Data Flow Testing, Slice-Based Testing and Mutation Testing

EECS 4313: Software Engineering Testing
Assignment 3
April 09, 2018

Authors

Daniel McVicar (213027479)

Glib Sitiugin (213036165)

Nisha Sharma (213251830)

Rijul Aggarwal (212691523)

Varsha Ragavendran (213193065)

Table of Contents

List of Figures

1.0 Task 1 – Borg Calendar

1.1 Method Under Test

```java
/**
 * generate a human readable string for a particular number of minutes
 *
 * @param mins - the number of minutes
 *
 * @return the string
 */
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;
}
```

1.2 Method Segmentation

`public static String minuteString(int mins) {`	A
`    int hours = mins / 60;`	B
`    int minsPast = mins % 60;`	C
`    String minutesString;`	D
`    String hoursString;`	E
`    if (hours > 1)`	F
`        hoursString = hours + " " + Resource.getResourceString("Hours");`	G
`    else if (hours > 0)`	H
`        hoursString = hours + " " + Resource.getResourceString("Hour");`	I
`    else         hoursString = "";`	J
`    if (minsPast > 1)`	K
`        minutesString = minsPast + " " + Resource.getResourceString("Minutes");`	L
`else if (minsPast > 0)`	M
`        minutesString = minsPast + " " + Resource.getResourceString("Minute");`	N
`else if (hours >= 1)`	O
`        minutesString = "";`	P

`else         minutesString = minsPast + " " +`
`Resource.getResourceString("Minutes");`
	Q

`    if (!hoursString.equals("") && !minutesString.equals(""))`	R
`        minutesString = " " + minutesString;`	S
`    return hoursString + minutesString;`	T

1.3 Program Graph

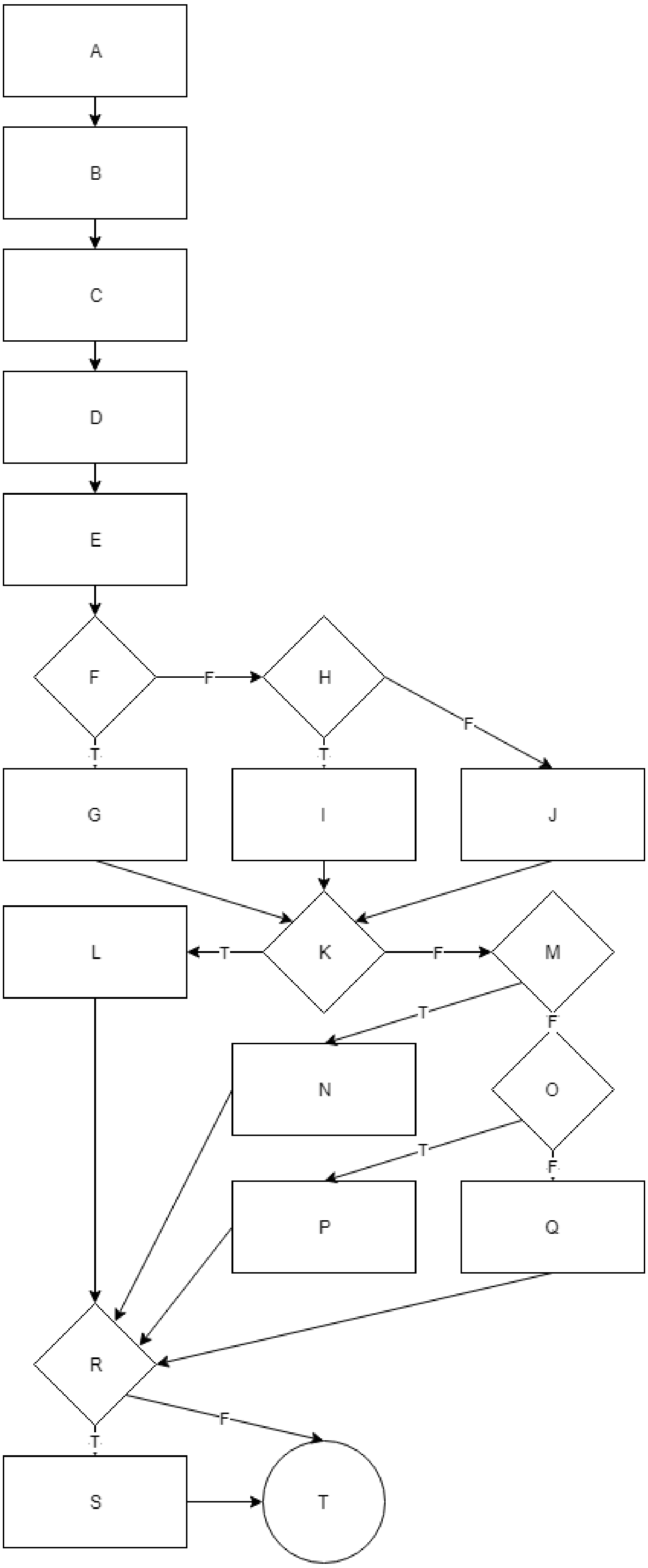


Figure # 1 : Program Graph of minuteString method

1.3.1 All-Defs

mins: AB
hours: BCDEF
minsPast: CDEFGK
minutesString: DEFGKL
hourString: EFG

1.3.2 All-Uses

mins: AB, ABC
hours: BCDEF, BCDEFG, BCDEFH, BCDEFHI, BCDEFHJKMO
minsPast: CDEFHJK, CDEFHJKL, CDEFHJKM, CDEFHJKMN, CDEFHJKMNOQ
minutesString: DEFHJKL, DEFHJKMN , DEFHJKMOP, DEFHJKMOQ, DEFHJKMOR, DEFHKMORS, ST
hourString: EFG, EFHI, EFHJ, EFHJKMOR, EFHJKMORT

1.3.3 All-P-Uses / Some-C-Uses

mins: AB, ABC
hours: BCDEF, BCDEFH, BCDEFHJKMO
minsPast: CDEFHJK, CDEFHJKM
minutesString:DEFHJKMOR
hourString: EFHJKMOR

1.3.4 All-C-Uses / Some-P-Uses

mins: AB, ABC
hours: BCDEFG, BCDEFHI
minsPast: CDEFHJKL, CDEFHJKMN, CDEFHJKMNOQ
minutesString: DEFHJKL, DEFHJKMN , DEFHKJMOP, DEFHJKMOQ, DEFHJKMORS, ST
hourString: EFG, EFHI, EFHJ, EFHJKMORT

1.4 Test Cases

```
/* All-Defs: AB
 * All-Uses: AB, ABC
 * All-P-Uses/Some-C-Uses:AB
 * All-C-Uses/Some-P-Uses: AB, ABC
 */
assertEquals("1 Minute", DateUtil.minuteString(1));

// All-Defs: BCDEF
assertEquals("2 Hours 30 Minutes", DateUtil.minuteString(150));

// All-Defs: CDEFHK
assertEquals("5 Minutes", DateUtil.minuteString(5));

// All-Defs: DEFHJKL
```

```
assertEquals("2 Minutes", DateUtil.minuteString(2));

// All-Defs: EFG
assertEquals("5 Hours", DateUtil.minuteString(300));
```

