



UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

BENCHMARKING SERVICE
SOFTWARE REQUIREMENTS SPECIFICATION

ProCODERS

BONGANI TSHELA - 14134790

HARRIS LESHABA - 15312144

JOSEPH LETSOALO - 15043844

MINAL PRAMLALL - 13288157

Contents

Table Of Contents	1
1 Project background.	2
2 Project vision.	2
3 Architectural Design	2
4 Project Scope	2
5 User management module	2
5.1 Scope	2
5.1.1 Domain model	3
5.1.2 Precondition Table	4
5.2 Service contracts	4
6 Display module	5
6.1 Scope	5
6.1.1 Domain model	6
6.2 Service contracts	6
6.3 Technologies	7
7 Bench-marking and VM initializing Module	7
7.1 Scope	7
7.2 Service Contracts.	8
7.3 Technologies.	8
8 System Deployment Diagram	9

1 Project background.

Bench marking seem to be a very common and useful application, very few bench-marking tools or services are available. Those that are available requires intricate configuration that may be beyond the reach of the developers, researchers, teachers and students who would like to use them. The development of a bench-marking service which can be used in a generic way would therefore be welcomed by a large potential user base.

2 Project vision.

The system provides a web interface for users to request and specify the bench-marking services they need. The services will be provided by executing the requested benchmarks on isolated machines where the side-effects that are not a concern of the specified benchmark is minimized. The system measures a variety of performance attributes such as CPU time, elapsed time, memory usage, power consumption, heat generation, etc. It accepts source code for a variety modern programming languages.

3 Architectural Design

The Benchmarking service will utilize a Cloud computing architecture, where it operates as software as a service (SaaS) rather than a business function as a service. Resources allocated will be approximately the amount needed, so if the user is charged for usage, then it would be for the actual amount of resources used.

4 Project Scope

The core of the system is a benchmarking tool which. The high level modules and their responsibilities are shown in Figure 1

5 User management module

5.1 Scope

The scope of the user management module is shown in Figure 1. The user management module is responsible for maintaining information about the registered users of the system, including the authority levels of each user. User can manage their benchmarking records like deleting and storing new records, view them and retrieving old algorithm they have stored.

