



Linnéuniversitetet
Kalmar Väst

Test Cases

|

Linnéuniversitetet
Kalmar Väst

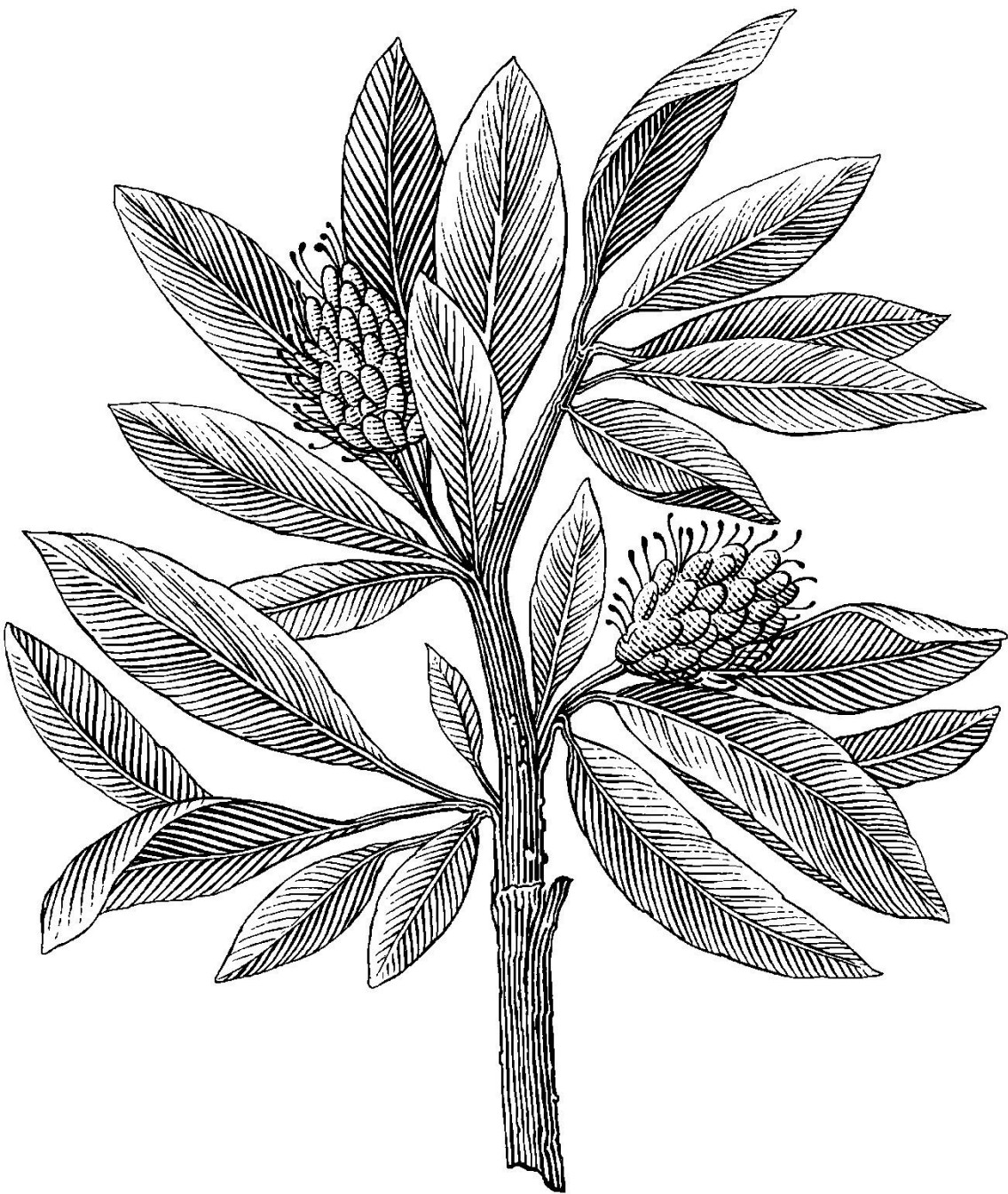


Table of Contents

Contents

1.Introduction	2
1.1General pre-condition	2
1.2Pre-condition for Postman.....	2
2.Manual Test Cases	3
TC 1.1.1 Start the server on a specific port number	3
TC1.1.2 Start the server and write a note in the access log.....	4
TC1.1.3 Server could not be started as port was taken	5
TC1.1.4 The web server could not be started due to restriction on the shared resource container.....	6
TC1.1.5 server could not start with a malformed sharing folder name (not a directory or a directory that does not exist)	7
TC1.1.6 server could not start without shared resource directory argument.	8
TC1.1.7 browser shows error “404 Not found” for using any path other	9
than “se/lnu/http/resources/inner”	9
TC1.1.8 Server could not start using a URL instead of directory for sharing resource argument.....	10
TC1.1.9 server could not start if port number is not first argument.....	11
TC 1.1.10 Server could not start with a port number less than 1.	12
TC1.1.11 Server could not start with a port number greater than 65535.	13
TC 1.1.12 Server could not start without initial arguments.....	13
TC 1.1.13 Server could not start with more than 2 initial arguments.....	14
TC 1.1.14 Server should start with one argument	15
TC 1.2.1 Server should stop when client wants to close the connection.....	16
TC1.2.2 Server should stop when user press CTRL + C	16
TC1.2.3 Server should not stop if user write anything other than “stop”	17
TC1.2.4 After stopping the server a note should be written in the access log file	18
TC 1.3.1 request a shared source- HTTP 200 ok.....	19
TC1.3.2 request shared resource – HTTP 403 Forbidden.....	20
TC 1.3.3 Request shared resource- HTTP 404 Not found	21
TC1.3.4 Request shared resource – HTTP 405 Method not Supported.....	22
TC1.4.1 The source code should be released under GPL-2.0.....	23
TC1.6.1 The web server should be responsive under high load.....	24
TC1.8.1 The access log should be viewable from a text editor.....	26

1.Introduction

This document is about all steps required to do manual tests. There are some general prerequisites necessary to run these manual tests, which are described in a separate sub-section. There would be a test case number for each test case to make it possible to track each test case relates to which of the mentioned requirements.

