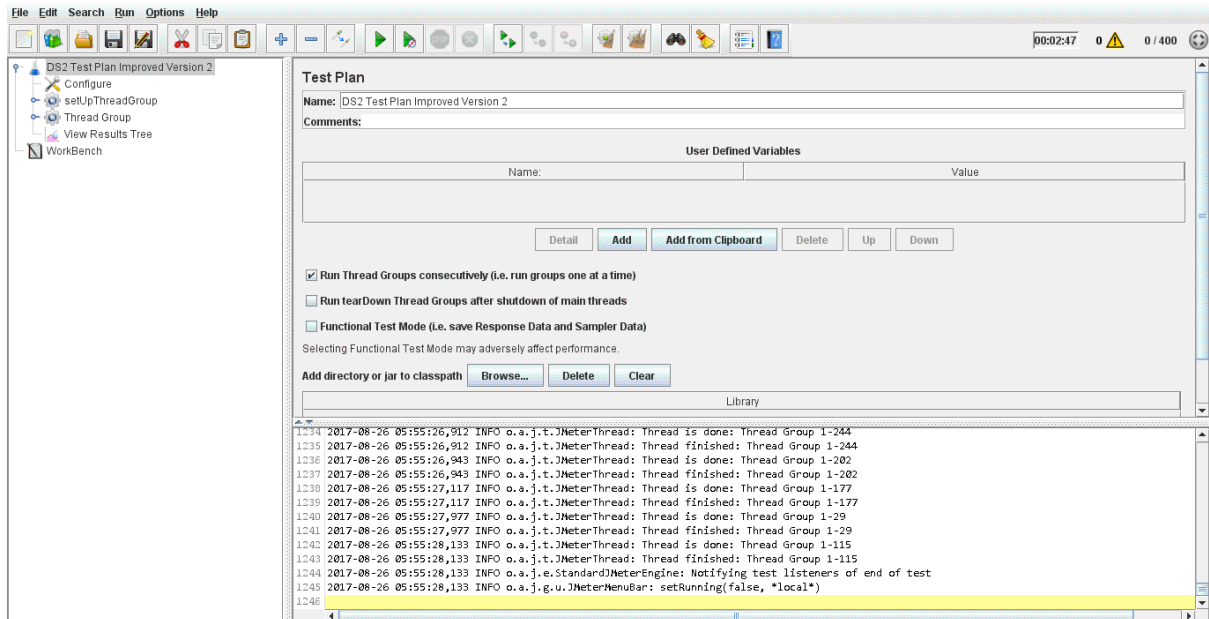


DS2 JMeter Test Plan Design

Contents

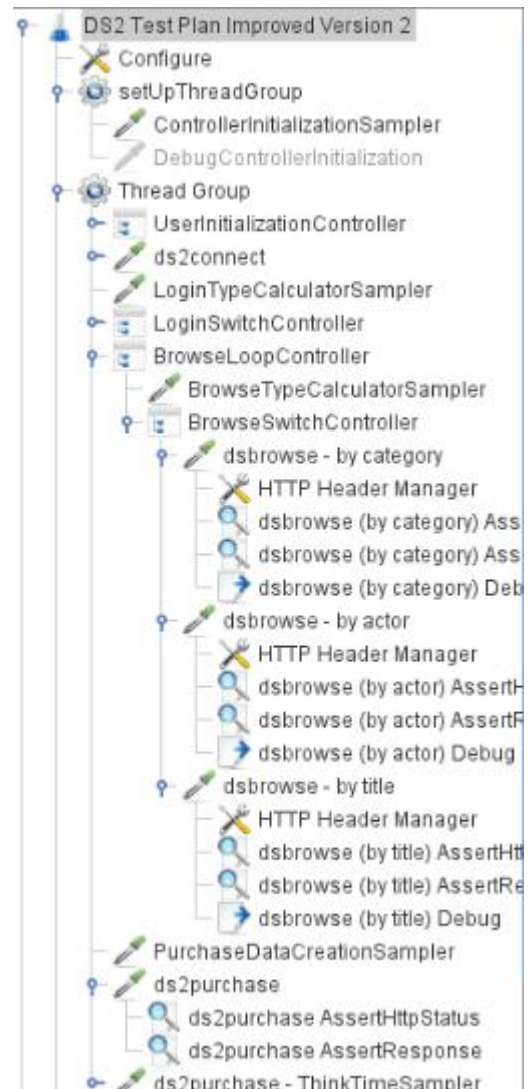
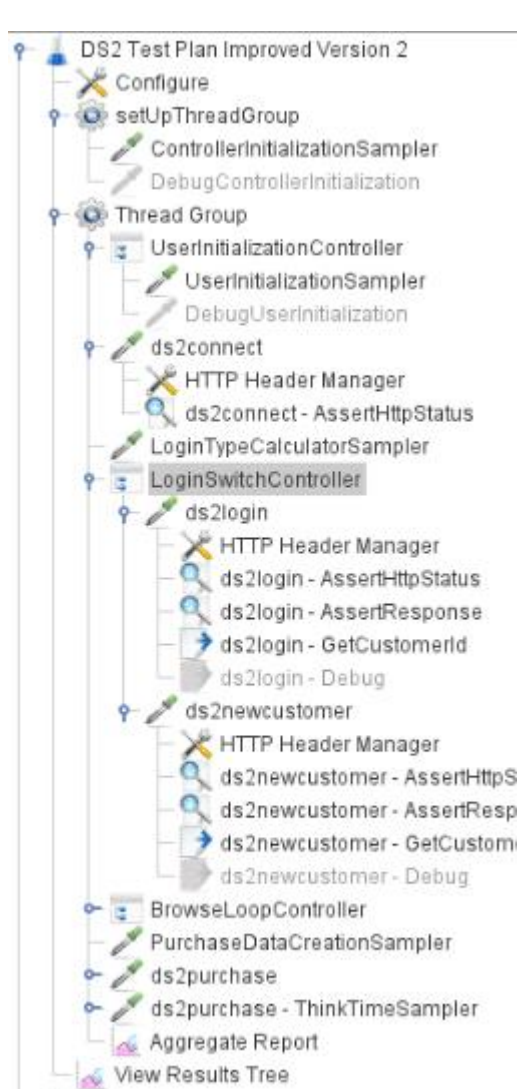
1	Test Plan.....	3
1.1	Configure.....	4
1.2	setUpThreadGroup	5
1.3	ThreadGroup.....	7
1.3.1	Step 1: UserInitializationController.....	7
1.3.2	Step 2: ds2connect.....	8
1.3.3	Step 3: LoginTypeCalculatorSampler	8
1.3.4	Step 4: LoginSwitchController.....	9
1.3.4.1	Step 4.A: ds2login.....	9
1.3.4.2	Step 4.B: ds2newcustomer	11
1.3.5	Step 5: BrowseLoopController	13
1.3.5.1	Step 5A: BrowseTypeCalculatorSampler.....	14
1.3.5.2	Step 5B: BrowseSwitchController	14
1.3.5.2.1	Step 5B1: dsbrowse - by category.....	15
1.3.5.2.2	Step 5B2: dsbrowse - by actor	16
1.3.5.2.3	Step 5B3: dsbrowse - by title	18
1.3.6	Step 6: PurchaseDataCreationSampler	20
1.3.7	Step 7: ds2purchase	20
1.3.8	Step 8: ds2purchase – ThinkTimeSampler	22
1.3.9	Aggregate Report	22
1.4	View Results Tree.....	23
2	Java Project	23
3	Test Runs	24
4	Additional Points.....	24