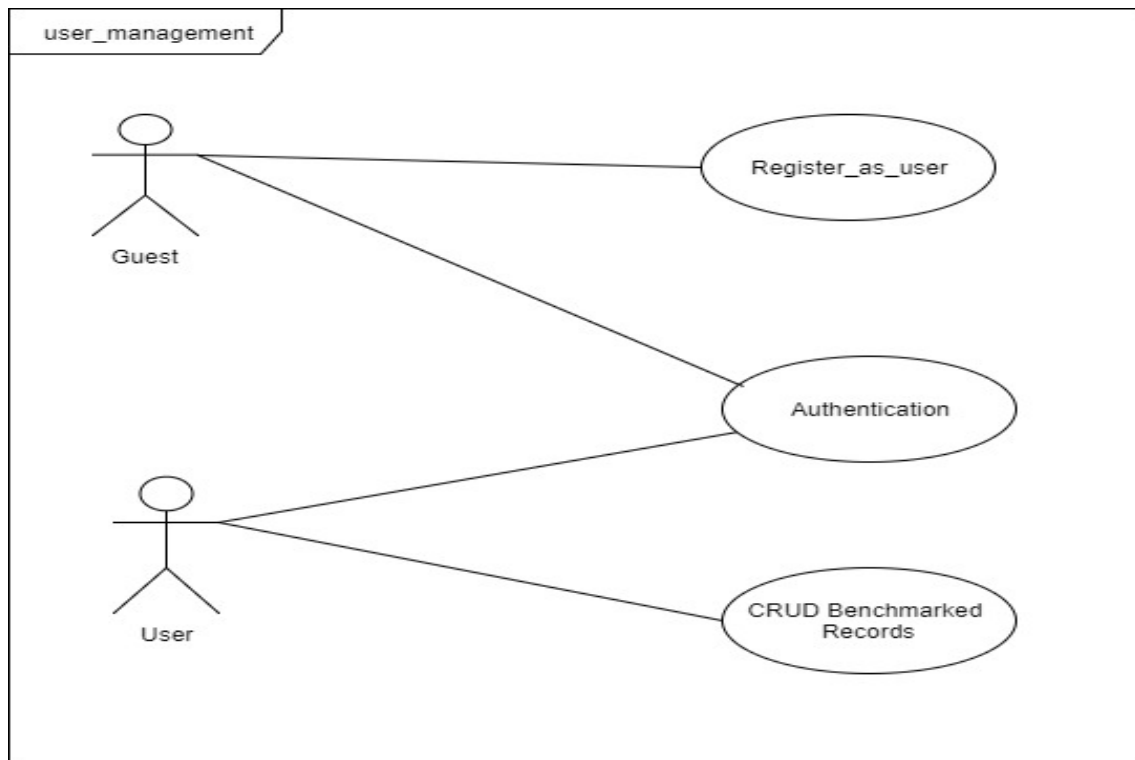


Figure 1 use case.

No information is stored for the guest user. When accessing the service, the user assumes the guest role without being logged in. The guest user may use public services and my register or log in. When a user is registered, the fields shown in the domain model are stored for the user using a unique automatically assigned ID. The user provides all other fields. The password is stored in encrypted format. The user can access all the services provided by the service The register as user use case is initiated by an end-user to create an account.

A user will not be created if the email specified in the request to create a user is already associated with an existing user. This is done to ensure that if a user is already associated with the email address, the user is given access to his/her profile instead of creating a new user.

Note that this use case will use the **getUser(userEmail)** service provided by the module to confirm if another user with that same email already exists in the system. A user should only be created if **getUser(userEmail)** throws the *noSuchUser* exception. After the account has been created a notification will be sent to the user. The system have two types of users: Guest user and Registered user.