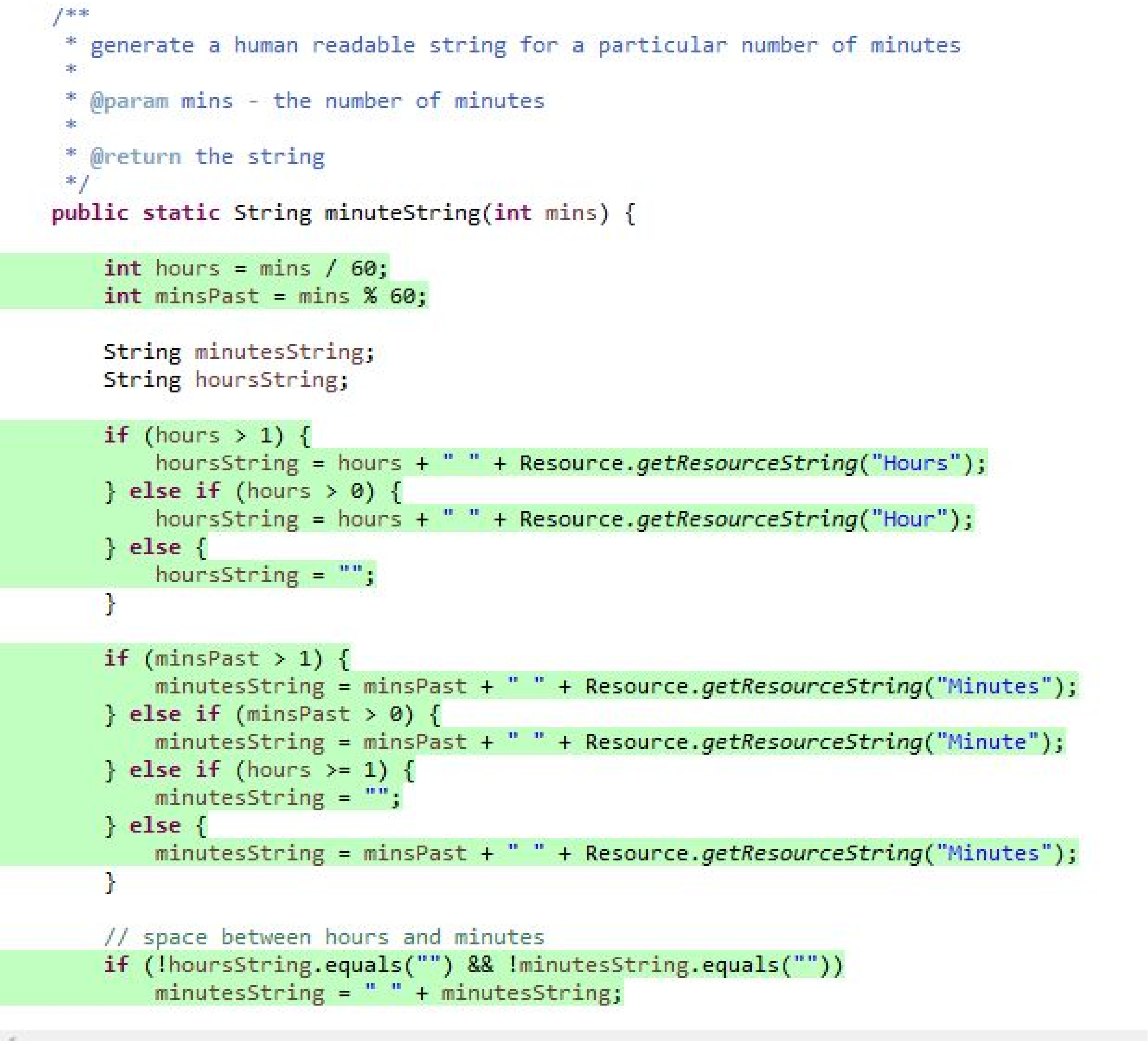
```
// All-Uses: BCDEF, BCDEFG, BCDEFH, BCDEFHI, BCDEFHJKMO
assertEquals("1 Hour 30 Minutes", DateUtil.minuteString(90));
assertEquals("2 Hours 30 Minutes", DateUtil.minuteString(150));
assertEquals("1 Minute", DateUtil.minuteString(1));

// All-Uses: CDEFHJK, CDEFHJKL, CDEFHJKM, CDEFHJKMN, CDEFHJKMNOQ
assertEquals("0 Minutes", DateUtil.minuteString(0));
assertEquals("1 Minute", DateUtil.minuteString(1));
assertEquals("2 Minutes", DateUtil.minuteString(2));

// All-Uses: DEFHJKL, DEFHJKMN , DEFHJKMOP, DEFHJKMOQ, DEFHJKMOR, DEFHKMORS, ST
assertEquals("1 Hour 1 Minute", DateUtil.minuteString(61));
assertEquals("1 Hour 2 Minutes", DateUtil.minuteString(62));
assertEquals("0 Minutes", DateUtil.minuteString(0));
assertEquals("1 Hour", DateUtil.minuteString(60));

// All-Uses: EFG, EFHI, EFHJ, EFHJKMOR, EFHJKMORT
assertEquals("2 Hours", DateUtil.minuteString(120));
assertEquals("1 Hour", DateUtil.minuteString(60));
assertEquals("0 Minutes", DateUtil.minuteString(0));
assertEquals("1 Hour 1 Minute", DateUtil.minuteString(61));
assertEquals("3 Hours 2 Minutes", DateUtil.minuteString(182));
```

```
// All-P-Uses/Some-C-Uses: BCDEFHJKMO
assertEquals("1 Hour", DateUtil.minuteString(60));

// All-P-Uses/Some-C-uses: CDEFHJK, CDEFHKM
assertEquals("15 Minutes", DateUtil.minuteString(15));
assertEquals("5 Hours 1 Minute", DateUtil.minuteString(301));

// All-P-Uses/Some-C-uses: DEFHJKMOR
assertEquals("7 Hours 1 Minute", DateUtil.minuteString(421));
assertEquals("1 Hour", DateUtil.minuteString(60));

// All-P-Uses/SomeC-Uses: EFHKMOR
assertEquals("7 Hours 1 Minute", DateUtil.minuteString(421));
assertEquals("1 Hour", DateUtil.minuteString(60));
```

```
// All-C-uses/Some-P-Uses: BCDEFG, BCDEFHI
assertEquals("1 Hour 2 Minutes", DateUtil.minuteString(62));
assertEquals("2 Hours 2 Minutes", DateUtil.minuteString(122));

// All-C-uses/Some-P-uses: CDEFHJKL, CDEFHJKMN, CDEFHJKMNOQ
assertEquals("1 Hour 10 Minutes", DateUtil.minuteString(70));
assertEquals("3 Hours 40 Minutes", DateUtil.minuteString(220));


// All-C-uses/Some-P-uses: DEFHJKL, DEFHJKMN , DEFHJKMOP, DEFHJKMOQ, DEFHJKMORS, ST
assertEquals("1 Hour 6 Minutes", DateUtil.minuteString(66));
assertEquals("2 Hours 1 Minute", DateUtil.minuteString(121));
assertEquals("1 Hour", DateUtil.minuteString(60));
assertEquals("0 Minutes", DateUtil.minuteString(0));
assertEquals("5 Minutes", DateUtil.minuteString(5));

// All-C-uses/Some-P-uses: EFG, EFHI, EFHJ, EFHJKMORT
assertEquals("1 Hour 6 Minutes", DateUtil.minuteString(66));
assertEquals("2 Hours 1 Minute", DateUtil.minuteString(121));
assertEquals("6 Minutes", DateUtil.minuteString(6));
```

1.5 Test Coverage Results

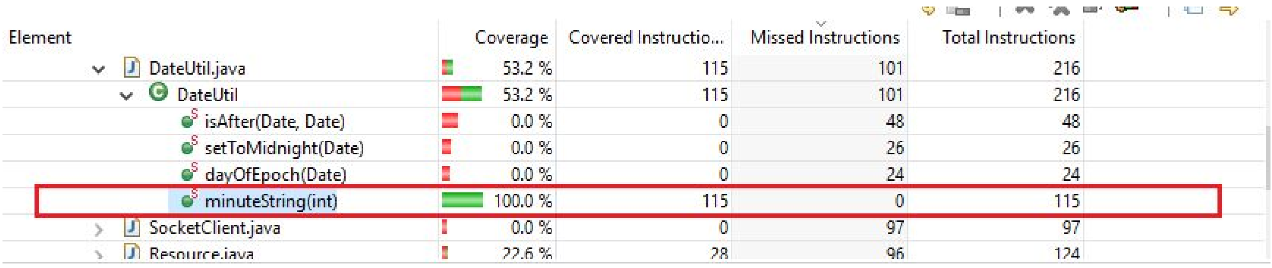
1.5.1 Summary



```
/**
 * generate a human readable string for a particular number of minutes
 *
 * @param mins - the number of minutes
 *
 * @return the string
 */
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;
```

Figure # 2 : Snapshot of minuteString method implementation

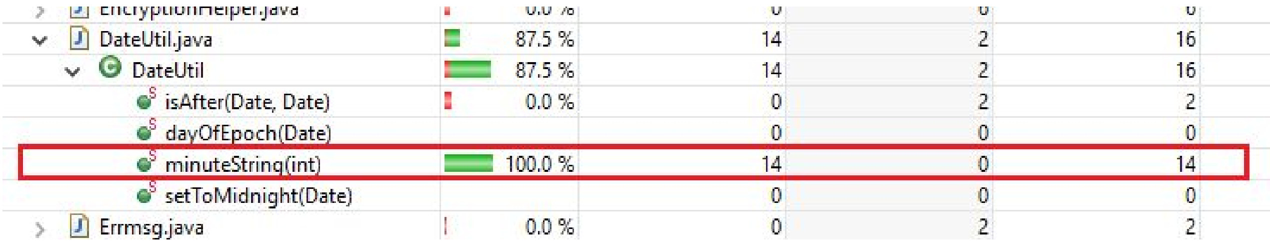
1.5.2 Instruction Coverage



Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
DateUtil.java	53.2 %	115	101	216
DateUtil	53.2 %	115	101	216
isAfter(Date, Date)	0.0 %	0	48	48
setToMidnight(Date)	0.0 %	0	26	26
dayOfEpoch(Date)	0.0 %	0	24	24
minuteString(int)	100.0 %	115	0	115
SocketClient.java	0.0 %	0	97	97
Resource.java	22.6 %	28	96	124

Figure # 3 : Instruction coverage of minuteString method in class DateUtil.java

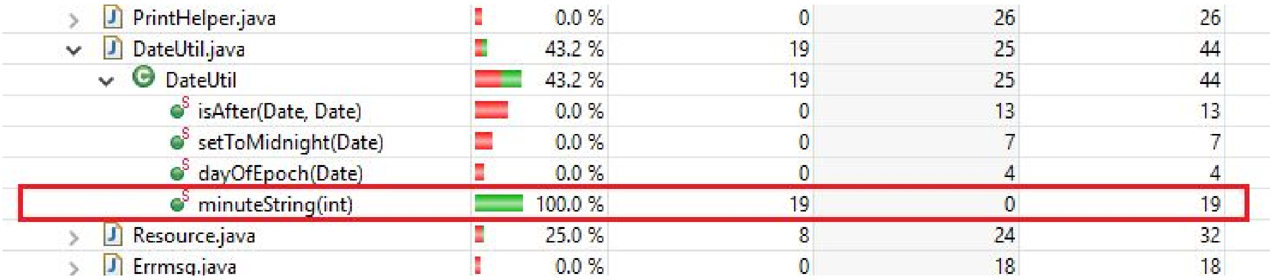
1.5.3 Branch Coverage



EncryptionHelper.java	0.0 %	0	0	0
DateUtil.java	87.5 %	14	2	16
DateUtil	87.5 %	14	2	16
isAfter(Date, Date)	0.0 %	0	2	2
dayOfEpoch(Date)		0	0	0
minuteString(int)	100.0 %	14	0	14
setToMidnight(Date)		0	0	0
Errmsg.java	0.0 %	0	2	2

Figure # 4 : Branch coverage of minuteString method in class DateUtil.java

1.5.4 Line Coverage



PrintHelper.java	0.0 %	0	26	26
DateUtil.java	43.2 %	19	25	44
DateUtil	43.2 %	19	25	44
isAfter(Date, Date)	0.0 %	0	13	13
setToMidnight(Date)	0.0 %	0	7	7
dayOfEpoch(Date)	0.0 %	0	4	4
minuteString(int)	100.0 %	19	0	19
Resource.java	25.0 %	8	24	32
Errmsg.java	0.0 %	0	18	18

Figure # 5 : Line coverage of minuteString method in class DateUtil.java

As can be seen, code coverage for the minuteString method is 100%. The coverage was 100% even before the data flow analysis as white box testing was performed for assignment 2. Regardless, test cases for data flow analysis and slicing were added to the existing suite for the test cases to ensure more rigorous test cases (see appendix).