1 Test Plan



The top level is the Test Plan. The name of the Test Plan is DS2 Test Plan Improved Version 2. The Test Plan has following parts.

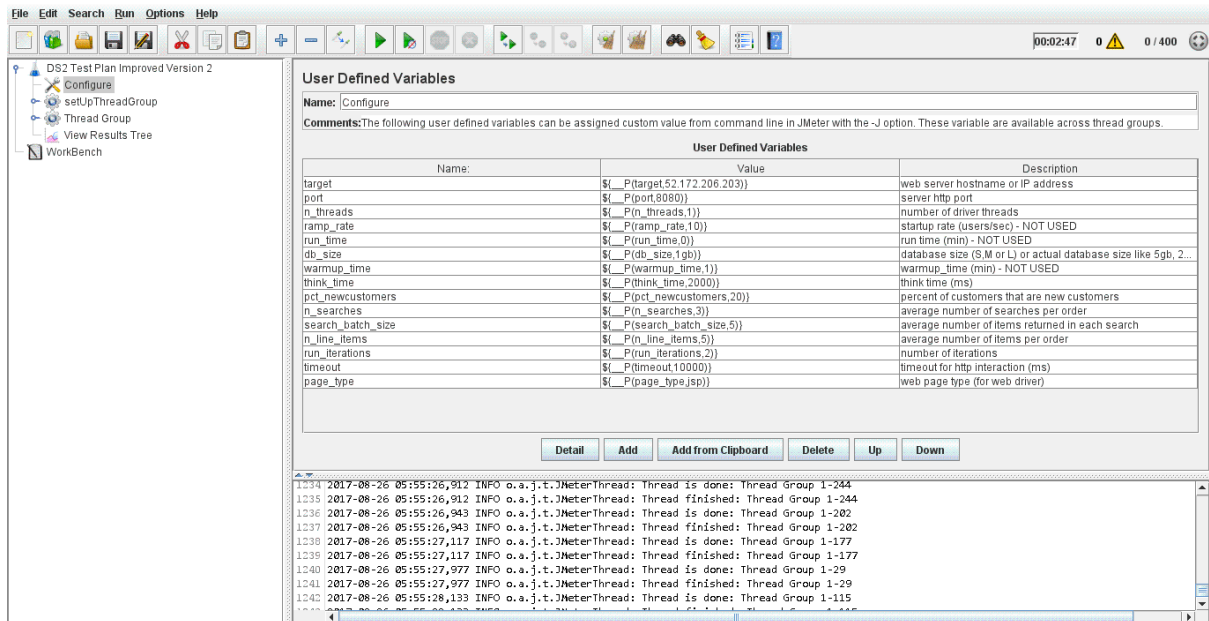
- Configure
- setUpThreadGroup
- ThreadGroup
- View Results Tree



To ensure that setUpThreadGroup runs before ThreadGroup, the Run Thread Groups consecutively property of the Test Plan is enabled.

1.1 Configure

Configure has JMeter user defined variables. The user defined variables are provided with default values. The values can also be set from the command line using the -J option.

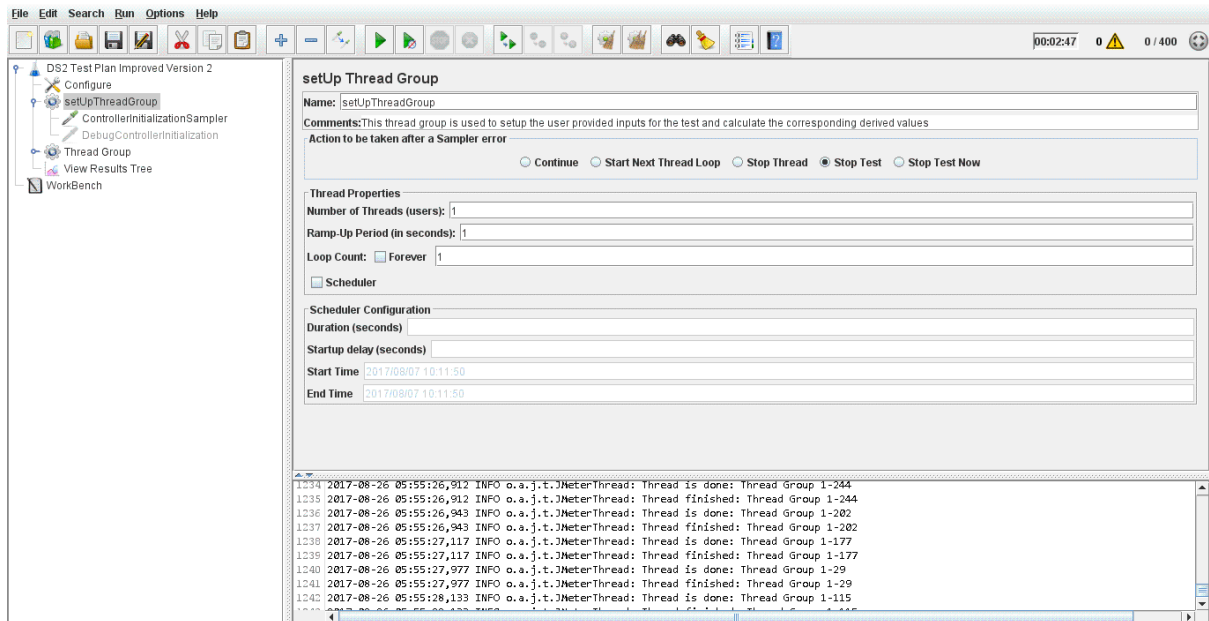


Following table provides the name, default value and description.