1.1General pre-condition

1. Install JDK 13 or higher.
2. Download the application MyWebServer from <https://github.com/dntoll/MyWebServer> and unzip it to a desired path.
3. Open Chrome browser.
4. Open Command Prompt in windows 10.
5. Change direction to where the unzipped MyWebServer-master is.
6. After changing direction to MyWebServer-master now write `cd MyWebServer-master` and press "enter".
7. Compile all the required java files using command `javac -sourcepath src -d bin -cp bin/se/lnu/http src/se/lnu/http/*.java`.
8. Change direction to bin by using command `cd bin` in Command Prompt.

1.2Pre-condition for Postman

- General pre-condition.
- Install Postman
- Run postman.

2.Manual Test Cases

TC 1.1.1 Start the server on a specific port number

Requirements under test:1, 1.1, 1.7

UC tested: use case1 Start Server

Test description

This manual test will test first part of main successful scenario of UC1(only starting the server).

Precondition

General precondition.

Test steps

- Run the command “java se.lnu.http.HTTPServerConsole 9000 se/lnu/http/resources/inner/”.
- In Chrome browser address bar write <http://localhost:9000> and press “enter”.

Expected result

(expected result shows server when started before starting browser and connecting client to server)

- In command prompt should be 3 lines as follow:

“HTTP Server object constructed

HTTP Server started

Accept”

- In browser:

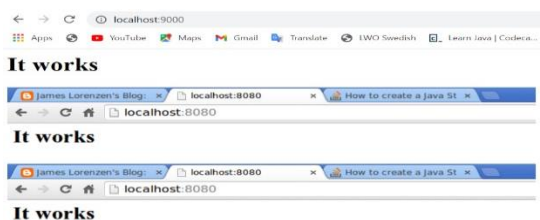
Shows a text “It works” and two provided images in path bin/se/lnu/http/resources/inner/images which are identical images saying “It works”

Actual results

In command prompt:

```
C:\Users\sas4n\Downloads\MyWebServer-master\MyWebServer-master\bin>java se.lnu.http.HTTPServerConsole 9000 se/lnu/http/r
resources/inner/
HTTP Server object constructed
HTTP Server started
Accept
```

In browser:



TC1.1.2 Start the server and write a note in the access log

requirements under test: 1, 1.1, 1.7.

Use case tested: use case1Start Server

Pre-condition

General pre-condition

Description

This test case tests manually whole main scenario of UC 1. This test case also tests the alternative scenario of UC 1 where “The access log could not be written to”

Test steps

- Run the command “java se.lnu.http.HTTPServerConsole 9000 se/lnu/http/resources/inner/”.
- In Chrome browser address bar write <http://localhost:9000> and press “enter”.

