Manual Test Cases

1.Revision History

Date	Version	Author	Description
2020-12-02	1.0	Test manager	Adding Manual Test Case.
2020-12-06	1.1	Test manager	Adding Manual Test Cases.
2020-12-07	1.2	Test Designer	Designing Manual Test Cases.
2020-12-08	1.3	Test Designer	Designing Manual Test Cases
2020-12-08	1.4	Test manager & Test Designer	Adding Manual Test Cases.
2020-12-09	1.5	Test manager & Test Designer	Adding Manual Test Cases.
2020-12-11	1.6	Test manager & Test Designer	Adding Manual Test Cases.

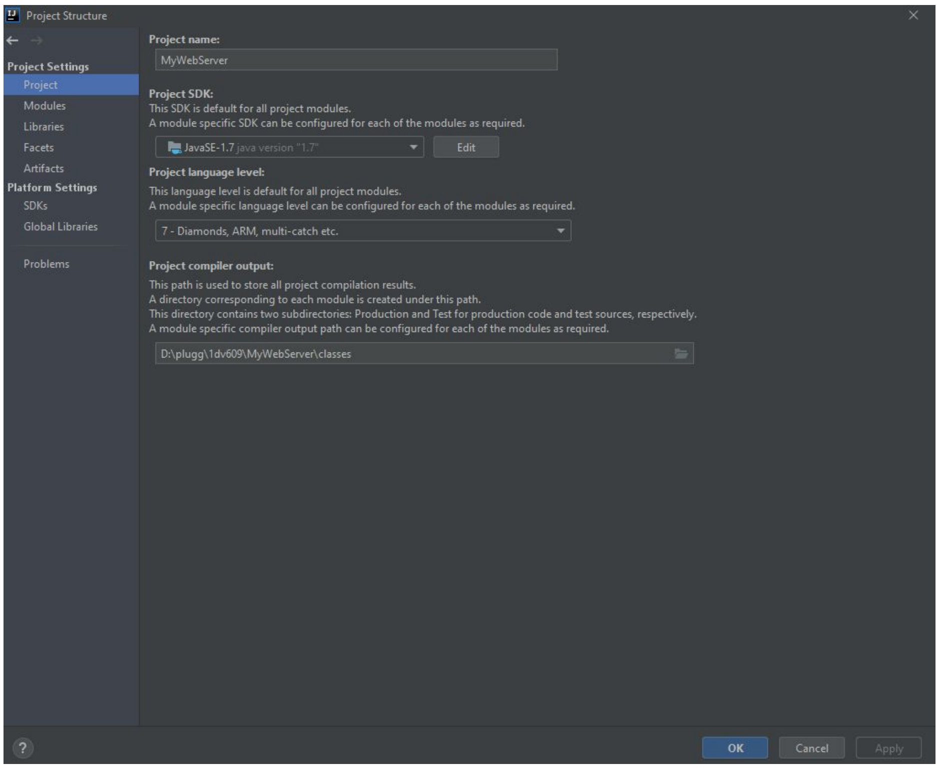
2. Manual Test Cases

Test Cases for Use Case 1, Start Server.

Test Case 1.1, Change Java settings in Intellij

Precondition	Download, Start Intellij and Install Java version 7 development kit (JDK).
Manual Test Steps	
Step 1	Open folder with name “MyWebServer”.
Step 2	Open File/Project Structure.
Step 3	Change Project SDK to java “version 1.7”, press Add SDK then add 1.7 jdk from your local computer.
Step 4	Change Project language level to “7 - Diamonds, ARM, multi-catch etc”.
Step 5	Press “Apply” to submit changes.

- **Expected result/output**



Project name:
MyWebServer

Project SDK:
This SDK is default for all project modules.
A module specific SDK can be configured for each of the modules as required.
JavaSE-1.7 java version "1.7"

Project language level:
This language level is default for all project modules.
A module specific language level can be configured for each of the modules as required.
7 - Diamonds, ARM, multi-catch etc.

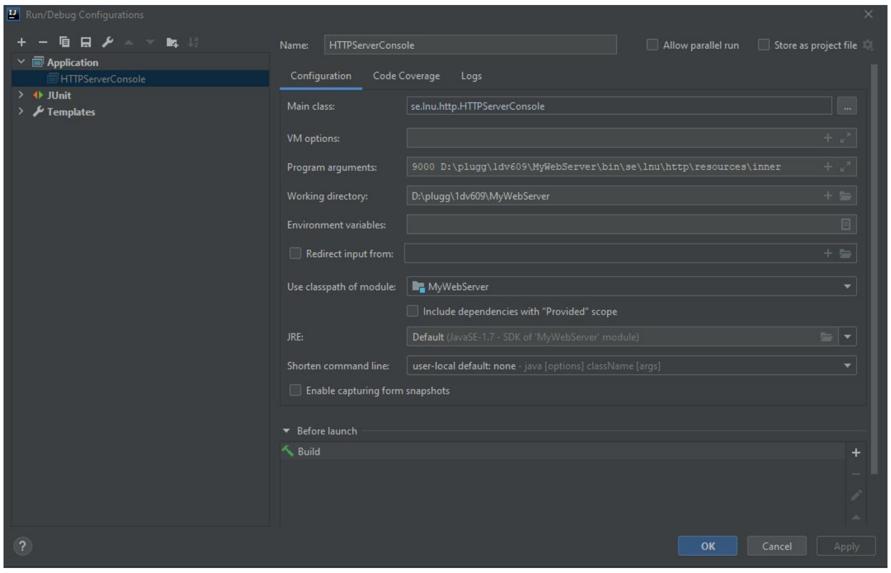
Project compiler output:
This path is used to store all project compilation results.
A directory corresponding to each module is created under this path.
This directory contains two subdirectories: Production and Test for production code and test sources, respectively.
A module specific compiler output path can be configured for each of the modules as required.
D:\plugg\1dv609\MyWebServer\classes

Test Case 1.2, Build project & Add program arguments

Precondition	Test Case 1.1, must be completed.
Manual Test Steps	
Step 1	Open Build and press "Build Project".
Step 2	Open the "src" folder.
Step 3	Right click on "HTTPServerConsole".
Step 4	Press Run "HTTPServerConsole.main()".
Step 5	Open Run/Edit Configurations.

Step 6	Make sure the Application and Name is “HTTPServerConsole”.
Step 7	Make sure the Main class is “se.lnu.http.HTTPServerConsole”.
Step 8	In Program arguments enter: “9000 “Project path”\MyWebServer\bin\se\lnu\http\resources\inner”.
Step 9	In the Working directory should have your “Project path”.
Step 10	Press Apply to submit changes.

- **Expected result/output**: “Run/Debug Configurations” in Intellij should look similar to the images except that Program arguments and Working directory File paths will be different.

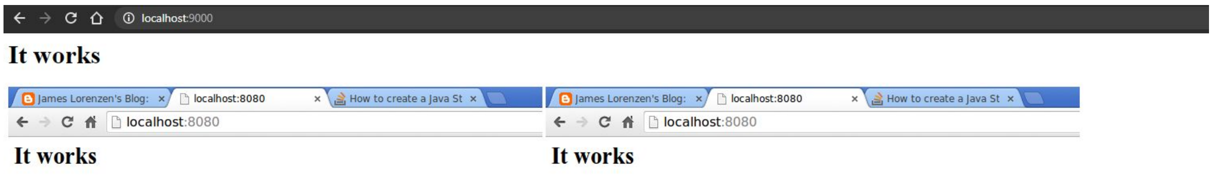


Name: HTTPServerConsole
Allow parallel run
Store as project file
Main class: se.lnu.http.HTTPServerConsole
VM options:
Program arguments: 9000 D:\plugg\1dv609\MyWebServer\bin\se\lnu\http\resources\inner
Working directory: D:\plugg\1dv609\MyWebServer
Environment variables:
Redirect input from:
Use classpath of module: MyWebServer
Include dependencies with "Provided" scope
JRE: Default (JavaSE-1.7 - SDK of 'MyWebServer' module)
Shorten command line: user-local default: none - java [options] className [args]
Enable capturing form snapshots

Test Case 1.3, Start the server in Intellij

Precondition	Test Case 1.2, must be completed.
Manual Test Steps	
Step 1	Press the "Run" icon in the top right.
Step 2	Open Google Chrome and enter "localhost:9000".