Variable	Default Value	Description
target	-	
port	8080	
n_threads	1	
ramp_rate	10	
run_time	0	
db_size	1gb	
warmup_time	1	
think_time	2000	
pct_newcustomers	20	
n_searches	3	
search_batch_size	5	
n_line_items	5	
run_iterations	2	
timeout	10000	
page_type	jsp	

1.2 setUpThreadGroup

setUpThreadGroup initializes the objects required in the run using the input value specified at the start of run. It also if any of the required parameters are missing. The thread executes only once i.e. the Number of Threads (Users) and Loop Count is 1.



The logic is coded in *ControllerInitializationSampler* which is a custom Java sampler class.

Java Request

Name:

Comments:

Classname: ▼

Send Parameters With the Request:

Name:	Value

The following values are checked.

- target
- port
- n_threads
- search_batch_size
- n_searches
- n_line_items
- pct_newcustomers
- db_size

The following values are internally derived using `db_size`.

- i_db_custom_size
- str_is_mb_gb
- ratio
- mult_cust_rows
- mult_ord_rows

- mult_prod_rows

The following values are calculated and stored for future use.

- customer_rows
- order_rows
- product_rows
- max_customer
- customer_rows

A singleton class *GlobalParametersSingleton* is initialized with following values.

- prod_array_size
- max_product
- max_customer
- pct_newcustomers
- n_searches
- search_batch_size
- n_line_items

This class is designed to be thread safe. However, the class is called only by a single thread.

1.3 ThreadGroup

ThreadGroup contains the entire test sequence.

1.3.1 Step 1: UserInitializationController

The first step is a Once Only Controller named *UserInitializationController*. The controller has one custom Java sampler *UserInitializationSampler*. This sample executes only once per every thread i.e. first iteration of the loop. The sampler creates a User object (one object is created per thread) and initializes it with a running sequence number. The sequence number is injected using an out of box JMeter function counter. FALSE indicates, that the counter is common across threads.

Java Request

Name: UserInitializationSampler

Comments:

Classname: ds2driver.jmeter.samplers.UserInitializationSampler

Send Parameters With the Request:

Name:	Value
userid	\${__counter(FALSE)}

Detail

Add

Add from Clipboard

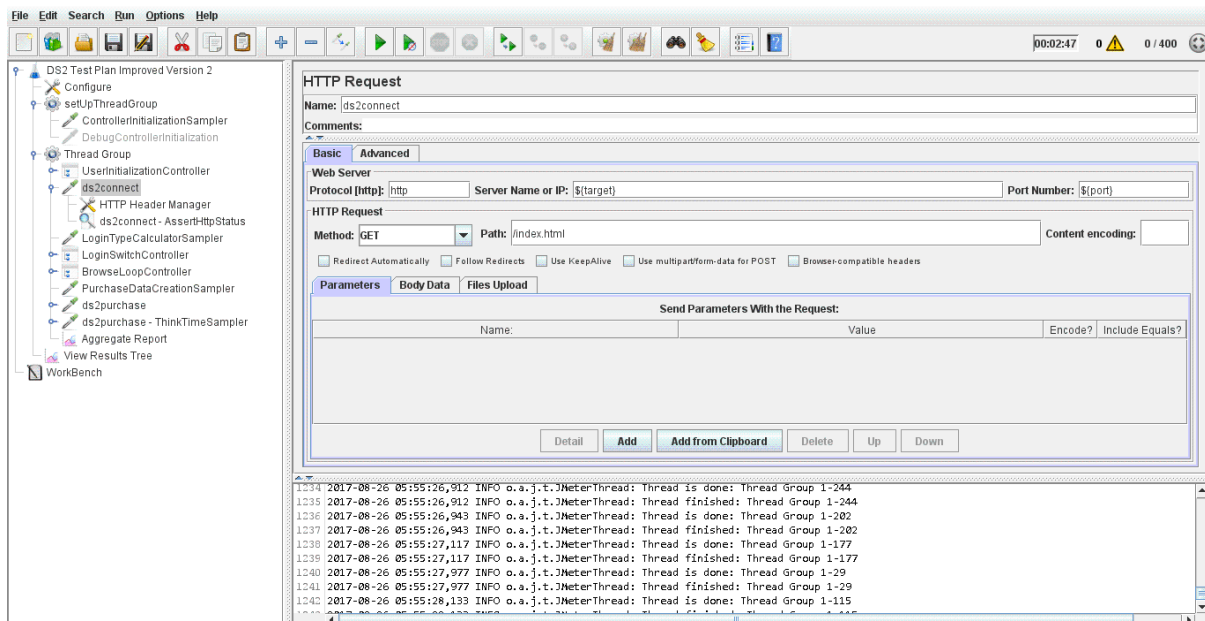
Delete

Up

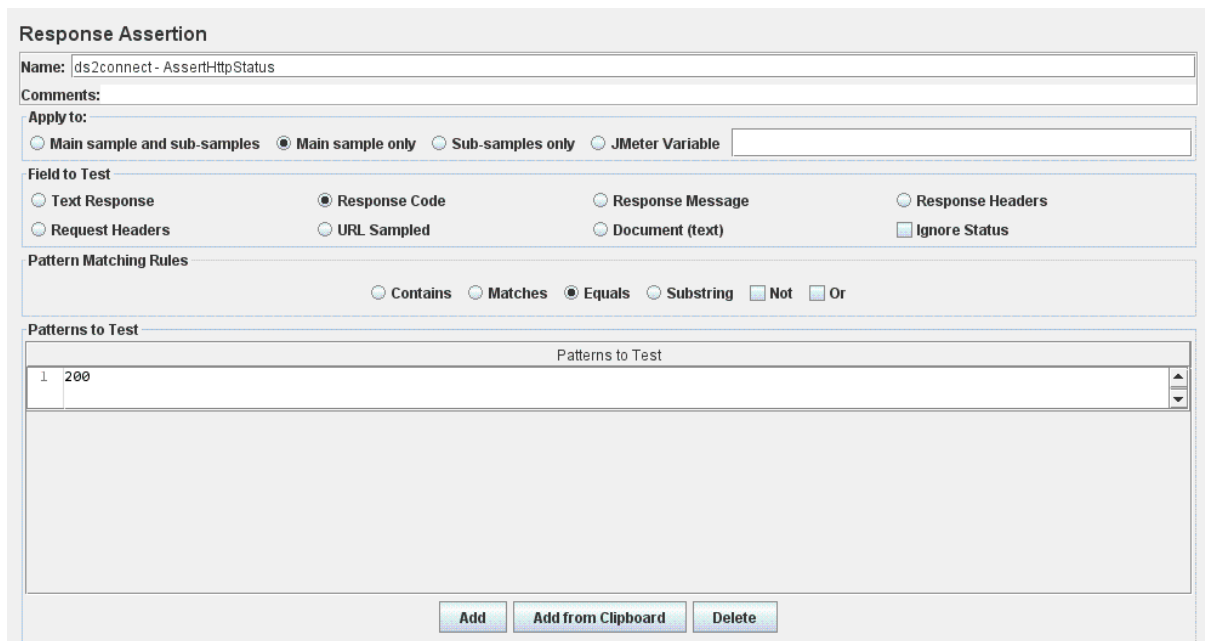
Down

1.3.2 Step 2: ds2connect

The second step is an out of box JMeter HTTP request sampler named *ds2connect*. The sampler executes `/index.html`. The default request headers sent are Accept-Language, Accept-Encoding, Accept and User-Agent.



The http response code is checked to see if it is 200.



1.3.3 Step 3: LoginTypeCalculatorSampler

The third step is a custom Java request sampler named *LoginTypeCalculatorSampler*. The sampler calculates the request type using randomization logic and the `pct_newcustomers` value. It can be an existing DS2 customer login or registration of a new DS2 customer. The data for both request types is created using internal functions `CreateLoginData` and `CreateLoginData` respectively. The number of iterations to be used in Step 5 is also calculated and stored in variable `n_browse`. The data is stored in the form of JMeter variables. JMeter variables are thread specific.

Java Request

Name: LoginTypeCalculatorSampler

Comments:

Classname: ds2driver.jmeter.samplers.LoginTypeCalculatorSampler ▼

Send Parameters With the Request:

Name:	Value

Detail Add Add from Clipboard Delete Up Down

1.3.4 Step 4: LoginSwitchController

In the fourth step, the *LoginSwitchController* either invokes *ds2login* or *ds2newcustomer* based on the value of request type calculated in the third step. The variable containing the request type is *islogin*.

Switch Controller

Name: LoginSwitchController

Comments:

Switch Value: \${islogin}

1.3.4.1 Step 4.A: ds2login

The step is an out of box JMeter HTTP request sampler named *ds2login*. The sampler executes */dslogin.jsp*.

The query parameters are sent the JMeter out of box option (Send Parameters With the Request). The default request headers sent are Accept-Language, Accept-Encoding, Accept and User-Agent. The query parameters are as follows.

- username
- password

HTTP Request

Name: ds2login

Comments:

Basic | **Advanced**

Web Server
 Protocol [http]: http Server Name or IP: \${target} Port Number: \${port}

HTTP Request
 Method: GET Path: /dslogin.\${page_type} Content encoding:

☐ Redirect Automatically ☐ Follow Redirects ☐ Use KeepAlive ☐ Use multipart/form-data for POST ☐ Browser-compatible headers

Parameters | **Body Data** | **Files Upload**

Send Parameters With the Request:

Name:	Value	Encode?	Include Equals?
username	\${username_in}	<input type="checkbox"/>	<input checked="" type="checkbox"/>
password	\${password_in}	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Detail Add Add from Clipboard Delete Up Down

First the http response code is checked to see if it is 200.

File Edit Search Run Options Help

00:02:47 0 0 / 400

DS2 Test Plan Improved Version 2

- Configure
- setUpThreadGroup
- ControllerInitializationSampler
- DebugControllerInitialization
- Thread Group
 - UserInitializationController
 - ds2connect
 - LoginTypeCalculatorSampler
 - LoginSwitchController
 - ds2login
 - HTTP Header Manager
 - ds2login - AssertHttpStatus
 - ds2login - AssertResponse
 - ds2login - GetCustomerId
 - ds2login - Debug
 - ds2newcustomer
 - HTTP Header Manager
 - ds2newcustomer - AssertHttpStatus
 - ds2newcustomer - AssertResponse
 - ds2newcustomer - GetCustomerId
 - ds2newcustomer - Debug
- BrowseLoopController
- PurchaseDataCreationSampler
- ds2purchase
- ds2purchase - ThinkTimeSampler
- Aggregate Report
- View Results Tree
- WorkBench

Response Assertion

Name: ds2login - AssertHttpStatus

Comments:

Apply to:

☐ Main sample and sub-samples ☒ Main sample only ☐ Sub-samples only ☐ JMeter Variable

Field to Test

☐ Text Response ☒ Response Code ☐ Response Message ☐ Response Headers

☐ Request Headers ☐ URL Sampled ☐ Document (text) ☐ Ignore Status

Pattern Matching Rules

☐ Contains ☐ Matches ☒ Equals ☐ Substring ☐ Not ☐ Or

Patterns to Test

1 200

Add Add from Clipboard Delete

1234 2017-08-26 05:55:26,912 INFO o.a.j.t.JMeterThread: Thread is done: Thread Group 1-244
 1235 2017-08-26 05:55:26,912 INFO o.a.j.t.JMeterThread: Thread finished: Thread Group 1-244
 1236 2017-08-26 05:55:26,943 INFO o.a.j.t.JMeterThread: Thread is done: Thread Group 1-202
 1237 2017-08-26 05:55:26,943 INFO o.a.j.t.JMeterThread: Thread finished: Thread Group 1-202
 1238 2017-08-26 05:55:27,117 INFO o.a.j.t.JMeterThread: Thread is done: Thread Group 1-177
 1239 2017-08-26 05:55:27,117 INFO o.a.j.t.JMeterThread: Thread finished: Thread Group 1-177
 1240 2017-08-26 05:55:27,977 INFO o.a.j.t.JMeterThread: Thread is done: Thread Group 1-29
 1241 2017-08-26 05:55:27,977 INFO o.a.j.t.JMeterThread: Thread finished: Thread Group 1-29
 1242 2017-08-26 05:55:28,133 INFO o.a.j.t.JMeterThread: Thread is done: Thread Group 1-115
 1243 2017-08-26 05:55:28,133 INFO o.a.j.t.JMeterThread: Thread finished: Thread Group 1-115

Next response text is checked to see if the following regular expression matches.