Expected results

(expected result shows server when started before starting browser and connecting client to server)

- In command prompt should be 3 lines as follow:

“HTTP Server object constructed

HTTP Server started

Accept”

A note in the access log was written, that the server was started.

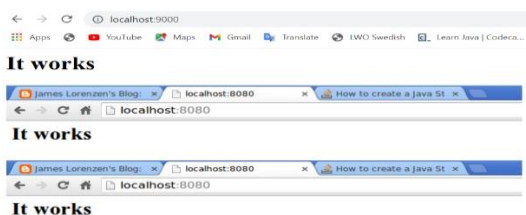
- In browser→Show a text “It works” and two provided images in path
bin/se/lnu/http/resources/inner/images which are identical images saying “It works”

Actual result

- In command prompt

```
C:\Users\sas4n\Downloads\MyWebServer-master\MyWebServer-master\bin>java se.lnu.http.HTTPServerConsole 9000 se/lnu/http/resources/inner/
HTTP Server object constructed
HTTP Server started
Accept
```

- In browser



There was not a file named “access log” to be written to.

TC1.1.3 Server could not be started as port was taken

Requirements under test: 1, 1.1, 1.7

Use case tested: use case 1 Start Server **Pre-condition**

- General pre-condition.
- Open another Command Prompt and do the general pre-condition steps 5-6-7-8.

Description

This test case, tests first alternative scenario of use case 1 where the chosen port is taken.

Test steps

- Run the command "java se.lnu.http.HTTPServerConsole 9000 se/lnu/http/resources/inner/".
- In Chrome browser address bar write <http://localhost:9000> and press "enter".
- In the second Command Prompt run command "java se.lnu.http.HTTPServerConsole 9000 se/lnu/http/resources/inner/".

Result expected

In second command prompt: "HTTP

Server object constructed

Port is taken".

Actual result

```
C:\Users\sas4n\Downloads\MyWebServer-master\MyWebServer-master\bin>java se.lnu.http.HTTPServerConsole 9000 se/lnu/http/r
esources/inner/
HTTP Server object constructed
Port is taken
```

TC1.1.4 The web server could not be started due to restriction on the shared resource container

Requirements under test: 1, 1.1, 1.7.

Use case tested: use case1 Start Server **Pre-condition**

- General pre-condition.
- Change the permission to resource folder as follow:
- Open Command Prompt as administrator
- Run command "cacls <file path to sharing resource> (in test lead's system C:\Users\sas4n\Downloads\MyWebServer-master\MyWebServer-master\bin\se\lnu\http\resources) /p everyone:n" to restrict the resource directory.

Test description

This test case checks if client can have access to a restricted folder or not and what would be the error message.

Test steps

- Run the command "java se.lnu.http.HTTPServerConsole 9000 se/lnu/http/resources/inner/".

Expected result

The system presents an error message: "No access to resource folder" **Actual**

result

```
Exception in thread "main" se.lnu.http.exceptions.NotADirectoryException: se.lnu.http.exceptions.NotADirectoryException:
directory does not exists
```


TC1.1.5 server could not start with a malformed sharing folder name (not a directory or a directory that does not exist)

Requirements under test : 1, 1.1, 1.7.

Use Case tested: use case 1 Start Server

Pre-condition

General pre-condition.

Test description

This test case checks if server can start with a directory that does not exist or when user write directory path wrong(misspelled).

Test steps

- Run the command “java se.lnu.http.HTTPServerConsole 9000
se/lnu/http/resourcedre/inner”

Expected result

System shows an error of

“Exception in thread "main" se.lnu.http.exceptions.NotADirectoryException:
se.lnu.http.exceptions.NotADirectoryException: directory does not exists”.

Actual result

```
C:\Users\sas4n\Downloads\MyWebServer-master\MyWebServer-master\bin>java se.lnu.http.  
ServerConsole 9000 se/lnu/http/resourcedre/inner  
Exception in thread "main" se.lnu.http.exceptions.NotADirectoryException: se.lnu.htt  
ceptions.NotADirectoryException: directory does not exists
```

TC1.1.6 server could not start without shared resource directory argument.

Requirements under test: 1, 1.1, 1.7.

Use Case tested: use case 1 Start Server

Pre-condition

General pre-condition.

Test description

This test case checks if server can start with only port number as an initial argument.

Test steps

- Run the command "java se.lnu.http.HTTPServerConsole 9000"

Expected result

System shows an error

"Exception in thread "main" se.lnu.http.exceptions.NotADirectoryException:
java.lang.ArrayIndexOutOfBoundsException: Index 1 out of bounds for length 1".