- **Expected result/output**: In the Google Chrome web browser, messages with "It works" is displayed and images of tabs is shown, the different tabs are called "James Lorenzen's Blog", "localhost:8080" and "How to create a Java St"



localhost:9000

It works

James Lorenzen's Blog: × | localhost:8080 × | How to create a Java St ×

localhost:8080

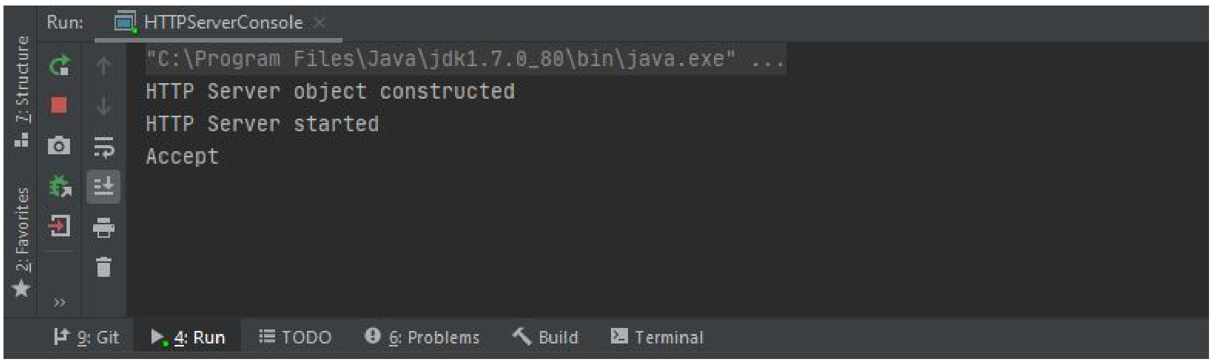
It works

James Lorenzen's Blog: × | localhost:8080 × | How to create a Java St ×

localhost:8080

It works

In the terminal, messages with "HTTP Server object constructed", "HTTP Server started" and "Accept" are shown.



```
"C:\Program Files\Java\jdk1.7.0_80\bin\java.exe" ...
HTTP Server object constructed
HTTP Server started
Accept
```

Test Case 1.4, Start Server in Intellij when socket is occupied

Precondition	Test Case 1.3
Manual Test Step	
Step 1	Open Intellij
Step 2	Press the "Run" icon in the top right.
Step 3	Open Google Chrome and enter "localhost:9000".

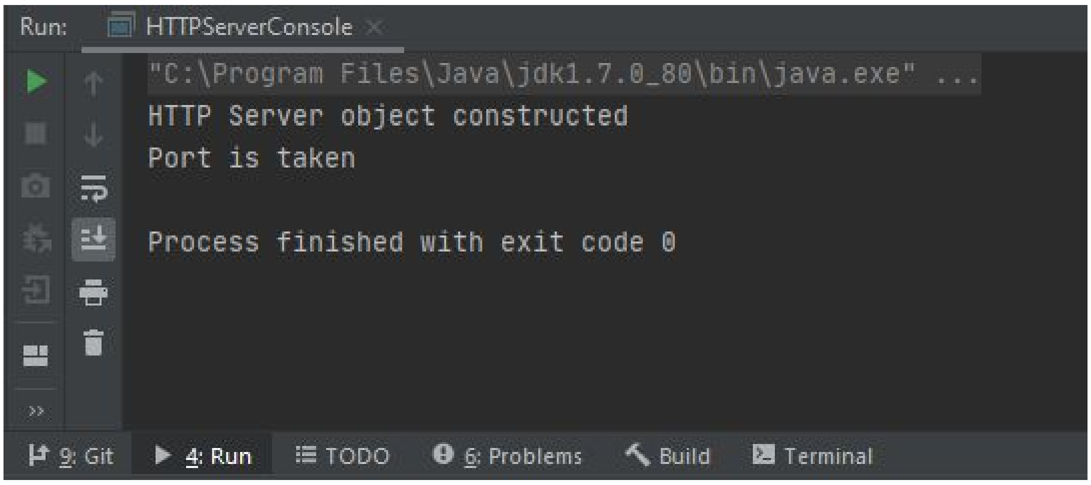
- **Expected result/output**:

  Message in Run "HTTP Server object constructed

  Port is taken

  Process finished with exit code 0

  " is displayed.



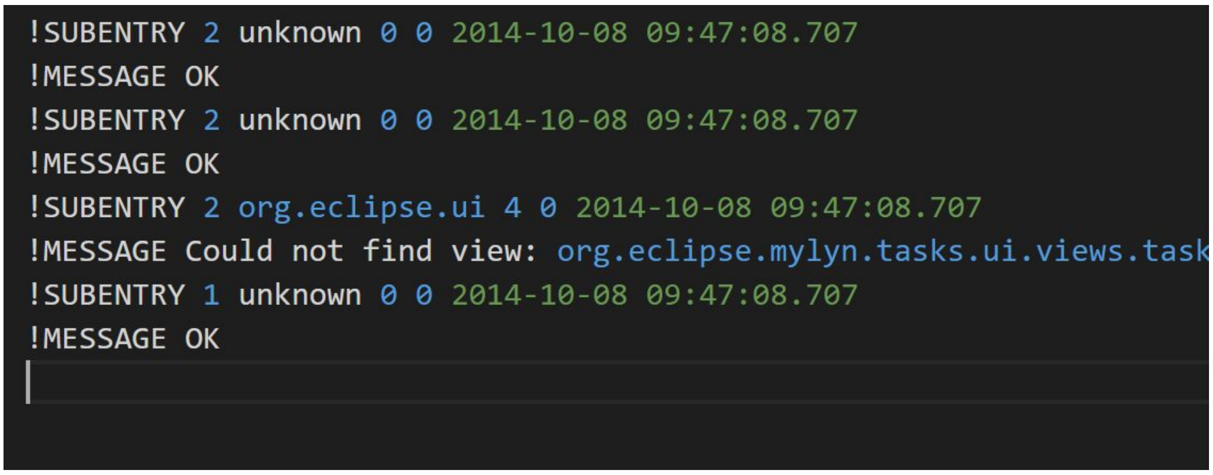
```
"C:\Program Files\Java\jdk1.7.0_80\bin\java.exe" ...
HTTP Server object constructed
Port is taken

Process finished with exit code 0
```

Test Case 1.5, The access log could not be written to

Precondition	Test Case 1.7 is completed.
Manual Test Steps	
Step 1	Go to folder with name “.metadata”
Step 2	Open file “.log”
Step 3	Check the logs

- **Expected result/output**:

  The .log file could not be written to the latest activity from 2014-10-08.



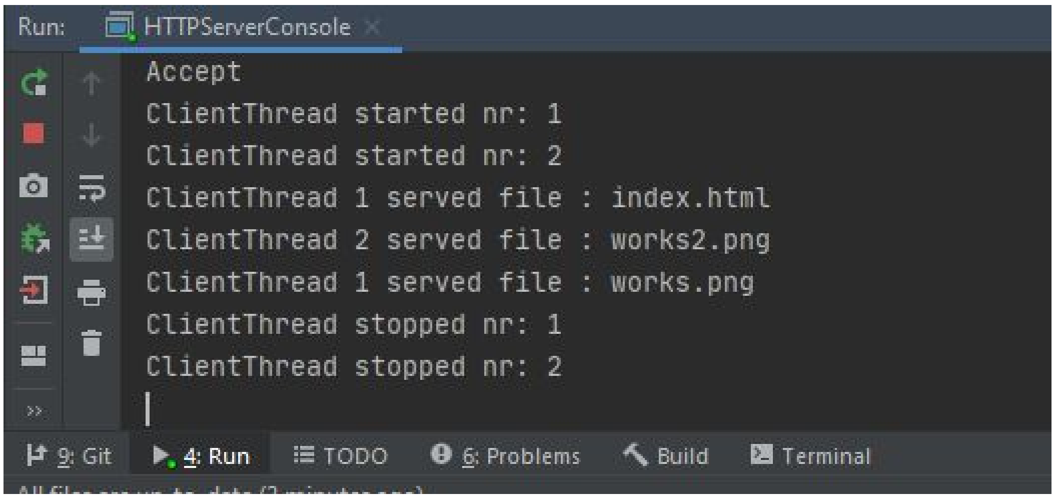
```
!SUBENTRY 2 unknown 0 0 2014-10-08 09:47:08.707
!MESSAGE OK
!SUBENTRY 2 unknown 0 0 2014-10-08 09:47:08.707
!MESSAGE OK
!SUBENTRY 2 org.eclipse.ui 4 0 2014-10-08 09:47:08.707
!MESSAGE Could not find view: org.eclipse.mylyn.tasks.ui.views.task
!SUBENTRY 1 unknown 0 0 2014-10-08 09:47:08.707
!MESSAGE OK
```

Message in the terminal “Cannot write to server log file log.txt” is shown.