NAME=customerid VALUE= ([0-9]+) >

Response Assertion

Name: ds2login - AssertResponse

Comments:

Apply to:

☐ Main sample and sub-samples ☒ Main sample only ☐ Sub-samples only ☐ JMeter Variable

Field to Test

☒ Text Response ☐ Response Code ☐ Response Message ☐ Response Headers

☐ Request Headers ☐ URL Sampled ☐ Document (text) ☐ Ignore Status

Pattern Matching Rules

☒ Contains ☐ Matches ☐ Equals ☐ Substring ☐ Not ☐ Or

Patterns to Test

Patterns to Test

1 NAME=customerid VALUE= ([0-9]+) >

Add Add from Clipboard Delete

Last the value of customer id is extracted using a regular expression extractor and stored in *customerid_out*.

Regular Expression Extractor

Name: ds2login - GetCustomerId

Comments:

Apply to:

☐ Main sample and sub-samples ☒ Main sample only ☐ Sub-samples only ☐ JMeter Variable

Field to check

☒ Body ☐ Body (unescaped) ☐ Body as a Document ☐ Response Headers ☐ Request Headers ☐ URL ☐ Response Code ☐ Response Message

Reference Name: customerid_out

Regular Expression: <INPUT TYPE=HIDDEN NAME=customerid VALUE= ([0-9]+) >

Template: \$0\$

Match No. (0 for Random):

Default Value: 0 ☐ Use empty default value

1.3.4.2 Step 4.B: ds2newcustomer

The step is an out of box JMeter HTTP request sampler named *ds2newcustomer*. The sampler executes /dsnewcustomer.jsp.

The query parameters are sent the JMeter out of box option (Send Parameters With the Request). The default request headers sent are Accept-Language, Accept-Encoding, Accept and User-Agent. The query parameters are as follows.

- firstname
- lastname
- address1
- address2
- city
- state
- zip
- country
- email
- phone

- creditcardtype
- creditcard
- ccexpmon
- ccexprry
- username
- password
- age
- income
- gender

Encoding ensures unsafe characters like spaces in text values get passed.

HTTP Request

Name: ds2newcustomer

Comments:

Basic | **Advanced**

Web Server

Protocol [http]: http Server Name or IP: \${target} Port Number: \${port}

HTTP Request

Method: GET Path: /dsnewcustomer.\${page_type} Content encoding:

☐ Redirect Automatically ☐ Follow Redirects ☐ Use KeepAlive ☐ Use multipart/form-data for POST ☐ Browser-compatible headers

Parameters | **Body Data** | **Files Upload**

Send Parameters With the Request:

Name:	Value	Encode?	Include Equals?
firstname	\${firstname_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
lastname	\${lastname_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
address1	\${address1_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
address2	\${address2_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
city	\${city_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
state	\${state_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Detail Add Add from Clipboard Delete Up Down

First the http response code is checked to see if it is 200.

Response Assertion

Name: ds2newcustomer - AssertHttpStatus

Comments:

Apply to:

☐ Main sample and sub-samples ☒ Main sample only ☐ Sub-samples only ☐ JMeter Variable

Field to Test

☐ Text Response ☒ Response Code ☐ Response Message ☐ Response Headers

☐ Request Headers ☐ URL Sampled ☐ Document (text) ☐ Ignore Status

Pattern Matching Rules

☐ Contains ☐ Matches ☒ Equals ☐ Substring ☐ Not ☐ Or

Patterns to Test

Patterns to Test
1 200

Add Add from Clipboard Delete

Next response text is checked to see if the following regular expression matches.

Note: this is slightly different from the expression for ds2login.

NAME=customerid VALUE=([0-9]+)>

Response Assertion

Name: ds2newcustomer - AssertResponse

Comments:

Apply to:

☐ Main sample and sub-samples ☒ Main sample only ☐ Sub-samples only ☐ JMeter Variable

Field to Test

☒ Text Response ☐ Response Code ☐ Response Message ☐ Response Headers

☐ Request Headers ☐ URL Sampled ☐ Document (text) ☐ Ignore Status

Pattern Matching Rules

☒ Contains ☐ Matches ☐ Equals ☐ Substring ☐ Not ☐ Or

Patterns to Test

Patterns to Test

1 NAME=customerid VALUE=([0-9]+)>

Add Add from Clipboard Delete

Last the value of customer id is extracted using a regular expression extractor and stored in *customerid_out*. The JMeter assigned real variable name is *customerid_out_g1*.

File Edit Search Run Options Help

DS2 Test Plan Improved Version 2

Configure

setUpThreadGroup

ControllerInitializationSampler

DebugControllerInitialization

Thread Group

UserInitializationController

ds2connect

LoginTypeCalculatorSampler

LoginSwitchController

ds2login

HTTP Header Manager

ds2login - AssertHttpStatus

ds2login - AssertResponse

ds2login - GetCustomerId

ds2login - Debug

ds2newcustomer

HTTP Header Manager

ds2newcustomer - AssertHttpStatus

ds2newcustomer - AssertResponse

ds2newcustomer - GetCustomerId

ds2newcustomer - Debug

BrowseLoopController

PurchaseDataCreationSampler

ds2purchase

ds2purchase - ThinkTimeSampler

Aggregate Report

View Results Tree

WorkBench

Regular Expression Extractor

Name: ds2newcustomer - GetCustomerId

Comments:

Apply to:

☐ Main sample and sub-samples ☒ Main sample only ☐ Sub-samples only ☐ JMeter Variable

Field to check

☒ Body ☐ Body (unescaped) ☐ Body as a Document ☐ Response Headers ☐ Request Headers ☐ URL ☐ Response Code ☐ Response Message

Reference Name: customerid_out

Regular Expression: <INPUT TYPE=HIDDEN NAME=customerid VALUE=([0-9]+)>

Template: \$0\$

Match No. (0 for Random):

Default Value: 0 ☐ Use empty default value

1234 2017-08-26 05:55:26,912 INFO o.a.j.t.JMeterThread: Thread is done: Thread Group 1-244

1235 2017-08-26 05:55:26,912 INFO o.a.j.t.JMeterThread: Thread finished: Thread Group 1-244

1236 2017-08-26 05:55:26,943 INFO o.a.j.t.JMeterThread: Thread is done: Thread Group 1-202

1237 2017-08-26 05:55:26,943 INFO o.a.j.t.JMeterThread: Thread finished: Thread Group 1-202

1238 2017-08-26 05:55:27,117 INFO o.a.j.t.JMeterThread: Thread is done: Thread Group 1-177

1239 2017-08-26 05:55:27,117 INFO o.a.j.t.JMeterThread: Thread finished: Thread Group 1-177

1.3.5 Step 5: BrowseLoopController

In the fifth step, the *BrowseLoopController* iterates *n_browse* times (calculated in Step 4).

