TEST CASES

Tables of Content

TC 1.1.1 - The web server should be responsive under high load

TestID: REQ1.1

Short description:

The web server should be responsive under high load of HTTP requests from multiple users at once.

Precondition(s):

The web server should be started and working

Test Steps:

- Run JMeter
- Set the test plan for the MyWebServer by adding Thread Groups.
- Set the Number of Threads to 1000 for a ramp-up period of 1 second and a loop of 1.
- Add HTTP request with correct URL and correct port.
- Add a listener (either “view result in the table” or “Graph results”).

Expected results:

The web server should not crash or fail, and the webserver should be responsive under high load of HTTP requests with a low level of latency.

TC 1.1.2 - The web server should be responsive under high load

TestID: REQ1.2

Short description:

The web server should be responsive under high load of HTTP requests from multiple users at once.

Precondition(s):

The web server should be started and working

Test Steps:

- Run JMeter
- Set the test plan for the MyWebServer by adding Thread Groups.
- Set the Number of Threads to 2000 for a ramp-up period of 1 second and a loop of 1.
- Add HTTP request with correct URL and correct port.
- Add a listener (either “view result in the table” or “Graph results”.

Expected results:

The web server should not crash or fail, and the webserver should be responsive under high load of HTTP requests with a low level of latency.

TC 1.2.1 - The web server must follow minimum requirements for HTTP 1.1 with Google Chrome

TestID: REQ2.1
Short description:
The web server should follow the minimum requirements of HTTP Protocol version 1.1 or later.

Preconditions:
The web server should be started

Test Steps:

- Open the web server on Google Chrome.
- Right Click and press on Inspect
- Click on the "Network" tab
- Refresh the page
- Click on "localhost"
- Click on "view source" next to "Response Headers"

Expected results:
It should show "HTTP/1.1 200 OK" just under the Response Headers.

TC 1.2.2 - The web server must follow minimum requirements for HTTP 1.1 with Curl

TestID: REQ2.2

Short description:
The web server should follow the minimum requirements of HTTP Protocol version 1.1 or later with cURL.

Preconditions:
The web server should be started

Test Steps:

- Install cURL
- Write command "curl -v website address" on Terminal / Command Line

Expected results:
It should show "HTTP/1.1 200 OK"

TC 1.3.1 - The web server must work on Windows 10

TestID : REQ3.1

Short description:

As the webserver has been built with Java, it ought to work on all Operating Systems. Taking into consideration that Java has been installed correctly.

Preconditions:

The web server should be started

Test Steps:

- Open any Browser on Mac, Windows or Linux.
- Go to the webserver.

Expected results:

The server works with Windows 10.

TC 1.3.2 - The web server must work on Mac OS Catalina

TestID : REQ3.2

Short description:

As the webserver has been built with Java, it is supposed to work on all Operating Systems. Taking into consideration that Java has been installed correctly.

Preconditions:

The web server should be started

Test Steps:

- Open any Browser on Mac.
- Go to the webserver.

Expected results:

The server works with Mac OS Catalina.

TC 1.4 - The source code should be released under GPL-2.0

TestID : REQ4.1

Short description:

The source code should be released under the GPL version 2.0. Control that the old software license allows for the program to be used

Preconditions:

None

Test Steps:

- Go through the source code of the application
- Search for a LICENCE file

Expected results:

The Licence file should state that the application is under GPL-2.0.

TC 1.5 - The access log should be viewable from a text editor

TestID : REQ5.1

Short description:

The system should contain a file that has an access log that can be viewable from a text editor.

Preconditions:

The server must be started and working. It has to show some content.

Test Steps:

- Go to the folder MyWebServer and search for a file log.txt
- Open the file .log in any text editor

Expected results:

It opens the .log file and the access log content is viewable in any text editor.

TC 1.6.1 - Start Server - The web server should start on the specific port

TestID : REQ6.1

Use Case Tested: UC1.2

Short description:

The web server should first ask for the socket port number and the shared resource container. Then when the administrator provides the socket port number and the shared resource container, the system should start the webserver on the given port and presents that the server was started

Preconditions:

Test Steps:

- Get a jar file of the src folder
- on the command line run java -jar MyWebServer.jar Port Number Resource location

Expected results:

The web server should start according to the given port and resource location

---

TC 1.6.2 -The web server should write into the access log file

TestID : REQ6.2

Use Case Tested: UC1.4

Short description:

Writes a note in the access log that the system started

Preconditions:

TC 1.6.1

Test Steps:

- Check access log for the note

Expected results:

The web server should start according to the given port and resource location and writes a note in the access log that the system has been started.

TC 1.6.3 - The web server should not start due to taken socket

TestID : REQ6.3

Use Case Tested: UC1.4a

Short description:
If a user tries to start the webserver on a port that is already taken, it should respond by an error message that stipulates that the port is already taken.

Preconditions:

Test Steps:

- Run the MyWebServer.jar with an already used port
- Check output

Expected results:
The web server output should be an error message like "Socket XX was taken" (XX is the socket number, Example "80").

