EECS 4313 Assignment 3
Data Flow Testing, Slice-Based Testing and Mutation Testing

Student Name — Student Number — EECS Account
Edward Vaisman — 212849857 — eddyv
Robin Bandzar — 212200531 — cse23028
Kirusanth Thiruchelvam — 212918298 — kirusant
Sadman Sakib Hasan — 212497509 — cse23152

April 7, 2018

Contents

List of Tables

List of Figures

1 BORG Calendar

1.1 Data Flow Testing

1.1.1 Chosen Method

- **Class**: *net.sf.borg.common.DateUtil.java*
- **Method**: *minuteString(int mins)*
- **Method Description**: This method generate a human reable string for a particular number of minutes. It returns the string in terms of hours or minutes or both hours and mintues.
  - **mins** - The first argument is of type integer.

Following is the code snippet of the *minuteString* method:

```
public static String minuteString(int mins) {

  int hours = mins / 60;
  int minsPast = mins % 60;

  String minutesString;
  String hoursString;

  if (hours > 1) {
    hoursString = hours + " " +
       Resource.getResourceString("Hours");
  } else if (hours > 0) {
    hoursString = hours + " " +
       Resource.getResourceString("Hour");
  } else {
    hoursString = "";
  }

  if (minsPast > 1) {
    minutesString = minsPast + " " +
       Resource.getResourceString("Minutes");
  } else if (minsPast > 0) {
    minutesString = minsPast + " " +
       Resource.getResourceString("Minute");
  } else if (hours >= 1) {
    minutesString = "";
  } else {
    minutesString = minsPast + " " +
       Resource.getResourceString("Minutes");
  }

  // space between hours and minutes
  if (!hoursString.equals("") && !minutesString.equals(""))
    minutesString = " " + minutesString;

  return hoursString + minutesString;
}
```

1.1.2 Variable set

Following are the list of variables used in the *minuteString* method along with their types:

Variable Name	Data Type	Is Primitive?
mins	int	Yes
hours	int	Yes
minsPast	int	Yes
minutesString	String	No
hoursString	String	No

Table 1: List of variables for the minuteString method

Note: The last two variables, *minutesString* and *hoursString*, are **NOT** primitive types, and will **NOT** be used to perform the data flow analysis as specified by the instructor.

1.1.3 Program Segmentation

The following table 2 contains the *minuteString* method broken down into segments:

public static string minuteString(int mins) {	A

int hours = mins / 60;
int minsPast = mins % 60;
	B

String minutesString;
String hoursString;
	C

if (hours > 1) {	D
hoursString = hours + ” ” + Resource.getResourceString(”Hours”);	E
} else if (hours > 0) {	F
hoursString = hours + ” ” + Resource.getResourceString(”Hour”);	G
} else { hoursString = ””; }	H
if (minsPast > 1) {	I
minutesString = minsPast + ” ” + Resource.getResourceString(”Minutes”);	J
} else if (minsPast > 0) {	K
minutesString = minsPast + ” ” + Resource.getResourceString(”Minute”);	L
} else if (hours >= 1) {	M
minutesString = ””;	N

} else {
minutesString = minsPast + ” ” + Resource.getResourceString(”Minutes”);
}
	O

if (!hoursString.equals(””) && !minutesString.equals(””))	P
minutesString = ” ” + minutesString;	Q
return hoursString + minutesString;	R

Table 2: Program Segmentation for the minuteString method

1.1.4 Program graph

The following diagram 1 represents the control flow graph for the *minuteString* method:

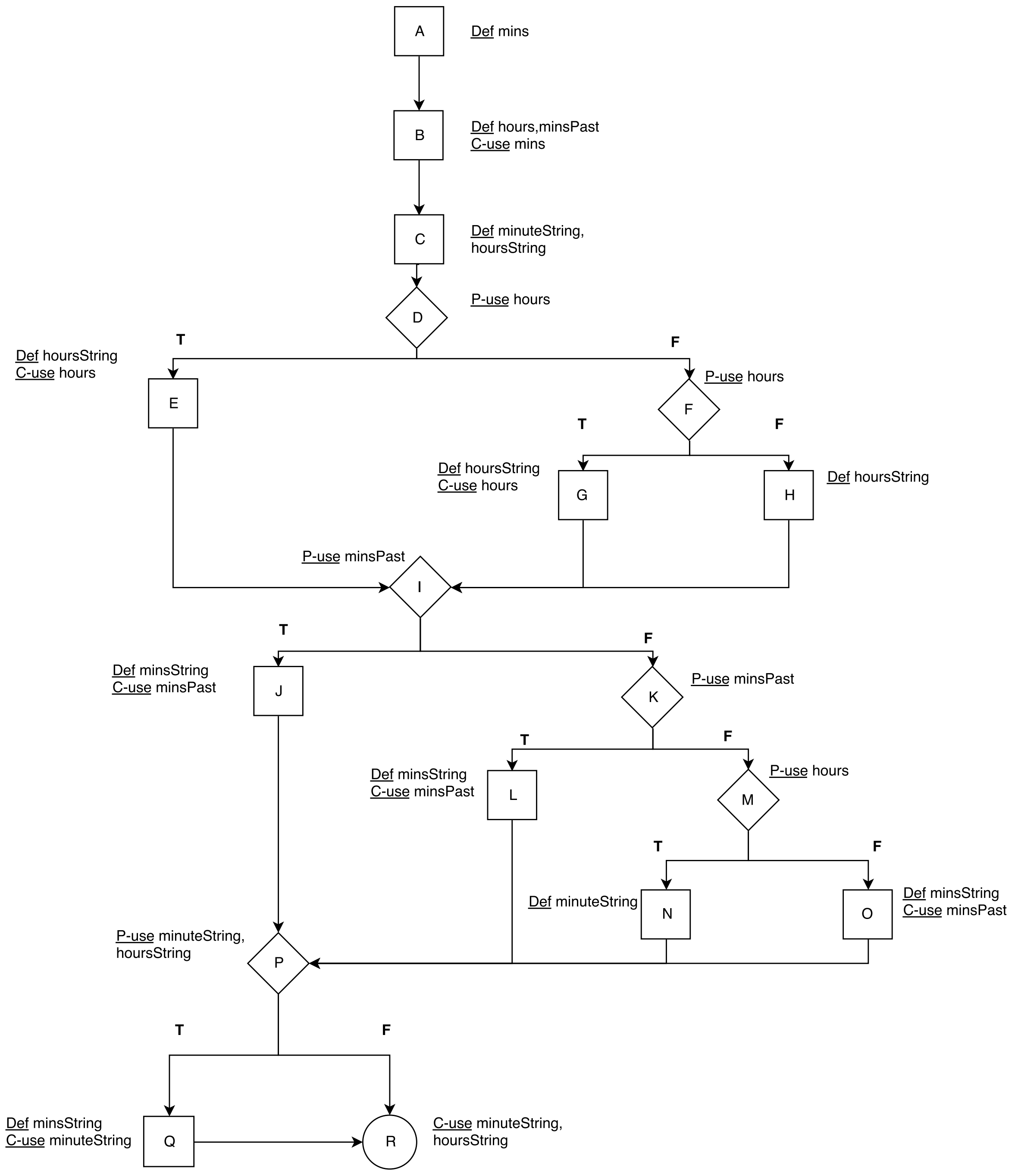


Figure 1: Control Flow Graph for the minuteString method

1.1.5 Data Flow Analysis

1. **du-paths for *mins***
   - **All-Defs**: *AB*
   - **All-C-Uses/Some-P-Uses**: *AB*
   - **All-P-Uses/Some-C-Uses**: *AB*
   - **All-Uses**: *AB*
2. **du-paths for *hours***
   - **All-Defs**: *BCD*
   - **All-C-Uses/Some-P-Uses**: *BCDE, BCDFG*
   - **All-P-Uses/Some-C-Uses**: *BCD, BCDF, BCDEIKM, BCDFGIKM, BCDFHIKM*
   - **All-Uses**: *BCD, BCDE, BCDF, BCDEIKM, BCDFG, BCDFGIKM, BCDFHIKM*
3. **du-paths for *minsPast***
   - **All-Defs**: *BCDEI*
   - **All-C-Uses/Some-P-Uses**: *BCDEIJ, BCDFGIJ, BCDFHIJ, BCDEIKL, BCDFGIKL, BCDFHIKL, BCDEIKMO, BCDFGIKMO, BCDFHIKMO*
   - **All-P-Uses/Some-C-Uses**: *BCDEI, BCDFGI, BCDFHI, BCDEIK, BCDFGIK, BCDFHIK*
   - **All-Uses**: *BCDEI, BCDEIJ, BCDFGIJ, BCDFHIJ, BCDEIKL, BCDFGIKL, BCDFHIKL, BCDEIKMO, BCDFGIKMO, BCDFHIKMO, BCDFGI, BCDFHI, BCDEIK, BCDFGIK, BCDFHIK*

1.1.6 Coverage Metrics

Following are the existing test cases from Assignment-2 and their coverage metrics using the data flow analysis for the *minuteString* method:

1. **Test Case 1**:
   - **Input**: mins = 60
   - **Path**: *ABCDEGIKMNPR*
   - **Coverage**: 100*(12/18) = 66.67%
2. **Test Case 2**:
   - **Input**: mins = 61

- **Path**: *ABCDFGIKLPQR*
- **Coverage**: 100*(14/18) = 77.78%

3. **Test Case 3**:
    - **Input**: mins = 75
    - **Path**: *ABCDFGIJPQR*
    - **Coverage**: 100*(15/18) = 83.33%
4. **Test Case 4**:
    - **Input**: mins = 180
    - **Path**: *ABCDEIKMNPR*
    - **Coverage**: 100*(16/18) = 88.88%
5. **Test Case 5**:
    - **Input**: mins = 121
    - **Path**: *ABCDEIKLPQR*
    - **Coverage**: 100*(16/18) = 88.88%
6. **Test Case 6**:
    - **Input**: mins = 145
    - **Path**: *ABCDEIJPQR*
    - **Coverage**: 100*(16/18) = 88.88%
7. **Test Case 7**:
    - **Input**: mins = 0
    - **Path**: *ABCDFHIOPR*
    - **Coverage**: 100*(18/18) = 100.0%

Rationale: The testing technique used for testing this method is *Equivalence Class Testing.* It is a suitable technique for this method since the argument is an integer which is an independent variable and the input range can be partitioned assuring disjointness and non-redundancy between each partition set. We have chosen the partition integer range based on usages of minute, minutes, hour, and hours. In order to partition the integer argument into hours and minutes, we divide the minutes by 60 to get the range of hours and the remainder (minutes % 60) to get the range of the minutes. By covering all the cases, where the given input returns a concatenation of hours and minutes string, we were able to reach a 100% coverage for this method.

1.2 Slice Testing

1.2.1 Chosen Method for Testing

- **Class**: *net.sf.borg.common.DateUtil.java*
- **Method**: *minuteString(int mins)*
- **Method Description**: This method generates a human readable string for a particular number of minutes. It returns the string in terms of hours or minutes or both hours and minutes.
  - **mins** - The first argument is of type integer.

Following is the code of the *minuteString* method:

```java
public static String minuteString(int mins) {

  int hours = mins / 60;
  int minsPast = mins % 60;

  String minutesString;
  String hoursString;

  if (hours > 1) {
   hoursString = hours + " " +
       Resource.getResourceString("Hours");
  } else if (hours > 0) {
   hoursString = hours + " " + Resource.getResourceString("Hour");
  } else {
   hoursString = "";
  }

  if (minsPast > 1) {
   minutesString = minsPast + " " +
       Resource.getResourceString("Minutes");
  } else if (minsPast > 0) {
   minutesString = minsPast + " " +
       Resource.getResourceString("Minute");
  } else if (hours >= 1) {
   minutesString = "";
  } else {
   minutesString = minsPast + " " +
       Resource.getResourceString("Minutes");
124     }
125
126     // space between hours and minutes
127     if (!hoursString.equals("") && !minutesString.equals(""))
128       minutesString = " " + minutesString;
129
130     return hoursString + minutesString;
131   }
```

1.2.2 Backward Slicing

Backward slicing is in the form of S(v,n) where the slices are code fragments that contribute to variable v at statement n. Slices are only done for primitive values and their All-defs and P-use paths defined in the data flow analysis part.