Loop Controller

Name:

Comments:

Loop Count: ☐ Forever

1.3.5.1 Step 5A: *BrowseTypeCalculatorSampler*

The *BrowseTypeCalculatorSampler* is a custom Java request sampler. The sampler calculates the browse type using randomization logic and the `pct_newcustomers` value. It can be browse by category, actor or title. The data for all browse types is created using internal functions `CreateActor` and `CreateTitle`. The data is stored in the form of JMeter variables. JMeter variables are thread specific.

Java Request

Name:

Comments:

Classname:

Send Parameters With the Request:

Name:	Value

1.3.5.2 Step 5B: *BrowseSwitchController*

In this step, the *BrowseSwitchController* either invokes *dsbrowse – by category*, *dsbrowse – by actor* or *dsbrowse – by title* based on the value of browse type calculated in the previous step. The variable containing the request type is `search_type`.

Switch Controller

Name:

Comments:

Switch Value

1.3.5.2.1 Step 5B1: dsbrowse - by category

The step is an out of box JMeter HTTP request sampler named *dsbrowse – by category*. The sampler executes /dsbrowse.jsp.

The query parameters are sent the JMeter out of box option (Send Parameters With the Request). The default request headers sent are Accept-Language, Accept-Encoding, Accept and User-Agent. The query parameters are as follows.

- browse_type
- browse_category
- browse_actor
- browse_title
- limit_null
- customer_id

The screenshot shows the JMeter HTTP Request sampler configuration window. The 'Name' field is 'dsbrowse - by category'. The 'Comments' field is empty. The 'Basic' tab is selected, showing the 'Web Server' section with 'Protocol (http): http', 'Server Name or IP: \${target}', and 'Port Number: \${port}'. The 'HTTP Request' section shows 'Method: GET' and 'Path: /dsbrowse.\${page_type}'. Below this are checkboxes for 'Redirect Automatically', 'Follow Redirects', 'Use KeepAlive', 'Use multipart/form-data for POST', and 'Browser-compatible headers'. The 'Parameters' tab is selected, showing a table for 'Send Parameters With the Request:'. The table has columns for 'Name', 'Value', 'Encode?', and 'Include Equals?'. The parameters are: browse_type (value: \${browse_type_in}), browse_category (value: \${browse_category_in}), browse_actor (value: \${browse_actor_in}), browse_title (value: \${browse_title_in}), limit_num (value: \${batch_size_in}), and customerid (value: \${customerid_out_g1}). The 'Encode?' and 'Include Equals?' checkboxes are checked for all parameters. At the bottom are buttons for 'Detail', 'Add', 'Add from Clipboard', 'Delete', 'Up', and 'Down'.

Name:	Value	Encode?	Include Equals?
browse_type	\${browse_type_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
browse_category	\${browse_category_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
browse_actor	\${browse_actor_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
browse_title	\${browse_title_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
limit_num	\${batch_size_in}	<input type="checkbox"/>	<input checked="" type="checkbox"/>
customerid	\${customerid_out_g1}	<input type="checkbox"/>	<input checked="" type="checkbox"/>

First the http response code is checked to see if it is 200.

Response Assertion

Name: dsbrowse (by category) AssertHttpStatus

Comments:

Apply to:

☐ Main sample and sub-samples ☒ Main sample only ☐ Sub-samples only ☐ JMeter Variable

Field to Test

☐ Text Response ☒ Response Code ☐ Response Message ☐ Response Headers

☐ Request Headers ☐ URL Sampled ☐ Document (text) ☐ Ignore Status

Pattern Matching Rules

☐ Contains ☐ Matches ☒ Equals ☐ Substring ☐ Not ☐ Or

Patterns to Test

Patterns to Test	
1	200

Add Add from Clipboard Delete

Next response text is checked to see if the following regular expression matches.

(<H2>Search Results</H2>|<H2>No DVDs Found</H2>)

Response Assertion

Name: ds2newcustomer - AssertResponse

Comments:

Apply to:

☐ Main sample and sub-samples ☒ Main sample only ☐ Sub-samples only ☐ JMeter Variable

Field to Test

☒ Text Response ☐ Response Code ☐ Response Message ☐ Response Headers

☐ Request Headers ☐ URL Sampled ☐ Document (text) ☐ Ignore Status

Pattern Matching Rules

☒ Contains ☐ Matches ☐ Equals ☐ Substring ☐ Not ☐ Or

Patterns to Test

Patterns to Test	
1	NAME=customerId VALUE=([0-9]+)>

Add Add from Clipboard Delete

1.3.5.2.2 Step 5B2: dsbrowse - by actor

The step is an out of box JMeter HTTP request sampler named *dsbrowse – by actor*. The sampler executes /dsbrowse.jsp.

The query parameters are sent the JMeter out of box option (Send Parameters With the Request). The default request headers sent are Accept-Language, Accept-Encoding, Accept and User-Agent. The query parameters are as follows.

- browse_type
- browse_category
- browse_actor
- browse_title

- limit_null
- customer_id

HTTP Request

Name: dsbrowse - by actor

Comments:

Basic Advanced

Web Server

Protocol [http]: http Server Name or IP: \${target} Port Number: \${port}

HTTP Request

Method: GET Path: /dsbrowse.\${page_type} Content encoding:

☐ Redirect Automatically ☐ Follow Redirects ☐ Use KeepAlive ☐ Use multipart/form-data for POST ☐ Browser-compatible headers

Parameters Body Data Files Upload

Send Parameters With the Request:

Name:	Value	Encode?	Include Equals?
browse_type	\${browse_type_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
browse_category	\${browse_category_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
browse_actor	\${browse_actor_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
browse_title	\${browse_title_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
limit_num	\${batch_size_in}	<input type="checkbox"/>	<input checked="" type="checkbox"/>
customerid	\${customerid_out_g1}	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Detail Add Add from Clipboard Delete Up Down

First the http response code is checked to see if it is 200.