1.6 Slices

1.6.1 Method Under Test

```
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;
}
```

1.6.2 Forward Slices

S(hours, 3)

```
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;
    }
```

S(minsPast, 4)

```
        int minsPast = mins % 60;

        String minutesString;

        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;
    }
```

S(minuteString, 6)

```
        String minutesString;

        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;
    }
```

S(hoursString, 7)

```
        String hoursString;

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

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

        return hoursString + minutesString;
    }
```

1.6.3 Backward Slices

S(hours, 3)

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

        int hours = mins / 60;

```

S(minsPast, 4)

```
        int minsPast = mins % 60;

```

S(minutesString, 18)

```

        String minutesString;

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

```

S(minutesString, 20)

```
        int minsPast = mins % 60;

        String minutesString;


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

S(minutesString, 22)

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

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

        String minutesString;

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

```

S(minutesString,24)

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

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

        String minutesString;

        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");
        }

```

S(minutesString, 29)

```
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;
    }
```

S(hoursString, 10)

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

        int hours = mins / 60;

        String hoursString;

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

```

S(hoursString, 12)

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

        int hours = mins / 60;

        String hoursString;

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

S(hoursString, 14)

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


        String hoursString;

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

```

1.6.4 Forward Slice Tests

```
// covers all forward slices
assertEquals("5 Hours 1 Minute", DateUtil.minuteString(301));
```

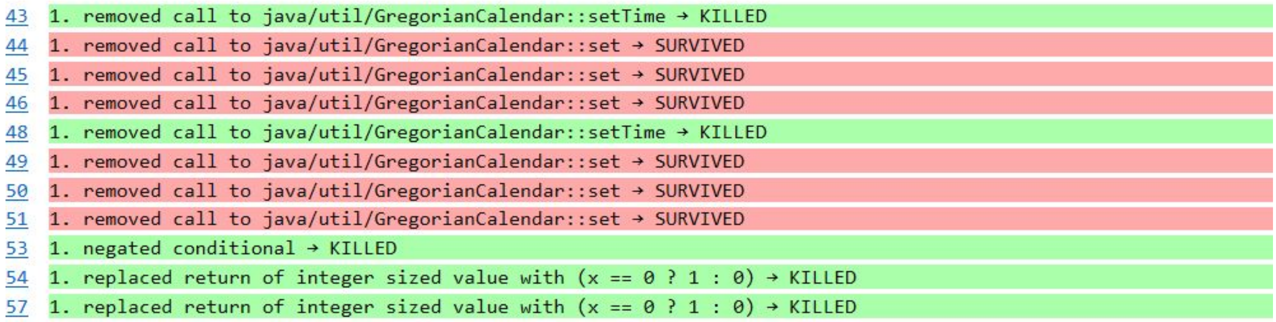
1.6.5 Backward Slice Tests

```
// Covers S(hours, 3), S(minsPast, 4), S(minutesString, 18)
assertEquals("5 Minutes", DateUtil.minuteString(5));

// Covers S(minutesString, 20)
assertEquals("1 Minute", DateUtil.minuteString(1));

// Covers S(minutesString, 22)
assertEquals("1 Hour", DateUtil.minuteString(60));

// Covers S(minutesString, 29)
assertEquals("0 Minutes", DateUtil.minuteString(0));

// Covers S(hoursString, 10)
assertEquals("0 Minutes", DateUtil.minuteString(0));

// Covers S(hoursString, 12)
assertEquals("5 Hours", DateUtil.minuteString(300));

// Covers S(hoursString, 14)
assertEquals("1 Hour", DateUtil.minuteString(60));

// Covers S(hoursString, 29)
assertEquals("5 Minutes", DateUtil.minuteString(5));
```

1.6.6 PIT Mutation Test

1.6.6.1 isAfter

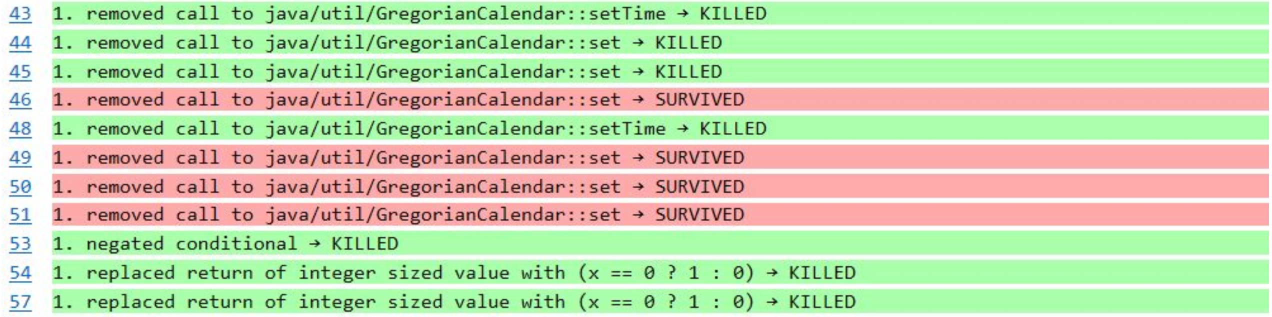
Before test addition



```
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 # 6 : Screenshot of PIT test before test addition of isAfter method

After test addition



```
43  1. removed call to java/util/GregorianCalendar::setTime → KILLED
44  1. removed call to java/util/GregorianCalendar::set → KILLED
45  1. removed call to java/util/GregorianCalendar::set → KILLED
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 # 7 : Screenshot of PIT test after test addition of isAfter method

There are 4 surviving mutants after addition of tests.The way the isAfter function is defined is that it sets the hours, minutes and seconds to zero before comparing. In addition, it sets the minutes for date 2 to be 10. These settings are not configurable/changeable.

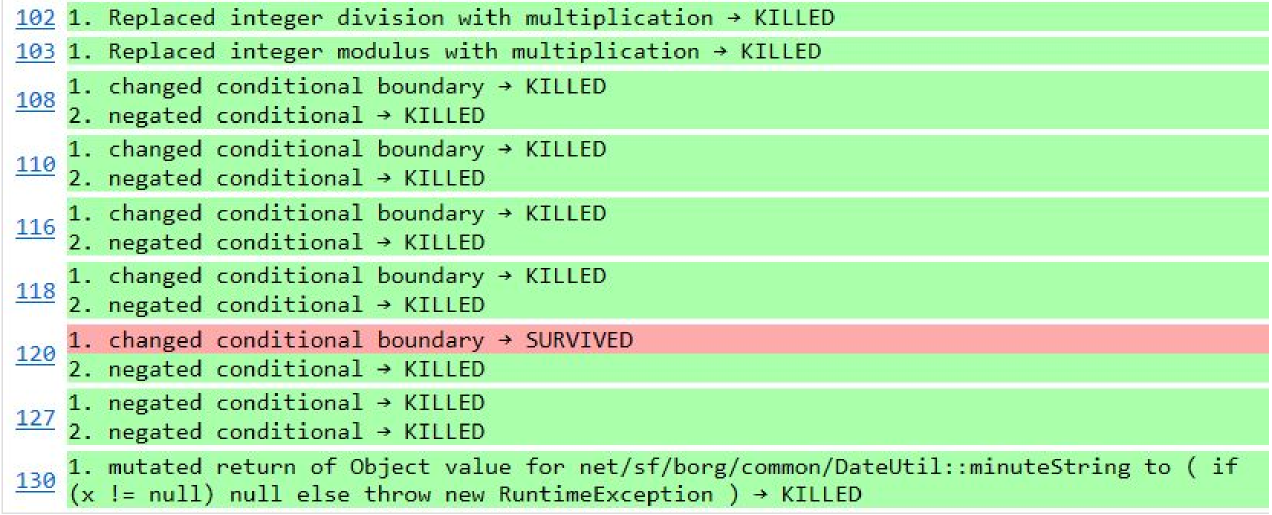
We can move date 1 forward in hours, minutes and seconds so that when comparison happens and the function is mutated to remove that respective header, it gives a different result than expected. 2 mutants (hour and minute setter) were killed, but the second can just be increased till 59 before it changes to a minute. Since date 2 already increases by 10 minutes, there is no way to exceed that in seconds. Hence this survives.

For the remaining 3 mutants, date 2 needs to have hour, minute, and second as negative values. Since it’s not allowed by the inherent Java Date API, those 3 mutants also survive.

To kill the 2 mutants, date 1 was increased by 1 hour and 20 mins respectively for additional 2 tests.

1.6.6.2 minuteString

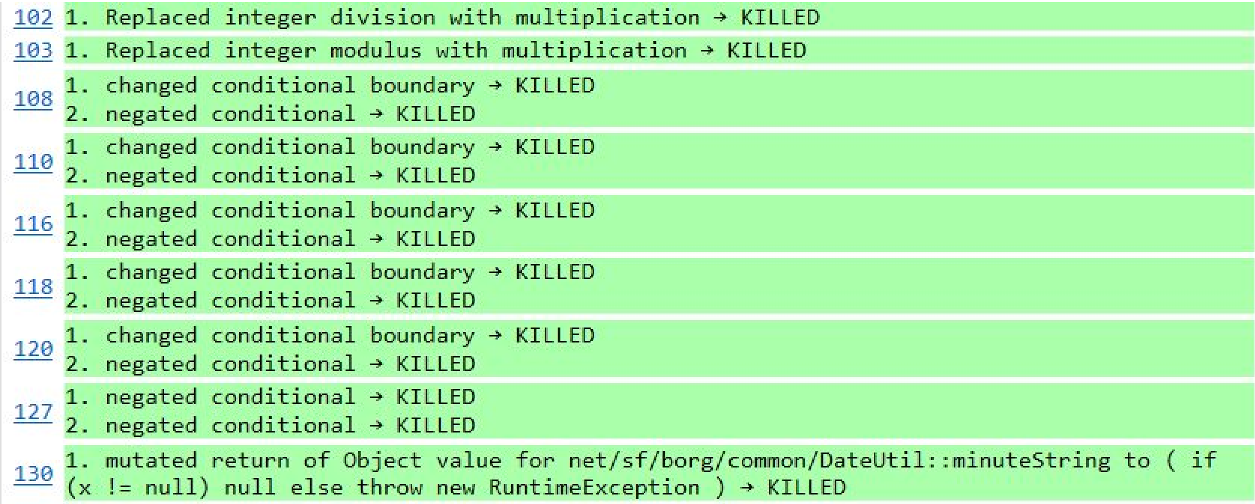
Before test addition



```
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 → SURVIVED
    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 # 8 : Screenshot of PIT test before test addition of minuteString method

After test addition



