System and Unit Test Report

BMI Dashboard, Team Dash, December 3, 2015

User Stories for Sprint 1

A. Parsing Data - As a developer, I need to convert the data to JSON or CSV or TSV so that I can use visualization tools to showcase values.

Satisfied by Test Case 2 in Data Acquisition

B. Storing into our local database - As a developer, I need to have a database system so I can store data as efficiently as possible.

Satisfied by Test Case 4 in Data Storage

Scenario:

1. start BMIDashboard app using following command:
    a. $ python -m SimpleHTTPServer 8080
    b. Open browser, refresh if need to
2. User should see static PV data graph with x-axis as time and y-axis as electricity generation.

User Stories for Sprint 2

A. Establish project functionality - As a developer, I need to display data in a simple manner to the visitor.

Satisfied by Test Case 1 in Data Visualization

B. Viewing previous data - As an exhibitor, I need to be able to see energy consumption of previous days so that I can compare different days.

Satisfied by Test Case 1 in Data Visualization and Test Cases 5 and 6 in Data Storage

C. Communicate with database - As a developer I need to learn MySQL so that I can connect our database to our web interface, through the use of visualization tools.

Satisfied by Test Case 4, 5 and 6 in Data Storage

Scenario:

1. start BMIDashboard app;
2. select ‘‘Temperature’;
3. User should see static temperature graph based on selection with x-axis as time and y-axis as degrees.

User Stories for Sprint 3

A. Connect Pipeline - As a developer, I need to connect data acquisition, insertion, and visualization together so that I can have a functional dashboard.

Satisfied through all Test Cases from Data Storage, Acquisition and Visualization

B. BMS Database Insertion - As a developer, I need to insert BMS data into our database and ensure that there are no duplicates and be able to access data from the web interface.

Satisfied by Test Case 4 in Data Storage

C. User Interface - As a user, I need to have a user interface so that I can easily use the dashboard.

Satisfied by Test Case 1 in Data Visualization

Scenario:

1. start BMIDashboard app;
2. select ‘2015-05-01’;
3. select ‘Temperature’;
4. User should see live temperature data with x-axis as time and y-axis as degrees and graph will update automatically every ten seconds.

A. PV Database Insertion - As a developer, I need to get rid of duplicate entries from the database so that we don’t have repeated data for the PV.

Scenario:

1. start BMIDashboard app;
2. select ‘PV’;
3. User should see live energy generation data with x-axis as time and y-axis as watts and graph will update automatically every ten seconds.

See Testing file for Unit Tests (test cases).

More System Testing

Test Case 1: Directories

The program looks in specific directories, with specific names to find files

```
Success:
       The program should find the specified directories

Fail:
       Crash with path error
```

Test Case 2: No Files

The program looks in the specified directory for .csv files

```
Success:
       The program should find files in the specified directories

Fail:
       Crash with file descriptor error
```

Test Case 3: File format

The programs in the specified directory have been cleaned (.csv) and are ESO files

```
Success:
       The program should find .csv files in the specified directories

Fail:
       Will not parse through the files properly if not ESO files,
       if not .csv the program will not parse
```

Test Case 4: No output Directories

The program looks for specified directories to output parsed files

```
Success:
	The program should find specified directories

Fail:
	Crash with path error
```

IF

1. the specified directories for input and output are there and properly labeled
2. there are files in the input directories
    a. they are .csv files
    b. they are ESO files (proper data)

THEN

The program should run properly

Inserting to database from CSV

Note: this applies to the functions insertToDB_BMS_R1IO, insertToDB_BMS_R2(), and insertToDB_PV(). All three functions require the same tests and throw the same exceptions.

Test Case 1: No entries in database.

```
Success:
	Should insert into the database without any problems. The functions do not have to
	check for the case of multiple entries into the database.

Fail:
	The database should remain empty since the entry was not inserted.
```

Test Case 2: Entry or Entries in database.

```
Success:
	Should insert into the database without any problems, unless the entry we are trying
	to insert is an entry already in the database.The functions should recognize
	duplicate entries and not insert that specific line of data again.

Fail: There should be duplicate entries of whatever data entry was not found using the query
check.
```

Printing from database to CSV

Note: This applies to printFromDB_PV, printFromDB_BMS_Combined, and printFromDB_BMS. All three functions require the same tests and throws the same exceptions when it fails.

Test Case 1: No Entries in database

```
Success:
	Should print out only title row at the beginning of the file and nothing else, since
	nothing is in the database. Loop will not run, therefore, nothing is printed.

Fail:
	An exception is thrown and the program quits.
```

Test Case 2: 1 Entry in database

```
Success:
	File will be created. Title row will be printed. Loop will run once to grab the data
	and print to CSV.

Fail:
	An exception is thrown and the program quits.
```

Test Case 3: Multiple Entries in database

```
Success:
	File will be created. Title row will be printed. Loop will run to grab the data from
	the database and print to CSV until the database returns null.

Fail:
	An exception is thrown and the program quits.
```

printfromDB_BMS Specific Test Case: Four entries in column array

```
Success:
	For loop will run four times. Once for each array index/column. One file will be
	generated per loop for a total of four files.
```

```
Fail:
       ArrayIndexOutOfBounds Exception is thrown and the program quits.
```

Data Display

The following tests are done by Venkata Karthik Thota for the display module.

Test Cases for Launching the Website.

