Software Requirements Specification Document

Performance **T**esting in the **C**loud, Team **30** and **K**avya Nerella **G** Pooja Shamili **S**hubham Ghiya **V**ishal Gupta

Brief problem statement

The project involves building a performance testing tool. The tool launches multiple virtual machines and from each of these machines, tests are performed on the target website. It is basically a platform to simulate a large number of users and check the performance of a website/web-application.

System requirements

Programming Language : Python

Frameworks/Libraries : Flask (Micro-Framework), Jinja2 (Templating Engine). Locust (Load testing tool), JMeter, AngularJs (MVW Framework), Multiprocessing, Threading, Gevent

Databases: MongoDb

VCS: Git

Collaboration Tool : Google Docs

Users profile

Webmasters, owners and developers of web based services like websites, apps and APIs that cater to large users and want to load test their product before pushing it into production are the main users of this product.

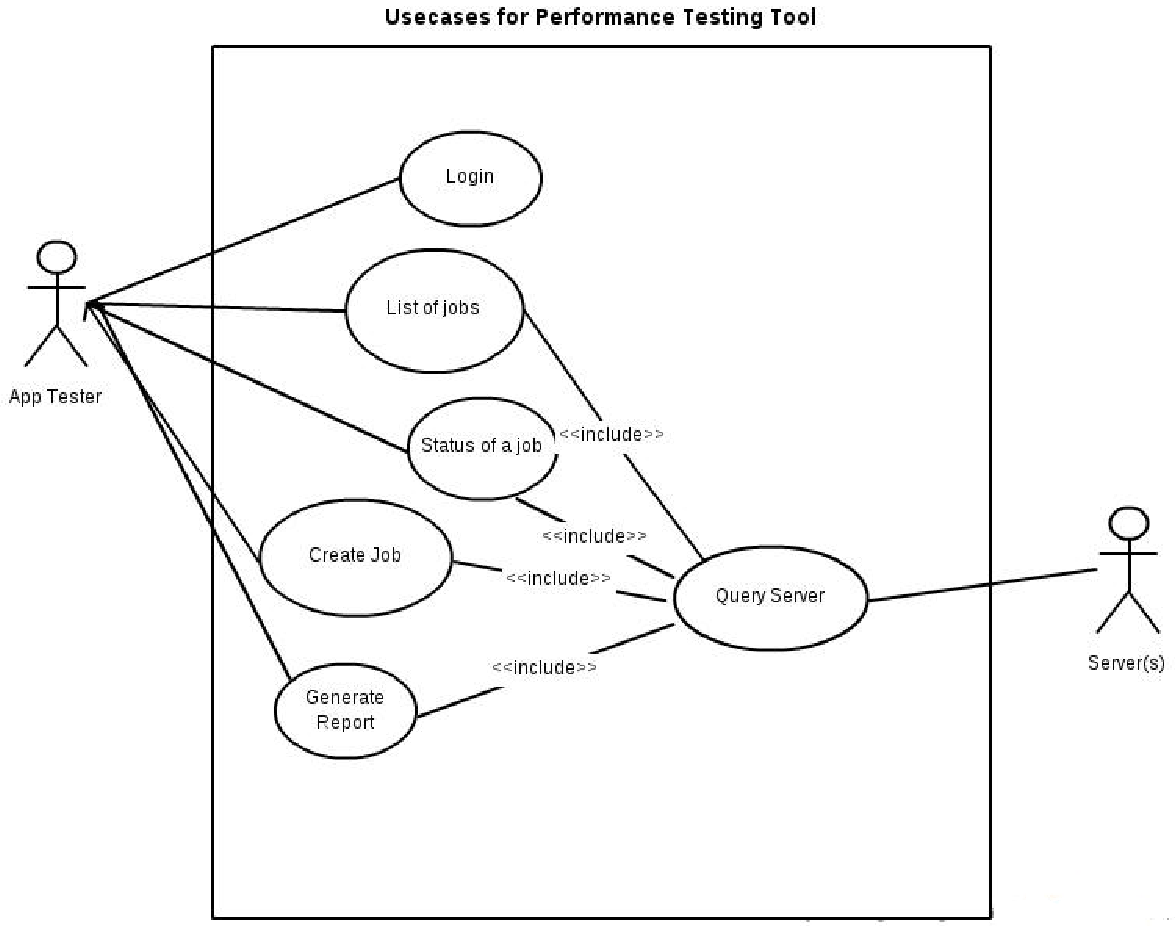
Research Requirements

1.	Locust	Learn, test and benchmark	R1
2.	Jmeter	Learn, test and benchmark	R1
3.	Threading	Learn, test and benchmark	R1
4.	Multiprocessing	Learn, test and benchmark	R1
5.	Gevent	Learn, test and benchmark	R1

Feature requirements (described using use cases)

No.	User Case Name	Description	Release
1.	Login	Customer will login by entering email id and password.	R2
2.	List of Jobs	All jobs that the user has started have to be displayed.	R2
3.	Show Status of a Job	Show the status of job the customer has started.	R2
4.	Create a new job	The user creates a new job based on his requirements.	R2
5.	Generate reports for a job	Every job has certain performance features associated which have to be aggregated and displayed.	R2

Use case diagram



Use case description

Use Case Number:	UC-01
Use Case Name:	Login
Overview:	Customer will login using his email id and password
Actors:	App Tester(customer)<<Primary>>
Pre condition:	Customer should have a valid email id.
Flow:	Main (success) Flow: 1. Customer enters email id. 2. Customer enters his password.
	Alternate Flows: 1. If email id or password entered is wrong i.e. not in the database ask customer to reenter email id and password.
Post Condition:	Customer will be redirected to his dashboard.

Use Case Number:	UC-02
Use Case Name:	List of jobs.
Overview:	All jobs that the user has started have to be displayed.