Actual result

```
C:\Users\sas4n\Downloads\MyWebServer-master\MyWebServer-master\bin>java se.lnu.http
ServerConsole 9000
Exception in thread "main" se.lnu.http.exceptions.NotADirectoryException: java.lang
yIndexOutOfBoundsException: Index 1 out of bounds for length 1
```

TC1.1.7 browser shows error “404 Not found” for using any path other than “se/lnu/http/resources/inner”

Requirements under test: 1, 1.1, 1.5, 1.7.

Use case tested: use case 1 Start Server

Pre-condition

General pre-condition.

Test description

This test case checks what happens when server starts with a directory other than specified sharing resource directory.

Test steps

- Run the command “java se.lnu.http.HTTPServerConsole 9000 se/lnu/http/resources”

Expected result

(expected result shows server when started before starting browser and connecting client to server)

Server starts successfully and show output:

“HTTP Server object constructed

HTTP Server started Accept”.

Browser display:

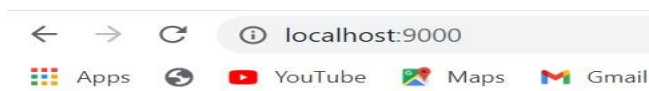
“404 Not found”.

Actual result

- In command prompt

```
C:\Users\sas4n\Downloads\MyWebServer-master\MyWebServer-master\bin>java se.lnu.http.HTTPServerConsole 9000 se/lnu/http/resources
HTTP Server object constructed
HTTP Server started
Accept
```

- Browser



404 Not found

TC1.1.8 Server could not start using a URL instead of directory for sharing resource argument.

Requirements under test: 1, 1.1, 1.7.

Use case tested: use case 1 Start Server

Pre-condition

General pre-condition.

Test description

This test case checks if server can start when path to the index.html file is used instead of path to the shared resource directory.

Test steps

- Run the command "java se.lnu.http.HTTPServerConsole 9000 se/lnu/http/resources/inner/index.html"

Expected result

System shows error:

"Exception in thread "main" se.lnu.http.exceptions.NotADirectoryException:

se.lnu.http.exceptions.NotADirectoryException: url is not a directory". **Actual**

result

```
C:\Users\sas4n\Downloads\MyWebServer-master\MyWebServer-master\bin>java se.lnu.http.HTTP
ServerConsole 9000 se/lnu/http/resources/inner/index.html
Exception in thread "main" se.lnu.http.exceptions.NotADirectoryException: se.lnu.http.ex
ceptions.NotADirectoryException: url is not a directory
```

TC1.1.9 server could not start if port number is not first argument

Requirements under test: 1, 1.1, 1.7.

Use case tested: use case 1 Start Server

Pre-condition

General pre-condition.

Test description

This test case checks if server can start when first argument is not a valid integer number.

Test steps

- Run the command "java se.lnu.http.HTTPServerConsole se/lnu/http/resources/inner/"

Expected result

System shows error: "Enter a valid port 1-65535 and a optional URL".

Actual result

```
C:\Users\sas4n\Downloads\MyWebServer-master\MyWebServer-master\bin>java se.lnu.http.HTTP
ServerConsole se/lnu/http/resources/inner/
Enter a valid port 1-65535 and a optional URL
```

TC 1.1.10 Server could not start with a port number less than 1.

Requirements under test: 1, 1.1, 1.7

Use case tested: use case 1 Start Server

Pre-condition

General pre-condition.

Test description

This test case if server can start with a port number less than 1.

Test steps

- Run the command "java se.lnu.http.HTTPServerConsole -1 se/lnu/http/resources/inner/"

Expected result

System shows error: "Enter a valid port 1-65535 and a optional URL".

Actual result

```
C:\Users\sas4n\Downloads\MyWebServer-master\MyWebServer-master\bin>java se.lnu.http.HTTP
ServerConsole -1 se/lnu/http/resources/inner/
Enter a valid port 1-65535 and a optional URL
```

TC1.1.11 Server could not start with a port number greater than 65535.

Requirements under test: 1, 1.1, 1.7

Use case tested: use case 1 Start Server

Pre-condition

General pre-condition.

Test description

This test case checks if server can start with a port number greater than 65535.

Test steps

- Run the command "java se.lnu.http.HTTPServerConsole 65536 se/lnu/http/resources/inner/"

Expected result

System shows error: "Enter a valid port 1-65535 and a optional URL".

