boolean or Error message	Successfull login	Yes

4.2 Java files Testing

All Java functions and subsections were tested with Junit (Unit testing). These functions deal with benchmarking the algorithms that the user uploads(Main functionality of the system).

4.2.1 CPUcalculation()

This function it is used to calculate what percentage the Algorithm uses of the CPU.

Function	Input	Expected Output	Output	Passing
CPUcalculatio()	tester.java, algorithm.java test.txt	Percentage usage of CPU	Percentage usage of CPU	Yes
CPUcalculatio()	tester.java, algorithm.java	Percentage usage of CPU	ERROR: no usage of the CPU	Yes
CPUcalculatio()	tester.java, test.txt	Percentage usage of CPU	Compilation ERROR	Yes
CPUcalculatio()	algorithm.java test.txt	Percentage usage of CPU	Compilation ERROR	Yes

4.2.2 RAMusage()

This function it is used to calculate what is the size of RAM does the Algorithm uses in bytes.

Function	Input	Expected Output	Output	Passing
RAMusage()	tester.java, algorithm.java test.txt	Size of Memory usage	Size of Memory usage	Yes
RAMusage()	tester.java, algorithm.java	Size of Memory usage	ERROR: no usage of the CPU	Yes
RAMusage()	tester.java, test.txt	Size of Memory usage	Compilation ERROR	Yes
RAMusage()	algorithm.java test.txt	Size of Memory usage	Compilation ERROR	Yes

4.3 Display module

The module is witten in PHP and other web scripting language. Given the results (Json strings) from the bench-marking module, the module graphically represents the results to make it easy for the use to see their results. The module was tested with Sahi Pro (Screenshots to follow) and it works fine under all test conditions it was put in.

4.4 Access Module

5 Conclusion.

The system was tested and most of the tests succeeded. The system was also tested for scalability and it performs well depending on the number of benchmarking nodes available. The intergrated system was also tested with different users (peers, clients, lectures). Through testing with different users it was found that the benchmarks are slower when the user uploads a seperate data-set file than when they give datasets in the testing java files/Main files.