**TCSS 360 Project State Document**

**Team: Users**

**Members:**

**Anh Nguyen (**[**anh2@uw.edu**](mailto:anh2@uw.edu)**)**

**Chris Kim (**[**cjkim00@uw.edu**](mailto:cjkim00@uw.edu)**)**

**Matthew Hwang (**[**matt95@uw.edu**](mailto:matt95@uw.edu)**)**

**Parker Olive (**[**olivep3@uw.edu**](mailto:olivep3@uw.edu)**)**

**Tyler Shupack (**[**tylers27@uw.edu**](mailto:tylers27@uw.edu)**)**

**Website:**

<https://rawgit.com/Zydico/TCSS360-Website/master/index.html>

**Table of Contents**

Introduction….…………………………………………………………. 3

Requirements Addressed………….……………………………………. 4

How to Run uDIY………………………………………………………. 6

Contributions……………………………………………………………. 7

Builds and Tests………………….……………………………………… 8

Source Catalog……………………………..……………………………. 9

Emergency Contacts…………………………………………………...… 10

**Introduction**

As we approach the deadline for this project, our group has finished implementing many of the features that we had set out to incorporate. Our project is functional - you can create a new project, save it, load it in, and export a human-readable version of the results screen. The application allows you to input a list of a variable amount of energy savings as well as material costs of the project, and returns the amount of time it would take for the project to “pay for itself”. Navigating through the pages is straightforward, and you have the option to skip to any page you wish from the menu bar at the top of the application. If we were to begin another two-week long sprint to further this application, we would likely begin with creating a database of items and their standard price rather than everything have to be manually input (which is something we had wanted to include but ran out of time for with this sprint) and further the data to show not just when the project recoups its own cost, but when you end up saving money overall compared to sticking with your old bill. We hit all of the major goals that we had set out for this iteration of the project.

**Requirements Addressed**

**Business Rules:**

BR1: The app must able to share information about a project.

BR2: The app must be able to work offline.

BR3: The app must be modular and allow for later changes easily.

BR4: The app must be able to collect information about a project.

|  |  |
| --- | --- |
| Project Requirements Addressed | |
| User Story | Implementation Status |
| \*As a DIYer, I want an app that calculates energy consumption. | Implemented - Follows BR2, BR4 |
| \*As a DIYer, I would like a material estimator feature, which could help figure out the cost of the materials. | Implemented - Follows BR2, BR4 |
| \*As a DIYer, I would like show in what amount of time the savings from a project will cover the costs of it. | Implemented - Follows BR1, BR2, BR4 |
| \*As a user of this tool, I want to be able to export the data it creates. | Implemented - Follows BR1, BR2 |
| As a DIYer, I would like a tool that can calculate the costs of a project | Implemented - Redundant with material estimator. |
| As a DIYer, I want to create a tool to help collect some data and measurements so that I can weigh costs versus benefits for smaller sized projects. | Unimplemented - Follows BR1, BR4, depending on implementation could either follow or be in conflict with BR2 |
| As a DIYer, I want to see the difficulty of a project. | Unimplemented - Follows BR1, could have some conflict with BR2 |
| As the provider of this tool, I want a way to update the app myself so that the information it provides is relevant. | Unimplemented - is connected to having a database for materials/energy usage. Conflict with BR2 |
| As a user of this tool, I would like it to be in the format of an online website where I can type in data and get helpful analysis. | Unimplemented - Conflict with BR2 |
| As the provider of this tool, I want a way for manufacturers and retailers to be promoted within the app. | Unimplemented - Requires BR3 |
| As the provider of this tool, I want to include references to contractors and professionals in the area. | Unimplemented - Conflict with BR2 |
| As a DIYer, I would like a “meter lens”, which would allow me to take a picture of an electric meter with my phone and figure out the reading. | Unimplemented - Logistical issues. |

\* Denotes key features that were top priority.