Test Case 1.6, Refresh Page

Precondition	Test Case 1.3 or Test Case 1.7
Manual Test Steps	
Step 1	Press F5 to refresh the page.

- **Expected result/output**:



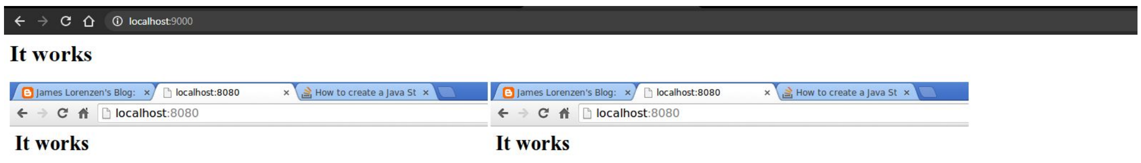
```
Accept
ClientThread started nr: 1
ClientThread started nr: 2
ClientThread 1 served file : index.html
ClientThread 2 served file : works2.png
ClientThread 1 served file : works.png
ClientThread stopped nr: 1
ClientThread stopped nr: 2
```

Test Case 1.7, Start Server in Visual Studio Code

Precondition	Download, run Visual Studio Code and install Java version 11 development kit (JDK). Download "Java Extension Pack" in Visual Studio Code.
Manual Test Steps	
Step 1	Open visual studio code.
Step 2	Open MyWebServer folder.
Step 3	Open terminal.
Step 4	Enter "cd bin" into the terminal.
Step 5	Enter "java se.lnu.http.HTTPServerConsole 9000 se/lnu/http/resources/inner".
Step 6	Enter "localhost:9000" in google chrome browser.

- **Expected result/output**:

In the Google Chrome web browser, messages with "It works" is displayed and images of tabs is shown, the different tabs are called "James Lorenzen's Blog", "localhost:8080" and "How to create a Java St"

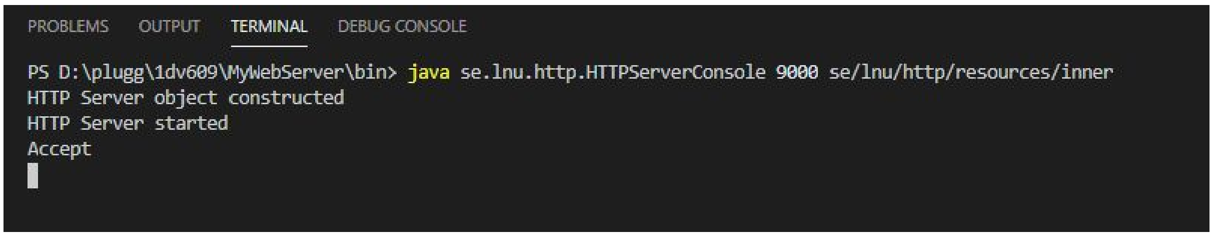


In the terminal, messages "PS D:\plugg\1dv609\MyWebServer\bin> java se.lnu.http.HTTPServerConsole 9000 se/lnu/http/resources/inner

HTTP Server object constructed

HTTP Server started

Accept”


PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
PS D:\plugg\1dv609\MyWebServer\bin> java se.lnu.http.HTTPServerConsole 9000 se/lnu/http/resources/inner
HTTP Server object constructed
HTTP Server started
Accept
```

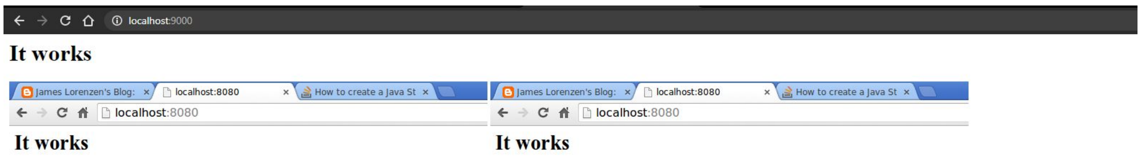
Test Case 1.8, Server still live when browser is reopened

Main scenario: Reopen browser.

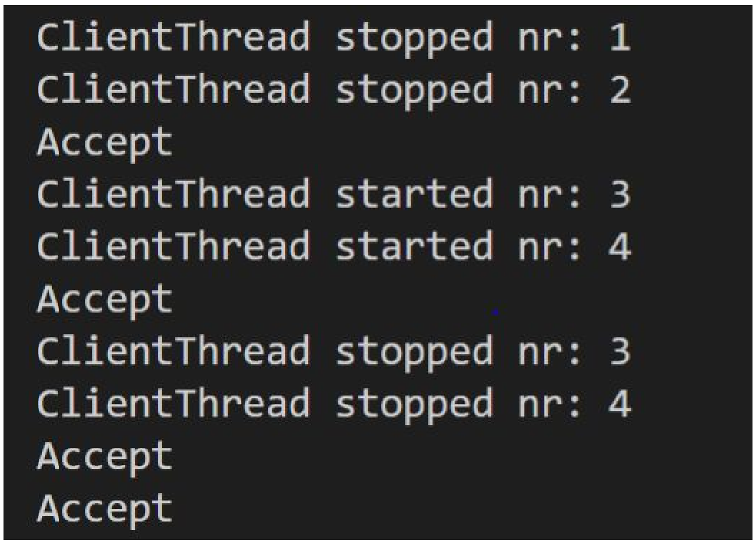
Precondition	Test Case 1.7 is completed
Manual Test Steps	
Step 1	Terminate google chrome browser on “localhost:9000”.
Step 2	open google chrome browser.
Step 3	Enter “localhost:9000”.

- **Expected result/output**:

In the Google Chrome web browser, messages with “It works” is displayed and images of tabs is shown, the different tabs are called “James Lorenzen’s Blog”, “localhost:8080” and “How to create a Java St”



Message in Visual Studio Terminal :



```
ClientThread stopped nr: 1
ClientThread stopped nr: 2
Accept
ClientThread started nr: 3
ClientThread started nr: 4
Accept
ClientThread stopped nr: 3
ClientThread stopped nr: 4
Accept
Accept
```

Test Case 1.9, Start Server in Visual Studio Code when socket is taken

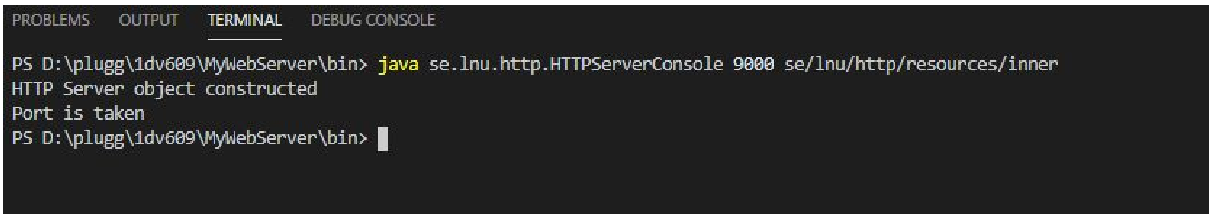
Precondition	Test Case 1.7
Manual Test Steps	
Step 1	Open visual studio code.
Step 2	Open MyWebServer folder.
Step 3	Open Visual Studio Terminal.
Step 4	Enter “cd bin” into the terminal.
Step 5	Enter “java se.lnu.http.HTTPServerConsole 9000 se/lnu/http/resources/inner”.

- **Expected result/output**:

  Message in the terminal:

“HTTP Server object constructed

Port is taken“