```
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
```

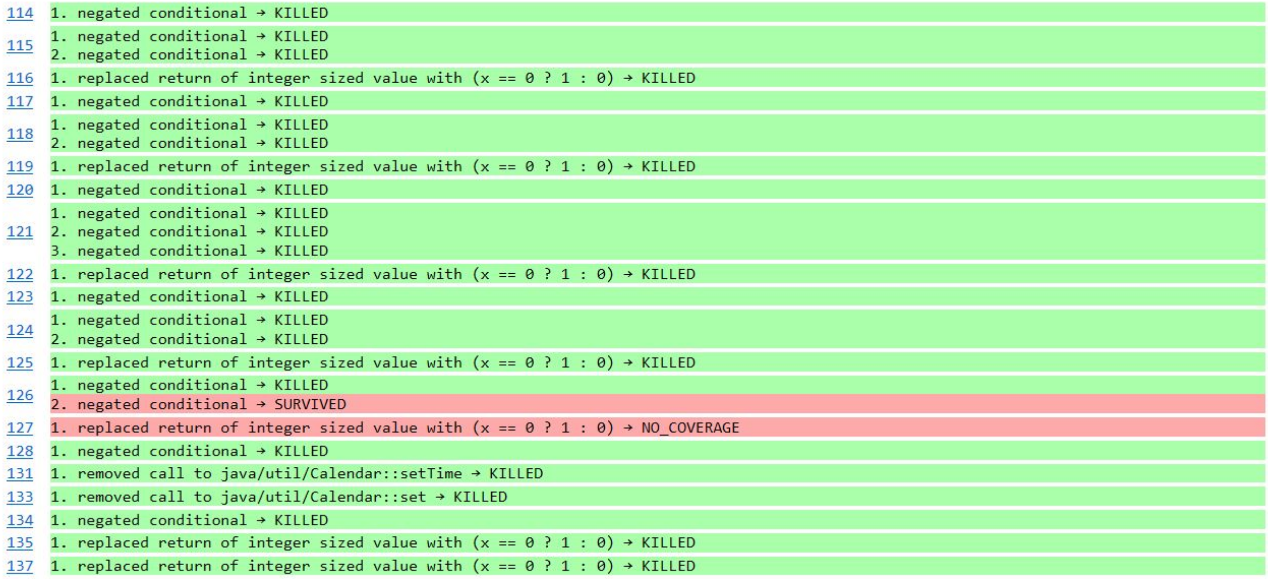
Figure # 9 : Screenshot of PIT test after test addition of minuteString method

As is evident, all mutants were killed for the specific method (minuteString) in DateUtil class. The added test cases were the decision flow and slicing tests.

Hence we can say the test cases kill all mutants.

1.6.6.3 isCompatible

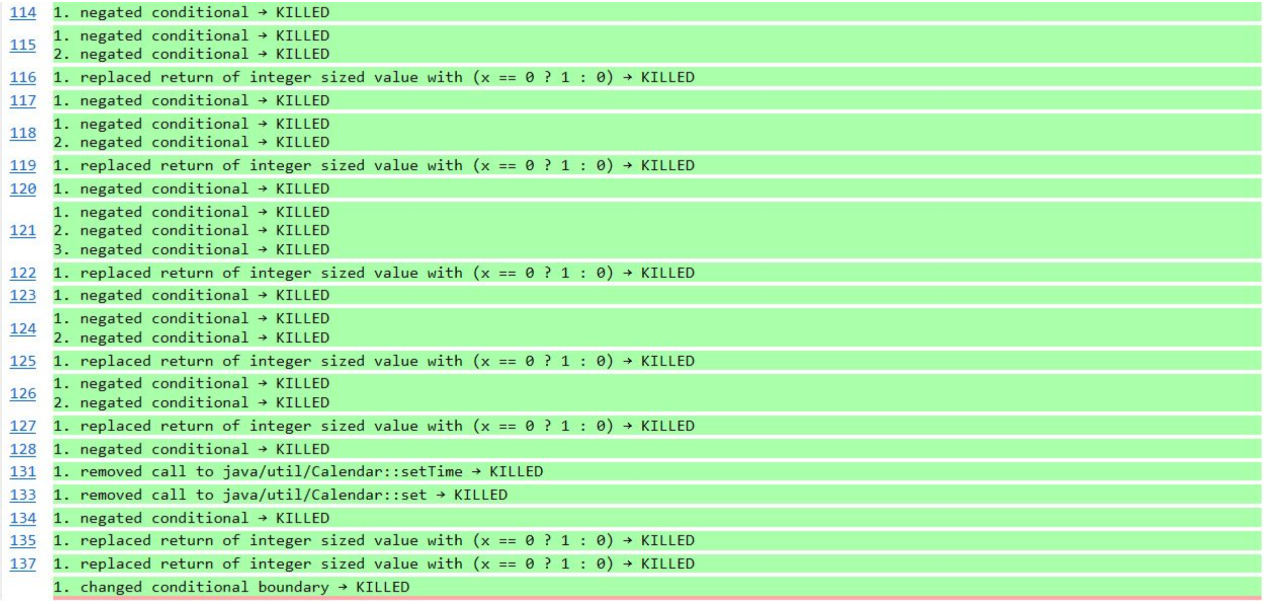
Before test addition



```
114 1. negated conditional → KILLED
115 1. negated conditional → KILLED
    2. negated conditional → KILLED
116 1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
117 1. negated conditional → KILLED
118 1. negated conditional → KILLED
    2. negated conditional → KILLED
119 1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
120 1. negated conditional → KILLED
121 1. negated conditional → KILLED
    2. negated conditional → KILLED
    3. negated conditional → KILLED
122 1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
123 1. negated conditional → KILLED
124 1. negated conditional → KILLED
    2. negated conditional → KILLED
125 1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
126 1. negated conditional → KILLED
    2. negated conditional → SURVIVED
127 1. replaced return of integer sized value with (x == 0 ? 1 : 0) → NO_COVERAGE
128 1. negated conditional → KILLED
131 1. removed call to java/util/Calendar::setTime → KILLED
133 1. removed call to java/util/Calendar::set → KILLED
134 1. negated conditional → KILLED
135 1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
137 1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Figure # 10 : Screenshot of PIT test before test addition of isCompatible method

After test addition



```
114  1. negated conditional → KILLED
115  1. negated conditional → KILLED
     2. negated conditional → KILLED
116  1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
117  1. negated conditional → KILLED
118  1. negated conditional → KILLED
     2. negated conditional → KILLED
119  1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
120  1. negated conditional → KILLED
     1. negated conditional → KILLED
121  2. negated conditional → KILLED
     3. negated conditional → KILLED
122  1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
123  1. negated conditional → KILLED
124  1. negated conditional → KILLED
     2. negated conditional → KILLED
125  1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
126  1. negated conditional → KILLED
     2. negated conditional → KILLED
127  1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
128  1. negated conditional → KILLED
131  1. removed call to java/util/Calendar::setTime → KILLED
133  1. removed call to java/util/Calendar::set → KILLED
134  1. negated conditional → KILLED
135  1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
137  1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
     1. changed conditional boundary → KILLED
```

Figure # 11 : Screenshot of PIT test after test addition of isCompatible method

As evident, the surviving mutant was killed by adding a test case that would evaluate the condition on line 126 as true and return false.

assertFalse(Repeat.isCompatible(new GregorianCalendar(2018, 4, 6), day_list_freq, daylist));

Hence we can say the test cases kill all mutants.

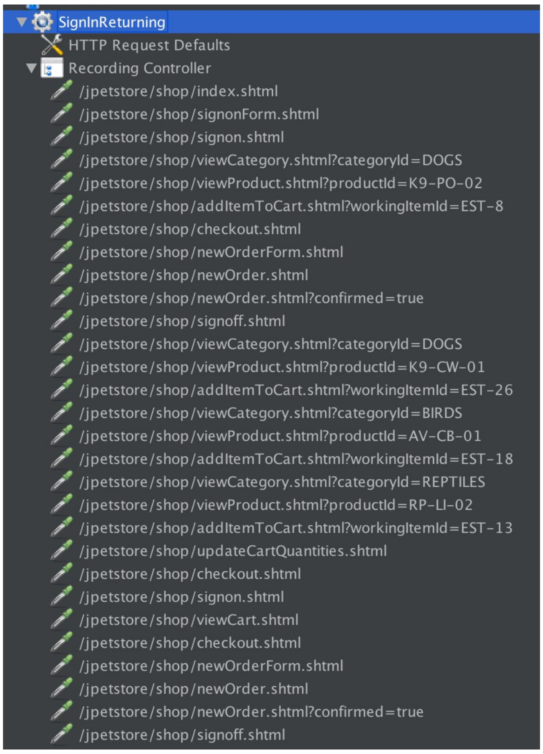
2.0 Task 2 – JPetStore

The following test scenarios were designed to measure the performance and perform load testing of the JPetStore e-commerce website. JMeter was used to carry out these tests.

2.1 Test Case Scenarios

2.1.1 Test Case #1 - Existing User Scenario

This test case investigates what happens when an existing user attempts to use the JPetStore system. The user is able to sign in, browse available products, select items and add them to their cart, update quantities in their cart, purchase everything in their cart, and sign out.



```
SignInReturning
  HTTP Request Defaults
  Recording Controller
    /jpetstore/shop/index.shtml
    /jpetstore/shop/signonForm.shtml
    /jpetstore/shop/signon.shtml
    /jpetstore/shop/viewCategory.shtml?categoryId=DOGS
    /jpetstore/shop/viewProduct.shtml?productId=K9-PO-02
    /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-8
    /jpetstore/shop/checkout.shtml
    /jpetstore/shop/newOrderForm.shtml
    /jpetstore/shop/newOrder.shtml
    /jpetstore/shop/newOrder.shtml?confirmed=true
    /jpetstore/shop/signoff.shtml
    /jpetstore/shop/viewCategory.shtml?categoryId=DOGS
    /jpetstore/shop/viewProduct.shtml?productId=K9-CW-01
    /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-26
    /jpetstore/shop/viewCategory.shtml?categoryId=BIRDS
    /jpetstore/shop/viewProduct.shtml?productId=AV-CB-01
    /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-18
    /jpetstore/shop/viewCategory.shtml?categoryId=REPTILES
    /jpetstore/shop/viewProduct.shtml?productId=RP-LI-02
    /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-13
    /jpetstore/shop/updateCartQuantities.shtml
    /jpetstore/shop/checkout.shtml
    /jpetstore/shop/signon.shtml
    /jpetstore/shop/viewCart.shtml
    /jpetstore/shop/checkout.shtml
    /jpetstore/shop/newOrderForm.shtml
    /jpetstore/shop/newOrder.shtml
    /jpetstore/shop/newOrder.shtml?confirmed=true
    /jpetstore/shop/signoff.shtml
```

