Fall 2020 Mid-Term Assignment

Software Verification and validation

Submitted By: Mudassir Ahmad Khan
ID# 14086
BSSE (6th Semester)

Submitted To: Mr. Zain Shaukat
(Lecturure)

Testing Tool : Apache Jmeter

Part: (a)

Pros of JMeter

- Easy installation: It can be installed on any desktop with Windows, Mac or Linux.
- It has a user-friendly interface or can be used in a command line interface.
- Test IDE allows test recording from browsers or native applications.
- Ability to extract data from popular response formats like HTML, JSON, XML or any textual format.
- Readily available plugins, for example, visualization plugin for data analysis.

Cons of JMeter

- Has a high learning curve thus it requires skilled testers.
- It doesn't support JavaScript and by extension doesn't automatically support AJAX requests.
- Complex applications that use dynamic content like CSRF tokens, or use JS to alter requests can be difficult to test using JMeter.
- Memory consumption is high in GUI mode which causes it gives out errors for a large number of users.

Part: (b)

Functionality

- Compatible with TCP
- 100% Java scripted
- Compatible with LDAP
- Data Analysis and Visualization
- Pluggable Samplers
- Compatible with Database via JDBC
- GUI Design and Interface
- Dynamic Input
- Compatible with SOAP / REST
- Result Analysis and Caches
- Compatible with Native Commands
- Highly Extensible Core
- Compatible with FTP
- Compatible with Web – HTTP, HTTPS
- Scriptable Samplers
- Compatible with Mail – SMTP(S)
- Compatible with POP3(S) and IMAP(S)
- Compatible with Message-oriented middleware via JMS

Part: (c)

Supporting Languages

- Web - HTTP, HTTPS (Java, NodeJS, PHP, ASP.NET)
- SOAP / REST Webservices
- FTP
- Database via JDBC
- LDAP
- Message-oriented middleware (MOM) via JMS
- Mail - SMTP(S), POP3(S) and IMAP(S)
- Native commands or shell scripts
- TCP
- Java

Part: (d)

Supporting Tests

Apache Jmeter Support performance, load and stress testing of a web application using JMeter Recording Tests (GUI mode) on Operating System.

1. Performance Testing:

- Performance Testing is crucial to determine that the web application under test will satisfy high load requirements. It can be used to analyze overall server performance under heavy load.
- Performance Testing checks the speed, response time, reliability, resource usage, scalability of a software program under their expected workload. The purpose of Performance Testing is not to find functional defects but to eliminate performance bottlenecks in the software or device.
- The focus of Performance Testing is checking a software program's

2. Load Testing:

- Load Testing: Modeling the expected usage by simulating multiple user access the Web services concurrently.
- Load testing is the process of putting the load through (HTTP, HTTPS, WebSocket etc) calls on any software system to determine its behavior under normal and high load conditions. Load test helps identify maximum requests a software system can handle. It helps determine the point of a bottleneck due to which application starts degrading performance. Load testing is effective when we simulate real user scenario. We need to know how our users behave on our application. Try to replicate their behavior using different API calls and generate load through many concurrent requests at our application for an extended period of time to understand application behavior changes.

3. Stress Testing:

- Stress Testing: Every web server has a maximum load capacity. When the load goes beyond the limit, the web server starts responding slowly and produce errors. The purpose of the Stress Testing is to find the maximum load the web server can handle.
- Stress Testing is a type of Software Testing that verifies the stability & reliability of the system. This test mainly measures the system on its robustness and error handling capabilities under extremely heavy load conditions.
- Stress Testing is done to make sure that the system would not crash under crunch situations. It even tests beyond the normal operating point and evaluates how the system works under those extreme conditions.