```
PS D:\plugg\1dv609\MyWebServer\bin> java se.lnu.http.HTTPServerConsole 9000 se/lnu/http/resources/inner
HTTP Server object constructed
Port is taken
PS D:\plugg\1dv609\MyWebServer\bin> 
```

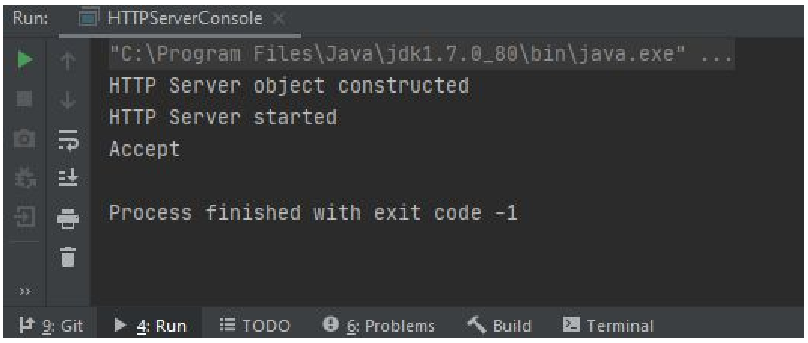
Test Cases for Use Case 2, Stop Server

Manual Test Case 2.1, Stop Server in Intellij,

Precondition	Test case 1.3 is completed.
Manual Test Steps	
Step 1	Press the stop button “Red square” in Intellij.

- **Expected result/output**:

  Message in Run is "Process finished with exit code -1"

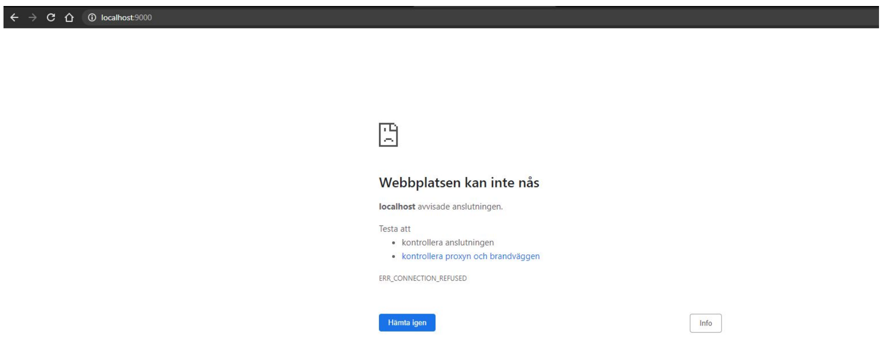


```
"C:\Program Files\Java\jdk1.7.0_80\bin\java.exe" ...
HTTP Server object constructed
HTTP Server started
Accept

Process finished with exit code -1
```

Test Case 2.2, Stop server by terminating Intellij

Precondition	Test case 1.3 is completed.
Manual Test Steps	
Step 1	Terminate intellij.

- **Expected result/output**:



Webbplatsen kan inte nås

localhost avvisade anslutningen.

Testa att
- kontrollera anslutningen
- kontrollera proxyn och brandväggen

ERR_CONNECTION_REFUSED

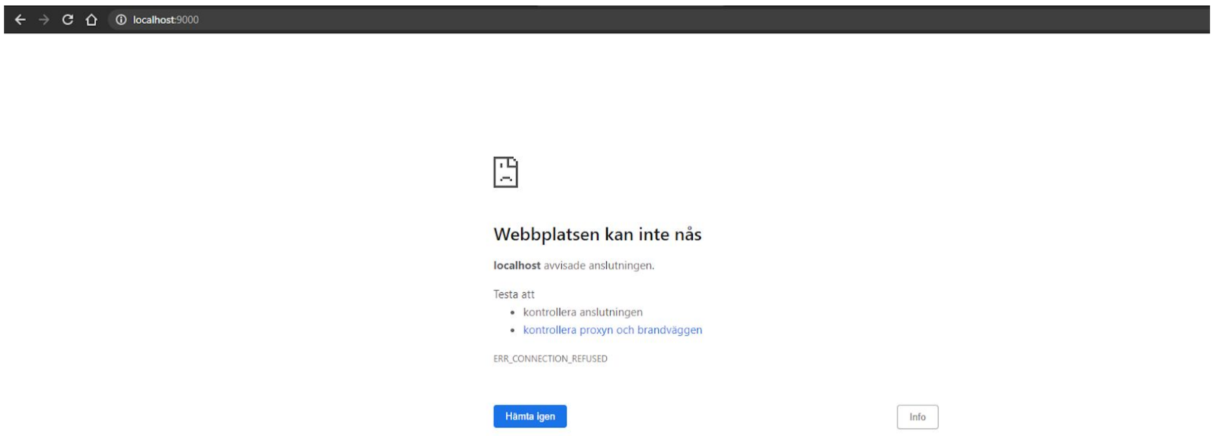
Test Case 2.3, Stop Server in Visual Studio Code

Precondition	Test case 1.9 is completed.
Manual Test Steps	
Step 1	Open terminal in Visual Studio Code.
Step 2	Enter CTRL + C into the terminal.

- **Expected result/output**:

  Server is stopped and cannot be accessed.

  In google chrome browser at localhost:9000.



localhost:9000

Webbplatsen kan inte nås

localhost avvisade anslutningen.

Testa att
- kontrollera anslutningen
- kontrollera proxyn och brandväggen

ERR_CONNECTION_REFUSED

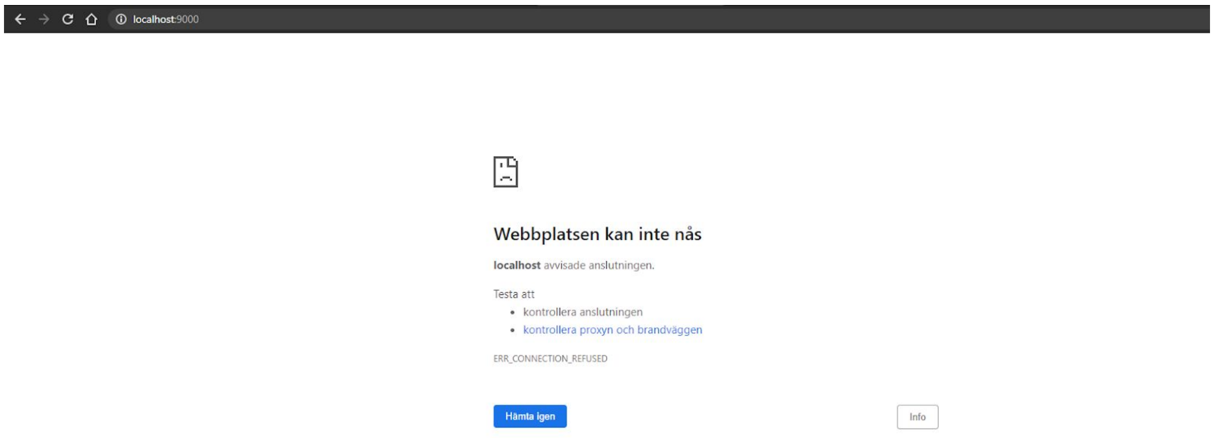
Test Case 2.4, Stop Server by terminating Visual Studio Code

Precondition	Test case 1.8 is completed.
Manual Test Steps	
Step 1	Terminate Visual Studio Code.

- **Expected result/output**:

Server is stopped and cannot be accessed.

In google chrome browser at localhost:9000.


localhost:9000

Webbplatsen kan inte nås

localhost avvisade anslutningen.

Testa att

- kontrollera anslutningen
- kontrollera proxyn och brandväggen

ERR_CONNECTION_REFUSED

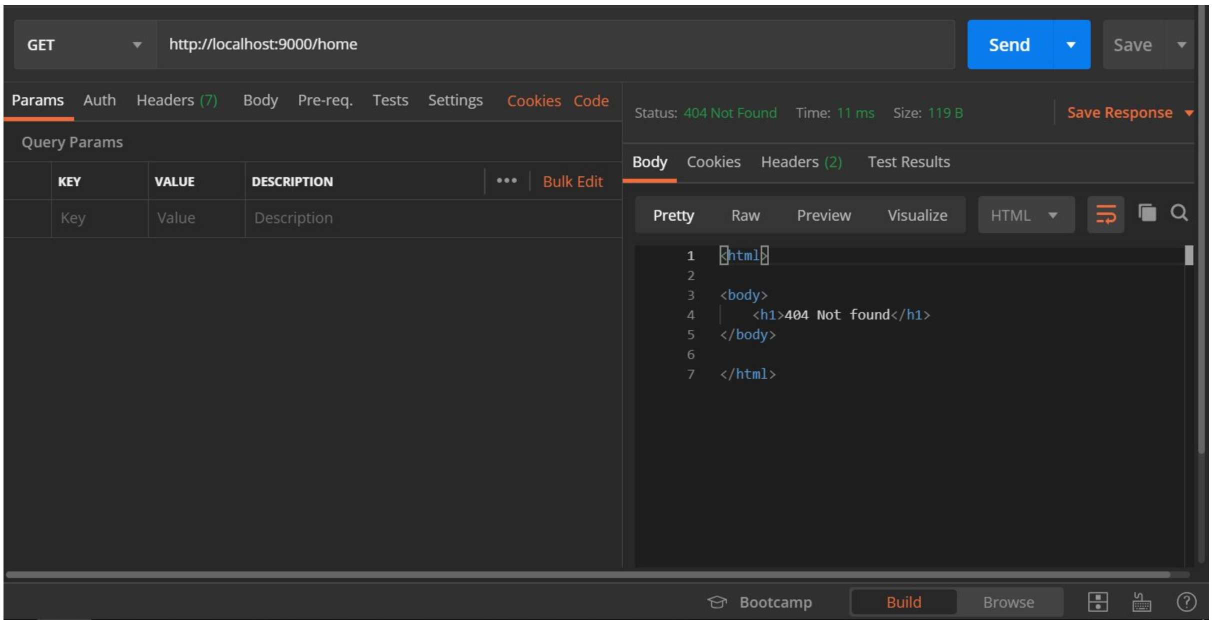
Test cases for Use Case 3, Request shared resource

Test Case 3.1, Web server shared resource cannot be found with faulty URL

Precondition	Test Case 1.8, download Postman
Manual Test Steps	
Step 1	Start the program Postman.
Step 2	Choose a “GET” request.
Step 3	Enter “http://localhost:9000/home” in the input.
Step 4	Press the “Send” button.

- **Expected result/output**:

  The GET request to http://localhost:9000/home will get Status 404 Not Found since the address does not currently exist in this content.



http://localhost:9000/home

Status: 404 Not Found Time: 11 ms Size: 119 B

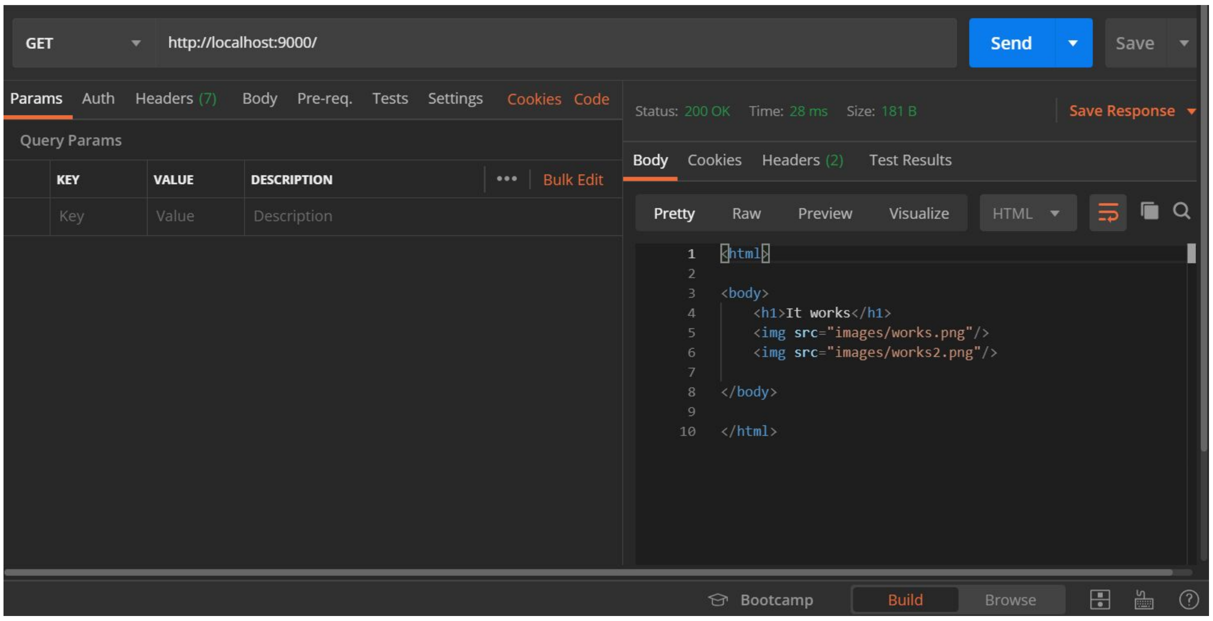
```
<html>