S(hours, 102)

```
100   public static String minuteString(int mins) {
101
102     int hours = mins / 60;
```

S(minsPast, 103)

```
100   public static String minuteString(int mins) {
101
102
103     int minsPast = mins % 60;
```

The following test case covers the two slices listed above and covers the All-def, P-use for *mins*.

```
assertEquals("1 Hour",DateUtil.minuteString(60));
```

S(hours, 108)

```
100  public static String minuteString(int mins) {
101
102    int hours = mins / 60;
103
104
105
106
107
108    if (hours > 1) {
109    }
```

S(hours, 120)

```
100  public static String minuteString(int mins) {
101
102    int hours = mins / 60;
103    int minsPast = mins % 60;
104
105
106
107
108
109
110
111
112
113
114
115
116    if (minsPast > 1) {
117
118    } else if (minsPast > 0) {
119
120    } else if (hours >= 1) {
121    }
```

The following test case covers the previous two slices for *hours*.

```
assertEquals("3 Hours",DateUtil.minuteString(180));
```

S(minsPast, 116)

```
100  public static String minuteString(int mins) {
101
102
103    int minsPast = mins % 60;
104
105
106
107
108
109
110
111
112
113
114
115
116    if (minsPast > 1) {
117
118    }
```

The following test case covers the previous slice for *minsPast*.

```
assertEquals("1 Hour 15 Minutes",DateUtil.minuteString(75));
```

S(hours, 110)

```
100  public static String minuteString(int mins) {
101
102    int hours = mins / 60;
103
104
105
106
107
108    if (hours > 1) {
109
110    } else if (hours > 0) {
111
112    }
```

The following test case covers the previous slice for *hours*.

```
assertEquals("1 Hour 1 Minute",DateUtil.minuteString(61));
```

S(minsPast, 118)

```
public static String minuteString(int mins) {


  int minsPast = mins % 60;












  if (minsPast > 1) {

  } else if (minsPast > 0) {


  }
```

The following test case covers the previous slice for *minsPast*.

```
assertEquals("1 Hour 1 Minute",DateUtil.minuteString(61));
```

S(hours, 112)

```
public static String minuteString(int mins) {

  int hours = mins / 60;
```

```
    if (hours > 1) {

    } else if (hours > 0) {

    } else {

    }
```

The following test case covers the previous slice for *hours*.

```
assertEquals("0 Minutes",DateUtil.minuteString(0));
```

S(hours, 122)

```
  public static String minuteString(int mins) {

    int hours = mins / 60;
    int minsPast = mins % 60;













    if (minsPast > 1) {

    } else if (minsPast > 0) {

    } else if (hours >= 1) {

    } else {

    }
```

S(minsPast, 122)

```
public static String minuteString(int mins) {

  int hours = mins / 60;
  int minsPast = mins % 60;













  if (minsPast > 1) {

  } else if (minsPast > 0) {

  } else if (hours >= 1) {

  } else {

  }
```

The following test cases covers the previous two slices.

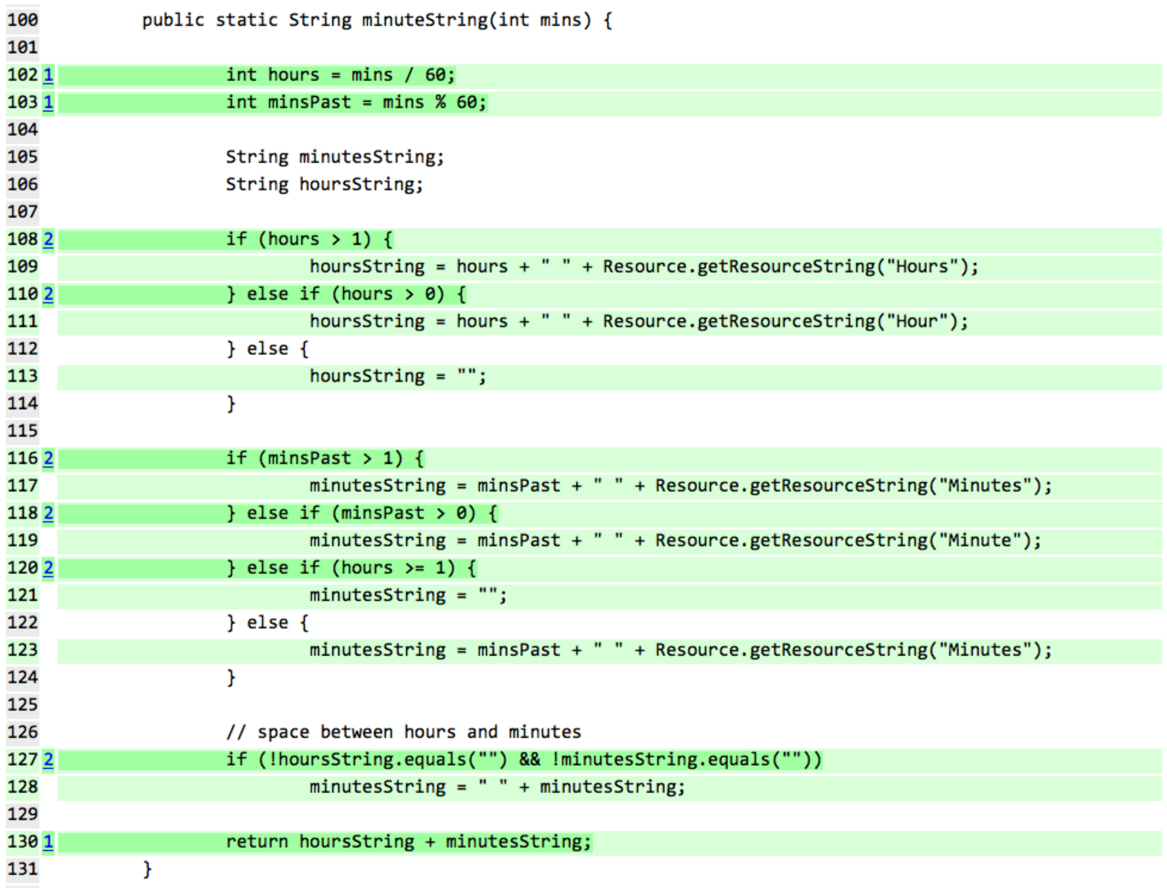
```
assertEquals("0 Minutes",DateUtil.minuteString(0));
```

This concludes all the backward slices related to the All-defs and P-uses of the primitive types in the *minuteString* function.

1.3 Mutation Testing

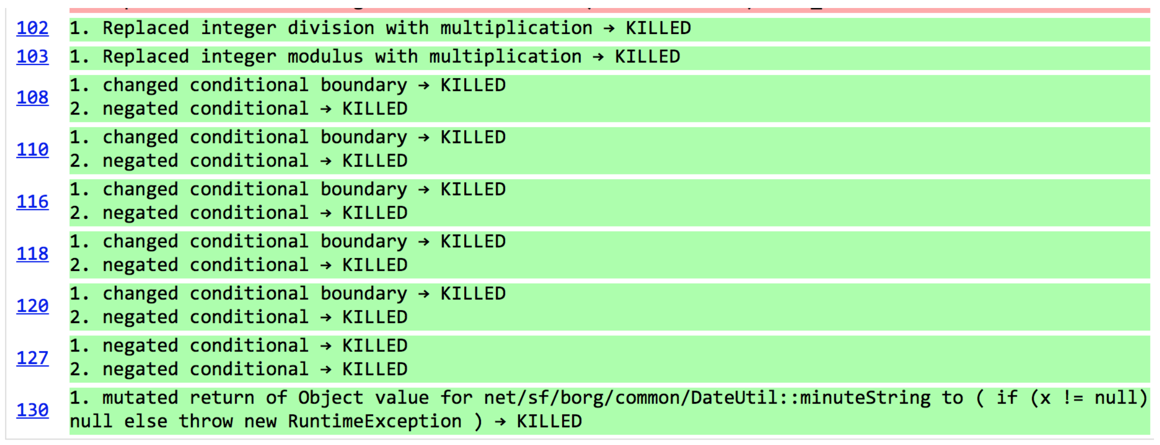
Mutation tests were run using the previous unit test suite that we created for assignment 2. The program used to run the Mutation tests was *Eclipse* with the *Pitclipse* plugin. The three methods that we tested are listed with their results in the following subsections.

1.3.1 minuteString Function



```
    public static String minuteString(int mins) {

1       int hours = mins / 60;
1       int minsPast = mins % 60;

        String minutesString;
        String hoursString;

2       if (hours > 1) {
            hoursString = hours + " " + Resource.getResourceString("Hours");
2       } else if (hours > 0) {
            hoursString = hours + " " + Resource.getResourceString("Hour");
        } else {
            hoursString = "";
        }

2       if (minsPast > 1) {
            minutesString = minsPast + " " + Resource.getResourceString("Minutes");
2       } else if (minsPast > 0) {
            minutesString = minsPast + " " + Resource.getResourceString("Minute");
2       } else if (hours >= 1) {
            minutesString = "";
        } else {
            minutesString = minsPast + " " + Resource.getResourceString("Minutes");
        }

        // space between hours and minutes
2       if (!hoursString.equals("") && !minutesString.equals(""))
            minutesString = " " + minutesString;

1       return hoursString + minutesString;
    }
```

Figure 2: Code for the minuteString method

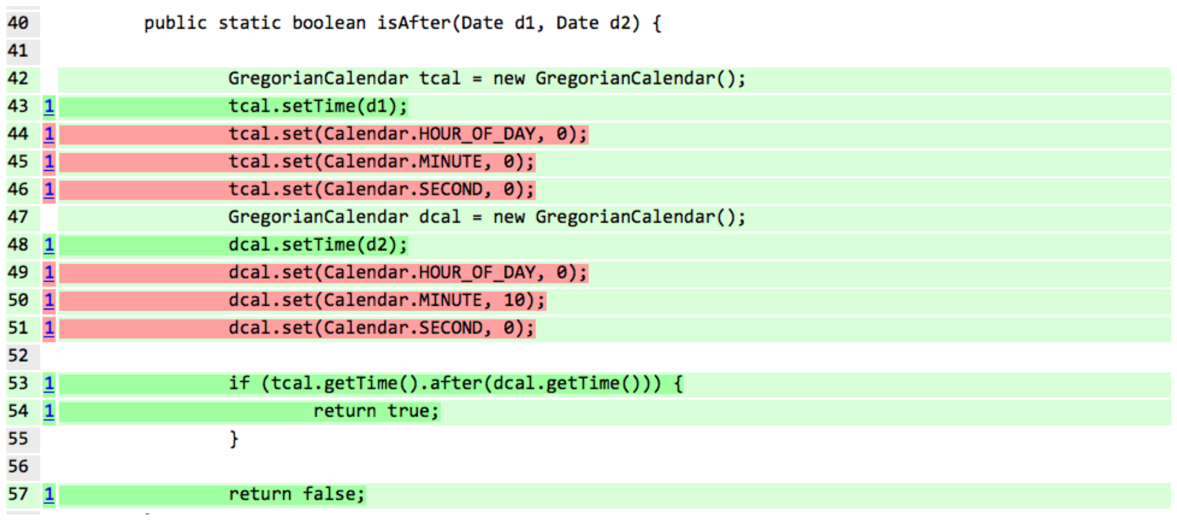


```
102  1. Replaced integer division with multiplication → KILLED
103  1. Replaced integer modulus with multiplication → KILLED
108  1. changed conditional boundary → KILLED
     2. negated conditional → KILLED
110  1. changed conditional boundary → KILLED
     2. negated conditional → KILLED
116  1. changed conditional boundary → KILLED
     2. negated conditional → KILLED
118  1. changed conditional boundary → KILLED
     2. negated conditional → KILLED
120  1. changed conditional boundary → KILLED
     2. negated conditional → KILLED
127  1. negated conditional → KILLED
     2. negated conditional → KILLED
130  1. mutated return of Object value for net/sf/borg/common/DateUtil::minuteString to ( if (x != null)
     null else throw new RuntimeException ) → KILLED
```

Figure 3: Mutations for the minuteString method

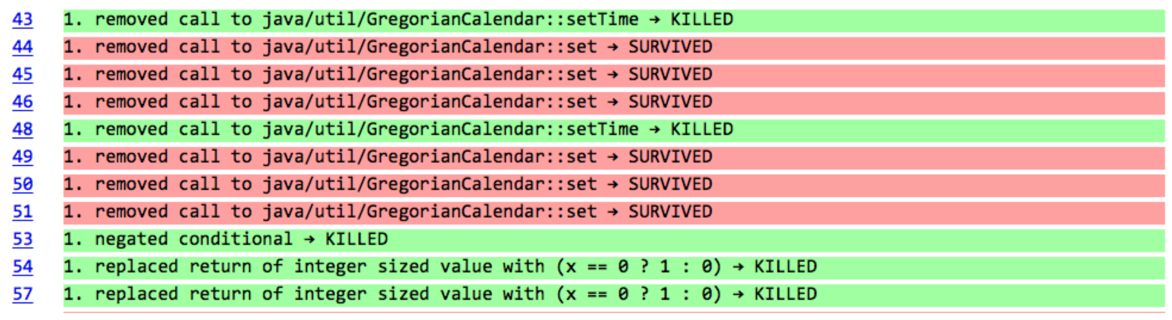
As one can see in Figures 2 and 3 that the previous tests effectively kill all the mutants so no further changes are needed.