Actual result

```
C:\Users\sas4n\Downloads\MyWebServer-master\MyWebServer-master\bin>java se.lnu.http.HTTP
ServerConsole 65536 se/lnu/http/resources/inner/
Enter a valid port 1-65535 and a optional URL
```

TC 1.1.12 Server could not start without initial arguments

Requirements under test: 1, 1.1, 1.7.

Use case tested: use case 1 Start Server

Pre-condition

General pre-condition.

Test description

This test case checks if server can start without any arguments.

Test steps

- Run the command "java se.lnu.http.HTTPServerConsole"

Expected result

System shows error: "Exception in thread "main"

se.lnu.http.exceptions.WrongNumberOfArgumentsException: This program should only have one or two arguments".

Actual result

```
C:\Users\sas4n\Downloads\MyWebServer-master\MyWebServer-master\bin>java se.lnu.http.HTTP
ServerConsole
Exception in thread "main" se.lnu.http.exceptions.WrongNumberOfArgumentsException: This
program should only have one or two arguments
```

TC 1.1.13 Server could not start with more than 2 initial arguments

Requirements under test: 1, 1.1, 1.7

Use case tested: use case 1 Start Server

Pre-condition

General pre-condition.

Test description

This test case check if server can start with more than 2 arguments.

Test steps

- Run the command “java se.lnu.http.HTTPServerConsole 9000 se/lnu/http/resources/inner/rt”

Expected result

System shows error:

“Exception in thread "main" se.lnu.http.exceptions.WrongNumberOfArgumentsException: This program should only have one or two arguments”.

Actual result

```
C:\Users\sas4n\Downloads\MyWebServer-master\MyWebServer-master\bin>java se.lnu.http.HTTP
ServerConsole 9000 se/lnu/http/resources/inner/ rt
Exception in thread "main" se.lnu.http.exceptions.WrongNumberOfArgumentsException: This
program should only have one or two arguments
```


TC 1.1.14 Server should start with one argument

Requirements under test: 1, 1.1, 1.7.

Use case tested: use case 1 Start Server

Pre-condition

General pre-condition.

Test description

This test case is for checking if the error message displayed by system when administrator try to start the server with more than 2 arguments which is “This program should only have one or two arguments” is correct or not.

Test steps

- Run the command “java se.lnu.http.HTTPServerConsole 9000”

Expected result

Server should run successfully.

Actual result

```
C:\Users\sas4n\Downloads\MyWebServer-master\MyWebServer-master\bin>java se.lnu.h
ttp.HTTPServerConsole 9000
Exception in thread "main" se.lnu.http.exceptions.NotADirectoryException: java.l
ang.ArrayIndexOutOfBoundsException: Index 1 out of bounds for length 1
```

TC 1.2.1 Server should stop when client wants to close the connection

Requirements under test: 1, 1.2, 1.7.

Use case tested: use case2 Stop Server

Pre-condition

The web server is started and running.

Description

This test case will check if server stopped after writing “stop” in command prompt or not(writing to log access file will check in next test case).

Test steps

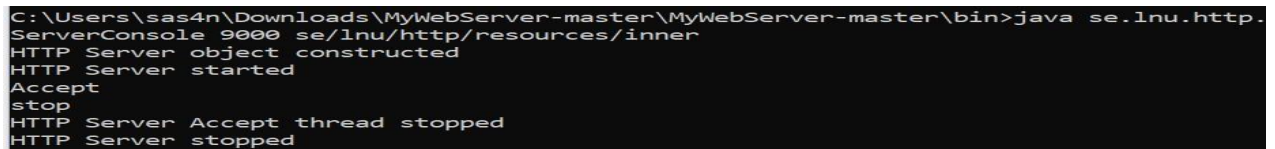
- Write “stop” in Command Prompt when server is running and press “enter”.

Expected result

Server stopped and following displayed as output:

“HTTP Server Accept thread stopped
HTTP Server stopped”.

Actual result



```
C:\Users\sas4n\Downloads\MyWebServer-master\MyWebServer-master\bin>java se.lnu.http.  
ServerConsole 9000 se/lnu/http/resources/inner  
HTTP Server object constructed  
HTTP Server started  
Accept  
stop  
HTTP Server Accept thread stopped  
HTTP Server stopped
```

TC1.2.2 Server should stop when user press CTRL + C

Requirements under test:1, 1.2, 1.7.

Use case tested: Use case 2 Stop Server

Pre-condition

The web server is started and running.

Description

This test case checks if server stops after pressing CTRL+C in command prompt.