<body>
    <h1>404 Not found</h1>
</body>

</html>
```

Test Case 3.2, Get web server shared resource with correct URL

Precondition	Test Case 1.8, download Postman
Manual Test Steps	
Step 1	Start the program Postman.
Step 2	Choose a “GET” request.
Step 3	Enter “http://localhost:9000/” in the input.
Step 4	Press the “Send” button.

- **Expected result/output**:

  The GET request to http://localhost:9000/ will get Status 200 OK. Which means the web server is up and running.



Status: 200 OK Time: 28 ms Size: 181 B

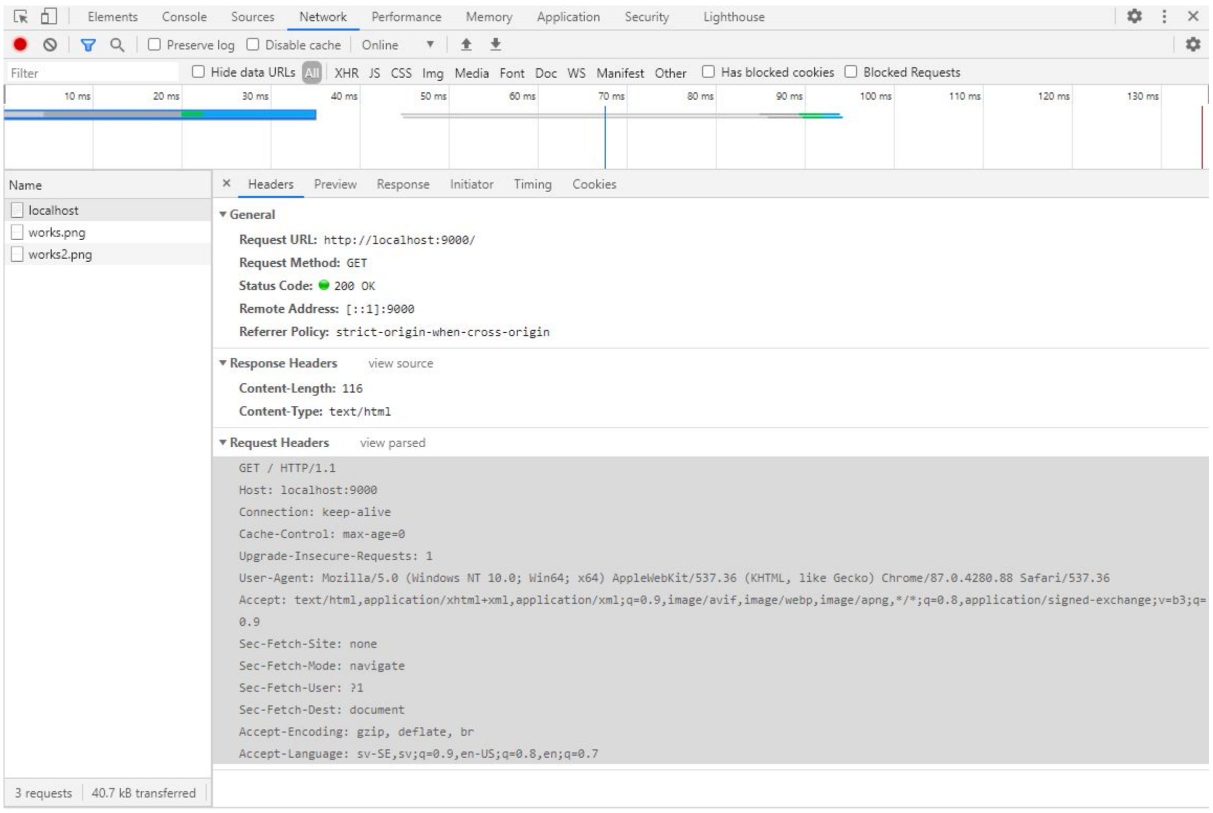
```
<html>