1.3.2 isAfter Function



```
	public static boolean isAfter(Date d1, Date d2) {

		GregorianCalendar tcal = new GregorianCalendar();
		tcal.setTime(d1);
		tcal.set(Calendar.HOUR_OF_DAY, 0);
		tcal.set(Calendar.MINUTE, 0);
		tcal.set(Calendar.SECOND, 0);
		GregorianCalendar dcal = new GregorianCalendar();
		dcal.setTime(d2);
		dcal.set(Calendar.HOUR_OF_DAY, 0);
		dcal.set(Calendar.MINUTE, 10);
		dcal.set(Calendar.SECOND, 0);

		if (tcal.getTime().after(dcal.getTime())) {
			return true;
		}

		return false;
```

Figure 4: Code for the isAfter method



```
43  1. removed call to java/util/GregorianCalendar::setTime → KILLED
44  1. removed call to java/util/GregorianCalendar::set → SURVIVED
45  1. removed call to java/util/GregorianCalendar::set → SURVIVED
46  1. removed call to java/util/GregorianCalendar::set → SURVIVED
48  1. removed call to java/util/GregorianCalendar::setTime → KILLED
49  1. removed call to java/util/GregorianCalendar::set → SURVIVED
50  1. removed call to java/util/GregorianCalendar::set → SURVIVED
51  1. removed call to java/util/GregorianCalendar::set → SURVIVED
53  1. negated conditional → KILLED
54  1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
57  1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Figure 5: Mutations for the isAfter method

Looking at the result of the mutation test result from Figure 5, it is inevitable that the method *isAfter* will still pass tests even after removing the lines 44-46 and 49-51. This means those lines (line 44-46 and line 49-51) can never have any affect on the expected output of the method.

This can be proved by adding a simple test case which creates two date objects and set their minute, hour and second to some value. No matter what value for minute, hour and second is passed with the Date object, the function will always create a Gregorian Calendar instance with values of 0 for the first date object. Similarly, it will create another Gregorian Calendar instance with values 0 for hours and seconds and 10 for minutes for the second date object.

Example test case:

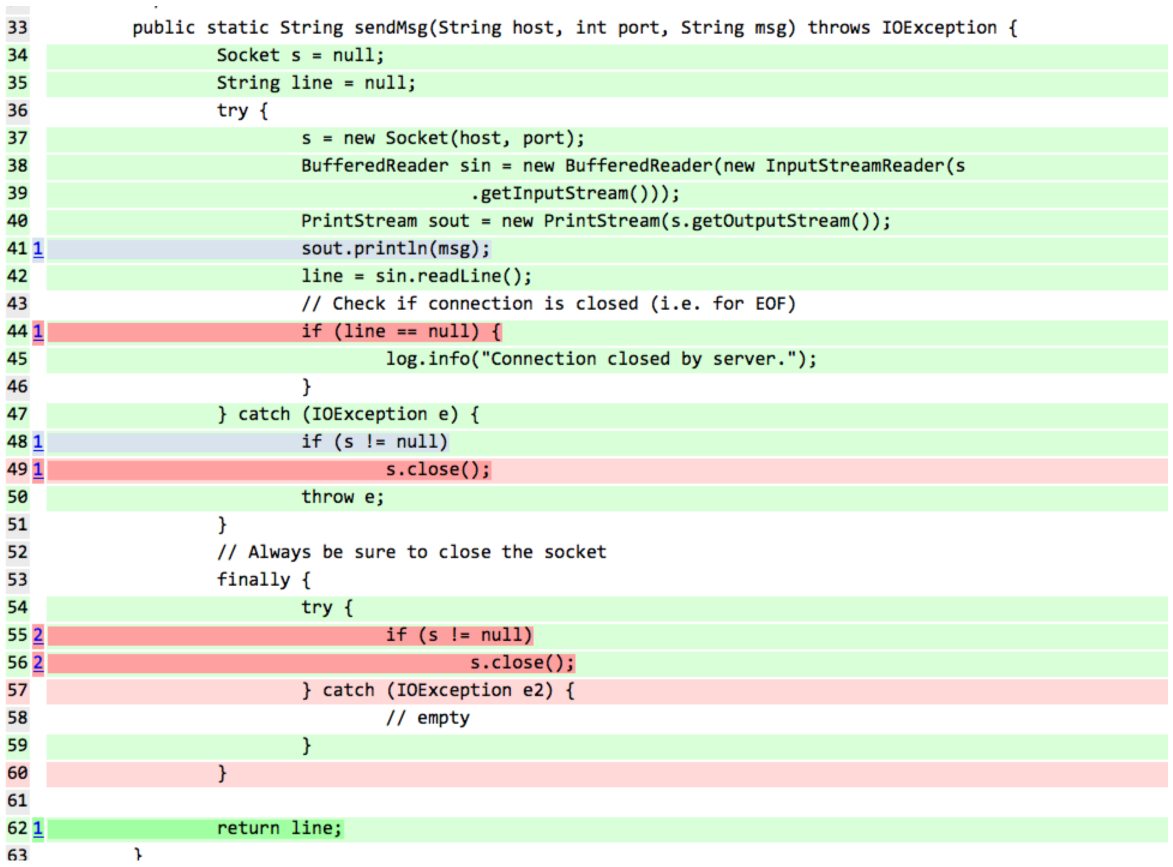
```
@Test
public void testIsAfterBug() {
  // Both happens at the same day, but at different hours
  // but the implementation ignores the hours, minutes
  // and seconds
  Date d1 = new Date( 2018 , 8 , 24 , 8 , 30 , 24 );
  Date d2 = new Date( 2018 , 8 , 24 , 9 , 30 , 24 );

  // This works correctly. The assert returns true
  assertFalse(DateUtil.isAfter(d1, d2));
  // This does not work correctly. The assert returns false,
  // when it should be true
  assertTrue(DateUtil.isAfter(d2, d1));
}
```

Bug Report:

- **Bug Report Title**: The *isAfter* function ignores hours, minutes and seconds fields from the passed in date object.
- **Reported by**: Sadman Sakib Hasan
- **Date reported**: Saturday, March 24, 2018
- **Program (or component) name**: BORG Calendar version 1.8.3
- **Configuration(s)**:
  System Info
  - Operating System: Windows 10 Pro 64-bit (10.0, Build 16299)
  - Language: English (Regional Setting: English)
  - System Manufacturer: Hewlett-Packard
  - System Model: HP 15 TouchSmart Notebook PC
  - BIOS: F.10
  - Processor: AMD A6-5200 APU with Radeon(TM) HD Graphics (4 CPUs), 2.0GHz
  - Memory: 6144MB RAM
  - Java Version: 1.8.0_151-b12
- **Report type**: Coding Error
- **Reproducibility**: 100%
- **Severity**: Medium
- **Problem summary**: The *isAfter* method automatically sets the HOUR_OF_DAY, MINUTE, and SECOND values of the newly created Gregorian Calendar object from the passed in dates d1 and d2 to a constant value.
- **Problem description**: The *isAfter* method in *net.sf.borg.common.DateUtil* does not take into account the hours, minutes, and seconds of the given date objects when comparing the two dates. A newly Gregorian Calendar object is created for both argument, d1 and d2, and the hours, minutes and seconds are set to 0 for *d1*'s calendar instance whereas the hours and seconds are set to 0 and minute set to 10 for *d2*'s calendar instance. This causes an issue when there are comparison between two dates that are in the same day, and hours, minutes and seconds differ as shown in the example test case 1.3.2.

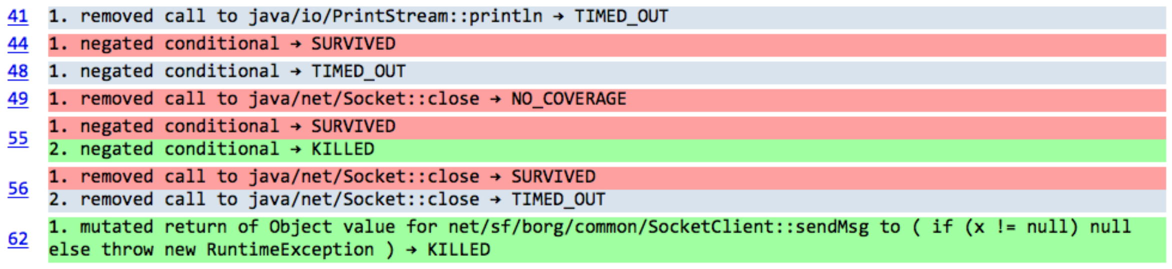
- **Steps to Reproduce**:
  - Create two date objects, d1 and d2, with the day set to the same value and hours, minutes and seconds set to different values.
  - Compare the two dates using *isAfter* function.
  - The *isAfter* function will always return false regardless of the fact if d1 is ahead on hours/minutes/seconds or vice versa.
- **New or old bug**: New

1.3.3 sendMsg Function



```
    public static String sendMsg(String host, int port, String msg) throws IOException {
        Socket s = null;
        String line = null;
        try {
            s = new Socket(host, port);
            BufferedReader sin = new BufferedReader(new InputStreamReader(s
                    .getInputStream()));
            PrintStream sout = new PrintStream(s.getOutputStream());
            sout.println(msg);
            line = sin.readLine();
            // Check if connection is closed (i.e. for EOF)
            if (line == null) {
                log.info("Connection closed by server.");
            }
        } catch (IOException e) {
            if (s != null)
                s.close();
            throw e;
        }
        // Always be sure to close the socket
        finally {
            try {
                if (s != null)
                    s.close();
            } catch (IOException e2) {
                // empty
            }
        }

        return line;
    }
```

Figure 6: Code for the sendMsg method

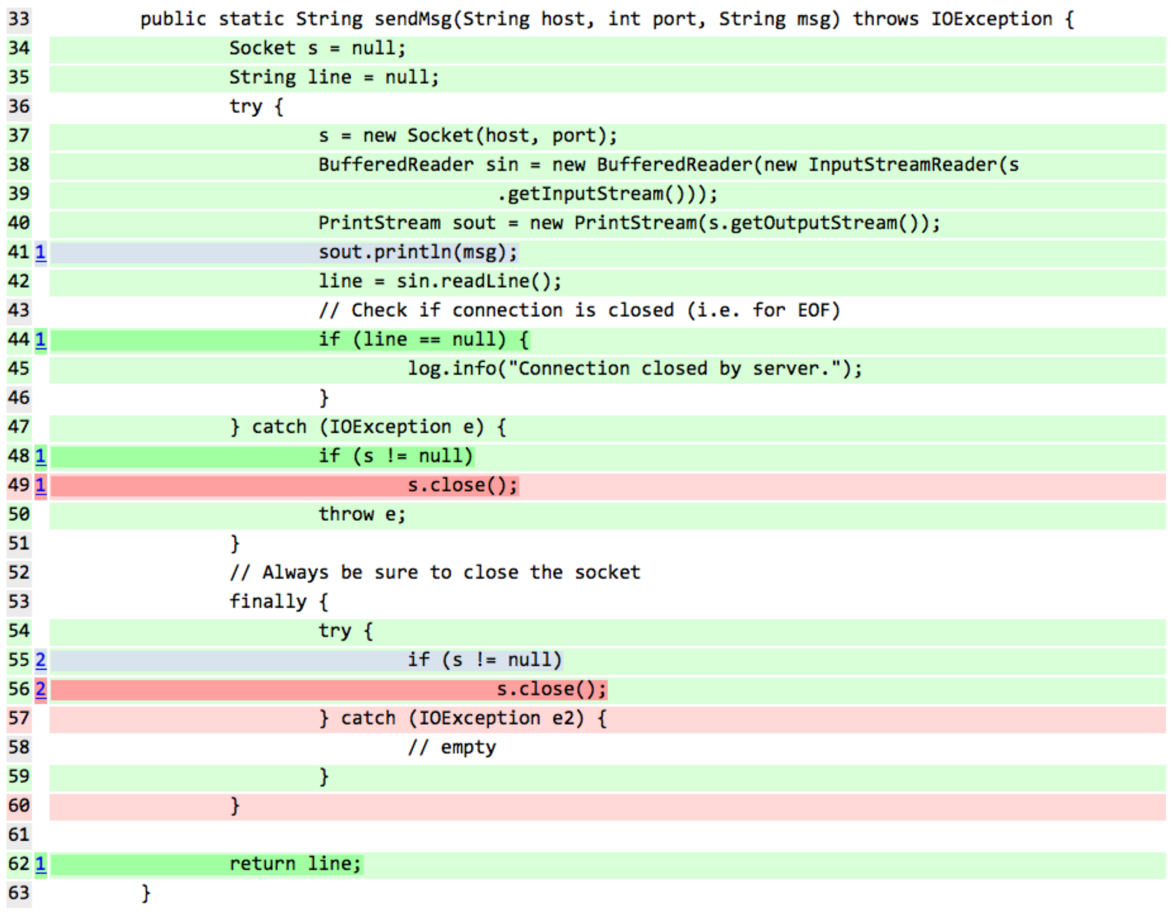