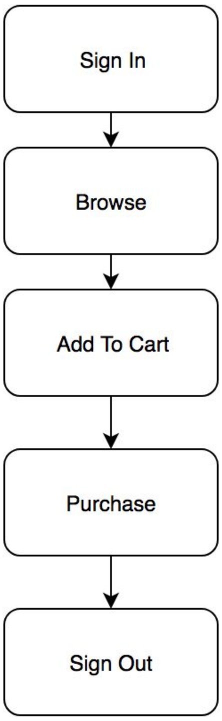
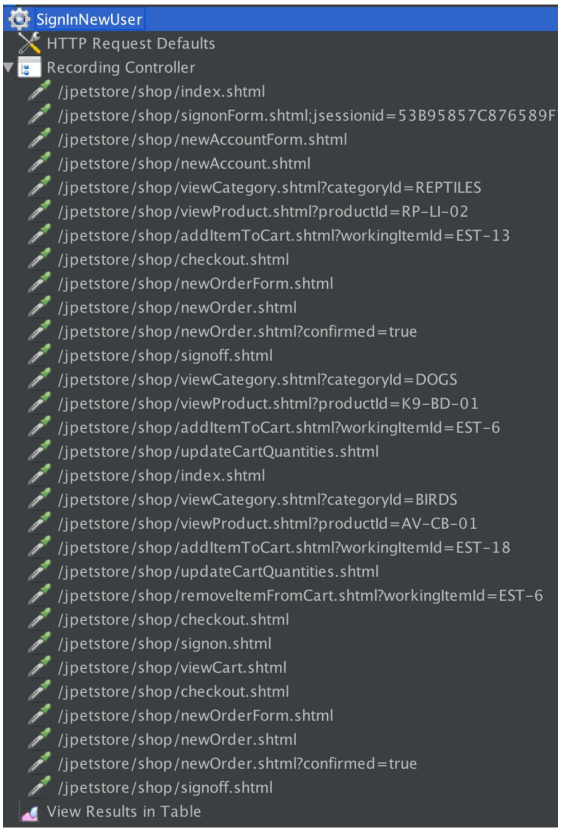


Figure # 12: Http Requests for Existing User Scenario and Use Case

2.1.2 Test Case #2 - New User Scenario

This case will investigate the use case of a new user to the JPetStore system. The user will have to create a new account before they are able to proceed to the storefront. Once an account has been created they will have the same options existing users have: they can browse available products, select items, add them to their cart, update quantities in their cart, purchase everything in their cart, and sign out.



```
SignInNewUser
  HTTP Request Defaults
  Recording Controller
    /jpetstore/shop/index.shtml
    /jpetstore/shop/signonForm.shtml;jsessionid=53B95857C876589F
    /jpetstore/shop/newAccountForm.shtml
    /jpetstore/shop/newAccount.shtml
    /jpetstore/shop/viewCategory.shtml?categoryId=REPTILES
    /jpetstore/shop/viewProduct.shtml?productId=RP-LI-02
    /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-13
    /jpetstore/shop/checkout.shtml
    /jpetstore/shop/newOrderForm.shtml
    /jpetstore/shop/newOrder.shtml
    /jpetstore/shop/newOrder.shtml?confirmed=true
    /jpetstore/shop/signoff.shtml
    /jpetstore/shop/viewCategory.shtml?categoryId=DOGS
    /jpetstore/shop/viewProduct.shtml?productId=K9-BD-01
    /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-6
    /jpetstore/shop/updateCartQuantities.shtml
    /jpetstore/shop/index.shtml
    /jpetstore/shop/viewCategory.shtml?categoryId=BIRDS
    /jpetstore/shop/viewProduct.shtml?productId=AV-CB-01
    /jpetstore/shop/addItemToCart.shtml?workingItemId=EST-18
    /jpetstore/shop/updateCartQuantities.shtml
    /jpetstore/shop/removeItemFromCart.shtml?workingItemId=EST-6
    /jpetstore/shop/checkout.shtml
    /jpetstore/shop/signon.shtml
    /jpetstore/shop/viewCart.shtml
    /jpetstore/shop/checkout.shtml
    /jpetstore/shop/newOrderForm.shtml
    /jpetstore/shop/newOrder.shtml
    /jpetstore/shop/newOrder.shtml?confirmed=true
    /jpetstore/shop/signoff.shtml
  View Results in Table
```

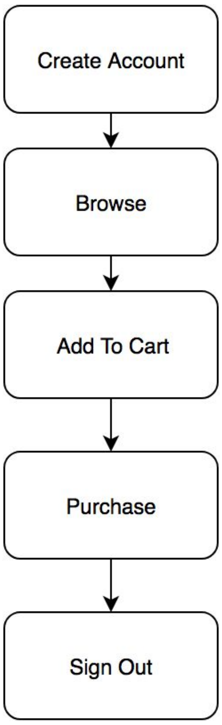
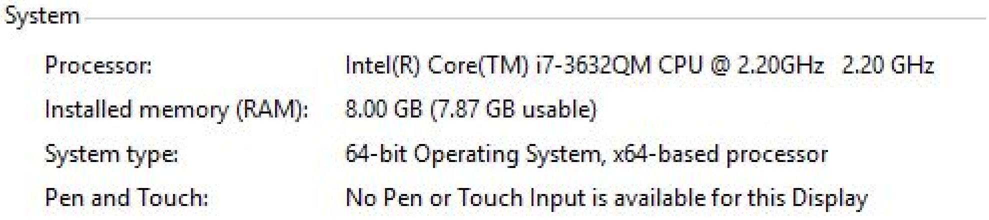


Figure # 13: Http Requests for New User Scenario and Use Case

2.2 Load Testing Results

Tests were run on windows machine with following configuration:



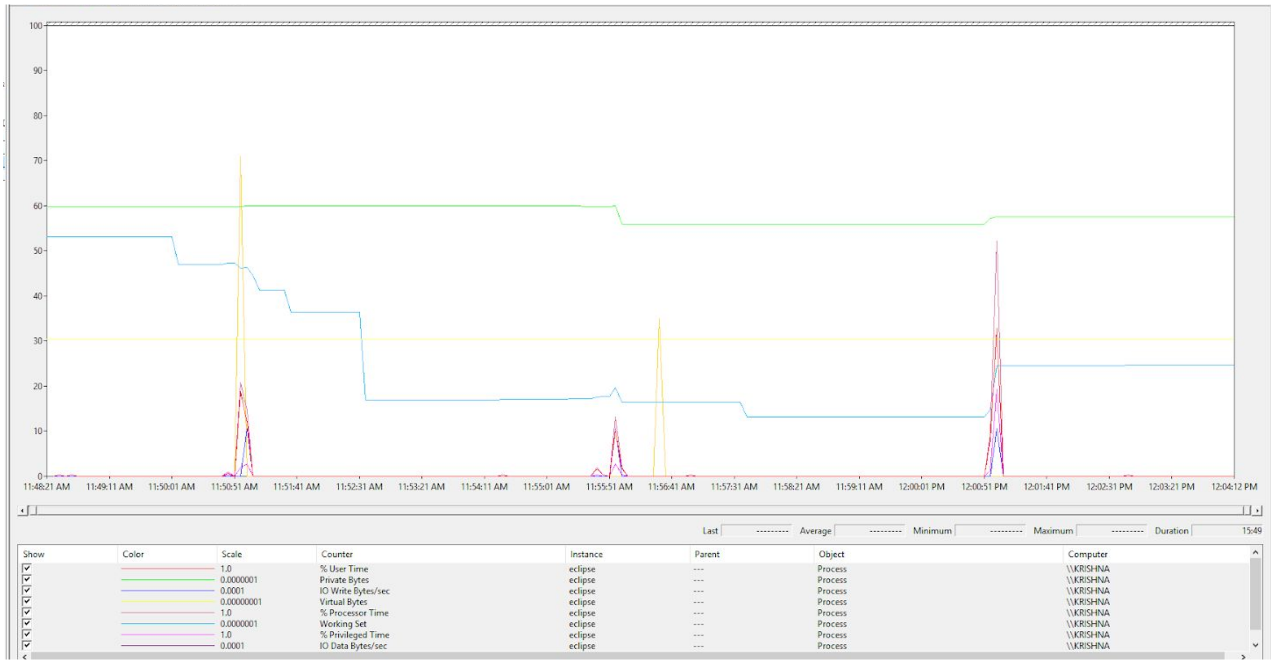
System

Processor:	Intel(R) Core(TM) i7-3632QM CPU @ 2.20GHz 2.20 GHz
Installed memory (RAM):	8.00 GB (7.87 GB usable)
System type:	64-bit Operating System, x64-based processor
Pen and Touch:	No Pen or Touch Input is available for this Display

Figure # 14: Machine Configuration

The load tests were run on the above 2 use cases on for 15 minutes each. For new user use cases the settings were 5 threads (users) with 1 ramp up and recurring loops until stopped at 15 minutes. For existing user scenario, the settings were 100 threads (users) with 1 ramp up and recurring loops until stopped at 15 minutes. There were no errors detected during the run and all of the requests were processed successfully indicating that the server system performed well. Screenshots from the respective runs are attached below with analysis.

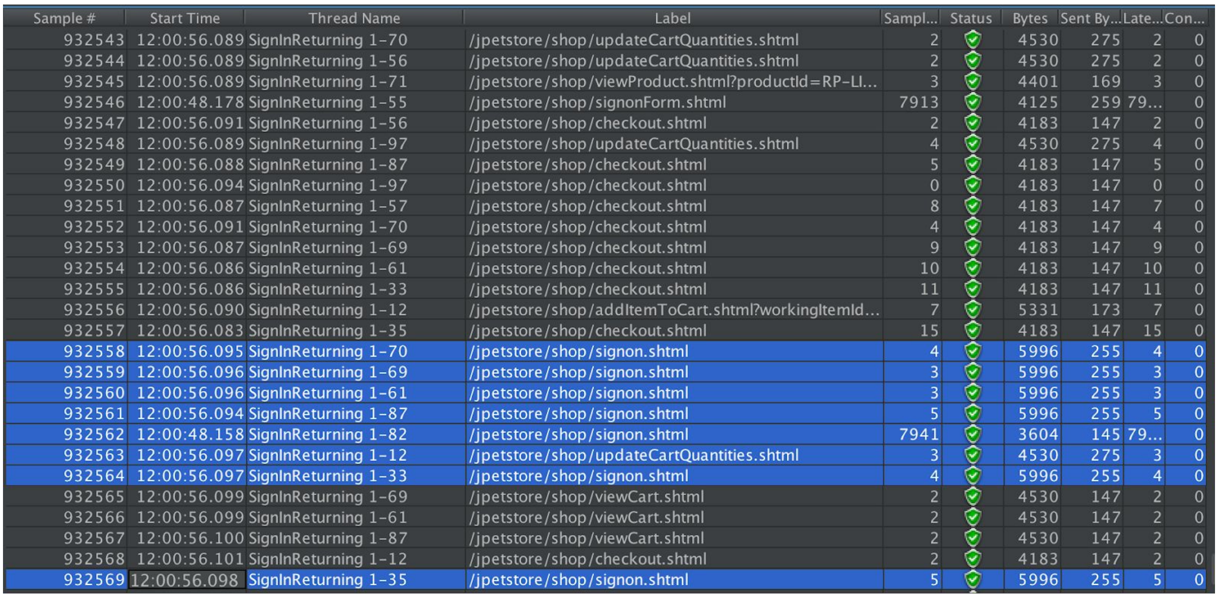
2.2.1 Test Case #1 - Existing User Scenario