<body>
    <h1>It works</h1>
    <img src="images/works.png"/>
    <img src="images/works2.png"/>

</body>

</html>
```

Test Case 3.3, Web server follows HTTP 1.1.

Precondition	Test case 1.7 is completed, Web server is up and running, as well as you have navigated to localhost:9000 in the browser.
Manual Test Steps	
Step 1	Right click anywhere in the browser on the localhost:9000 page.
Step 2	Press “inspektera”/”inspect”.
Step 3	Go to Network in the menu.
Step 4	Press F5 to refresh the page.
Step 5	Press on localhost below “name” in Network.
Step 6	In Headers go to “Request Headers”.
Step 7	Press “view source” right of “Request Headers”.

- **Expected result/output**:

  In the browser:



localhost
works.png
works2.png

3 requests | 40.7 kB transferred

General

```
Request URL: http://localhost:9000/
Request Method: GET
Status Code: ● 200 OK
Remote Address: [::1]:9000
Referrer Policy: strict-origin-when-cross-origin
```

Response Headers view source

```
Content-Length: 116
Content-Type: text/html
```

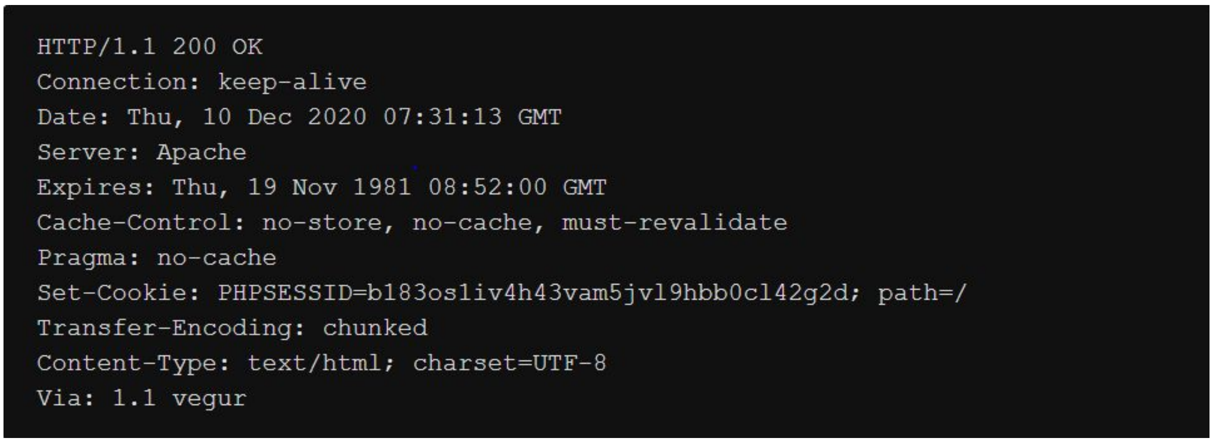
Request Headers view parsed

```
GET / HTTP/1.1
Host: localhost:9000
Connection: keep-alive
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=
0.9
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: sv-SE,sv;q=0.9,en-US;q=0.8,en;q=0.7
```

Test Case 3.4, Web server follows HTTP 1.1

Precondition	Test case 1.9 is completed, Web server is up and running.
Manual Test Steps	
Step 1	Deploy the web server on a public address.
Step 2	Go to https://redbot.org/
Step 3	Enter the URL that the web server is hosted on.

- **Expected result/output**: A verification has been done regarding what standard of the HTTP the server is using (1.1.).