```
41  1. removed call to java/io/PrintStream::println → TIMED_OUT
44  1. negated conditional → SURVIVED
48  1. negated conditional → TIMED_OUT
49  1. removed call to java/net/Socket::close → NO_COVERAGE
55  1. negated conditional → SURVIVED
    2. negated conditional → KILLED
56  1. removed call to java/net/Socket::close → SURVIVED
    2. removed call to java/net/Socket::close → TIMED_OUT
62  1. mutated return of Object value for net/sf/borg/common/SocketClient::sendMsg to ( if (x != null) null
    else throw new RuntimeException ) → KILLED
```

Figure 7: Mutations for the sendMsg method

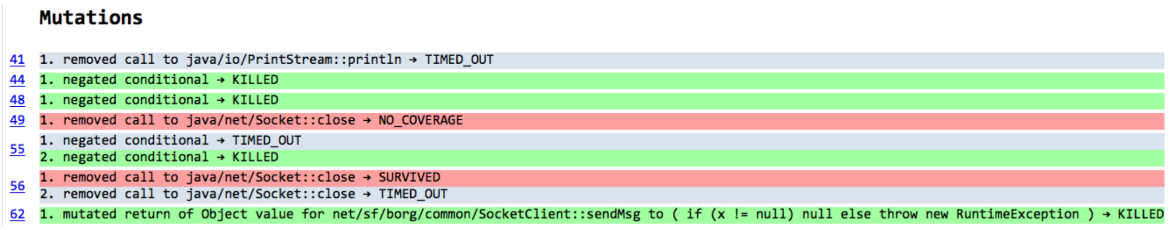
The results show that not all mutants have been killed. From Figure 6 we can see that our mutation testing results can be possibly improved if more tests on the server and socket state are created.

After Additional Test Cases



```
33      public static String sendMsg(String host, int port, String msg) throws IOException {
34              Socket s = null;
35              String line = null;
36              try {
37                      s = new Socket(host, port);
38                      BufferedReader sin = new BufferedReader(new InputStreamReader(s
39                                      .getInputStream()));
40                      PrintStream sout = new PrintStream(s.getOutputStream());
41 1                    sout.println(msg);
42                      line = sin.readLine();
43                      // Check if connection is closed (i.e. for EOF)
44 1                    if (line == null) {
45                              log.info("Connection closed by server.");
46                      }
47              } catch (IOException e) {
48 1                    if (s != null)
49 1                            s.close();
50                      throw e;
51              }
52              // Always be sure to close the socket
53              finally {
54                      try {
55 2                            if (s != null)
56 2                                    s.close();
57                      } catch (IOException e2) {
58                              // empty
59                      }
60              }
61
62 1            return line;
63      }
```

Figure 8: Updated code results for the sendMsg method



Mutations

```
41  1. removed call to java/io/PrintStream::println → TIMED_OUT
44  1. negated conditional → KILLED
48  1. negated conditional → KILLED
49  1. removed call to java/net/Socket::close → NO_COVERAGE
55  1. negated conditional → TIMED_OUT
    2. negated conditional → KILLED
56  1. removed call to java/net/Socket::close → SURVIVED
    2. removed call to java/net/Socket::close → TIMED_OUT
62  1. mutated return of Object value for net/sf/borg/common/SocketClient::sendMsg to ( if (x != null) null else throw new RuntimeException ) → KILLED
```

Figure 9: Updated mutations for the sendMsg method

The test case shown below was added and improved results to what is shown in Figures 8 and 9. This test case kills the mutant at lines 44, 48 without increasing line coverage. Thus, showing how test cases built for coverage alone are not sufficient.

The remaining bugs all relate to the *close* function of the Socket object instantiated in the *sendMsg* method. When Pitclipse mutates the *close* function and removes it, this action causes a timeout. Which is why the *sendMsg* method is defensively programmed to ensure that the socket connection is closed after the message is sent. Due to the amount of defensive programming and case of timeouts, it is not possible to achieve 100% only using mutation testing for the method sendMsg.

Listing 1: Additional test case for sendMsg

```java
@Test
public void checkServerAlive() {
String msg = null;
SocketServer ss;
String response;
   try {
      ss = new SocketServer(2922, this);
      response = SocketClient.sendMsg("localhost", 2922, msg);
      assertTrue(ss.isAlive());
   } catch (IOException e) {
      // TODO Auto-generated catch block
      e.printStackTrace();
   }
}
```

2 JPetStore

After exploring the JPetStore system, we came up with some realistic non-trivial test scenarios that can be carried out for load testing using JMeter. The following subsections cover each scenarios, description on how it was load tested and the result analysis of the load test.

Following are the system specifications for which the load test was conducted under:

- **Operating System**: Windows 10 Pro 64-bit (10.0, Build 16299)
- **Language**: English (Regional Setting: English)
- **System Manufacturer**: Hewlett-Packard
- **System Model**: HP 15 TouchSmart Notebook PC
- **BIOS**: F.10
- **Processor**: AMD A6-5200 APU with Radeon(TM) HD Graphics (4 CPUs), 2.0GHz
- **Memory**: 6144MB RAM
- **Java Version**: 1.8.0_151-b12
- **Apache Tomcat Version**: 7.0.85
- **JMeter Version**: 2.11 r1554548

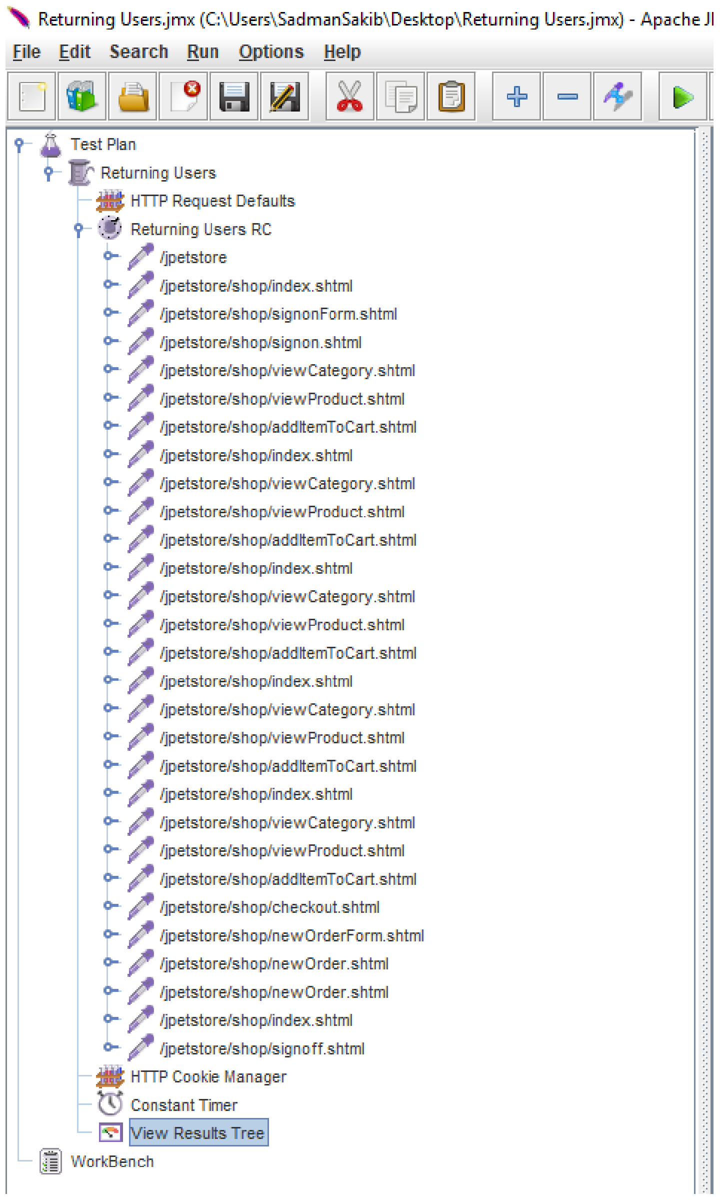
2.1 Scenario 1: Returning User

2.1.1 Overview

The first test scenario is for an existing user in the system. The testing scenario will consist of a returning user logging in, selecting one of each of the 5 possible items sold in JPetStore, adding the items to the cart, performing a checkout of the cart and finally logging out. The following describes an exact breakdown of the steps the load test will carry out:

- Access the JPetStore Homepage (`http://localhost:8080/jpetstore/`).
- Click *Enter the Store*.
- Click on the *Sign in* button.
- Enter sign-in credentials and click *Login* By default, we will be using *j2ee* user for this scenario.
- Go to the *Fish* section, select a fish item, add it to the cart and return to the main menu.
- Go to the *Dogs* section, select a dog item, add it to the cart and return to the main menu.
- Go to the *Reptiles* section, select a reptile item, add it to the cart and return to the main menu.
- Go to the *Cats* section, select a cat item, add it to the cart and return to the main menu.
- Go to the *Birds* section, select a bird item, add it to the cart.
- Proceed to checkout and follow the steps until the order has been placed.
- Return back to the main menu and sign out.

The following images depicts the Recording Controller for the test case scenario and the pages our load test will navigate through per each iteration:


Returning Users.jmx (C:\Users\SadmanSakib\Desktop\Returning Users.jmx) - Apache JI

```
Test Plan
  Returning Users
    HTTP Request Defaults
    Returning Users RC
      /jpetstore
      /jpetstore/shop/index.shtml
      /jpetstore/shop/signonForm.shtml
      /jpetstore/shop/signon.shtml
      /jpetstore/shop/viewCategory.shtml
      /jpetstore/shop/viewProduct.shtml
      /jpetstore/shop/addItemToCart.shtml
      /jpetstore/shop/index.shtml
      /jpetstore/shop/viewCategory.shtml
      /jpetstore/shop/viewProduct.shtml
      /jpetstore/shop/addItemToCart.shtml
      /jpetstore/shop/index.shtml
      /jpetstore/shop/viewCategory.shtml
      /jpetstore/shop/viewProduct.shtml
      /jpetstore/shop/addItemToCart.shtml
      /jpetstore/shop/index.shtml
      /jpetstore/shop/viewCategory.shtml
      /jpetstore/shop/viewProduct.shtml
      /jpetstore/shop/addItemToCart.shtml
      /jpetstore/shop/index.shtml
      /jpetstore/shop/viewCategory.shtml
      /jpetstore/shop/viewProduct.shtml
      /jpetstore/shop/addItemToCart.shtml
      /jpetstore/shop/checkout.shtml
      /jpetstore/shop/newOrderForm.shtml
      /jpetstore/shop/newOrder.shtml
      /jpetstore/shop/newOrder.shtml
      /jpetstore/shop/index.shtml
      /jpetstore/shop/signoff.shtml
    HTTP Cookie Manager
    Constant Timer
    View Results Tree
WorkBench
```

Figure 10: Recording Controller for Returning User Scenario

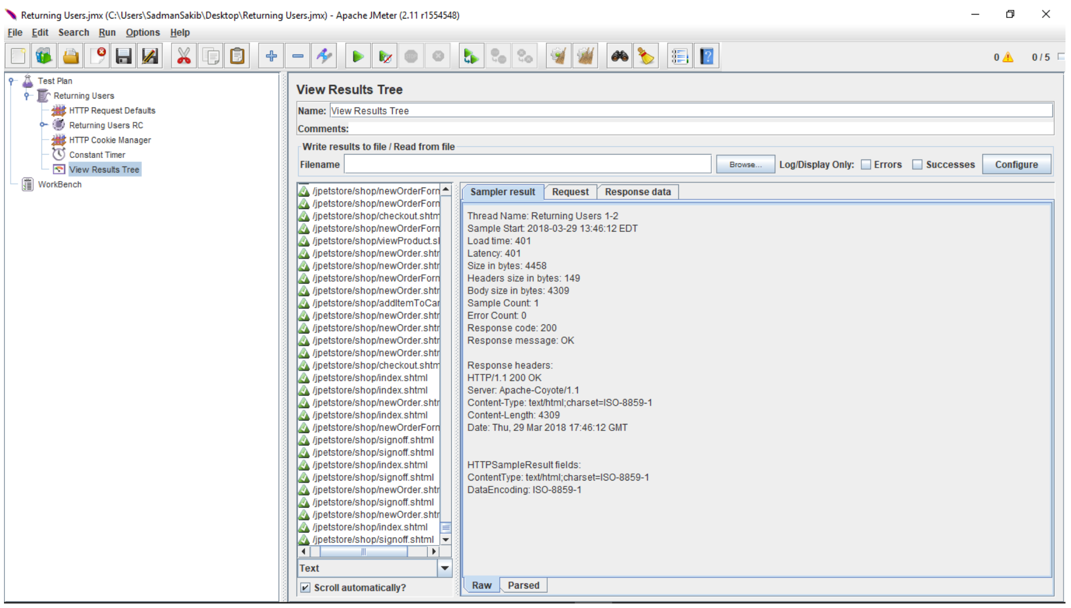
2.1.2 Load Test Properties

Following are the load test properties applied for testing this scenario:

- **Number of Thread (users)**: 5
- **Ramp-up Period (in seconds)**: 10
- **Loop Count**: 30
- **Thread Delay (in milliseconds)**: 1000

2.1.3 Executing Load Test

After setting up the load test plan using JMeter, we executed the test. The test run was for approximately 18 minutes completing all 30 iterations. The following diagrams show the result tree of the test run.



View Results Tree

Name: View Results Tree

Comments:

Write results to file / Read from file

Filename

☑ Scroll automatically?