Test steps

- Press CTRL + C in command prompt to stop the server.

Expected result

Server stops.

Actual result Server

stops.

TC1.2.3 Server should not stop if user write anything other than “stop”

Requirements under test: 1, 1.2, 1.7

Use case tested: use case 2 Stop Server

Pre-condition

The web server is started and running.

Description

This test case will check if server stops after writing something other than “stop” in Command Prompt or not (writing to log access file will check in next test case).

Test steps

- Write “hello” in Command Prompt and press “enter”.
- Write “do not stop” in Command Prompt and press “enter”.
- Write “stoppp” in Command Prompt and press “enter”.

Expected result

Server does not stop.

Actual result

Server does not stop

```
C:\Users\sas4n\Downloads\MyWebServer-master\MyWebServer-master\bin>java se.lnu.h
ttp.HTTPServerConsole 9000 se/lnu/http/resources/inner/
HTTP Server object constructed
HTTP Server started
Accept
hello
do not stop
stoppp
-
```

TC1.2.4 After stopping the server a note should be written in the access log file

Requirements under test: 1, 1.2, 1.7

Use case tested: use case 2 Stop Server

Pre-condition

The web server is started and running.

Description

This test case checks if a note is written in access log file after stopping the server.

Test steps

- Write “stop” and press “Enter” in Command Prompt to start server

Expected result

A note is written in access log file regarding server is stopped.

Actual results

Access log file does not exist in provided codes and files.

TC 1.3.1 request a shared source- HTTP 200 ok

Requirements under test: 1, 1.3, 1.5, 1.7.

Use case tested: Use case 3 Request shared resource

Pre-condition

Pre-condition for Postman

Description

This test case check if a status code of 200 displayed after a successful request sent.

Test steps

- Choose “GET” option in Postman.
- Write <http://localhost:9000> in URL text-field and press “Send”.

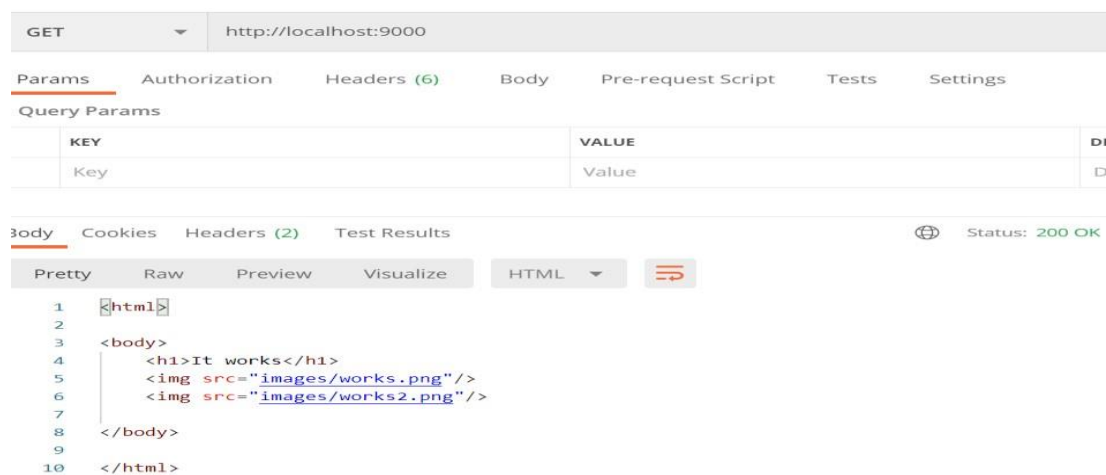
Expected result

- Postman shows status code “200 Ok”.
- Postman shows:

“ <h1>It works</h1>

” in “Body” tab.

Actual result



TC1.3.2 request shared resource – HTTP 403 Forbidden

Requirements under test: 1, 1.3, 1.5, 1.7.

Use case tested: use case 3 Request shared resource

Pre-condition

Pre-condition for Postman

Description

This test case checks if a status code of 403 displayed after a forbidden request sent by client.

Test steps

- Choose “GET” option in Postman.
- Write <http://localhost:9000/./AcceptThread.class> in URL text-field and press “Send”.

Expected result

- Postman displays a status code of “403 Forbidden”.
- Postman displays “<h1>403 Forbidden</h1>” in “Body” tab.