Response Assertion

Name: dsbrowse (by actor) AssertHttpStatus

Comments:

Apply to:

☐ Main sample and sub-samples ☒ Main sample only ☐ Sub-samples only ☐ JMeter Variable

Field to Test

☐ Text Response ☒ Response Code ☐ Response Message ☐ Response Headers

☐ Request Headers ☐ URL Sampled ☐ Document (text) ☐ Ignore Status

Pattern Matching Rules

☐ Contains ☐ Matches ☒ Equals ☐ Substring ☐ Not ☐ Or

Patterns to Test

Patterns to Test

1	200
---	-----

Add Add from Clipboard Delete

Next response text is checked to see if the following regular expression matches.

(<H2>Search Results</H2>|<H2>No DVDs Found</H2>)

Response Assertion

Name: dsbrowse (by actor) AssertResponse

Comments:

Apply to:

☐ Main sample and sub-samples
 ☒ Main sample only
 ☐ Sub-samples only
 ☐ JMeter Variable

Field to Test

☒ Text Response
 ☐ Response Code
 ☐ Response Message
 ☐ Response Headers

☐ Request Headers
 ☐ URL Sampled
 ☐ Document (text)
 ☐ Ignore Status

Pattern Matching Rules

☒ Contains
 ☐ Matches
 ☐ Equals
 ☐ Substring
 ☐ Not
 ☐ Or

Patterns to Test

	Patterns to Test
1	{<H2>Search Results</H2> <H2>No DVDs Found</H2>}

Add Add from Clipboard Delete

1.3.5.2.3 Step 5B3: dsbrowse - by title

The step is an out of box JMeter HTTP request sampler named *dsbrowse – by title*. The sampler executes /dsbrowse.jsp.

The query parameters are sent the JMeter out of box option (Send Parameters With the Request). The default request headers sent are Accept-Language, Accept-Encoding, Accept and User-Agent. The query parameters are as follows.

- browse_type
- browse_category
- browse_actor
- browse_title
- limit_null
- customer_id

HTTP Request

Name: dsbrowse - by title

Comments:

Basic | **Advanced**

Web Server

Protocol [http]: http Server Name or IP: \${target} Port Number: \${port}

HTTP Request

Method: GET Path: /dsbrowse.\${page_type} Content encoding:

☐ Redirect Automatically ☐ Follow Redirects ☐ Use KeepAlive ☐ Use multipart/form-data for POST ☐ Browser-compatible headers

Parameters | **Body Data** | **Files Upload**

Send Parameters With the Request:

Name:	Value	Encode?	Include Equals?
browse_type	\${browse_type_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
browse_category	\${browse_category_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
browse_actor	\${browse_actor_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
browse_title	\${browse_title_in}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
limit_num	\${batch_size_in}	<input type="checkbox"/>	<input checked="" type="checkbox"/>
customerid	\${customerid_out_g1}	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Detail Add Add from Clipboard Delete Up Down

First the http response code is checked to see if it is 200.

Response Assertion

Name: dsbrowse (by title) AssertHttpStatus

Comments:

Apply to:

☐ Main sample and sub-samples ☒ Main sample only ☐ Sub-samples only ☐ JMeter Variable

Field to Test

☐ Text Response ☒ Response Code ☐ Response Message ☐ Response Headers

☐ Request Headers ☐ URL Sampled ☐ Document (text) ☐ Ignore Status

Pattern Matching Rules

☐ Contains ☐ Matches ☒ Equals ☐ Substring ☐ Not ☐ Or

Patterns to Test

Patterns to Test
1 200

Add Add from Clipboard Delete

Next response text is checked to see if the following regular expression matches.

(<H2>Search Results</H2>|<H2>No DVDs Found</H2>)

Response Assertion

Name: dsbrowse (by title) AssertResponse

Comments:

Apply to:

☐ Main sample and sub-samples ☒ Main sample only ☐ Sub-samples only ☐ JMeter Variable

Field to Test

☒ Text Response ☐ Response Code ☐ Response Message ☐ Response Headers

☐ Request Headers ☐ URL Sampled ☐ Document (text) ☐ Ignore Status

Pattern Matching Rules

☒ Contains ☐ Matches ☐ Equals ☐ Substring ☐ Not ☐ Or

Patterns to Test

	Patterns to Test
1	{<H2>Search Results</H2> <H2>No DVDs Found</H2>}

Add Add from Clipboard Delete

1.3.6 Step 6: PurchaseDataCreationSampler

The sixth step is a custom Java request sampler *PurchaseDataCreationSampler*. The sampler calculates the items and respective quantities to be purchased using randomization logic.

- Randomize number of cart items with average n_line_items
- For each cart item randomly select product_id using weighted prod_array

The data is stored in the form of JMeter variable *purchaseitemsqty*. JMeter variables are thread specific.

Java Request

Name: PurchaseDataCreationSampler

Comments:

Classname: ds2driver.jmeter.samplers.PurchaseDataCreationSampler

Send Parameters With the Request:

Name:	Value
-------	-------

Detail Add Add from Clipboard Delete Up Down

1.3.7 Step 7: ds2purchase

The seventh step is an out of box JMeter HTTP request sampler named *ds2purchase*. The sampler executes */dspurchase.jsp*.

The query parameters are sent the JMeter out of box option (Send Parameters With the Request). The default request headers sent are Accept-Language, Accept-Encoding, Accept and User-Agent. The query parameters are as follows.

- confirmpurchase
- customerid
- purchaseitemsqty

HTTP Request

Name: ds2purchase

Comments:

Basic | **Advanced**

Web Server

Protocol [http]: http Server Name or IP: \${target} Port Number: \${port}

HTTP Request

Method: GET Path: chase.\${page_type}?confirmpurchase=yes&customerid=\${customerid_out_g1}&\${purchaseitemsqty} Content encoding:

☐ Redirect Automatically ☐ Follow Redirects ☐ Use KeepAlive ☐ Use multipart/form-data for POST ☐ Browser-compatible headers

Parameters | **Body Data** | **Files Upload**

Send Parameters With the Request:

Name	Value	Encode?	Include Equals?

Detail Add Add from Clipboard Delete Up Down

First the http response code is checked to see if it is 200.

Response Assertion

Name: ds2purchase AssertHttpStatus

Comments:

Apply to:

☐ Main sample and sub-samples ☒ Main sample only ☐ Sub-samples only ☐ JMeter Variable

Field to Test

☐ Text Response ☒ Response Code ☐ Response Message ☐ Response Headers

☐ Request Headers ☐ URL Sampled ☐ Document (text) ☐ Ignore Status

Pattern Matching Rules

☐ Contains ☐ Matches ☒ Equals ☐ Substring ☐ Not ☐ Or

Patterns to Test

Patterns to Test
1 200

Add Add from Clipboard Delete

Next response text is checked to see if the following regular expression matches.