Test Case 1: Localhost 8888

Start localhost 8888 by typing in Terminal:

```
$ python -m SimpleHTTPServer 8888 &
$ open http://localhost:8888
```

```
Success:
       Should see a webpage that is running/rendered from index.html. If you see a network
       error prompt, then wait a few seconds or refresh the page.

Fail:
       A network error is prompted to the user
```

Test Case 2: http://kthotav.github.io/CMPS115/

```
Success:
       Should see a webpage that is running/rendered from index.html.

Fail:
       A network error is prompted to the user. GitHub Pages server is at fault.
```

Test Cases for Display Localhost and GitHub Pages

Test Case 1: Clicking on Dates on the navigation bar on any page.


```
Success:
       Returns to the homepage index.html

Fail:
       Localhost Error
       Error response
       Error code 404.
       Message: File not found.
       Error code explanation: 404 = Nothing matches the given URI.

       OR

       GitHub Pages Error
       404
       File not found
       The site configured at this address does not contain the requested file.

       If this is your site, make sure that the filename case matches the URL.

       For root URLs (like http://example.com/) you must provide an index.html file.

       Read the full documentation for more information about using GitHub Pages.
```

Test Case 2: Clicking on BMI Dashboard on the navigation bar on any page.


```
Success:
       Returns to the homepage index.html

Fail:
       Localhost Error
       Error response
       Error code 404.
       Message: File not found.
       Error code explanation: 404 = Nothing matches the given URI.

       OR


       GitHub Pages Error
```

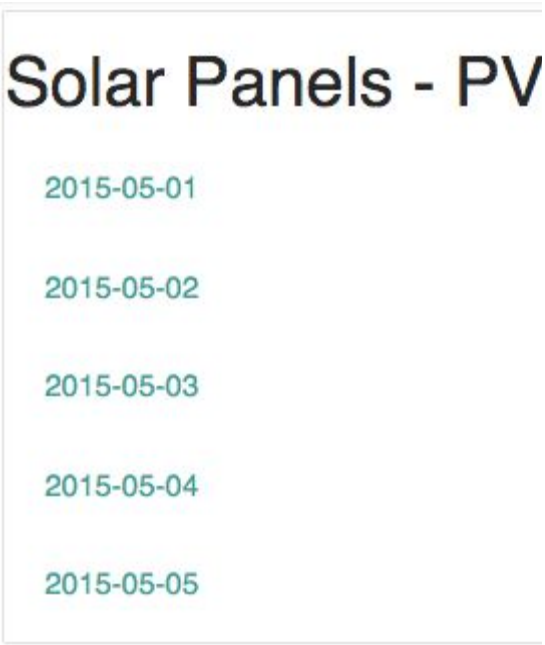
404
File not found
The site configured at this address does not contain the requested file.

If this is your site, make sure that the filename case matches the URL.

For root URLs (like http://example.com/) you must provide an index.html file.

Read the full documentation for more information about using **GitHub Pages**.

Test Case 3: Clicking on any one of the dates listed under Solar Panels (PV)




```
Success:
       Returns the page pv1.html for 2015-05-01 |
       Returns the page pv2.html for 2015-05-02 |
       Returns the page pv3.html for 2015-05-03 |
       Returns the page pv4.html for 2015-05-04 |
       Returns the page pv5.html for 2015-05-05

Fail:
       Localhost Error
       Error response
       Error code 404.
       Message: File not found.
       Error code explanation: 404 = Nothing matches the given URI.

       OR

       GitHub Pages Error
       404
       File not found
```

The site configured at this address does not contain the requested file.

If this is your site, make sure that the filename case matches the URL.

For root URLs (like http://example.com/) you must provide an index.html file.

Read the full documentation for more information about using **GitHub Pages**.

Test Case 4: Click on the TEMPERATURE, PAC, and VAC buttons in any one of the five pv*.html files.


```
Success:
        Returns a line chart different from the previous graph. The y-axis is updated to the
        attribute that is clicked on.

Fail:
        Localhost Error
        The graph does not update or display the line. Either the chart remains with no line
        plot, D3 error or network error

        OR

        GitHub Pages Error
        The graph does not update or display the line. Either the chart remains with no line
        plot, D3 error or network error.
```

Test Case 5: Click on the TEMPERATURE, RELATIVE HUMIDITY, CO 2, and SENSIBLE HEAT buttons in any one of the five bms*.html files.


```
Success:
       Returns a line chart different from the previous graph. The y-axis is updated to the
       attribute that is clicked on.

Fail:
       Localhost Error
       The graph does not update or display the line. Either the chart remains with no line
       plot, D3 error or network error

       OR

       GitHub Pages Error
       The graph does not update or display the line. Either the chart remains with no line
       plot, D3 error or network error
```

Test Cases for pv*.html files and bms1.html files

Test Case 1: PV (Solar Panels) and BMS Views are rendered properly.

```
Success:
       Returns the default line chart for Temperature attribute.

Fail:
       Localhost Error
       Empty/blank page is displayed. Possibility of Either the chart remains with no line
       plot, D3 error or network error

       OR

       GitHub Pages Error
       The graph does not update or display the line. Either the chart remains with no line
```

```
        plot, D3 error or network error
```

Test Case 2: Loading D3.js

```
Success:

Fail:
        Localhost Error

        Uncaught ReferenceError: d3 is not defined


        OR

        GitHub Pages Error

        Uncaught ReferenceError: d3 is not defined
```

Test Cases for pvGraph*.html files and graphBMS*.html files

Test Case 1: D3 functions

```
Success:

Fail:
        Localhost Error

        Uncaught ReferenceError: <variable name> is not defined.

        OR

        GitHub Pages Error

        Uncaught ReferenceError: <variable name> is not defined.
```

Test 2: Loading data (CSV files).

```
Success:

Fail:
        Localhost Error

        GET http://localhost:8888/BSOne.csv 404 (File not found) |

        OR

        GitHub Pages Error

        GET <web address>BSOne.csv 404 (File not found)
```

Test 3: CSV Variables.