5.1.1 Domain model

The users management domain model is shown in Figure 2

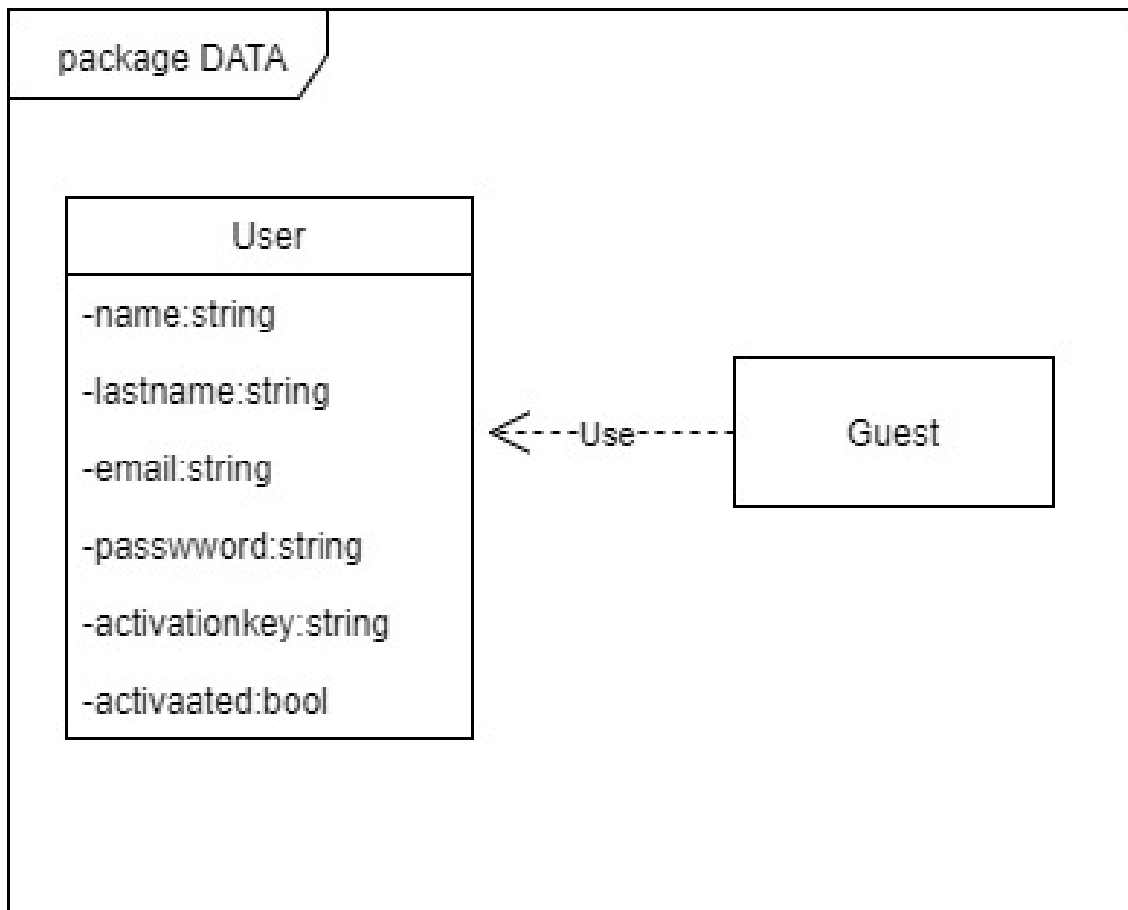


Figure 2 class diagram.

5.1.2 Precondition Table

CU1 getUserEmail(email)

Precondition: userEmail is associated to register user	
Actor: Guest user	System: user module
1. Guest user enter detail	3. System add the user into the database
2. Guest click register	4. System allow the user access to the other services

Figure 3 class diagram.

5.2 Service contracts

The following services should be provided

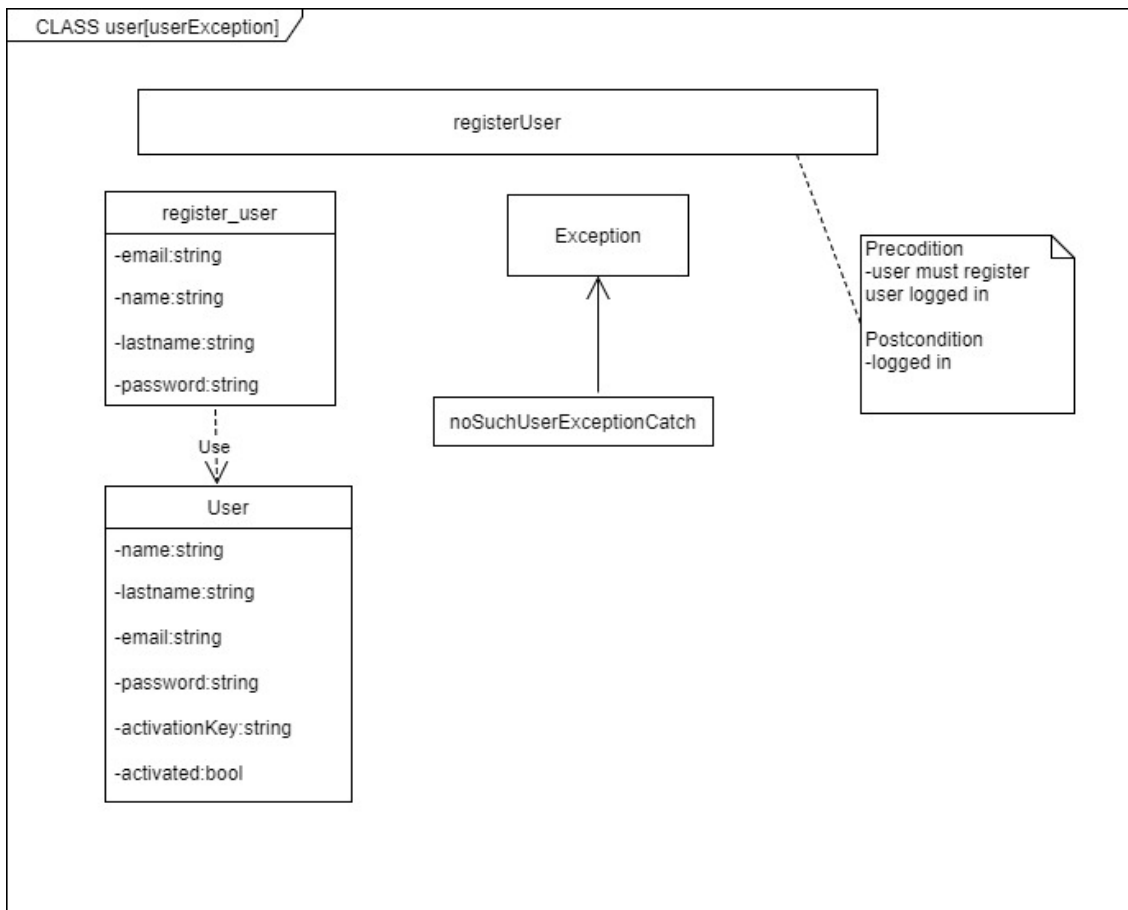


Figure 4 class diagram.