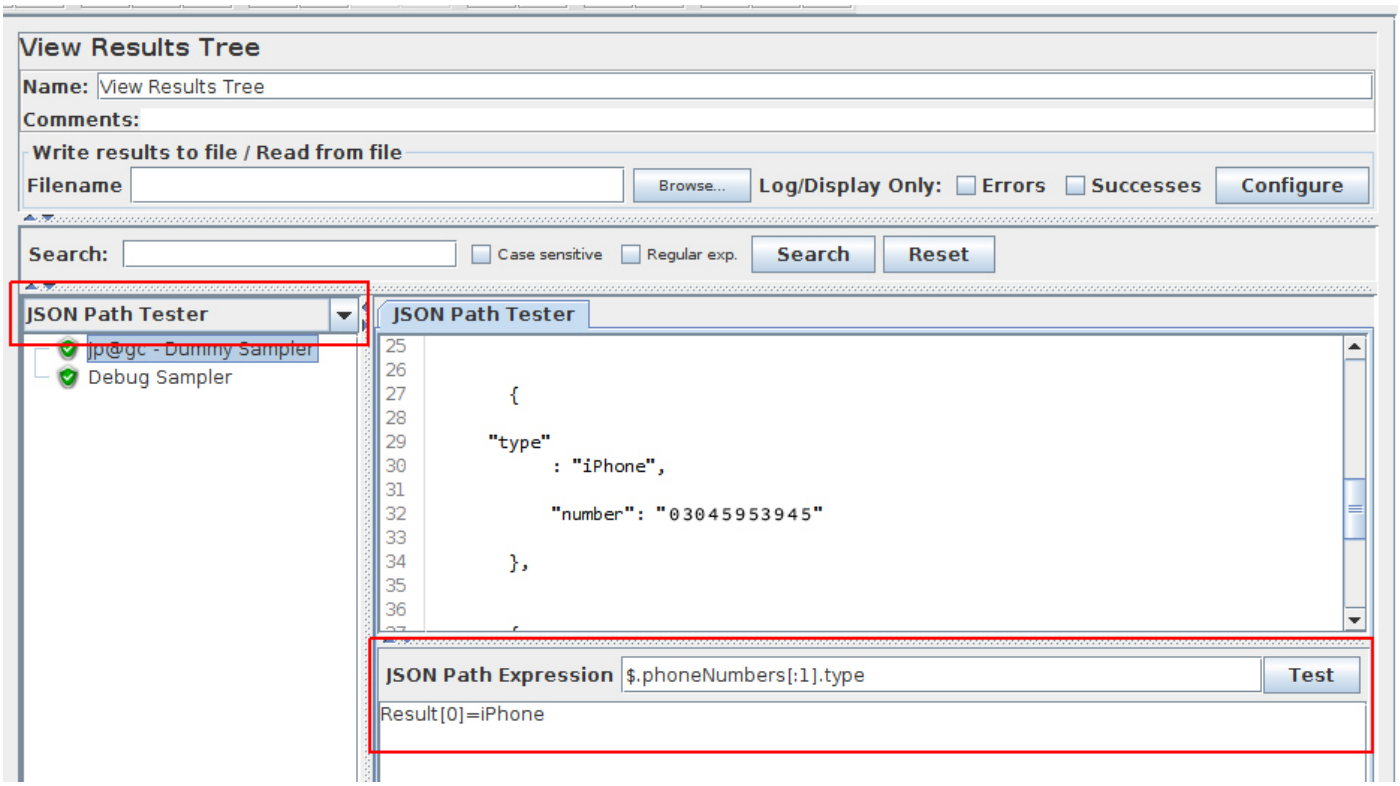
Part: (e)

Write a short(faulty) code, Test using this tool, and show the bugs in the code

```
{
 “firstName”: “ahmad”,
 “lastName” : “khan”,
 “age”      : 22,
 “phoneNumbers”: [
   {
    “type”  : “iPhone”,
    “number”: “03045953945”
   },
   {
    “type”  : “home”,
    “number”: “03045953945”
   }
 ]
}
```

Bug:

In the Code above, i’m trying to extract the first phone number type by using JsonPath $.phoneNumbers[:1].type. It matches the iPhone value in the sample document:



Thank you!