**How to Run uDIY**

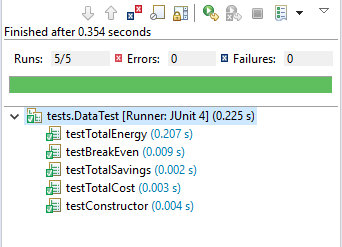
1. Start up the jar executable
2. Click on “New Project”
3. Put in the name of the project, such as “Installing Solar Panels”
4. Input your current energy usage in Kilowatt-Hours, such as 1234 KwH in the “Old Energy” section, and the expected new energy usage in Kilowatt-Hours, such as 1100 KwH in the “New Energy” section.
5. If there are additional components that relate to energy consumption, you can add another line with the “Add” button underneath the energy section, and fill in the information in the same manner as in #4.
6. In the materials section, add in the name of a material you need, such as “Solar Panel” in the “Item Name” section, and put in the unit price of that item, such as 20000 in the “Price/Unit” section, and the number of that item that you need, such as 4 in the “Quantity” section.
7. If additional materials are needed, you can add more by pressing the “Add” button and repeating the same procedure as in #6.
8. When everything is complete, press the “Done” button on the bottom right to go to the results page.
9. From the results page, you can decide to either save or export the project from the menu buttons at the top left. Saving will create a file that you will be able to open, while export will create a file that the user can easily read.
10. If you wish to go back to make changes to the project, press the “Back” button on the results page.
11. If you wish to open an existing project or create a new one, simply press the correct button from the menu on the top left.

**Contributions**

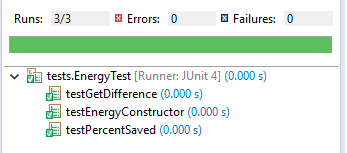
|  |  |
| --- | --- |
| **Team Member** | **Class / Method**  **Worked On** |
| **Parker Olive** | **Item(**author**), Energy(**author**), FileIO(**author**), GUI(**createActionListeners, minor changes**), DataTest(**testConstructor**), Data(**totalCost minor changes**)** |
| **Matthew Hwang** | **Data (**addEnergy, addItem, totalCost, totalNewEnergy, totalOldEnergy, totalSavings, breakEven**), Driver** (Author - All)**, Energy** (focusLost minor changes)**, FileIO** (setProjectName, open minor changes, getProjectName, export minor changes, **GUI** (Author - Most), **Item** (focusLost minor changes), **EnergyTest** (testEnergyConstructor) |
| **Tyler Shupack** | **Graph** (Author - All), **GUI** (createEditPage/prepareResults minor changes), **EnergyTest** (Author - almost all tests), **ItemTest** (Author, almost all tests), **DataTest** (Author - majority of tests) |
| **Anh Nguyen** | **GUI** (openProject, exit) |
| **Chris Kim** | **Data**(constructor, totalCost, totalSavings, breakEven, getItemList, getEnergyList) **DataTest**(testTotalCost) |