Show	Color	Scale	Counter	Instance	Parent	Object	Computer
☑	🟥	1.0	% User Time	eclipse	---	Process	\\KRISHNA
☑	🟩	0.0000001	Private Bytes	eclipse	---	Process	\\KRISHNA
☑	🟦	0.0001	IO Write Bytes/sec	eclipse	---	Process	\\KRISHNA
☑	🟨	0.00000001	Virtual Bytes	eclipse	---	Process	\\KRISHNA
☑	🟥	1.0	% Processor Time	eclipse	---	Process	\\KRISHNA
☑	🟦	0.0000001	Working Set	eclipse	---	Process	\\KRISHNA
☑	🟪	1.0	% Privileged Time	eclipse	---	Process	\\KRISHNA
☑	🟪	0.0001	IO Data Bytes/sec	eclipse	---	Process	\\KRISHNA

Figure # 15: Performance metrics while executing Existing User test case scenario

There are several spikes in performance throughout the test. We will use the spike taking place between 12:00:51 PM and 12:01:41 PM as an example to investigate the cause of these spikes. Figure 16 shows the logs of the executed paths for the Existing User scenario. The highlighted requests show that an unusually large number of POST requests from different threads being executed at the same time. These requests require the database to accept incoming data and update itself, which requires more time than other requests on average. All of the performance spikes have a similar pattern: a higher than normal number of POST requests.

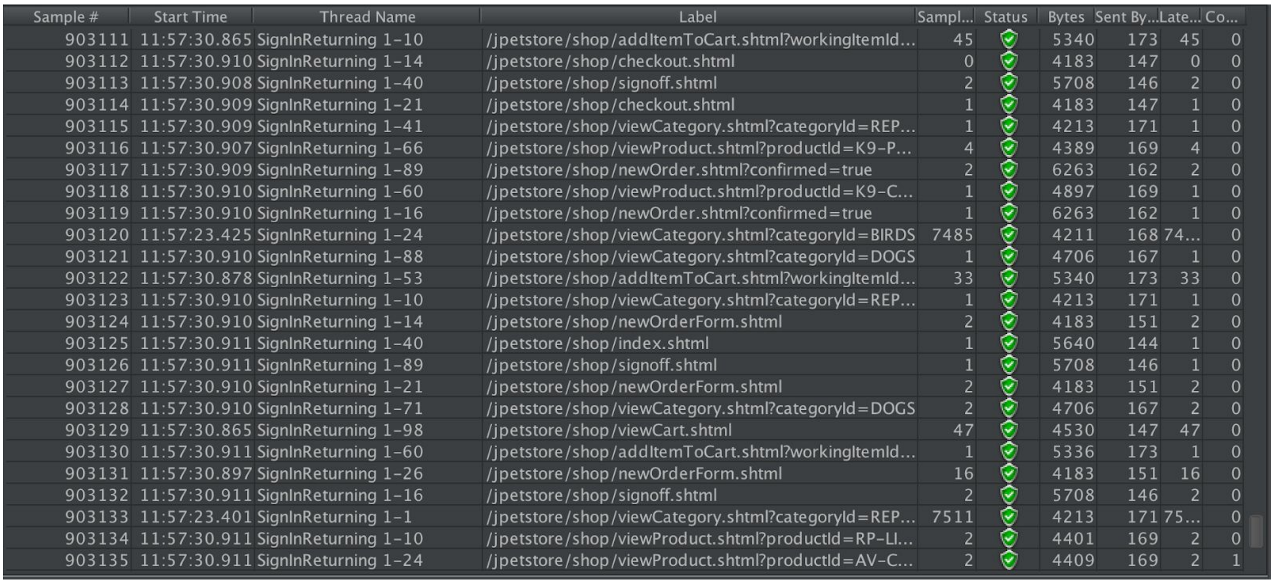


Sample #	Start Time	Thread Name	Label	Sampl...	Status	Bytes	Sent By...	Late...	Con...
932543	12:00:56.089	SignInReturning 1-70	/jpetstore/shop/updateCartQuantities.shtml	2	✓	4530	275	2	0
932544	12:00:56.089	SignInReturning 1-56	/jpetstore/shop/updateCartQuantities.shtml	2	✓	4530	275	2	0
932545	12:00:56.089	SignInReturning 1-71	/jpetstore/shop/viewProduct.shtml?productId=RP-LI...	3	✓	4401	169	3	0
932546	12:00:48.178	SignInReturning 1-55	/jpetstore/shop/signonForm.shtml	7913	✓	4125	259	79...	0
932547	12:00:56.091	SignInReturning 1-56	/jpetstore/shop/checkout.shtml	2	✓	4183	147	2	0
932548	12:00:56.089	SignInReturning 1-97	/jpetstore/shop/updateCartQuantities.shtml	4	✓	4530	275	4	0
932549	12:00:56.088	SignInReturning 1-87	/jpetstore/shop/checkout.shtml	5	✓	4183	147	5	0
932550	12:00:56.094	SignInReturning 1-97	/jpetstore/shop/checkout.shtml	0	✓	4183	147	0	0
932551	12:00:56.087	SignInReturning 1-57	/jpetstore/shop/checkout.shtml	8	✓	4183	147	7	0
932552	12:00:56.091	SignInReturning 1-70	/jpetstore/shop/checkout.shtml	4	✓	4183	147	4	0
932553	12:00:56.087	SignInReturning 1-69	/jpetstore/shop/checkout.shtml	9	✓	4183	147	9	0
932554	12:00:56.086	SignInReturning 1-61	/jpetstore/shop/checkout.shtml	10	✓	4183	147	10	0
932555	12:00:56.086	SignInReturning 1-33	/jpetstore/shop/checkout.shtml	11	✓	4183	147	11	0
932556	12:00:56.090	SignInReturning 1-12	/jpetstore/shop/addItemToCart.shtml?workingItemId...	7	✓	5331	173	7	0
932557	12:00:56.083	SignInReturning 1-35	/jpetstore/shop/checkout.shtml	15	✓	4183	147	15	0
932558	12:00:56.095	SignInReturning 1-70	/jpetstore/shop/signon.shtml	4	✓	5996	255	4	0
932559	12:00:56.096	SignInReturning 1-69	/jpetstore/shop/signon.shtml	3	✓	5996	255	3	0
932560	12:00:56.096	SignInReturning 1-61	/jpetstore/shop/signon.shtml	3	✓	5996	255	3	0
932561	12:00:56.094	SignInReturning 1-87	/jpetstore/shop/signon.shtml	5	✓	5996	255	5	0
932562	12:00:48.158	SignInReturning 1-82	/jpetstore/shop/signon.shtml	7941	✓	3604	145	79...	0
932563	12:00:56.097	SignInReturning 1-12	/jpetstore/shop/updateCartQuantities.shtml	3	✓	4530	275	3	0
932564	12:00:56.097	SignInReturning 1-33	/jpetstore/shop/signon.shtml	4	✓	5996	255	4	0
932565	12:00:56.099	SignInReturning 1-69	/jpetstore/shop/viewCart.shtml	2	✓	4530	147	2	0
932566	12:00:56.099	SignInReturning 1-61	/jpetstore/shop/viewCart.shtml	2	✓	4530	147	2	0
932567	12:00:56.100	SignInReturning 1-87	/jpetstore/shop/viewCart.shtml	2	✓	4530	147	2	0
932568	12:00:56.101	SignInReturning 1-12	/jpetstore/shop/checkout.shtml	2	✓	4183	147	2	0
932569	12:00:56.098	SignInReturning 1-35	/jpetstore/shop/signon.shtml	5	✓	5996	255	5	0

Figure # 16: Executed Paths in Existing User Scenario causing spike in performance

There are also long periods of time during the test where many requests are being processed but there are no spikes in performance. As an example, the time period 11:57:00 AM - 12:00:01 PM is shown below in Figure 17. During this time there are are a number GET requests being processed, but no POST requests. GET requests retrieve information but do not require any information to be updated. This means GET requests result in a much lower load on the server. Additionally, GET requests require fewer parameters to be passed to the server which results in a lower average packet size.

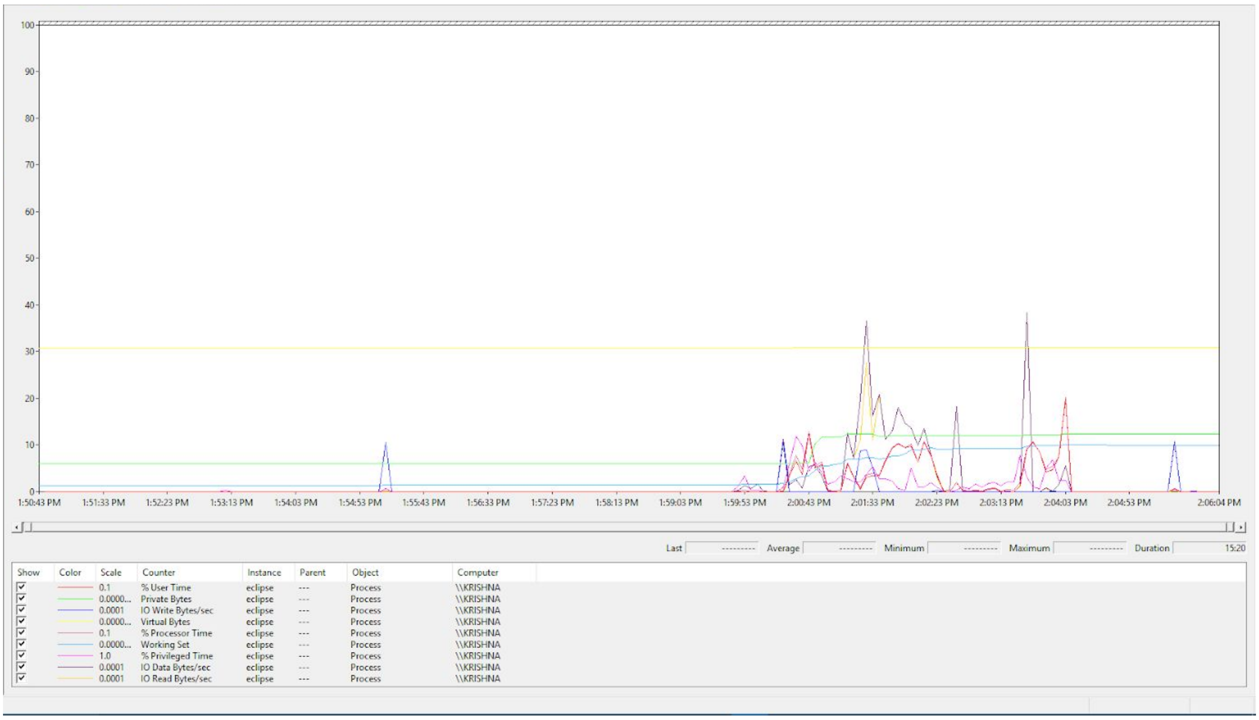
Overall the system performed well under load.