6 Display module

6.1 Scope

The display module construts graphs, charts and tables to represent the benchmark results, and how different algorithms compare against one another in terms of perfolrmance.

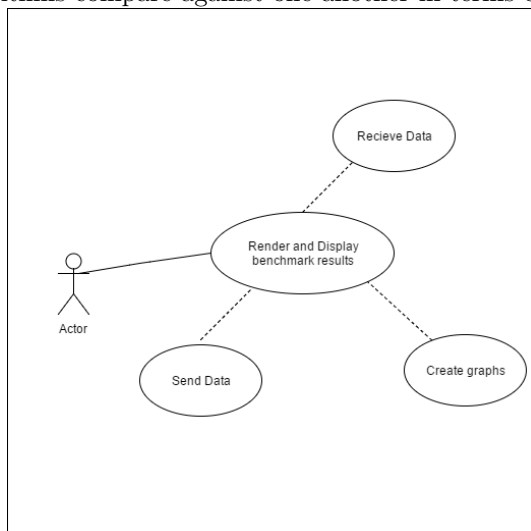


Figure 5 use case.

The display module concerns itself with rendering and displaying the results of benchmarks. The display module concerns itself primarily about creating graphs, charts and tables to represent the benchmarking results

in a easily readable manner.

6.1.1 Domain model

The Display domain model is shown in Figure 6

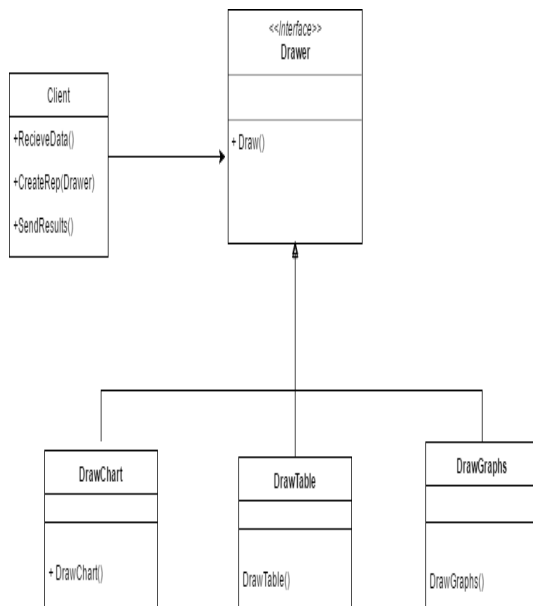


Figure 6 class diagram.

The domain model indicates how the client will use the Drawer to create the specific representation of the benchmark results.