```
/jpetstore/shop/newOrderForm
/jpetstore/shop/newOrderForm
/jpetstore/shop/checkout.shtm
/jpetstore/shop/newOrderForm
/jpetstore/shop/viewProduct.sl
/jpetstore/shop/newOrder.shtr
/jpetstore/shop/newOrder.shtr
/jpetstore/shop/newOrderForm
/jpetstore/shop/newOrder.shtr
/jpetstore/shop/addItemToCar
/jpetstore/shop/newOrder.shtr
/jpetstore/shop/newOrder.shtr
/jpetstore/shop/newOrder.shtr
/jpetstore/shop/newOrder.shtr
/jpetstore/shop/checkout.shtm
/jpetstore/shop/index.shtml
/jpetstore/shop/index.shtml
/jpetstore/shop/newOrder.shtr
/jpetstore/shop/index.shtml
/jpetstore/shop/newOrderForm
/jpetstore/shop/signoff.shtml
/jpetstore/shop/signoff.shtml
/jpetstore/shop/index.shtml
/jpetstore/shop/signoff.shtml
/jpetstore/shop/newOrder.shtr
/jpetstore/shop/signoff.shtml
/jpetstore/shop/newOrder.shtr
/jpetstore/shop/index.shtml
/jpetstore/shop/signoff.shtml
```

```
Thread Name: Returning Users 1-2
Sample Start: 2018-03-29 13:46:12 EDT
Load time: 401
Latency: 401
Size in bytes: 4458
Headers size in bytes: 149
Body size in bytes: 4309
Sample Count: 1
Error Count: 0
Response code: 200
Response message: OK

Response headers:
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/html;charset=ISO-8859-1
Content-Length: 4309
Date: Thu, 29 Mar 2018 17:46:12 GMT


HTTPSampleResult fields:
ContentType: text/html;charset=ISO-8859-1
DataEncoding: ISO-8859-1
```

Figure 11: View Results Tree for Returning User Scenario

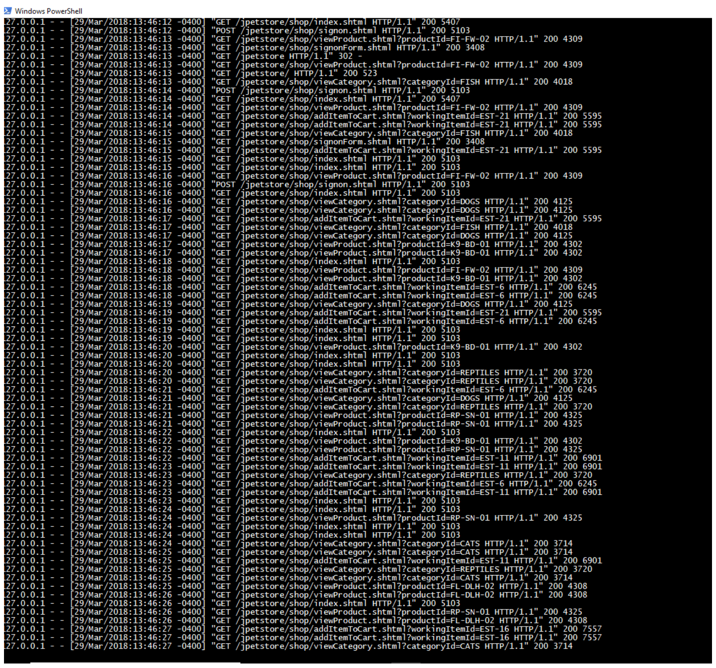
2.1.4 Analysing Load Test

The following statistics were gathered from the Apache access logs. The access log file was parsed to get the start of the load test and the end of the load test and the total time of the load test execution.

Within that timeframe, the total number of requests was found by simply executing: *wc -l ruser_log.txt*. The total number of GET and POST requests were found by executing *grep -c -w "GET" ruser_log.txt* and *grep -c -w "POST" ruser_log.txt* respectively.

- **Test Duration**: Approximately 18 minutes
- **Total Number of Requests**: 4714
- **Request Rate**: 4.36 requests/second
- **Number of GET Requests**: 4399 (93% of all requests)
- **Number of POST Requests**: 315 (7% of all requests)
- **Success HTTP Codes**: 200 (Success) and 302 (Found)
- **Failure HTTP Codes**: None

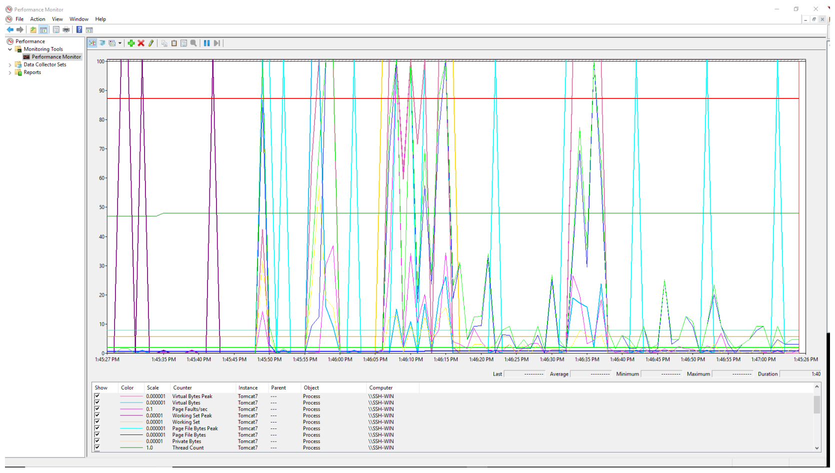
Following is a snapshot of the Apache access log:



```
27.0.0.1 - - [29/Mar/2018:13:46:12 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5407
27.0.0.1 - - [29/Mar/2018:13:46:12 -0400] "POST /jpetstore/shop/signon.shtml HTTP/1.1" 200 5103
27.0.0.1 - - [29/Mar/2018:13:46:13 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=FI-FW-02 HTTP/1.1" 200 4309
27.0.0.1 - - [29/Mar/2018:13:46:13 -0400] "GET /jpetstore/shop/signonForm.shtml HTTP/1.1" 200 3408
27.0.0.1 - - [29/Mar/2018:13:46:13 -0400] "GET /jpetstore HTTP/1.1" 302 -
27.0.0.1 - - [29/Mar/2018:13:46:13 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=FI-FW-02 HTTP/1.1" 200 4309
27.0.0.1 - - [29/Mar/2018:13:46:13 -0400] "GET /jpetstore/ HTTP/1.1" 200 523
27.0.0.1 - - [29/Mar/2018:13:46:13 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=FISH HTTP/1.1" 200 4018
27.0.0.1 - - [29/Mar/2018:13:46:14 -0400] "POST /jpetstore/shop/signon.shtml HTTP/1.1" 200 5103
27.0.0.1 - - [29/Mar/2018:13:46:14 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5407
27.0.0.1 - - [29/Mar/2018:13:46:14 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=FI-FW-02 HTTP/1.1" 200 4309
27.0.0.1 - - [29/Mar/2018:13:46:14 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-21 HTTP/1.1" 200 5595
27.0.0.1 - - [29/Mar/2018:13:46:14 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-21 HTTP/1.1" 200 5595
27.0.0.1 - - [29/Mar/2018:13:46:15 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=FISH HTTP/1.1" 200 4018
27.0.0.1 - - [29/Mar/2018:13:46:15 -0400] "GET /jpetstore/shop/signonForm.shtml HTTP/1.1" 200 3408
27.0.0.1 - - [29/Mar/2018:13:46:15 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-21 HTTP/1.1" 200 5595
27.0.0.1 - - [29/Mar/2018:13:46:15 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
27.0.0.1 - - [29/Mar/2018:13:46:15 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
27.0.0.1 - - [29/Mar/2018:13:46:16 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=FI-FW-02 HTTP/1.1" 200 4309
27.0.0.1 - - [29/Mar/2018:13:46:16 -0400] "POST /jpetstore/shop/signon.shtml HTTP/1.1" 200 5103
27.0.0.1 - - [29/Mar/2018:13:46:16 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
27.0.0.1 - - [29/Mar/2018:13:46:16 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=DOGS HTTP/1.1" 200 4125
27.0.0.1 - - [29/Mar/2018:13:46:16 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=DOGS HTTP/1.1" 200 4125
27.0.0.1 - - [29/Mar/2018:13:46:17 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-21 HTTP/1.1" 200 5595
27.0.0.1 - - [29/Mar/2018:13:46:17 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=FISH HTTP/1.1" 200 4018
27.0.0.1 - - [29/Mar/2018:13:46:17 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=DOGS HTTP/1.1" 200 4125
27.0.0.1 - - [29/Mar/2018:13:46:17 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=K9-BD-01 HTTP/1.1" 200 4302
27.0.0.1 - - [29/Mar/2018:13:46:17 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=K9-BD-01 HTTP/1.1" 200 4302
27.0.0.1 - - [29/Mar/2018:13:46:18 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
27.0.0.1 - - [29/Mar/2018:13:46:18 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=FI-FW-02 HTTP/1.1" 200 4309
27.0.0.1 - - [29/Mar/2018:13:46:18 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=K9-BD-01 HTTP/1.1" 200 4302
27.0.0.1 - - [29/Mar/2018:13:46:18 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-6 HTTP/1.1" 200 6245
27.0.0.1 - - [29/Mar/2018:13:46:18 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-6 HTTP/1.1" 200 6245
27.0.0.1 - - [29/Mar/2018:13:46:19 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=DOGS HTTP/1.1" 200 4125
27.0.0.1 - - [29/Mar/2018:13:46:19 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-21 HTTP/1.1" 200 5595
27.0.0.1 - - [29/Mar/2018:13:46:19 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-6 HTTP/1.1" 200 6245
27.0.0.1 - - [29/Mar/2018:13:46:19 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
27.0.0.1 - - [29/Mar/2018:13:46:19 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
27.0.0.1 - - [29/Mar/2018:13:46:20 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=K9-BD-01 HTTP/1.1" 200 4302
27.0.0.1 - - [29/Mar/2018:13:46:20 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
27.0.0.1 - - [29/Mar/2018:13:46:20 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
27.0.0.1 - - [29/Mar/2018:13:46:20 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=REPTILES HTTP/1.1" 200 3720
27.0.0.1 - - [29/Mar/2018:13:46:20 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=REPTILES HTTP/1.1" 200 3720
27.0.0.1 - - [29/Mar/2018:13:46:21 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-6 HTTP/1.1" 200 6245
27.0.0.1 - - [29/Mar/2018:13:46:21 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=DOGS HTTP/1.1" 200 4125
27.0.0.1 - - [29/Mar/2018:13:46:21 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=REPTILES HTTP/1.1" 200 3720
27.0.0.1 - - [29/Mar/2018:13:46:21 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=RP-SN-01 HTTP/1.1" 200 4325
27.0.0.1 - - [29/Mar/2018:13:46:21 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=RP-SN-01 HTTP/1.1" 200 4325
27.0.0.1 - - [29/Mar/2018:13:46:22 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
27.0.0.1 - - [29/Mar/2018:13:46:22 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=K9-BD-01 HTTP/1.1" 200 4302
27.0.0.1 - - [29/Mar/2018:13:46:22 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=RP-SN-01 HTTP/1.1" 200 4325
27.0.0.1 - - [29/Mar/2018:13:46:22 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-11 HTTP/1.1" 200 6901
27.0.0.1 - - [29/Mar/2018:13:46:23 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-11 HTTP/1.1" 200 6901
27.0.0.1 - - [29/Mar/2018:13:46:23 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=REPTILES HTTP/1.1" 200 3720
27.0.0.1 - - [29/Mar/2018:13:46:23 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-6 HTTP/1.1" 200 6245
27.0.0.1 - - [29/Mar/2018:13:46:23 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-11 HTTP/1.1" 200 6901
27.0.0.1 - - [29/Mar/2018:13:46:23 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
27.0.0.1 - - [29/Mar/2018:13:46:24 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
27.0.0.1 - - [29/Mar/2018:13:46:24 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=RP-SN-01 HTTP/1.1" 200 4325
27.0.0.1 - - [29/Mar/2018:13:46:24 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
27.0.0.1 - - [29/Mar/2018:13:46:24 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
27.0.0.1 - - [29/Mar/2018:13:46:24 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=CATS HTTP/1.1" 200 3714
27.0.0.1 - - [29/Mar/2018:13:46:25 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=CATS HTTP/1.1" 200 3714
27.0.0.1 - - [29/Mar/2018:13:46:25 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-11 HTTP/1.1" 200 6901
27.0.0.1 - - [29/Mar/2018:13:46:25 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=REPTILES HTTP/1.1" 200 3720
27.0.0.1 - - [29/Mar/2018:13:46:25 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=CATS HTTP/1.1" 200 3714
27.0.0.1 - - [29/Mar/2018:13:46:25 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=FL-DLH-02 HTTP/1.1" 200 4308
27.0.0.1 - - [29/Mar/2018:13:46:26 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=FL-DLH-02 HTTP/1.1" 200 4308
27.0.0.1 - - [29/Mar/2018:13:46:26 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
27.0.0.1 - - [29/Mar/2018:13:46:26 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=RP-SN-01 HTTP/1.1" 200 4325
27.0.0.1 - - [29/Mar/2018:13:46:26 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=FL-DLH-02 HTTP/1.1" 200 4308
27.0.0.1 - - [29/Mar/2018:13:46:27 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-16 HTTP/1.1" 200 7557
27.0.0.1 - - [29/Mar/2018:13:46:27 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-16 HTTP/1.1" 200 7557
27.0.0.1 - - [29/Mar/2018:13:46:27 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=CATS HTTP/1.1" 200 3714
```

Figure 12: Apache Access Log Snapshot for Returning User Scenario

Following is a snapshot of the Windows Performance Monitor and Java Monitor Console during the load test execution:



Show	Color	Scale	Counter	Instance	Parent	Object	Computer
✓	🟪	0.000001	Virtual Bytes Peak	Tomcat7	---	Process	\\SSH-WIN
✓	🟦	0.000001	Virtual Bytes	Tomcat7	---	Process	\\SSH-WIN
✓	🟪	0.1	Page Faults/sec	Tomcat7	---	Process	\\SSH-WIN
✓	🟪	0.00001	Working Set Peak	Tomcat7	---	Process	\\SSH-WIN
✓	🟨	0.00001	Working Set	Tomcat7	---	Process	\\SSH-WIN
✓	🟦	0.000001	Page File Bytes Peak	Tomcat7	---	Process	\\SSH-WIN
✓	🟦	0.000001	Page File Bytes	Tomcat7	---	Process	\\SSH-WIN
✓	🟨	0.00001	Private Bytes	Tomcat7	---	Process	\\SSH-WIN
✓	🟩	1.0	Thread Count	Tomcat7	---	Process	\\SSH-WIN

Figure 13: Windows Performance Monitor Snapshot for Returning User Scenario

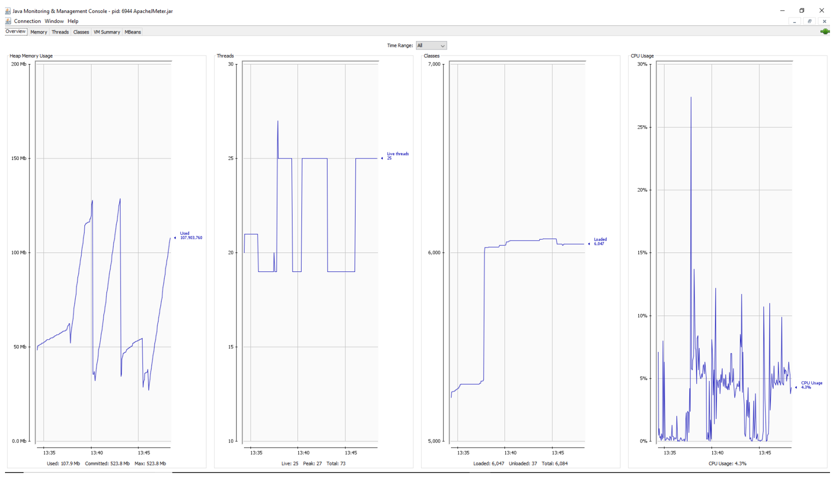

Java Monitoring & Management Console - pid: 6944 Apache/JMeter.jar
Connection Window Help
Overview Memory Threads Classes VM Summary MBeans
Time Range: All
Heap Memory Usage
Threads
Classes
CPU Usage
Used: 107.9 Mb Committed: 523.8 Mb Max: 523.8 Mb
Live: 25 Peak: 27 Total: 73
Loaded: 6,047 Unloaded: 37 Total: 6,084
CPU Usage: 4.3%

Figure 14: JConsole Snapshot for Returning User Scenario

Conclusion: The load test scenario for returning user made about 4714 requests and the test ran for approximately 18 minutes. Despite some natural high spikes on the performance monitor and java monitor, the load test was carried out successfully without any crashes or unexpected behaviour in the application.

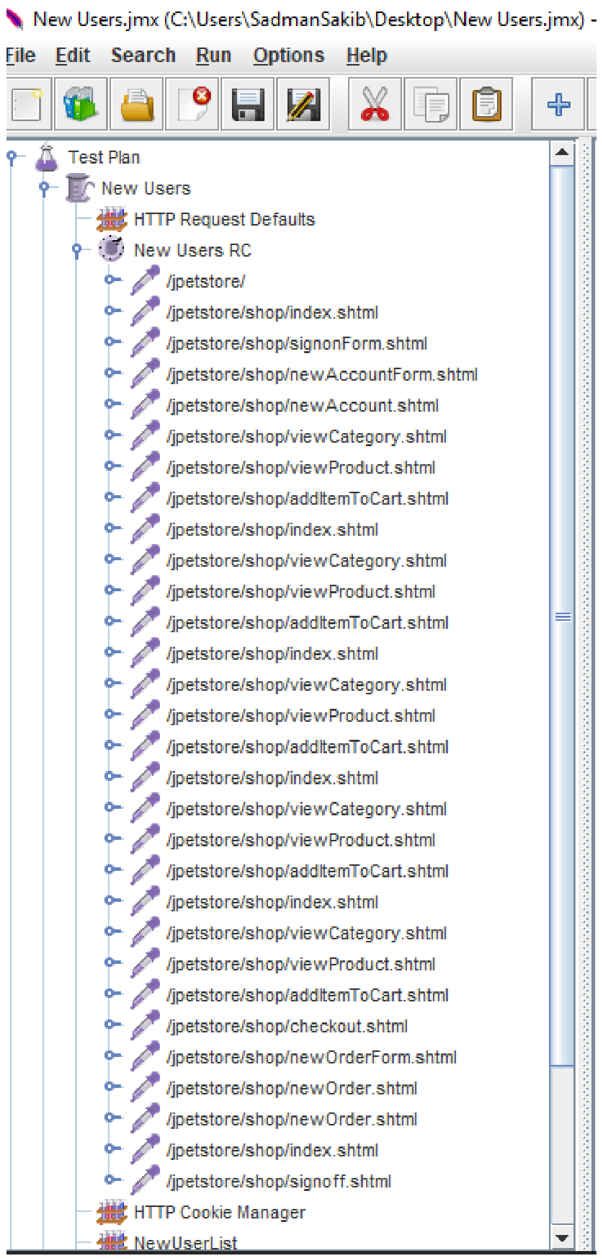
2.2 Scenario 2: New User

2.2.1 Overview

The second test scenario is for a new user in the system. The testing scenario will consist of registering a new user, selecting one of each of the 5 possible items sold in JPetStore, adding the items to the cart, performing a checkout of the cart and finally logging out. The following describes an exact breakdown of the steps the load test will carry out:

- Access the JPetStore Homepage (`http://localhost:8080/jpetstore/`).
- Click *Enter the Store*.
- Click on the *Sign in* button.
- Click *Register now*.
- Enter the sign-up credentials and click *Create Account*. The username and password for signing up would be loaded from a CSV file, whereas the other fields will be supplied the value of *abc*.
- Go to the *Fish* section, select a fish item, add it to the cart and return to the main menu.
- Go to the *Dogs* section, select a dog item, add it to the cart and return to the main menu.
- Go to the *Reptiles* section, select a reptile item, add it to the cart and return to the main menu.
- Go to the *Cats* section, select a cat item, add it to the cart and return to the main menu.
- Go to the *Birds* section, select a bird item, add it to the cart.
- Proceed to checkout and follow the steps until the order has been placed.
- Return back to the main menu and sign out.

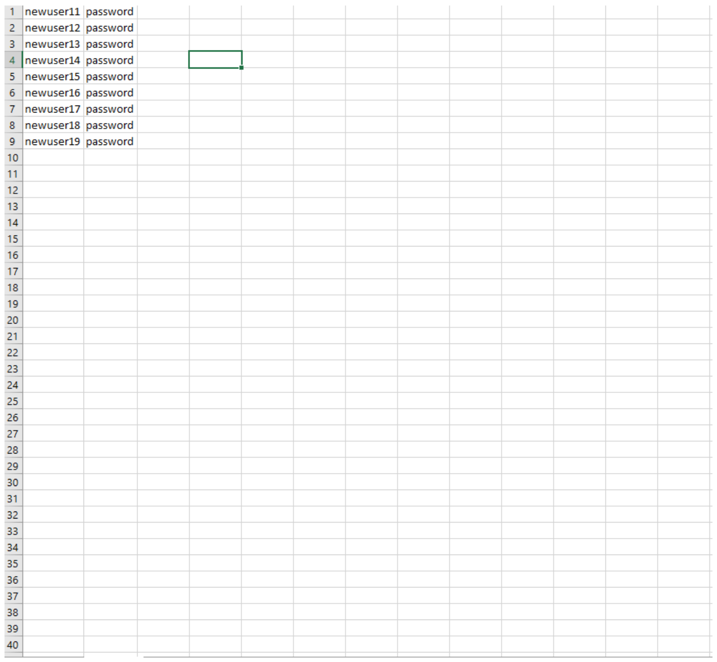
The following images depicts the Recording Controller for the test case scenario and the pages our load test will navigate through per each iteration:


New Users.jmx (C:\Users\SadmanSakib\Desktop\New Users.jmx) -
File Edit Search Run Options Help

Test Plan
New Users
HTTP Request Defaults
New Users RC
/jpetstore/
/jpetstore/shop/index.shtml
/jpetstore/shop/signonForm.shtml
/jpetstore/shop/newAccountForm.shtml
/jpetstore/shop/newAccount.shtml
/jpetstore/shop/viewCategory.shtml
/jpetstore/shop/viewProduct.shtml
/jpetstore/shop/addItemToCart.shtml
/jpetstore/shop/index.shtml
/jpetstore/shop/viewCategory.shtml
/jpetstore/shop/viewProduct.shtml
/jpetstore/shop/addItemToCart.shtml
/jpetstore/shop/index.shtml
/jpetstore/shop/viewCategory.shtml
/jpetstore/shop/viewProduct.shtml
/jpetstore/shop/addItemToCart.shtml
/jpetstore/shop/index.shtml
/jpetstore/shop/viewCategory.shtml
/jpetstore/shop/viewProduct.shtml
/jpetstore/shop/addItemToCart.shtml
/jpetstore/shop/index.shtml
/jpetstore/shop/viewCategory.shtml
/jpetstore/shop/viewProduct.shtml
/jpetstore/shop/addItemToCart.shtml
/jpetstore/shop/checkout.shtml
/jpetstore/shop/newOrderForm.shtml
/jpetstore/shop/newOrder.shtml
/jpetstore/shop/newOrder.shtml
/jpetstore/shop/index.shtml
/jpetstore/shop/signoff.shtml
HTTP Cookie Manager
NewUserList

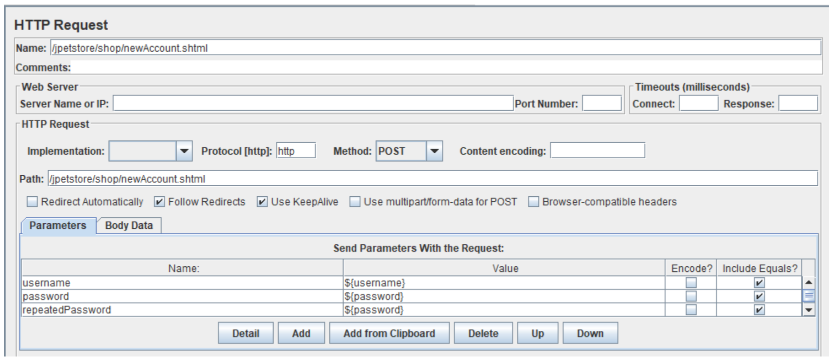
Figure 15: Recording Controller for New User Scenario

The following images depicts the snapshot of the CSV file containing usernames and password for registering new users. On the POST request for registering a new user we load the credentials and use them through variable names such as $\{username\} and $\{password\}.



1	newuser11	password
2	newuser12	password
3	newuser13	password
4	newuser14	password
5	newuser15	password
6	newuser16	password
7	newuser17	password
8	newuser18	password
9	newuser19	password
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20		
21		
22		
23		
24		
25		
26		
27		
28		
29		
30		
31		
32		
33		
34		
35		
36		
37		
38		
39		
40		

Figure 16: CSV Credential File for New User Scenario



HTTP Request

Name: /jpetstore/shop/newAccount.shtml

Comments:

Web Server

Server Name or IP: Port Number:

Timeouts (milliseconds)

Connect: Response:

HTTP Request

Implementation: Protocol [http]: http Method: POST Content encoding:

Path: /jpetstore/shop/newAccount.shtml

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

Parameters | Body Data

Send Parameters With the Request:

Name:	Value	Encode?	Include Equals?
username	${username}	☐	☑
password	${password}	☐	☑
repeatedPassword	${password}	☐	☑

Figure 17: HTTP Post Request for New User Scenario

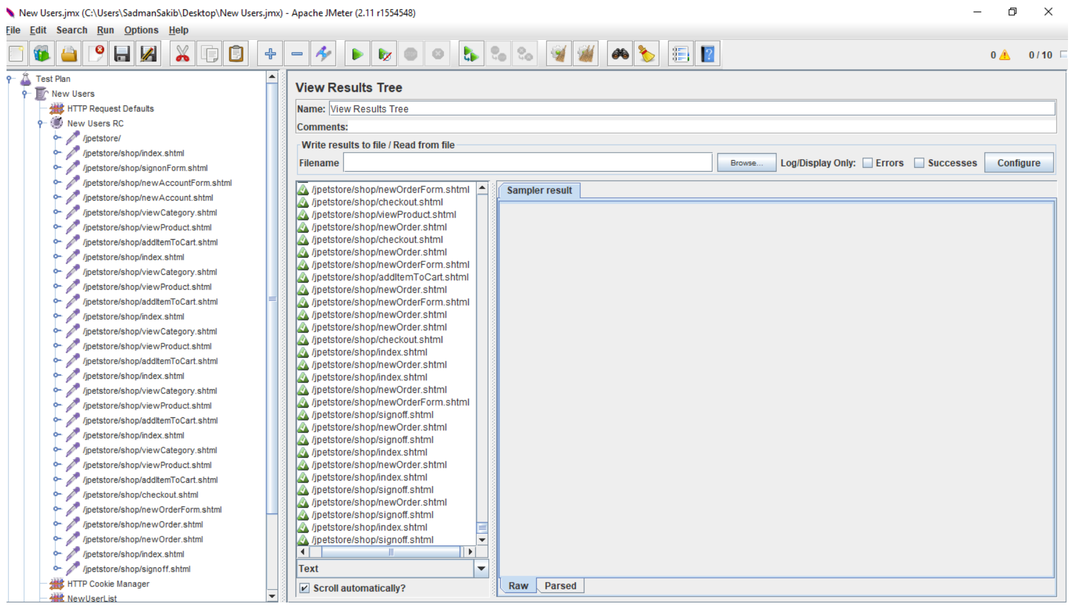
2.2.2 Load Test Properties

Following are the load test properties applied for testing this scenario:

- **Number of Thread (users)**: 5
- **Ramp-up Period (in seconds)**: 5
- **Loop Count**: 30
- **Thread Delay (in milliseconds)**: 1000

2.2.3 Executing Load Test

After setting up the load test plan using JMeter, we executed the test. The test run was for approximately 15 minutes completing all 30 iterations. The following diagrams show the result tree of the test run.



New Users.jmx (C:\Users\SadmanSakib\Desktop\New Users.jmx) - Apache JMeter (2.11 r1554548)