Sample #	Start Time	Thread Name	Label	Sampl...	Status	Bytes	Sent By...	Late...	Co...
903111	11:57:30.865	SignInReturning 1-10	/jpetstore/shop/addItemToCart.shtml?workingItemId...	45	✓	5340	173	45	0
903112	11:57:30.910	SignInReturning 1-14	/jpetstore/shop/checkout.shtml	0	✓	4183	147	0	0
903113	11:57:30.908	SignInReturning 1-40	/jpetstore/shop/signoff.shtml	2	✓	5708	146	2	0
903114	11:57:30.909	SignInReturning 1-21	/jpetstore/shop/checkout.shtml	1	✓	4183	147	1	0
903115	11:57:30.909	SignInReturning 1-41	/jpetstore/shop/viewCategory.shtml?categoryId=REP...	1	✓	4213	171	1	0
903116	11:57:30.907	SignInReturning 1-66	/jpetstore/shop/viewProduct.shtml?productId=K9-P...	4	✓	4389	169	4	0
903117	11:57:30.909	SignInReturning 1-89	/jpetstore/shop/newOrder.shtml?confirmed=true	2	✓	6263	162	2	0
903118	11:57:30.910	SignInReturning 1-60	/jpetstore/shop/viewProduct.shtml?productId=K9-C...	1	✓	4897	169	1	0
903119	11:57:30.910	SignInReturning 1-16	/jpetstore/shop/newOrder.shtml?confirmed=true	1	✓	6263	162	1	0
903120	11:57:23.425	SignInReturning 1-24	/jpetstore/shop/viewCategory.shtml?categoryId=BIRDS	7485	✓	4211	168	74...	0
903121	11:57:30.910	SignInReturning 1-88	/jpetstore/shop/viewCategory.shtml?categoryId=DOGS	1	✓	4706	167	1	0
903122	11:57:30.878	SignInReturning 1-53	/jpetstore/shop/addItemToCart.shtml?workingItemId...	33	✓	5340	173	33	0
903123	11:57:30.910	SignInReturning 1-10	/jpetstore/shop/viewCategory.shtml?categoryId=REP...	1	✓	4213	171	1	0
903124	11:57:30.910	SignInReturning 1-14	/jpetstore/shop/newOrderForm.shtml	2	✓	4183	151	2	0
903125	11:57:30.911	SignInReturning 1-40	/jpetstore/shop/index.shtml	1	✓	5640	144	1	0
903126	11:57:30.911	SignInReturning 1-89	/jpetstore/shop/signoff.shtml	1	✓	5708	146	1	0
903127	11:57:30.910	SignInReturning 1-21	/jpetstore/shop/newOrderForm.shtml	2	✓	4183	151	2	0
903128	11:57:30.910	SignInReturning 1-71	/jpetstore/shop/viewCategory.shtml?categoryId=DOGS	2	✓	4706	167	2	0
903129	11:57:30.865	SignInReturning 1-98	/jpetstore/shop/viewCart.shtml	47	✓	4530	147	47	0
903130	11:57:30.911	SignInReturning 1-60	/jpetstore/shop/addItemToCart.shtml?workingItemId...	1	✓	5336	173	1	0
903131	11:57:30.897	SignInReturning 1-26	/jpetstore/shop/newOrderForm.shtml	16	✓	4183	151	16	0
903132	11:57:30.911	SignInReturning 1-16	/jpetstore/shop/signoff.shtml	2	✓	5708	146	2	0
903133	11:57:23.401	SignInReturning 1-1	/jpetstore/shop/viewCategory.shtml?categoryId=REP...	7511	✓	4213	171	75...	0
903134	11:57:30.911	SignInReturning 1-10	/jpetstore/shop/viewProduct.shtml?productId=RP-LI...	2	✓	4401	169	2	0
903135	11:57:30.911	SignInReturning 1-24	/jpetstore/shop/viewProduct.shtml?productId=AV-C...	2	✓	4409	169	2	1

Figure # 17: Executed Paths in Existing User Scenario causing no spike in performance

2.2.2 Test Case #2 - New User Scenario

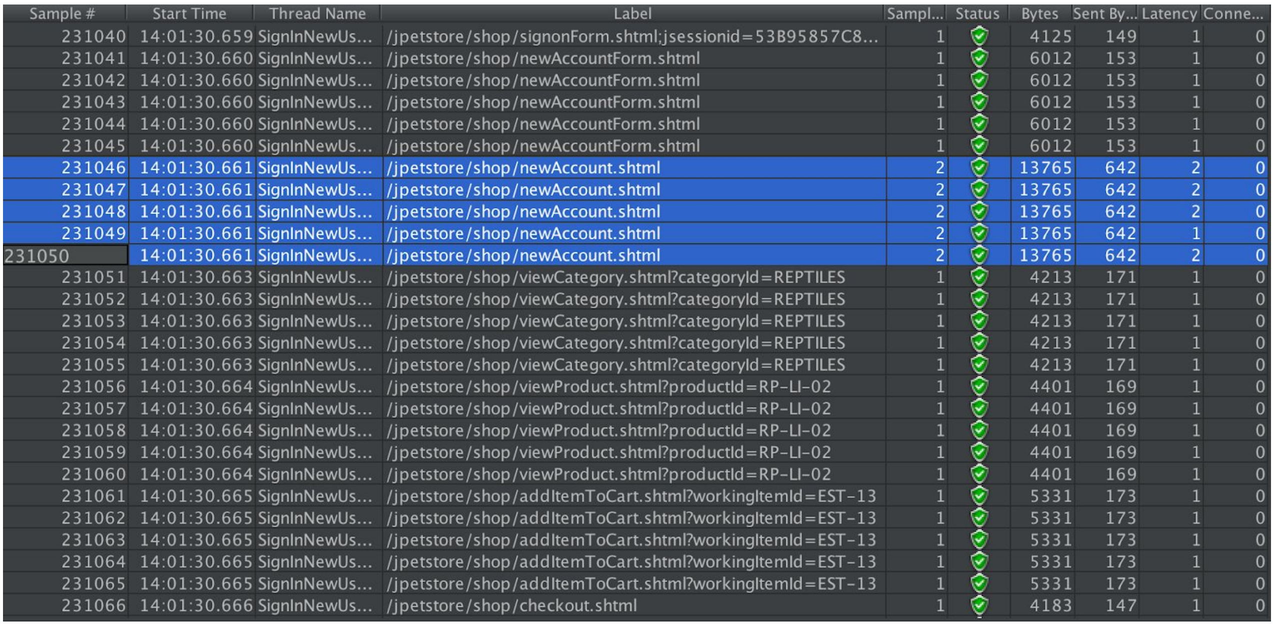


Last --------- Average --------- Minimum --------- Maximum --------- Duration 15:20

Show	Color	Scale	Counter	Instance	Parent	Object	Computer
☑	—	0.1	% User Time	eclipse	---	Process	\\KRISHNA
☑	—	0.0000...	Private Bytes	eclipse	---	Process	\\KRISHNA
☑	—	0.0001	IO Write Bytes/sec	eclipse	---	Process	\\KRISHNA
☑	—	0.0000...	Virtual Bytes	eclipse	---	Process	\\KRISHNA
☑	—	0.1	% Processor Time	eclipse	---	Process	\\KRISHNA
☑	—	0.0000...	Working Set	eclipse	---	Process	\\KRISHNA
☑	—	1.0	% Privileged Time	eclipse	---	Process	\\KRISHNA
☑	—	0.0001	IO Data Bytes/sec	eclipse	---	Process	\\KRISHNA
☑	—	0.0001	IO Read Bytes/sec	eclipse	---	Process	\\KRISHNA

Figure # 18: Performance metrics while executing New User test case scenario

There is some interesting activity approximately between 2:01:00 PM and 2:02:23 PM. Looking at Figure 19, at 2:01:30 PM there are multiple POST activities occurring simultaneously. The /jpetstore/shop/newAccount.shtml path in particular was executed by multiple threads (users) at the same time. This is a POST action, which requests our web server (apache-tomcat) to accept the data that was being sent as parameters enclosed in the body of the request message. This is why there is a spike in the performance monitor screenshot show in Figure 18.