Actual result

GET http://localhost:9000/./AcceptThread.class

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (2) Test Results

Status: 403 Forbidden Time: 16 ms

403 Forbidden

The request was a legal request, but the server is refusing to respond to it. Unlike a 401 Unauthorized response, authenticating will make no difference.

```
1 <html>
2
3 <body>
4   <h1>403 Forbidden</h1>
5 </body>
6
7 </html>
```

TC 1.3.3 Request shared resource- HTTP 404 Not found

Requirements under test: 1, 1.3, 1.5, 1.7.

Use case tested: use case 3 Request shared resource

Pre-condition

Pre-condition for Postman.

Description

This test case checks if a status code of 404 displayed after a client sent a request for a file or path which does not exist or is not accessible now. **Test steps**

- Choose “GET” option in Postman.
- Write <http://localhost:9000/accessLog.txt> in URL text-field and press “Send”.

Expected result

- Postman displays status code of “404 Not Found”
- Postman displays “<h1>404 Not found</h1>” in “Body” tab.

Actual result

The screenshot shows the Postman interface for a GET request to `http://localhost:9000/accessLog.txt`. The status is **404 Not Found** with a response time of 25 ms. The response body is displayed in HTML format, showing the following structure:

```
1 <html>
2
3 <body>
4   <h1>404 Not found</h1>
5 </body>
6
7 </html>
```

A message box on the right states: "404 Not Found. The requested resource could not be found but may be available again in the future. Subsequent requests by the client are permissible."

TC1.3.4 Request shared resource – HTTP 405 Method not Supported

Requirements under test: 1, 1.3, 1.5, 1.7

Use case tested: use case 3 Request shared resource

Pre-condition

Pre-condition for Postman.

Description

This test case checks if any request other than “GET” is supported or not. **Test**

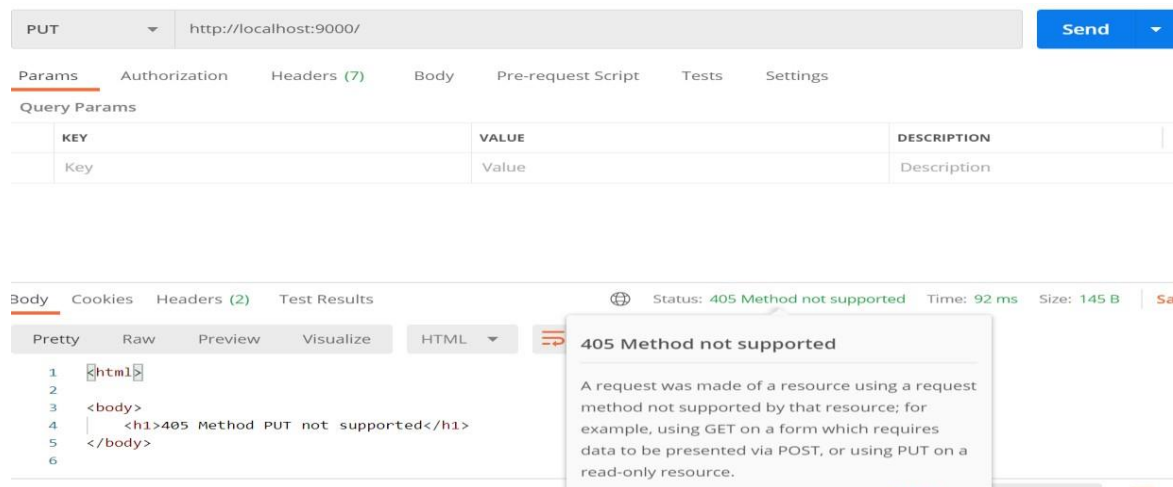
steps

- Choose “PUT” option in Postman.
- Write http://localhost:9000/ in URL text-field and press “Send”.

Expected result

- Postman displays status code of “405 Method not supported”
- Postman displays “<h1>405 Method Put not supported</h1>” in “Body” tab.

Actual result



TC1.4.1 The source code should be released under GPL-2.0

Requirements under test: 1, 1.4.

Pre-condition

General pre-condition steps 2-4-5-6.

Test steps

- Run command "type LICENSE" in Command Prompt.

Expected result

Source code released under GPL-2.0.