6.2 Service contracts

The following services should be provided

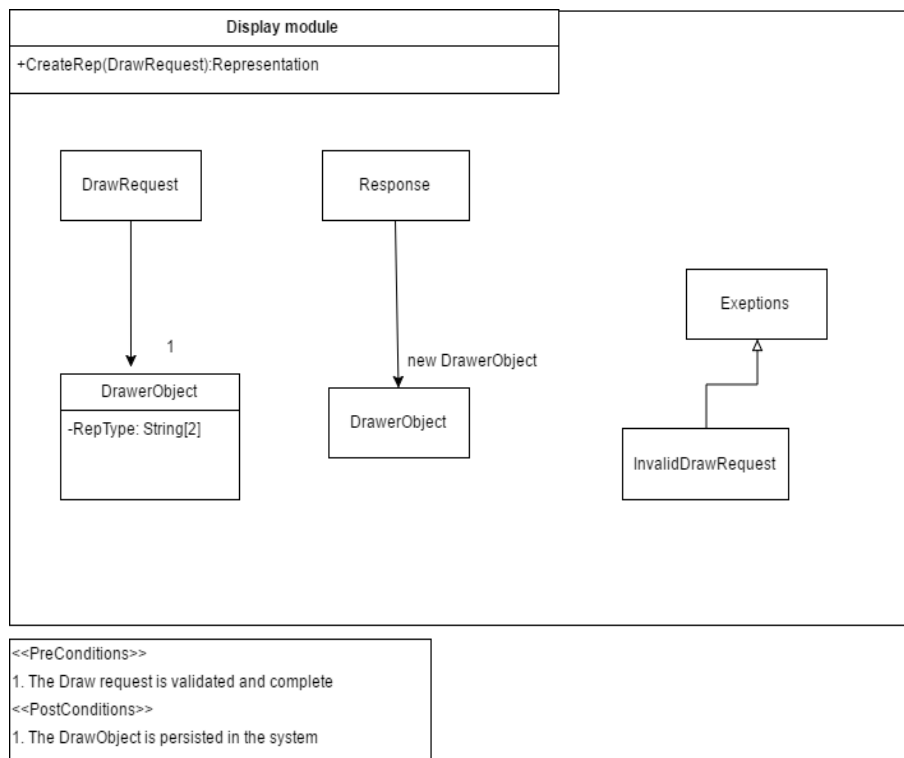


Figure 7 Service Contracts.

The service contract specified above will be refused under one condition. If the draw request fails the validation protocol, then the contract can be refused. Otherwise, the DrawObject will be persisted and stored in the system.

6.3 Technologies

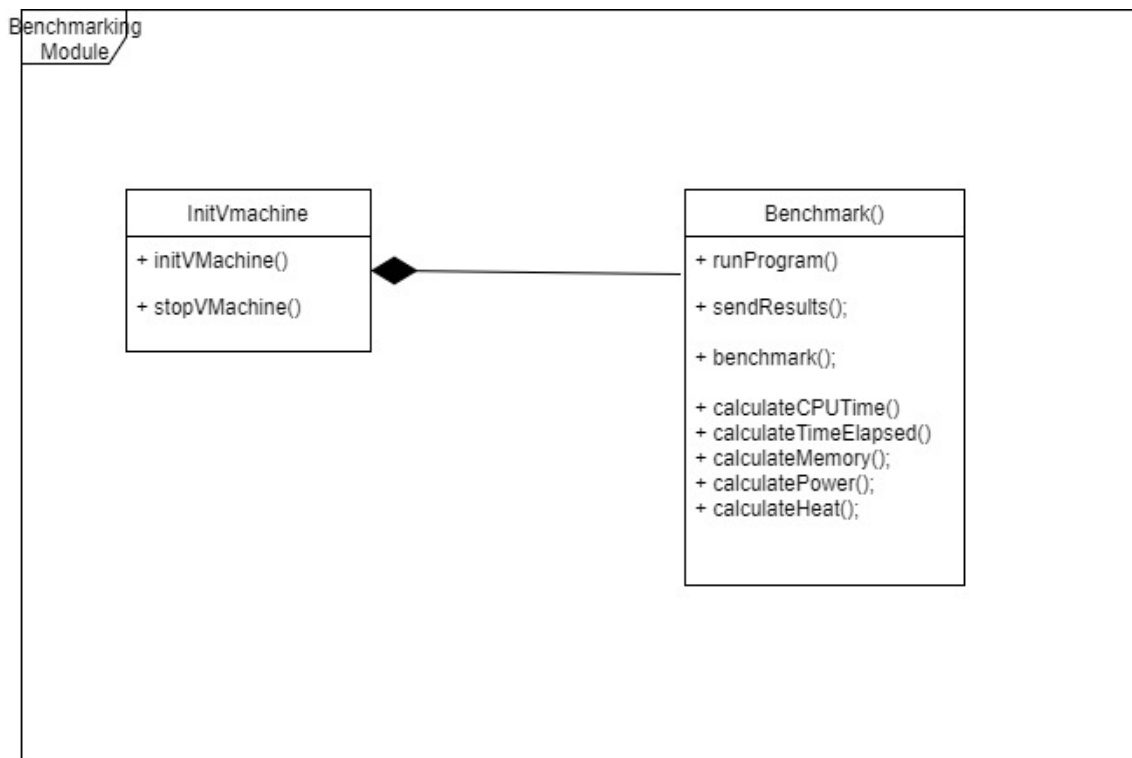
When it comes to Display technologies, one in particular is of note. JFreeChart - A free 100JFreeChart's extensive feature set includes:

- 1.A consistent and well-documented API, supporting a wide range of chart types;
- 2.A flexible design that is easy to extend, and targets both server-side and client-side applications;
- 3.Support for many output types, including Swing and JavaFX components, image files (including PNG and JPEG), and vector graphics file formats (including PDF, EPS and SVG);
- 4.JFreeChart is open source or, more specifically, free software. It is distributed under the terms of the GNU Lesser General Public Licence (LGPL), which permits use in proprietary applications.

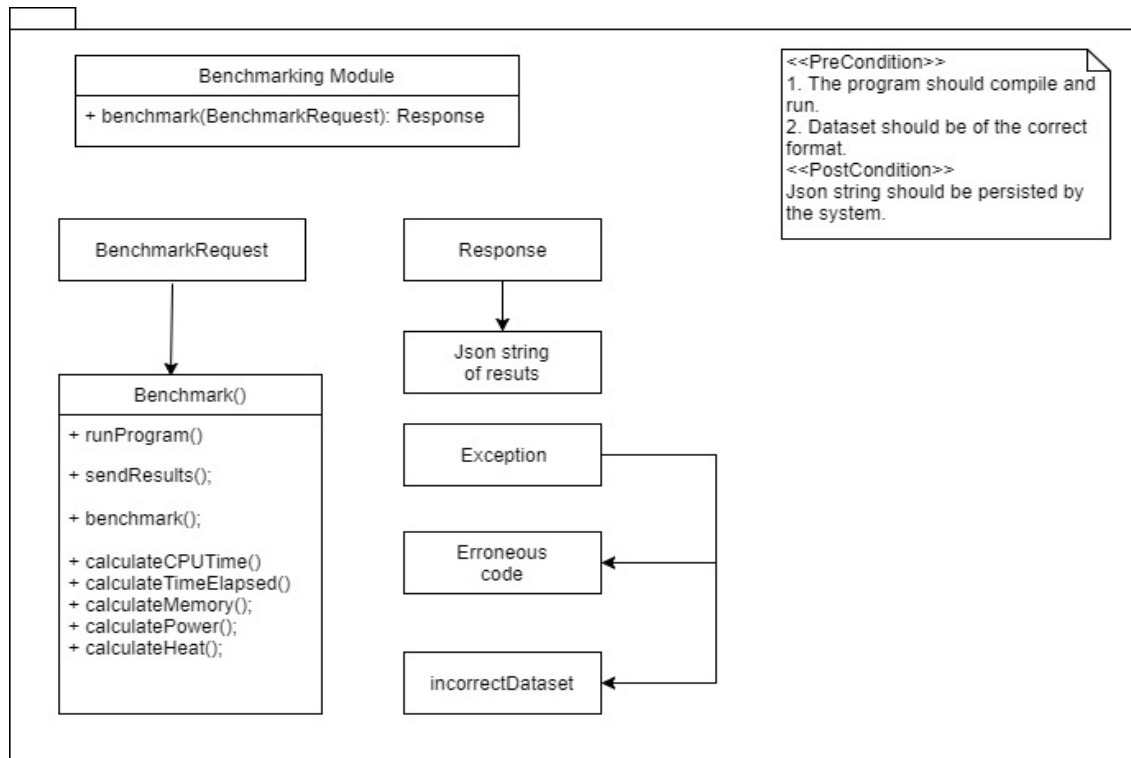
7 Bench-marking and VM initializing Module

7.1 Scope

The module initializes and deploys VMs based on the request(s). On the VMs we will have a Java program that runs other programs, that is compile them, run them with the user specified data sets and then benchmark the program or algorithms. This module will initialize VMs on request and can initialize them concurrently within seconds.



7.2 Service Contracts.



7.3 Technologies.

Osv - the open source operating system designed for the cloud. Built from the ground up for effortless deployment and management, with superior performance.

Unikernels - specialized, single address space machine images constructed by using library operating systems(Osv in our case).We select, from a modular stack, the minimal set of libraries which correspond to the OS constructs required for our application to run.

8 System Deployment Diagram