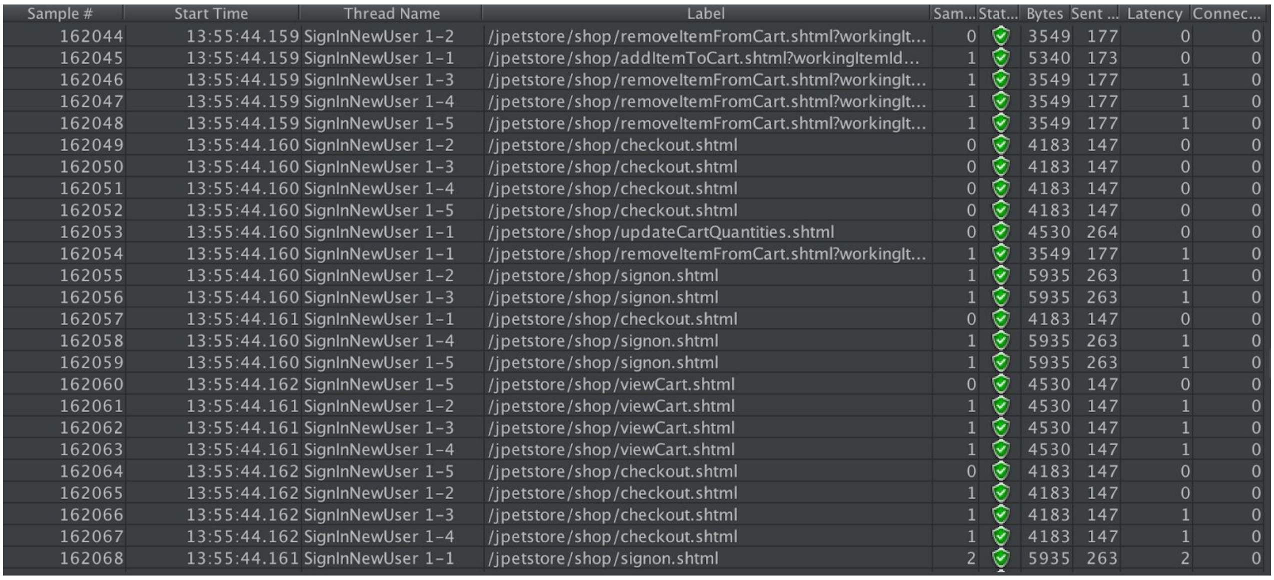


Sample #	Start Time	Thread Name	Label	Sampl...	Status	Bytes	Sent By...	Latency	Conne...
231040	14:01:30.659	SignInNewUs...	/jpetstore/shop/signonForm.shtml;jsessionid=53B95857C8...	1	✓	4125	149	1	0
231041	14:01:30.660	SignInNewUs...	/jpetstore/shop/newAccountForm.shtml	1	✓	6012	153	1	0
231042	14:01:30.660	SignInNewUs...	/jpetstore/shop/newAccountForm.shtml	1	✓	6012	153	1	0
231043	14:01:30.660	SignInNewUs...	/jpetstore/shop/newAccountForm.shtml	1	✓	6012	153	1	0
231044	14:01:30.660	SignInNewUs...	/jpetstore/shop/newAccountForm.shtml	1	✓	6012	153	1	0
231045	14:01:30.660	SignInNewUs...	/jpetstore/shop/newAccountForm.shtml	1	✓	6012	153	1	0
231046	14:01:30.661	SignInNewUs...	/jpetstore/shop/newAccount.shtml	2	✓	13765	642	2	0
231047	14:01:30.661	SignInNewUs...	/jpetstore/shop/newAccount.shtml	2	✓	13765	642	2	0
231048	14:01:30.661	SignInNewUs...	/jpetstore/shop/newAccount.shtml	2	✓	13765	642	2	0
231049	14:01:30.661	SignInNewUs...	/jpetstore/shop/newAccount.shtml	2	✓	13765	642	1	0
231050	14:01:30.661	SignInNewUs...	/jpetstore/shop/newAccount.shtml	2	✓	13765	642	2	0
231051	14:01:30.663	SignInNewUs...	/jpetstore/shop/viewCategory.shtml?categoryId=REPTILES	1	✓	4213	171	1	0
231052	14:01:30.663	SignInNewUs...	/jpetstore/shop/viewCategory.shtml?categoryId=REPTILES	1	✓	4213	171	1	0
231053	14:01:30.663	SignInNewUs...	/jpetstore/shop/viewCategory.shtml?categoryId=REPTILES	1	✓	4213	171	1	0
231054	14:01:30.663	SignInNewUs...	/jpetstore/shop/viewCategory.shtml?categoryId=REPTILES	1	✓	4213	171	1	0
231055	14:01:30.663	SignInNewUs...	/jpetstore/shop/viewCategory.shtml?categoryId=REPTILES	1	✓	4213	171	1	0
231056	14:01:30.664	SignInNewUs...	/jpetstore/shop/viewProduct.shtml?productId=RP-LI-02	1	✓	4401	169	1	0
231057	14:01:30.664	SignInNewUs...	/jpetstore/shop/viewProduct.shtml?productId=RP-LI-02	1	✓	4401	169	1	0
231058	14:01:30.664	SignInNewUs...	/jpetstore/shop/viewProduct.shtml?productId=RP-LI-02	1	✓	4401	169	1	0
231059	14:01:30.664	SignInNewUs...	/jpetstore/shop/viewProduct.shtml?productId=RP-LI-02	1	✓	4401	169	1	0
231060	14:01:30.664	SignInNewUs...	/jpetstore/shop/viewProduct.shtml?productId=RP-LI-02	1	✓	4401	169	1	0
231061	14:01:30.665	SignInNewUs...	/jpetstore/shop/addItemToCart.shtml?workingItemId=EST-13	1	✓	5331	173	1	0
231062	14:01:30.665	SignInNewUs...	/jpetstore/shop/addItemToCart.shtml?workingItemId=EST-13	1	✓	5331	173	1	0
231063	14:01:30.665	SignInNewUs...	/jpetstore/shop/addItemToCart.shtml?workingItemId=EST-13	1	✓	5331	173	1	0
231064	14:01:30.665	SignInNewUs...	/jpetstore/shop/addItemToCart.shtml?workingItemId=EST-13	1	✓	5331	173	1	0
231065	14:01:30.665	SignInNewUs...	/jpetstore/shop/addItemToCart.shtml?workingItemId=EST-13	1	✓	5331	173	1	0
231066	14:01:30.666	SignInNewUs...	/jpetstore/shop/checkout.shtml	1	✓	4183	147	1	0

Figure # 19: Executed Paths in New User Scenario causing spike in performance

There is a low level of activity between 1:55:43 PM - 1:59:03 PM. From Figure 20, we can see that the most common requests are GET requests. Very few POST requests are executed per millisecond (i.e /jpetstore/shop/addItemToCart.shtml @ 1:55:44.159, next at 1:55:44.160 /jpetstore/shop/updateCartQuantities.shtml). The previous case showed that POST requests have a significantly higher affect on server performance than GET requests. Because of this, there is no performance spike seen while executing these requests.

Overall the system performed well under load.



Sample #	Start Time	Thread Name	Label	Sam...	Stat...	Bytes	Sent ...	Latency	Connec...
162044	13:55:44.159	SignInNewUser 1-2	/jpetstore/shop/removeItemFromCart.shtml?workingIt...	0	✓	3549	177	0	0
162045	13:55:44.159	SignInNewUser 1-1	/jpetstore/shop/addItemToCart.shtml?workingItemId...	1	✓	5340	173	0	0
162046	13:55:44.159	SignInNewUser 1-3	/jpetstore/shop/removeItemFromCart.shtml?workingIt...	1	✓	3549	177	1	0
162047	13:55:44.159	SignInNewUser 1-4	/jpetstore/shop/removeItemFromCart.shtml?workingIt...	1	✓	3549	177	1	0
162048	13:55:44.159	SignInNewUser 1-5	/jpetstore/shop/removeItemFromCart.shtml?workingIt...	1	✓	3549	177	1	0
162049	13:55:44.160	SignInNewUser 1-2	/jpetstore/shop/checkout.shtml	0	✓	4183	147	0	0
162050	13:55:44.160	SignInNewUser 1-3	/jpetstore/shop/checkout.shtml	0	✓	4183	147	0	0
162051	13:55:44.160	SignInNewUser 1-4	/jpetstore/shop/checkout.shtml	0	✓	4183	147	0	0
162052	13:55:44.160	SignInNewUser 1-5	/jpetstore/shop/checkout.shtml	0	✓	4183	147	0	0
162053	13:55:44.160	SignInNewUser 1-1	/jpetstore/shop/updateCartQuantities.shtml	0	✓	4530	264	0	0
162054	13:55:44.160	SignInNewUser 1-1	/jpetstore/shop/removeItemFromCart.shtml?workingIt...	1	✓	3549	177	1	0
162055	13:55:44.160	SignInNewUser 1-2	/jpetstore/shop/signon.shtml	1	✓	5935	263	1	0
162056	13:55:44.160	SignInNewUser 1-3	/jpetstore/shop/signon.shtml	1	✓	5935	263	1	0
162057	13:55:44.161	SignInNewUser 1-1	/jpetstore/shop/checkout.shtml	0	✓	4183	147	0	0
162058	13:55:44.160	SignInNewUser 1-4	/jpetstore/shop/signon.shtml	1	✓	5935	263	1	0
162059	13:55:44.160	SignInNewUser 1-5	/jpetstore/shop/signon.shtml	1	✓	5935	263	1	0
162060	13:55:44.162	SignInNewUser 1-5	/jpetstore/shop/viewCart.shtml	0	✓	4530	147	0	0
162061	13:55:44.161	SignInNewUser 1-2	/jpetstore/shop/viewCart.shtml	1	✓	4530	147	1	0
162062	13:55:44.161	SignInNewUser 1-3	/jpetstore/shop/viewCart.shtml	1	✓	4530	147	1	0
162063	13:55:44.161	SignInNewUser 1-4	/jpetstore/shop/viewCart.shtml	1	✓	4530	147	1	0
162064	13:55:44.162	SignInNewUser 1-5	/jpetstore/shop/checkout.shtml	0	✓	4183	147	0	0
162065	13:55:44.162	SignInNewUser 1-2	/jpetstore/shop/checkout.shtml	1	✓	4183	147	0	0
162066	13:55:44.162	SignInNewUser 1-3	/jpetstore/shop/checkout.shtml	1	✓	4183	147	1	0
162067	13:55:44.162	SignInNewUser 1-4	/jpetstore/shop/checkout.shtml	1	✓	4183	147	1	0
162068	13:55:44.161	SignInNewUser 1-1	/jpetstore/shop/signon.shtml	2	✓	5935	263	2	0

Figure # 20: Executed Paths in New User Scenario causing no spike in performance

References

[1] "mikeberger/borg_calendar", *GitHub*, 2018. [Online]. Available: https://github.com/mikeberger/borg_calendar/tree/master/BORGCalendar. [Accessed: 01- Mar- 2018].

Appendix

Specification of the Selected Java Methods

Following are the specifications for three Java methods selected for the purpose of this assignment.

Selected method one

public static boolean isCompatible(Calendar date, String freq,Collection<Integer> daylist)

This method checks if the given date is valid with a certain repeating event frequency. If the date is valid for the given frequency, the function returns true. Otherwise, it returns false. Frequency strings are generated by providing an integer to the getFreqString(Int i) method. For our test the key frequencies will be:

- Weekday
- Weekend
- Monday, Wednesday, Friday
- Tuesday, Thursday
- Last day of the month.

And less importantly:

- Once, Daily, Weekly, Biweekly, Monthly, or Yearly.

Selected method two

public static boolean isAfter(Date d1, Date d2)

This method checks if one date falls on a later calendar day than another. The method takes two date objects as arguments. If the first date falls on a later day than second date, the method returns true otherwise it returns false. The method utilizes "after" method from "`java.util.Calendar`" class which compares two Calendar time objects; it returns whether this Calendar represents a time after the time represented by the specified Object.

Selected method three

public static String minuteString(int mins)

This method takes in the number of minutes as an integer and returns a human readable String representation of it in hours and minutes. This method expects non-negative integer value of minutes. The returned String takes care of singular or plural representation of hours or minutes in the returned String as well.