Actors:	App Tester(customer)<<Primary>>, Server<<Secondary>>
Pre condition:	The App Tester must be logged into the system and must have selected a “View Current Jobs” option.
Flow:	Main (success) Flow: 1. The customer queries the system about all the jobs. 2. The servers return a list of jobs along with their status. Alternate flow: 1. If there are no jobs, the server gives the customer an option to start a new job .
Post Condition:	A list of current jobs shall be displayed.

Use Case Number:	UC-03
Use Case Name:	Show Status of a Job
Overview:	The customer will come to know the status of each job in detail.
Actors:	App Tester(customer)<<Primary>>, Server<<Secondary>>
Pre condition:	The AppTester must be logged into the system and should select a particular job to see its details.
Flow:	Main (success) Flow: 1. The customer queries the system about the status of the job. 2. The server returns the status of various tasks of a job in a particular order in which they are supposed to be done.
Post Condition:	Customer will be able to view the status of various tasks i.e. status of a job.

Use Case Number:	UC-04
Use Case Name:	Creating a new job.
Overview:	The user creates a new job based on his requirements.
Actors:	App Tester(customer)<<Primary>>
Pre condition:	The App tester must be logged in and must have selected the “New Test” option.
Flow:	**Main Flow:** 1. The website tester specifies the URL he wants to test. 2. He can also mention the number of concurrent requests to be made to the target web site. 3. Without any specific mention , a default value is taken. **Alternate Flow:** 1. Failure of creating a job due to whatsoever reasons will be recorded and taken care by the system administrator.
Post Condition:	A message briefly describing the created job is displayed to confirm with the user.

Use Case Number:	UC-05
Use Case Name:	Generate reports
Overview:	Every job has certain performance features associated which have to be aggregated and displayed.
Actors:	App Tester(Customer)<<Primary>>
Pre condition:	The job for which reports are being generated should have been completed.
Flow:	Main (success) Flow: 1. Customer is redirected to a new page. 2. Server is queried for all data pertaining to a job. 3. The new page displays key metrics like no.of requests, failures, avg response time and no. of requests/sec. Graphs depicting various aspects of load and stress testing are displayed.
Post Condition:	Customer is able to view the results he/she wanted.