```
HTTP/1.1 200 OK
Connection: keep-alive
Date: Thu, 10 Dec 2020 07:31:13 GMT
Server: Apache
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Set-Cookie: PHPSESSID=b183os1iv4h43vam5jvl9hbb0cl42g2d; path=/
Transfer-Encoding: chunked
Content-Type: text/html; charset=UTF-8
Via: 1.1 vegur
```

Test Case 4.0, Stress Test web server.

Precondition	Downloaded and started Intellij.
Manual Test Steps	
Step 1	Open the folder with the name “MyWebServer”.
Step 2	Open folder path “Test/se.lnu.http/integration”.
Step 3	Right click on the file “StressTest”.
Step 4	Press Run ‘StressTest’.

- **Expected result/output**:
  StressTest runs and passes. This shows that the web server can handle multiple requests at a short time span.

Test Case 4.1, Check Web Server status with HTTPServerTest

Precondition	Download and started Intellij
Manual Test Steps	
Step 1	Open the folder with the name “MyWebServer”.
Step 2	Open folder path “Test/se.lnu.http/integration”.
Step 3	Right click on the file “HTTPServerTest”.
Step 4	Press Run ‘HTTPServerTest’

- **Expected result/output**:
  HTTPServerTest runs and passes, checks when a web server is live and returns a 404 which stands for page “not found”.

Test Case 4.3, Check content on web server

Precondition	Download and started Intellij
Manual Test Steps	
Step 1	Open the folder with the name “MyWebServer”.
Step 2	Open folder path “Test/se.lnu.http/response”.
Step 3	Right click on the file “ContentTypeTest”.
Step 4	Press Run ‘ContentTypeTest’’

- **Expected result/output**:
  ContentTypeTest runs and passes, the test reads
  the file different html and checks that the content is correct.

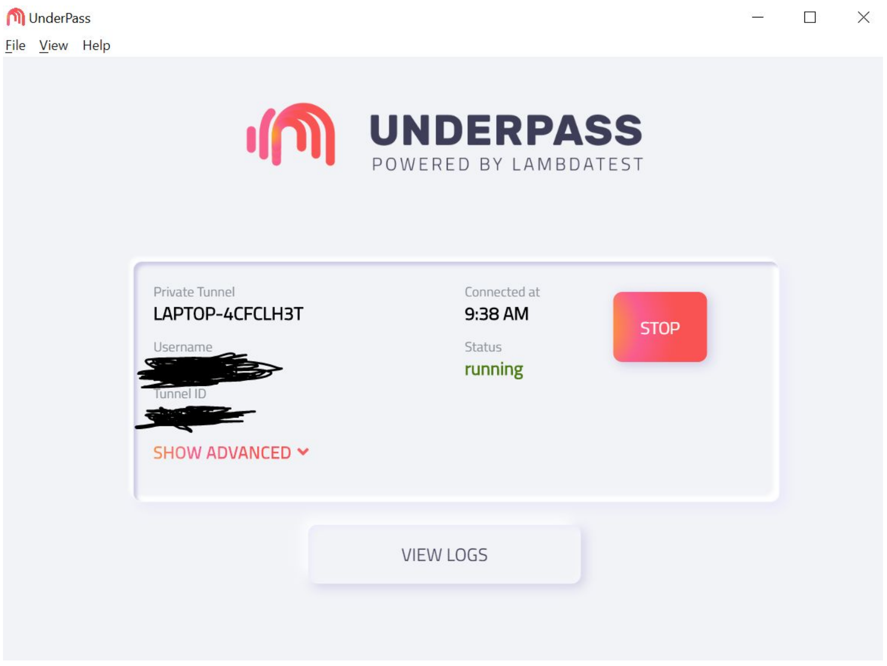
Test Case 4.4, Setting up lambdaTest Configure Tunnel to test web server in a different browser.

Precondition	Create an account on lambdaTest and enter the lambdaTest webpage.
Manual Test Steps	
Step 1	Go to “Real Time Testing”
Step 2	Start by going to “Configure Tunnel” to fix a setup locally.
Step 3	Go to “COMMAND LINE” and copy user and key.
Step 4	Open “DESKTOP APP”.
Step 5	Press “LAUNCH APP”.

Step 6	Enter the user and key that was copied.
Step 7	Press Start in the underpass.
Step 8	Go back to the lambdaTest page and refresh.

- **Expected result/output**:

  How it should look in UNDERPASS.



Private Tunnel
LAPTOP-4CFCLH3T

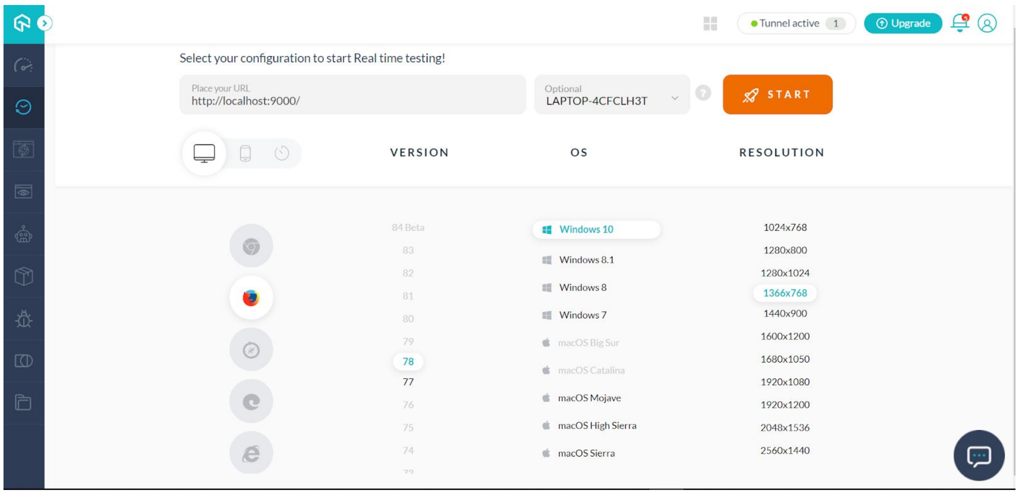
Connected at
9:38 AM

Username
[redacted]

Status
running

Tunnel ID
[redacted]

How it should look in LambdaTest.



Select your configuration to start Real time testing!

Place your URL
http://localhost:9000/

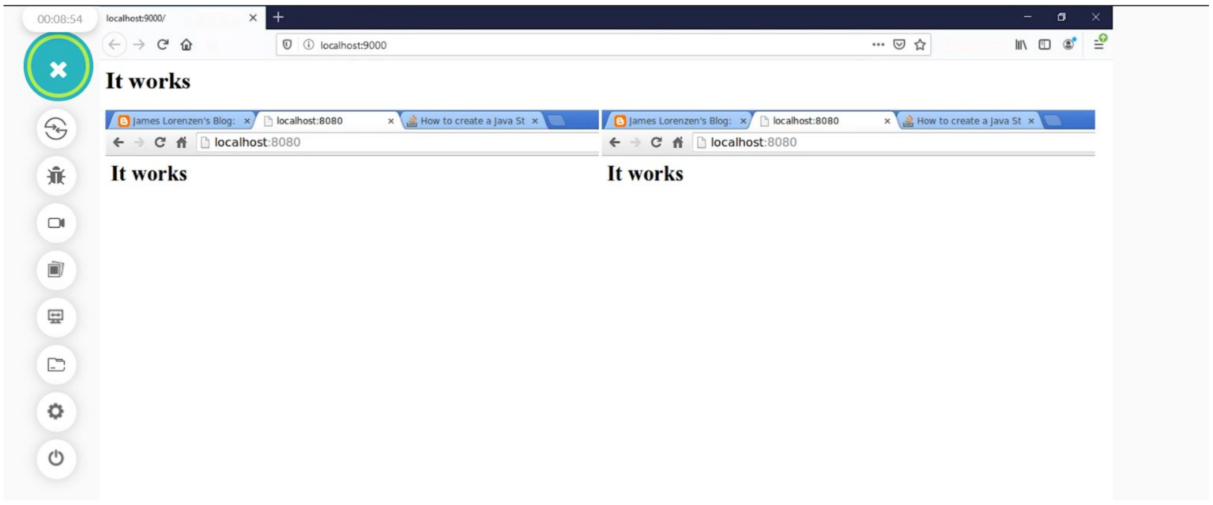
Optional
LAPTOP-4CFCLH3T

VERSION	OS	RESOLUTION
84 Beta	Windows 10	1024x768
83	Windows 8.1	1280x800
82	Windows 8	1280x1024
81	Windows 7	1366x768
80	macOS Big Sur	1440x900
79	macOS Catalina	1600x1200
78	macOS Mojave	1680x1050
77	macOS High Sierra	1920x1080
76	macOS Sierra	1920x1200
75		2048x1536
74		2560x1440

Test Case 4.5, Testing web server locally in Firefox with lambdaTest

Precondition	Test Case 4.4 and Test Case 1.7 is completed.