**Builds and Tests**

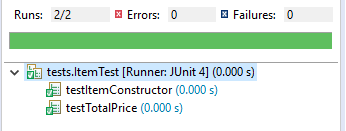
DataTest



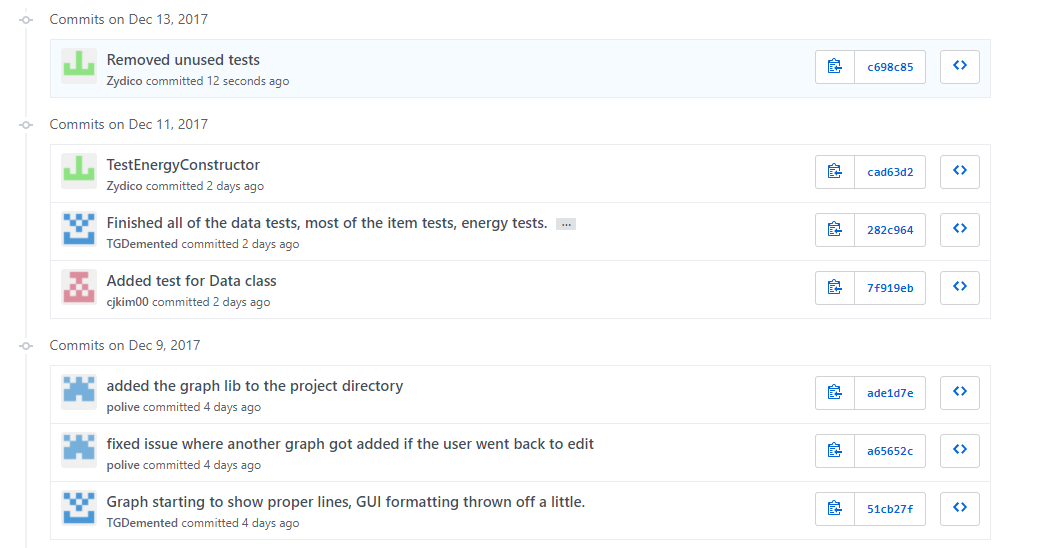
EnergyTest

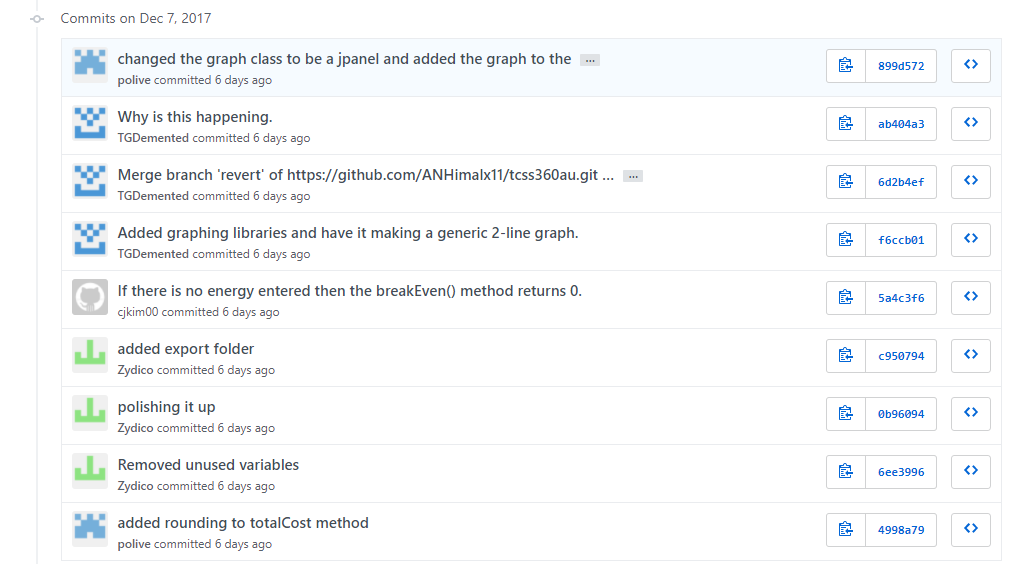


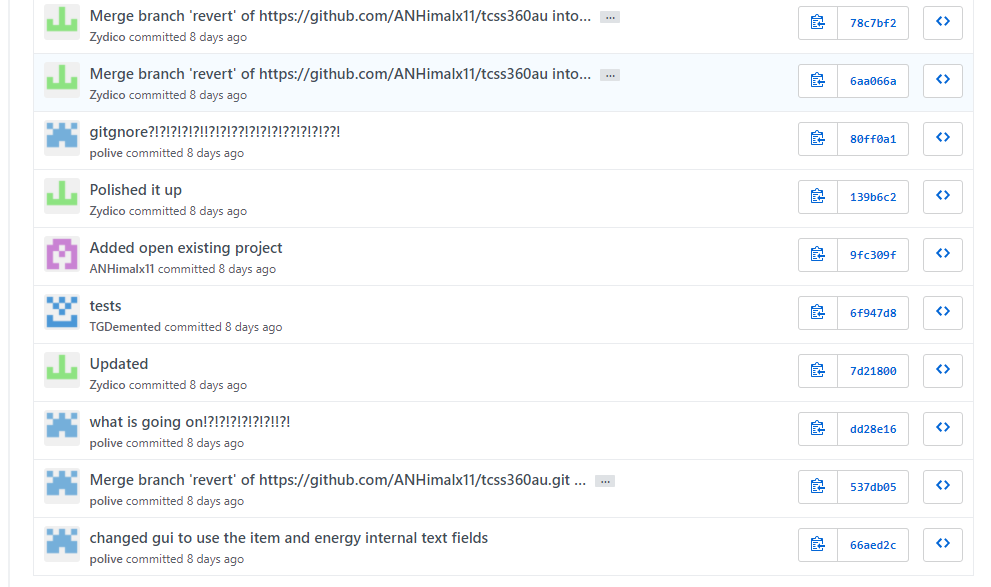
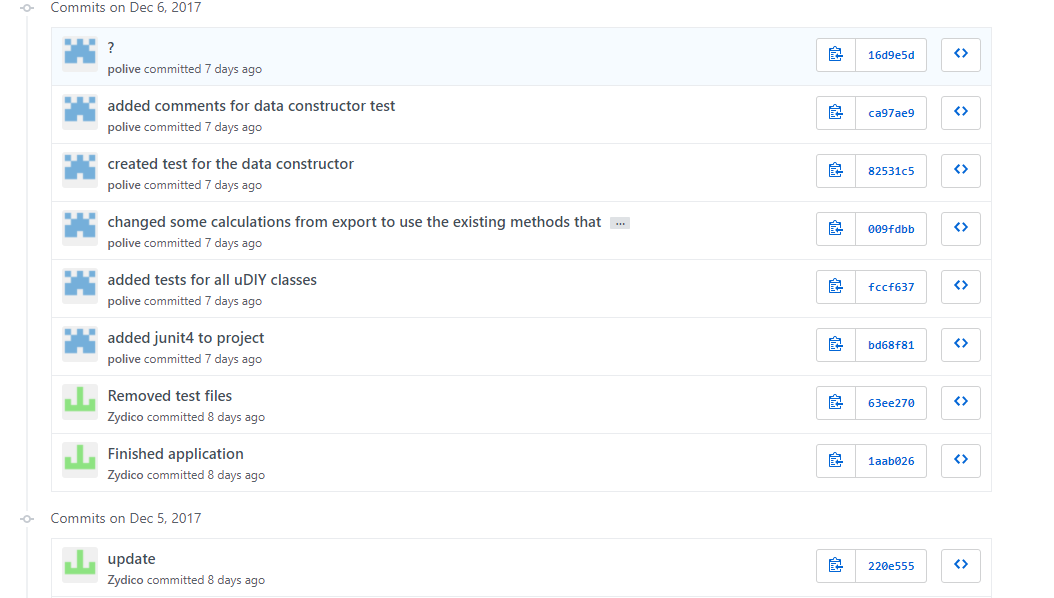
ItemTest



Git Build History







And more available here: <https://github.com/ANHimalx11/tcss360au/commits/revert>

**Source Catalog**

* **Driver.java: Runs the program**
* **Item.java: Saves the items that the user enters**
* **Energy.java: Saves the energy values the user enters**
* **Data.java: Uses the values in Item and Energy to calculate values**
* **FileIO.java: Deals with saving the values and data. Also is able to open the file and put the data back in**
* **GUI.java: The gui for the project**
* **Graph.java: Outputs a graph onto the final screen showing the number of days to break even after completing the project**
* **ItemTest.java: Tests for the Item class**
* **EnergyTest.java: Tests for the Energy class**
* **DataTest.java: Tests for the Data class**

**Emergency Contacts**

Matthew Hwang 206-372-0329 matt95@uw.edu

Tyler Shupack 623-640-5316 tylers27@uw.edu