(<H2>Order Completed Successfully --- ORDER NUMBER: ([0-9]+)</H2>
<H3>Insufficient stock - order not processed</H3>)

Aggregate Report

Name:

Comments:

Write results to file / Read from file

Filename Log/Display Only: ☐ Errors ☐ Successes

Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Max	Error %	Throughput	Received K...	Sent KB/sec
TOTAL	0	0	0	0	0	0	922337203...	-92233720...	0.00%	0/hour	0.00	0.00

☐ Include group name in label? ☒ Save Table Header

1.4 View Results Tree

Displays granular details of errors. Only the error option has been enabled.

View Results Tree

Name:

Comments:

Write results to file / Read from file

Filename Log/Display Only: ☒ Errors ☐ Successes

Search: ☐ Case sensitive ☐ Regular exp.

Text

☐ Scroll automatically?

Raw

2 Java Project

The DS2JMeter project contains the custom Java samplers and beans.

- Beans
 - GlobalConstants
 - GlobalParametersSingleton
 - User

- Samplers
 - BrowseTypeCalculatorSampler
 - ControllerInitializationSampler
 - LoginTypeCalculatorSampler
 - PurchaseDataCreationSampler
 - UserInitializationSampler

Following external jars are required by the project. The jars are present either in the /lib or /lib/ext of the JMeter folder.

- ApacheJMeter_core
- ApacheJMeter_java
- slf4j-api-1.7.25
- slf4j-ext-1.7.25

DS2JMeter is compiled and exported as JMeterClassesForDS2.jar to the /lib/ext folder of JMeter.

3 Test Runs

Results of test run with 400 virtual users and 10 iterations are enclosed. Default input parameters were used. The run was executed on the Microsoft Azure cloud with three VMS.

- VM1: Basic A2 (2 vcpus, 3.5 GB memory), Windows OS, Apache JMeter 3.2
- VM2: Basic A1 (1 vcpu, 1.75 GB memory), CentOS 6.8, Tomcat 9.0.0.M22 with DS2 jsp web application, JDK 1.8, threads = 800
- VM3: Basic A1 (1 vcpu, 1.75 GB memory), CentOS 6.8, MySQL with DS2 database, connections = 800

```
innodb_buffer_pool_size = 1G
innodb_log_file_size = 256M
innodb_log_buffer_size = 8M #default value
innodb_flush_method = O_DIRECT
```

4 Additional Points

Following features are not available [can be considered in the next release].

- ramp_rate
- run_time
- warmup_time

Following logic is not available [can be considered in the next release]

- Initialization of random function with very randomized seed


```

        // Create random stream r with very randomized seed
        Random rtemp = new Random ( Userid * 1000 ); // Temporary
seed
        // For multi-thread runs sleep between 0 - 10 second to
spread out Ticks (100 nsecs)
        if ( Controller.n_threads > 1 ) Thread.Sleep ( rtemp.Next (
10000 ) );
        long DTNT = DateTime.Now.Ticks;
        uint lowDTNT = ( uint ) ( 0x00000000ffffffff & DTNT );
        uint rev_lowDTNT = 0; // take low 32 bits of Tick counter
and reverse them
        for ( i = 0 ; i < 32 ; i++ ) rev_lowDTNT = rev_lowDTNT | (
( 0x1 & ( lowDTNT >> i ) ) << ( 31 - i ) );
        //Console.Error.WriteLine("DTNT= 0x{0:x16} lowDTNT= 0x
{1:x8} rev_lowDTNT= 0x{2:x8}", DTNT, lowDTNT, rev_lowDTNT);
        r = new Random ( ( int ) rev_lowDTNT );

```

- Loop for new user creation if an “already exists” error is returned [can be considered in the next release]

```

|
        if ( customerid_out == 0 ) Console.Error.WriteLine (
"User name {0} already exists" , username_in );
        } while ( customerid_out == 0 ); // end of do/while
try newcustomer

```

The possibility is reduced by further randomizing the user name. The new user names are not used in the login logic.

```

        //added some more randomization with alphabets to avoid the logic in
//existing workload generator which keeps generating new customer
//until find a userid that doesn't exist
        String username_in = "newuser" + ( char ) ( 97 + user.nextInt ( 26 ) ) + ( char )
( 97 + user.nextInt ( 26 ) ) + i_user;

```

- Detailed parsing of responses was not included, as it did not appear to be used in the driver logic. Existing driver logic example is given below. JMeter assertions have been added to act as verification points.

```

        i_row = 0;
        str_acc = str_acc.Substring(str_acc.IndexOf("<TABLE"));
// Snip off everything up to <TABLE> tag
        str_acc = str_acc.Substring(4 + str_acc.IndexOf
("<TR>")); // Skip first <TR> tag
        while (str_acc.IndexOf("<TR>") > 0)
        {
            str_acc = str_acc.Substring(str_acc.IndexOf("<TR>"));
// Find <TR> tag
            str_acc = str_acc.Substring(6 + str_acc.IndexOf
("VALUE"));
            ind_e = str_acc.IndexOf(">");
            prod_id_out[i_row] = Convert.ToInt32(str_acc.Substring
(0, ind_e).Trim(''));
            str_acc = str_acc.Substring(4 + str_acc.IndexOf
("<TD>")); // Find <TD> tag
            ind_e = str_acc.IndexOf("<");
            title_out[i_row] = str_acc.Substring(0, ind_e);
            str_acc = str_acc.Substring(4 + str_acc.IndexOf
("<TD>")); // Find <TD> tag
            ind_e = str_acc.IndexOf("<");
            actor_out[i_row] = str_acc.Substring(0, ind_e);
            str_acc = str_acc.Substring(4 + str_acc.IndexOf
("<TD>")); // Find <TD> tag
            ind_e = str_acc.IndexOf("<");
            price_out[i_row] = Convert.ToDecimal(str_acc.Substring
(0, ind_e));
            ++i_row;
            ++rows_returned;
        }
    }
}

```

- Changed the data point O'DONNELL to O DONNEL as it was resulting in a MySQL error because of the single quote.
- The existing logic has a constant think time only at the end of all the steps. However, the practitioner can experiment with the timer – think time after every step, to simulate more realistic patterns.
- Currently the following input values are stored in the *User* bean. These will be moved to the *GlobalParametersSingleton* bean as the values remain same for all the threads [can be considered in the next release].
- Performance metrics calculated by the existing driver is not included [can be considered in the next release]