Actual result

It can be seen source code released under MIT license.

```
C:\Users\sas4n\Downloads\MyWebServer-master\MyWebServer-master>type LICENSE
The MIT License (MIT)

Copyright (c) 2014 Daniel Toll

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

TC1.6.1 The web server should be responsive under high load

Requirement under test: 1.6

Pre-condition

- General pre-condition
- Download Apache Jmeter(zip file)
- Unzip file to a desired destination
- Open unzipped file and go to “bin” direction.
- Double click on “jmeter.bat”

Test description

This test case checks if server can handle massive number of clients (100) without difficulty.

Test steps

- Right click on” test plan” then choose Thread (Users)→Thread Group.
- Right click on “Thread Group” then Add→Sampler→HTTP Request.
- Right click on “Thread Group” then Add→Listener→View Results in Table.
- On Thread Group menu in Number of Threads (Users) field enter “100”.
- On HTTP Request menu on Server Address or IP field enter “localhost” and in Port number enter “9000”.
- Save the file.
- Run application.

Expected result

Server should continue running without any problem and status in results table in Jmeter should be green. **Actual result**

In Command Prompt

```

ClientThread started nr: 92
ClientThread 92 served file : index.html
ClientThread stopped nr: 92
Accept
ClientThread started nr: 93
ClientThread 93 served file : index.html
ClientThread stopped nr: 93
Accept
ClientThread started nr: 94
ClientThread 94 served file : index.html
ClientThread stopped nr: 94
Accept
ClientThread started nr: 95
ClientThread 95 served file : index.html
ClientThread stopped nr: 95
Accept
ClientThread started nr: 96
ClientThread 96 served file : index.html
ClientThread stopped nr: 96
Accept
ClientThread started nr: 97
ClientThread 97 served file : index.html
ClientThread stopped nr: 97
Accept
ClientThread started nr: 98
ClientThread 98 served file : index.html
ClientThread stopped nr: 98
Accept
ClientThread started nr: 99
Accept
ClientThread started nr: 100
ClientThread 99 served file : index.html
ClientThread stopped nr: 99
ClientThread 100 served file : index.html
ClientThread stopped nr: 100

```

In Jmeter:

Write results to file / Read from file										
Filename				Browse...		Log/Display Only:		<input type="radio"/> Errors	<input type="radio"/> Successes	Configure
Sample #	Start Time	Thread Name	Label	Sample Time(...)	Status	Bytes	Sent Bytes	Latency	Connect Time...	
81	22:44:16.338	Thread Group...	HTTP Request	47		174	116	47	2	
82	22:44:16.328	Thread Group...	HTTP Request	67		174	116	67	1	
83	22:44:16.359	Thread Group...	HTTP Request	48		174	116	48	2	
84	22:44:16.349	Thread Group...	HTTP Request	66		174	116	66	1	
85	22:44:16.379	Thread Group...	HTTP Request	51		174	116	51	1	
86	22:44:16.368	Thread Group...	HTTP Request	65		174	116	65	2	
87	22:44:16.399	Thread Group...	HTTP Request	52		174	116	52	2	
88	22:44:16.389	Thread Group...	HTTP Request	68		174	116	68	1	
89	22:44:16.419	Thread Group...	HTTP Request	53		174	116	53	1	
90	22:44:16.409	Thread Group...	HTTP Request	73		174	116	73	1	
91	22:44:16.428	Thread Group...	HTTP Request	62		174	116	62	1	
92	22:44:16.449	Thread Group...	HTTP Request	50		174	116	50	2	
93	22:44:16.439	Thread Group...	HTTP Request	70		174	116	70	1	
94	22:44:16.459	Thread Group...	HTTP Request	57		174	116	57	1	
95	22:44:16.480	Thread Group...	HTTP Request	47		174	116	47	1	
96	22:44:16.469	Thread Group...	HTTP Request	65		174	116	65	1	
97	22:44:16.499	Thread Group...	HTTP Request	47		174	116	47	1	
98	22:44:16.489	Thread Group...	HTTP Request	63		174	116	63	1	
99	22:44:16.519	Thread Group...	HTTP Request	45		174	116	45	1	
100	22:44:16.509	Thread Group...	HTTP Request	62		174	116	62	1	

After selecting Errors radio button to show only errors the result would be

Write results to file / Read from file

Filename

Browse...

Log/Display Only: ☒ Errors ☐ Successes

Configure

Sample #	Start Time	Thread Name	Label	Sample Time(...	Status	Bytes	Sent Bytes	Latency	Connect Time...

TC1.8.1 The access log should be viewable from a text editor.

Requirement under test: 1.8.

Pre-condition

General pre-condition step 2.

Test description

After searching in provided source code, a file named “access log” could not be found.

Expected result

A file named “access log” exist in source code which can be edited using text editor.

Actual result

There is no file named “access log” in source code.