```
Test Plan
  New Users
    HTTP Request Defaults
    New Users RC
      /jpetstore/
      /jpetstore/shop/index.shtml
      /jpetstore/shop/signonForm.shtml
      /jpetstore/shop/newAccountForm.shtml
      /jpetstore/shop/newAccount.shtml
      /jpetstore/shop/viewCategory.shtml
      /jpetstore/shop/viewProduct.shtml
      /jpetstore/shop/addItemToCart.shtml
      /jpetstore/shop/index.shtml
      /jpetstore/shop/viewCategory.shtml
      /jpetstore/shop/viewProduct.shtml
      /jpetstore/shop/addItemToCart.shtml
      /jpetstore/shop/index.shtml
      /jpetstore/shop/viewCategory.shtml
      /jpetstore/shop/viewProduct.shtml
      /jpetstore/shop/addItemToCart.shtml
      /jpetstore/shop/index.shtml
      /jpetstore/shop/viewCategory.shtml
      /jpetstore/shop/viewProduct.shtml
      /jpetstore/shop/addItemToCart.shtml
      /jpetstore/shop/index.shtml
      /jpetstore/shop/viewCategory.shtml
      /jpetstore/shop/viewProduct.shtml
      /jpetstore/shop/addItemToCart.shtml
      /jpetstore/shop/checkout.shtml
      /jpetstore/shop/newOrderForm.shtml
      /jpetstore/shop/newOrder.shtml
      /jpetstore/shop/newOrder.shtml
      /jpetstore/shop/index.shtml
      /jpetstore/shop/signoff.shtml
    HTTP Cookie Manager
    NewUserList
```

View Results Tree

Name: View Results Tree
Comments:
Write results to file / Read from file
Filename: Log/Display Only: ☐ Errors ☐ Successes Configure

```
/jpetstore/shop/newOrderForm.shtml
/jpetstore/shop/checkout.shtml
/jpetstore/shop/viewProduct.shtml
/jpetstore/shop/newOrder.shtml
/jpetstore/shop/checkout.shtml
/jpetstore/shop/newOrder.shtml
/jpetstore/shop/newOrderForm.shtml
/jpetstore/shop/addItemToCart.shtml
/jpetstore/shop/newOrder.shtml
/jpetstore/shop/newOrderForm.shtml
/jpetstore/shop/newOrder.shtml
/jpetstore/shop/newOrder.shtml
/jpetstore/shop/checkout.shtml
/jpetstore/shop/index.shtml
/jpetstore/shop/newOrder.shtml
/jpetstore/shop/index.shtml
/jpetstore/shop/newOrder.shtml
/jpetstore/shop/newOrderForm.shtml
/jpetstore/shop/signoff.shtml
/jpetstore/shop/newOrder.shtml
/jpetstore/shop/signoff.shtml
/jpetstore/shop/index.shtml
/jpetstore/shop/newOrder.shtml
/jpetstore/shop/index.shtml
/jpetstore/shop/signoff.shtml
/jpetstore/shop/newOrder.shtml
/jpetstore/shop/signoff.shtml
/jpetstore/shop/index.shtml
/jpetstore/shop/signoff.shtml
```

Sampler result

Text

☑ Scroll automatically?

Figure 18: View Results Tree for New User Scenario

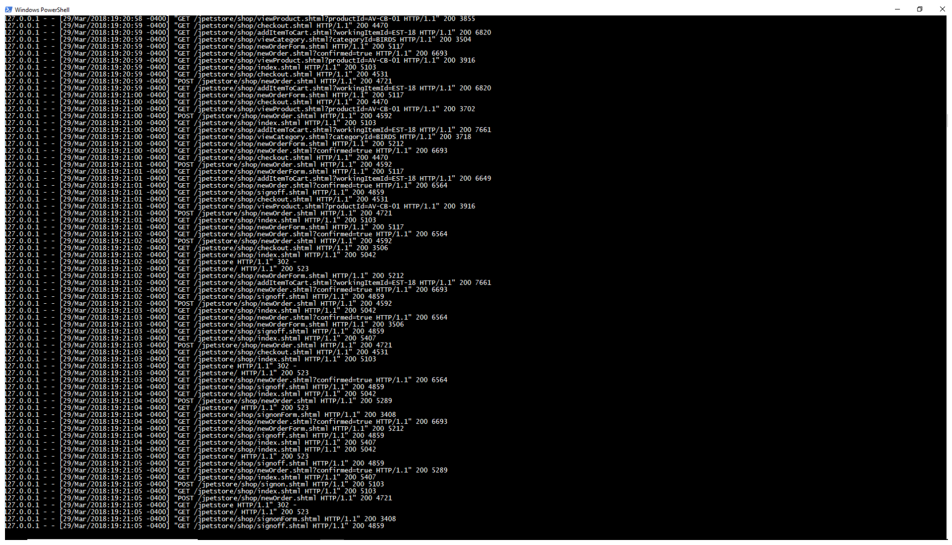
2.2.4 Analysing Load Test

The following statistics were gathered from the Apache access logs. The access log file was parsed to get the start of the load test and the end of the load test and the total time of the load test execution.

Within that timeframe, the total number of requests was found by simply executing: *wc -l nuser_log.txt*. The total number of GET and POST requests were found by executing *grep -c -w "GET" nuser_log.txt* and *grep -c -w "POST" nuser_log.txt* respectively.

- **Test Duration**: Approximately 15 minutes
- **Total Number of Requests**: 7500
- **Request Rate**: 8.33 requests/second
- **Number of GET Requests**: 7000 (93% of all requests)
- **Number of POST Requests**: 500 (7% of all requests)
- **Success HTTP Codes**: 200 (Success) and 302 (Found)
- **Failure HTTP Codes**: None

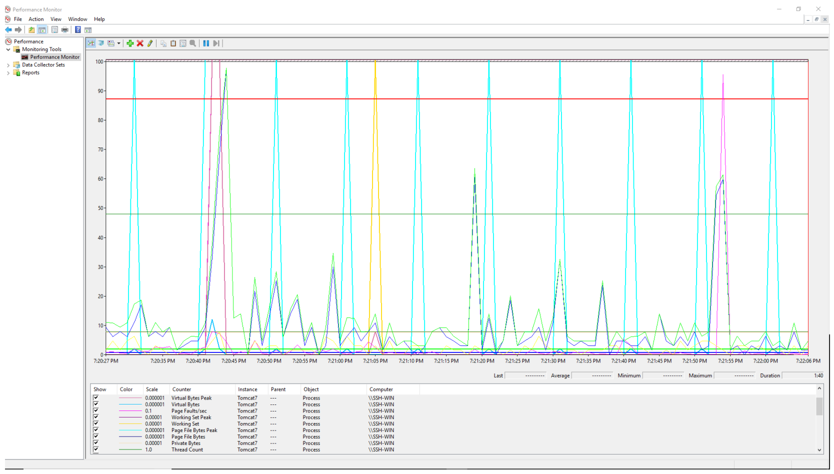
Following is a snapshot of the Apache access log:



```
127.0.0.1 - - [29/Mar/2018:19:20:58 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=AV-CB-01 HTTP/1.1" 200 3855
127.0.0.1 - - [29/Mar/2018:19:20:59 -0400] "GET /jpetstore/shop/checkout.shtml HTTP/1.1" 200 4470
127.0.0.1 - - [29/Mar/2018:19:20:59 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-18 HTTP/1.1" 200 6820
127.0.0.1 - - [29/Mar/2018:19:20:59 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=BIRDS HTTP/1.1" 200 3504
127.0.0.1 - - [29/Mar/2018:19:20:59 -0400] "GET /jpetstore/shop/newOrderForm.shtml HTTP/1.1" 200 5117
127.0.0.1 - - [29/Mar/2018:19:20:59 -0400] "GET /jpetstore/shop/newOrder.shtml?confirmed=true HTTP/1.1" 200 6693
127.0.0.1 - - [29/Mar/2018:19:20:59 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=AV-CB-01 HTTP/1.1" 200 3916
127.0.0.1 - - [29/Mar/2018:19:20:59 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
127.0.0.1 - - [29/Mar/2018:19:20:59 -0400] "GET /jpetstore/shop/checkout.shtml HTTP/1.1" 200 4531
127.0.0.1 - - [29/Mar/2018:19:20:59 -0400] "POST /jpetstore/shop/newOrder.shtml HTTP/1.1" 200 4721
127.0.0.1 - - [29/Mar/2018:19:20:59 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-18 HTTP/1.1" 200 6820
127.0.0.1 - - [29/Mar/2018:19:21:00 -0400] "GET /jpetstore/shop/newOrderForm.shtml HTTP/1.1" 200 5117
127.0.0.1 - - [29/Mar/2018:19:21:00 -0400] "GET /jpetstore/shop/checkout.shtml HTTP/1.1" 200 4470
127.0.0.1 - - [29/Mar/2018:19:21:00 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=AV-CB-01 HTTP/1.1" 200 3702
127.0.0.1 - - [29/Mar/2018:19:21:00 -0400] "POST /jpetstore/shop/newOrder.shtml HTTP/1.1" 200 4592
127.0.0.1 - - [29/Mar/2018:19:21:00 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
127.0.0.1 - - [29/Mar/2018:19:21:00 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-18 HTTP/1.1" 200 7661
127.0.0.1 - - [29/Mar/2018:19:21:00 -0400] "GET /jpetstore/shop/viewCategory.shtml?categoryId=BIRDS HTTP/1.1" 200 3718
127.0.0.1 - - [29/Mar/2018:19:21:00 -0400] "GET /jpetstore/shop/newOrderForm.shtml HTTP/1.1" 200 5212
127.0.0.1 - - [29/Mar/2018:19:21:00 -0400] "GET /jpetstore/shop/newOrder.shtml?confirmed=true HTTP/1.1" 200 6693
127.0.0.1 - - [29/Mar/2018:19:21:00 -0400] "GET /jpetstore/shop/checkout.shtml HTTP/1.1" 200 4470
127.0.0.1 - - [29/Mar/2018:19:21:01 -0400] "POST /jpetstore/shop/newOrder.shtml HTTP/1.1" 200 4592
127.0.0.1 - - [29/Mar/2018:19:21:01 -0400] "GET /jpetstore/shop/newOrderForm.shtml HTTP/1.1" 200 5117
127.0.0.1 - - [29/Mar/2018:19:21:01 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-18 HTTP/1.1" 200 6649
127.0.0.1 - - [29/Mar/2018:19:21:01 -0400] "GET /jpetstore/shop/newOrder.shtml?confirmed=true HTTP/1.1" 200 6564
127.0.0.1 - - [29/Mar/2018:19:21:01 -0400] "GET /jpetstore/shop/signoff.shtml HTTP/1.1" 200 4859
127.0.0.1 - - [29/Mar/2018:19:21:01 -0400] "GET /jpetstore/shop/checkout.shtml HTTP/1.1" 200 4531
127.0.0.1 - - [29/Mar/2018:19:21:01 -0400] "GET /jpetstore/shop/viewProduct.shtml?productId=AV-CB-01 HTTP/1.1" 200 3916
127.0.0.1 - - [29/Mar/2018:19:21:01 -0400] "POST /jpetstore/shop/newOrder.shtml HTTP/1.1" 200 4721
127.0.0.1 - - [29/Mar/2018:19:21:01 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
127.0.0.1 - - [29/Mar/2018:19:21:01 -0400] "GET /jpetstore/shop/newOrderForm.shtml HTTP/1.1" 200 5117
127.0.0.1 - - [29/Mar/2018:19:21:02 -0400] "GET /jpetstore/shop/newOrder.shtml?confirmed=true HTTP/1.1" 200 6564
127.0.0.1 - - [29/Mar/2018:19:21:02 -0400] "POST /jpetstore/shop/newOrder.shtml HTTP/1.1" 200 4592
127.0.0.1 - - [29/Mar/2018:19:21:02 -0400] "GET /jpetstore/shop/checkout.shtml HTTP/1.1" 200 3506
127.0.0.1 - - [29/Mar/2018:19:21:02 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5042
127.0.0.1 - - [29/Mar/2018:19:21:02 -0400] "GET /jpetstore HTTP/1.1" 302 -
127.0.0.1 - - [29/Mar/2018:19:21:02 -0400] "GET /jpetstore/ HTTP/1.1" 200 523
127.0.0.1 - - [29/Mar/2018:19:21:02 -0400] "GET /jpetstore/shop/newOrderForm.shtml HTTP/1.1" 200 5212
127.0.0.1 - - [29/Mar/2018:19:21:02 -0400] "GET /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-18 HTTP/1.1" 200 7661
127.0.0.1 - - [29/Mar/2018:19:21:02 -0400] "GET /jpetstore/shop/newOrder.shtml?confirmed=true HTTP/1.1" 200 6693
127.0.0.1 - - [29/Mar/2018:19:21:02 -0400] "GET /jpetstore/shop/signoff.shtml HTTP/1.1" 200 4859
127.0.0.1 - - [29/Mar/2018:19:21:02 -0400] "POST /jpetstore/shop/newOrder.shtml HTTP/1.1" 200 4592
127.0.0.1 - - [29/Mar/2018:19:21:03 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5042
127.0.0.1 - - [29/Mar/2018:19:21:03 -0400] "GET /jpetstore/shop/newOrder.shtml?confirmed=true HTTP/1.1" 200 6564
127.0.0.1 - - [29/Mar/2018:19:21:03 -0400] "GET /jpetstore/shop/newOrderForm.shtml HTTP/1.1" 200 3506
127.0.0.1 - - [29/Mar/2018:19:21:03 -0400] "GET /jpetstore/shop/signoff.shtml HTTP/1.1" 200 4859
127.0.0.1 - - [29/Mar/2018:19:21:03 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5407
127.0.0.1 - - [29/Mar/2018:19:21:03 -0400] "POST /jpetstore/shop/newOrder.shtml HTTP/1.1" 200 4721
127.0.0.1 - - [29/Mar/2018:19:21:03 -0400] "GET /jpetstore/shop/checkout.shtml HTTP/1.1" 200 4531
127.0.0.1 - - [29/Mar/2018:19:21:03 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
127.0.0.1 - - [29/Mar/2018:19:21:03 -0400] "GET /jpetstore HTTP/1.1" 302 -
127.0.0.1 - - [29/Mar/2018:19:21:03 -0400] "GET /jpetstore/ HTTP/1.1" 200 523
127.0.0.1 - - [29/Mar/2018:19:21:03 -0400] "GET /jpetstore/shop/newOrder.shtml?confirmed=true HTTP/1.1" 200 6564
127.0.0.1 - - [29/Mar/2018:19:21:04 -0400] "GET /jpetstore/shop/signoff.shtml HTTP/1.1" 200 4859
127.0.0.1 - - [29/Mar/2018:19:21:04 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5042
127.0.0.1 - - [29/Mar/2018:19:21:04 -0400] "POST /jpetstore/shop/newOrder.shtml HTTP/1.1" 200 5289
127.0.0.1 - - [29/Mar/2018:19:21:04 -0400] "GET /jpetstore/ HTTP/1.1" 200 523
127.0.0.1 - - [29/Mar/2018:19:21:04 -0400] "GET /jpetstore/shop/signonForm.shtml HTTP/1.1" 200 3408
127.0.0.1 - - [29/Mar/2018:19:21:04 -0400] "GET /jpetstore/shop/newOrder.shtml?confirmed=true HTTP/1.1" 200 6693
127.0.0.1 - - [29/Mar/2018:19:21:04 -0400] "GET /jpetstore/shop/newOrderForm.shtml HTTP/1.1" 200 5212
127.0.0.1 - - [29/Mar/2018:19:21:04 -0400] "GET /jpetstore/shop/signoff.shtml HTTP/1.1" 200 4859
127.0.0.1 - - [29/Mar/2018:19:21:04 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5407
127.0.0.1 - - [29/Mar/2018:19:21:04 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5042
127.0.0.1 - - [29/Mar/2018:19:21:05 -0400] "GET /jpetstore/ HTTP/1.1" 200 523
127.0.0.1 - - [29/Mar/2018:19:21:05 -0400] "GET /jpetstore/shop/signoff.shtml HTTP/1.1" 200 4859
127.0.0.1 - - [29/Mar/2018:19:21:05 -0400] "GET /jpetstore/shop/newOrder.shtml?confirmed=true HTTP/1.1" 200 5289
127.0.0.1 - - [29/Mar/2018:19:21:05 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5407
127.0.0.1 - - [29/Mar/2018:19:21:05 -0400] "POST /jpetstore/shop/signon.shtml HTTP/1.1" 200 5103
127.0.0.1 - - [29/Mar/2018:19:21:05 -0400] "GET /jpetstore/shop/index.shtml HTTP/1.1" 200 5103
127.0.0.1 - - [29/Mar/2018:19:21:05 -0400] "POST /jpetstore/shop/newOrder.shtml HTTP/1.1" 200 4721
127.0.0.1 - - [29/Mar/2018:19:21:05 -0400] "GET /jpetstore HTTP/1.1" 302 -
127.0.0.1 - - [29/Mar/2018:19:21:05 -0400] "GET /jpetstore/ HTTP/1.1" 200 523
127.0.0.1 - - [29/Mar/2018:19:21:05 -0400] "GET /jpetstore/shop/signonForm.shtml HTTP/1.1" 200 3408
127.0.0.1 - - [29/Mar/2018:19:21:05 -0400] "GET /jpetstore/shop/signoff.shtml HTTP/1.1" 200 4859
```

Figure 19: Apache Access Log Snapshot for New User Scenario

Following is a snapshot of the Windows Performance Monitor and Java Monitor Console during the load test execution:



Show	Color	Scale	Counter	Instance	Parent	Object	Computer
✓	—	0.000001	Virtual Bytes Peak	Tomcat7	---	Process	\\SSH-WIN
✓	—	0.000001	Virtual Bytes	Tomcat7	---	Process	\\SSH-WIN
✓	—	0.1	Page Faults/sec	Tomcat7	---	Process	\\SSH-WIN
✓	—	0.00001	Working Set Peak	Tomcat7	---	Process	\\SSH-WIN
✓	—	0.00001	Working Set	Tomcat7	---	Process	\\SSH-WIN
✓	—	0.000001	Page File Bytes Peak	Tomcat7	---	Process	\\SSH-WIN
✓	—	0.000001	Page File Bytes	Tomcat7	---	Process	\\SSH-WIN
✓	—	0.00001	Private Bytes	Tomcat7	---	Process	\\SSH-WIN
✓	—	1.0	Thread Count	Tomcat7	---	Process	\\SSH-WIN

Figure 20: Windows Performance Monitor Snapshot for New User Scenario

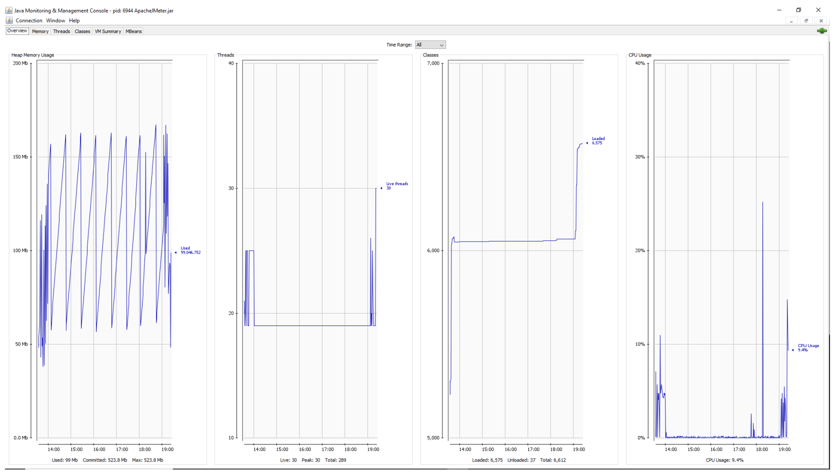

Java Monitoring & Management Console - pid: 6944 ApacheJMeter.jar
Connection Window Help
Overview Memory Threads Classes VM Summary MBeans
Time Range: All
Heap Memory Usage
Threads
Classes
CPU Usage
Used: 99 Mb Committed: 523.8 Mb Max: 523.8 Mb
Live: 30 Peak: 30 Total: 289
Loaded: 6,575 Unloaded: 37 Total: 6,612
CPU Usage: 9.4%

Figure 21: JConsole Snapshot for New User Scenario

Conclusion: The load test scenario for new user made about 7500 requests and the test ran for approximately 15 minutes. Despite some natural high spikes on the performance monitor and java monitor, the load test was carried out successfully without any crashes or unexpected behaviour in the application.