Manual Test Steps	
Step 1	In input “Place your URL” enter http://localhost:9000/.
Step 2	Choose browser press Firefox.
Step 3	Choose OS press Windows 10
Step 4	Press “Start”

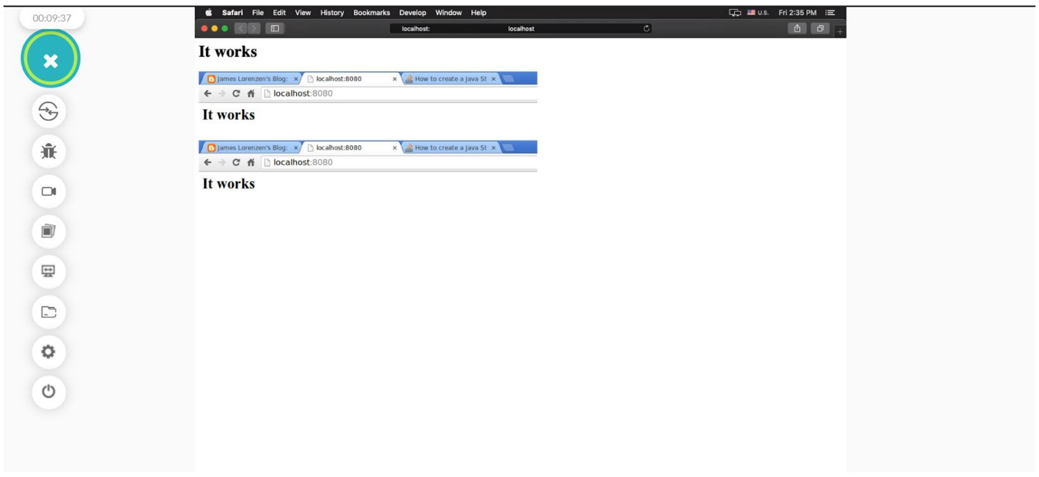
- **Expected result/output**:



Test Case 4.6, Testing web server locally in Safari and OS Mac with lambdaTest

Precondition	Test Case 4.4 and Test Case 1.7 is completed.
Manual Test Steps	
Step 1	In input "Place your URL" enter http://localhost:9000/.
Step 2	Choose browser press Safari.
Step 3	Choose OS press machOS Mojave
Step 4	Press "Start"

- **Expected result/output**:


00:09:37
Safari File Edit View History Bookmarks Develop Window Help
U.S. Fri 2:35 PM
localhost: localhost

It works

James Lorenzen's Blog: localhost:8080 How to create a Java St
localhost:8080

It works

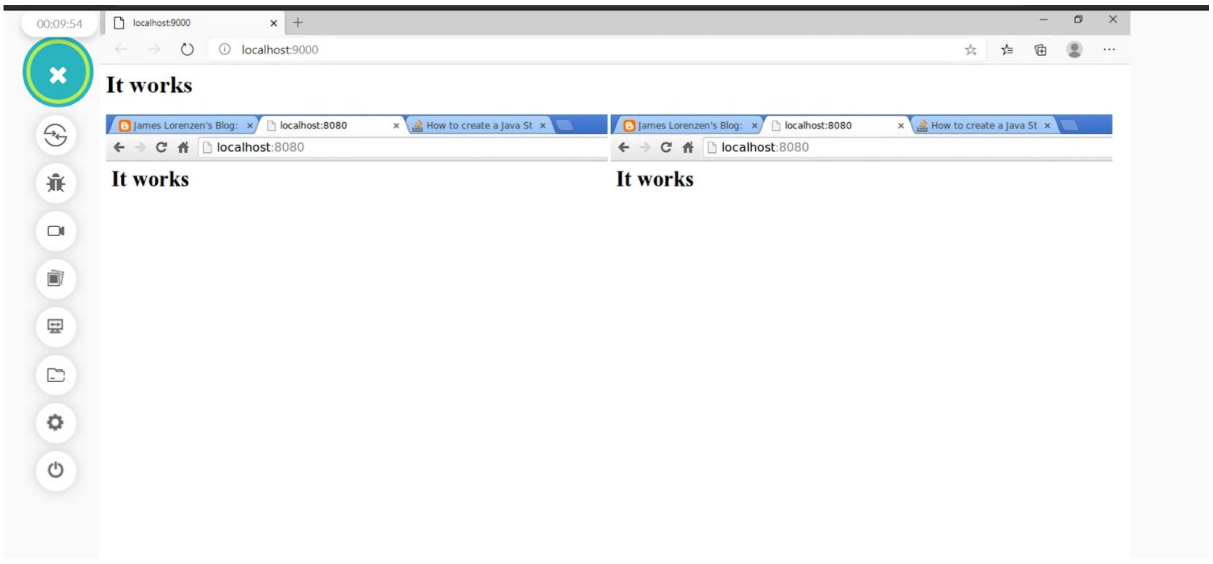
James Lorenzen's Blog: localhost:8080 How to create a Java St
localhost:8080

It works

Test Case 4.7, Testing web server locally in Edge with lambdaTest

Precondition	Test Case 4.4 and Test Case 1.7 is completed.
Manual Test Steps	
Step 1	In input "Place your URL" enter http://localhost:9000/.
Step 2	Choose browser press Edge.
Step 3	Choose OS press Windows 10.
Step 4	Press "Start".

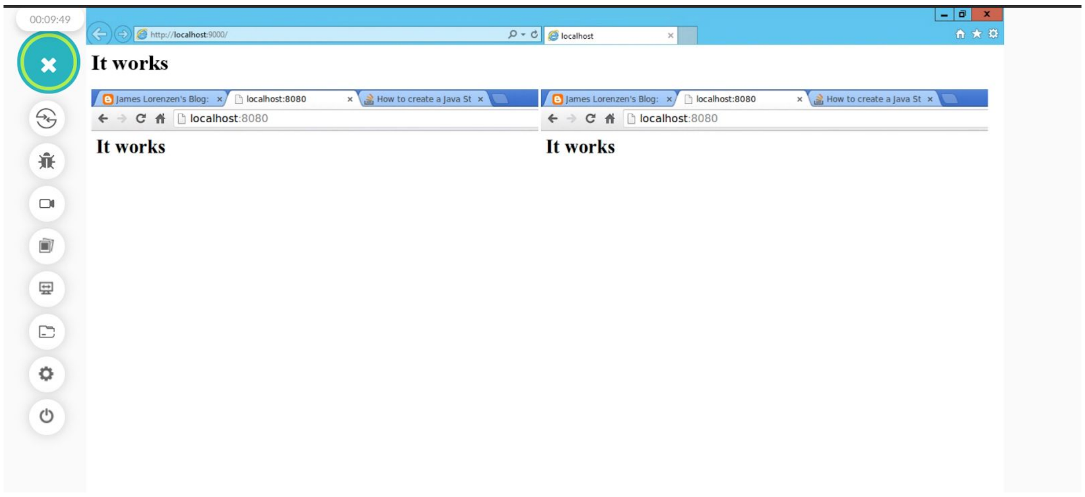
- **Expected result/output**:



Test Case 4.8, Testing web server locally in Internet Explorer with lambdaTest

Precondition	Test Case 4.4 and Test Case 1.7 is completed.
Manual Test Steps	
Step 1	In input “Place your URL” enter http://localhost:9000/.
Step 2	Choose browser press IE.
Step 3	Choose OS press Windows 8.1.
Step 4	Press “Start”.

- **Expected result/output**:



It works

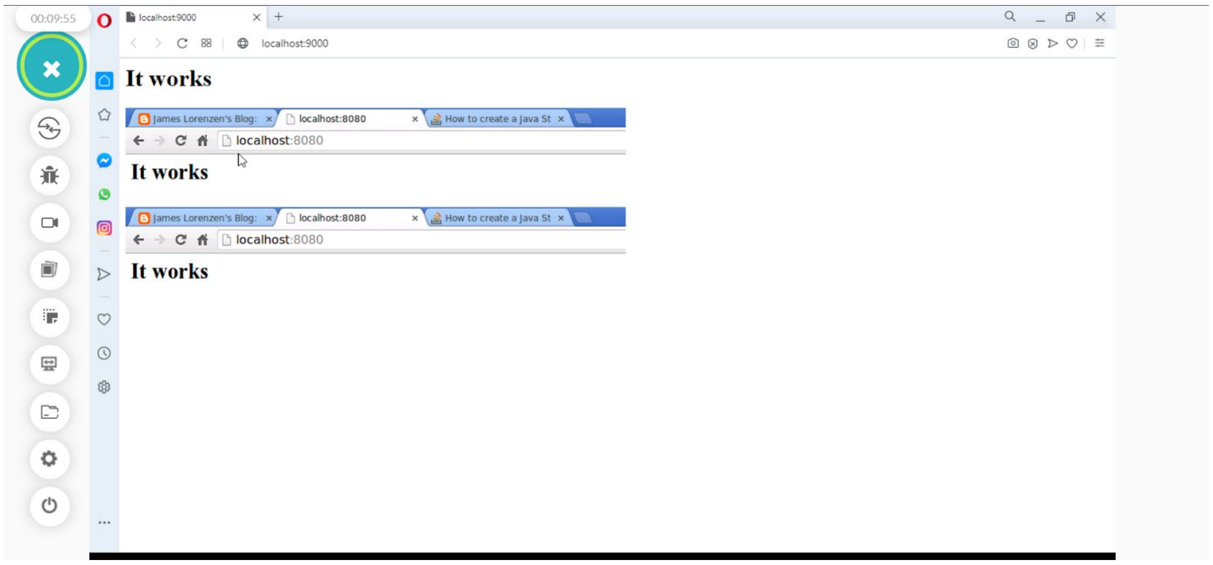
It works

It works

Test Case 4.9, Testing web server locally in Opera with lambdaTest

Precondition	Test Case 4.4 and Test Case 1.7 is completed.
Manual Test Steps	
Step 1	In input “Place your URL” enter http://localhost:9000/.
Step 2	Choose browser press Opera.
Step 3	Choose OS press Windows 10.
Step 4	Press “Start”.

- **Expected result/output**:



3.Time Log

Role	**Assignment**	**Description**	**Estimated Time(H)**	**Actual Time(H)**	**Date**
Test manager	Plan documentation for manual test cases.	Create structure for Manual Test Cases documentation	3	2	2020-12-02
Test manager	Adding manual test cases to start server and stop server.	Creating Manual Test Cases.	8	8	2020-12-07
Test designer	Adding manual test cases for HTTP standard	Designing Manual Test Cases.	4	4	2020-12-07
Test designer	Adding manual test cases for request share resource	Designing Manual Test Cases.	4	4	2020-12-08
Test manager	Adding manual test cases for start/stop server and request share resource.	Designing Manual Test Cases.	4	4	2020-12-08
Test designer	Adding Manual Test cases.	Adding Manual Test Cases.	4	4	2020-12-9
Test manager	Adding Manual Test Cases.	Adding Manual Test Cases.	4	4	2020-12-9
Test designer	Adding Manual Test	Adding Manual Test	4	4	2020-12-11

	Cases.	Cases.			
Test manager	Adding Manual Test Cases.	Adding Manual Test Cases.	4	4	2020-12-11