TC 1.6.4 - The web server should not start due to restrictions on the shared resource container

TestID : REQ6.4

Use Case Tested: UC1.4b

Short description:
If the user tries to start the webserver on a resource that has restricted rights, the system should present an error message saying that it cannot access the folder

Test Steps:

- Delete the folder resources
- Start the server with a path to resources

Expected results:
The web server output should be an error message like "No access to folder XX" (XX is the shared resource container provided, Example "\var\www").

TC 1.6.5 - Server Start - Access log could not be written to

TestID : REQ6.5
Use Case Tested: UC1.4c

Short description:
The web server should show a message that says that it cannot write to the server log file log.txt
Preconditions:

Test Steps:
- Delete the log file
- Start the webserver
- Check for an output message about accessing the log file

Expected results:
The web server output should be an error message like “Cannot write to server log file log.txt”

TC 1.7 - The web server should stop when asked to stop

TestID : REQ7.1

Use Case Tested: UC2

Short description:
When the administrator stops the web server, the system should stop and display a message that informs the administrator that the server has been successfully stopped.

Preconditions:
MyWebServer should be started

Test Steps:
- In the terminal, write stop to stop the server
- Check the given output message

Expected results:
The web server should display a message that tells the user that the server has been successfully stopped.

TC 1.8.1 - The web server should display the requested resource Status 200 and write to the log

TestID : REQ8.1

Use Case Tested: UC3.2

Short description:
When the browser wants to access a shared resource, the system should deliver that resource to the browser and a success message should be written to the access log.

Preconditions:
MyWebServer should be started

Test Steps:
- Open the browser and access the website
- Check if the website output content
- Check the output in the access log

Expected results:
The web server should make the website's content available when we check in the browser and a success message ought to be written in the access log.

TC 1.8.2 - The web server should give 404 Not Found

TestID : REQ8.2

Use Case Tested: UC3.2a

Short description: An error message should be shown when accessing a source that does not exist in the server.

Preconditions:
The web server must be started.

Test Steps:
- Access a URL that does not exist in the website or delete the index.html file
- Check the error message shown on the browser.

Expected results:
The error message “404 Not Found” should be displayed on the browser.

TC 1.8.3 - The web server should give 403 Forbidden

TestID : REQ8.3

Use Case Tested: UC3.2b

Short description: An error message should be shown when accessing a source that the server does not have permission.

Preconditions:
The web server must be started. A source with restrained permission should be present.

Test Steps:
- Run JMeter
- Set the test plan for the MyWebServer by adding Thread Groups.
- Set the Number of Threads to 1 for a ramp-up period of 1 second and a loop of 1.
- Add HTTP request with the correct Server Name, pathname (/../secret.html) and correct port.
- Change the Method to GET
- Add a listener (View Results tree).

Expected results: The error message “403 Forbidden” should be shown on the web browser.

TC 1.8.4 - The web server should give 400 Bad request

TestID : REQ8.4

Use Case Tested: UC3.2c

Short description:
An error message should be shown when the resource requested is invalid or malformed

Preconditions:
The web server should be started

Test Steps:

- Install cURL
- Write command **curl -d “h1=hey” localhost:9000/** on Terminal / Command Line

Expected results:
The error message “400 Bad Request” should be shown

TC 1.8.5 - The web server should encounter an error when processing a request

TestID : REQ8.5

Use Case Tested: UC3.2d

Short description:
An error message should be shown when the server encountered an error when trying to process the request

Preconditions:
The web server should be started

Test Steps:

- Run JMeter
- Set the test plan for the MyWebServer by adding Thread Groups.
- Set the Number of Threads to 1 for a ramp-up period of 1 second and a loop of 1.
- Add HTTP request with correct URL and correct port.
- Change the Method to POST
- Add a listener (View Results tree).

Expected results:
The error message “405 Request Not allowed” should be visible

---

TCJU 2 - Integration

TestID : REQ9.1

Short description:
The web server should be responsive under high load of HTTP requests from multiple users at once.

Preconditions:
The web server should be started

Test Steps:

- Run the integration package with JUnit test

Expected results:
All tests should pass with a print out that tells how many times the server has been successful or failed or threw an exception

---

TCJU 2.1 - Response

TestID : REQ9.2

Short description:
The web server should follow the minimum requirements of HTTP Protocol version 1.1 or later.

Preconditions:
The web server should be started

Test Steps:

- Run the response package with JUnit

Expected results:
All tests should pass

TCJU 2.2 - View

TestID : REQ9.3

Short description:
The web server should follow the minimum requirements of HTTP Protocol version 1.1 or later.

Preconditions:
The web server should be started

Test Steps:

- Run the response package with JUnit

Expected results:
All tests should pass

TCJU 2.3 - All the JUnit tests suite

TestID : REQ9.4

Short description:
Group of automated tests that will allow confirming the correct functioning of the server through the JUnit tests suite.

Preconditions:
The web server should be started

Test Steps:
- Run the response package with JUnit

Expected results:
All tests should pass

TA - Acceptance Testing

TestID: REQ10

Short description:
The web server should be tested and ready to use. This test allows us to check if the system meets the company's requirements and tells whether this technology is acceptable for delivery.

Test Steps:
- Run JUnit
- Do Manual test with MyWebServer application

Expected results:
All tests should succeed