

TEAM TORNADO



PROJECT TEAM MEMBERS

Eric Thomas Hoover
Minh Pham
Curran Mohan Seam
Sharanjit Singh

PROJECT SUBMITTED TO

Mr. Jeffrey Weiss

TEAM EMAIL

ttornado360@gmail.com

WEBSITE URL

<https://github.com/teamtornado/TCSS360-Project>

Table of Contents:

I.	Summary Page	
	A. Introduction.....	Page 3
II.	Requirements Addressed	
	A. User Stories	Page 4-5
	B. Business Rules	Page 5
III.	Build and Test	
	A. Unit Testing and GitHub	
	Screenshots.....	Page 6
IV.	How to Run the Application	
	A. Setup.....	Page 7
VI.	Additional Notes.....	Page 8-10
	A. Contributions	
	B. Source Catalog	
	C. Emergency Contacts	

B. Introduction:

Our team's final design for the application is made in a way to facilitate/guide to an end user for their DIY project. It follows a step-by-step procedure approach for a DIY project that the end user can specify using a variety of given data. Our design focuses on a data driven implementation that allows the user or developers to expand or redirect what sort of projects are supported - be it a house renovation, vehicle refab, or dog leash patterning. With this area of emphasis, we are able to support vastness options any niche project might have for our app.

With our data driven structures, we can utilize the provided end user information in a more efficient manner. That is, we can store, parse, and process user information in ways of more value and accuracy. In our design, an end-user can easily add items to their project along with their respective fields. This ease of use regarding item and field additions is conducive towards the simple free-flowing and DIY oriented approach of the application. Furthermore, items follow a broad-to-specific format. That is, if a user wants to add an item to their DIY project, they can easily look for the category they want to add, say appliance, and then simply add a child type of an appliance, say a stove. Once the user has found an item they wish to add to their project, they can fill in the fields provided from the editable database for that specific item.

With this design in mind, an end user can create, view, and edit the project. Lastly, the app runs in an offline experience with the option to be able to sign in and out of an email - this includes the editable database of fields and items. We

believe this tandem of creation, revision, and viewing is the very core of a well rounded and usable DIY-project application.

C. Requirements Addressed

User Stories:

Note: Main features are **bolded**.

- **US01:** As a user, I want to be able to view my project history so I can evaluate how I have progressed. **(YES)**
- **US02:** As a user, I can input the current state of a feature of my house and be advised on the costs and energy savings of different projects for improving the feature. **(Partial - there is no advisement built-in)**
- **US03:** The user can edit the underlying structure of available items and fields to tailor their experience in case the developers had not thought to support their particular niche project. **(Partial - if the user decides to learn the schemaDatabase.txt syntax, they can edit it. It's not complicated, but not a shippable experience. As the developers though, this could be a workable solution).**
- **US04:** As a DIYer, I would like to be able to make calculations and measurements important for heat and energy savings. **(NO)**
- **US05:** As a privacy minded user, I want the stored information kept within a local storage so I can maintain my privacy personally with less reliance on others. **(YES)**
- **US06:** As a DIYer, I want to store additional notes about measurements or miscellaneous information to keep track. **(YES)**

- US07: As a user, I want to be able to export the information stored on my internal database as a printable document so I can view it traditionally and share the information with others. **(YES)**
- US08: As a user with limited personal time to work on these projects, I want to be able to view the information already stored on the app on my different devices, so I'm not restricted to one device. **(Partial - If you have a flash drive, the saved projects can be loaded from any other app, but I wouldn't call this streamlined)**
- US09: As an indecisive user, if I add an item I decide I do not want while creating or editing a project, I can delete said item. **(YES)**
- US10: As a user, I can use my email within the user settings within the project, and have this information persist between runs. **(YES)**
- US11: As a user who likes my own settings, I can set the window size and have it persist between runs. **(YES)**

Business Rules:

- BR01. Keep projects within a specified budget. **(YES)**
- BR02. Finish project by the end of the quarter. **(YES)**

D. How to run the application

To run this application first double click the TornadoProjectApp.jar. You will be greeted by the home screen. The menu bar at the top of the application houses a *help* button and *setting* button. Under *setting* you can set the frame size, export your user settings, and sign in/out of you email. Under *help* is our about page.

To create a project, click *Create Project*. Here, enter a project name, state (location), budget and project description. Only when these are filled can you continue forward. You can go back to the home screen at any time by clicking the back button. Click next to continue to the item selection screen.

This page allows you to add and delete items as you please. The items selection is on the right side, the left side is the rundown of what your project consists of, so far. To add an item, click on the dropdown bar that says *appliance*. With this bar, you can choose an item type. We have Appliance, Heating, Lighting, and Renovation. Once one of these is clicked, you can select a subcategory of these. For each item that you select, you can then enter relevant units and measurements to be more specific. If you don't like one of the items you added, simply click the delete item button and select which item you want to delete. If you are lost at any point in the item selection menu, click the reset button to bring the item selection bar back to the beginning. Once done adding items, click *next* to save the file.

You will be brought back to the homescreen once the project is successfully saved. Open the file you have just created by clicking *Open Project* button. When clicked, you may choose any saved project to view - a test project is also provided. Here, You can view your project and all of the items and values. If you want to edit

the project, click the *edit* button and you will go to the creation screen. The values currently in your project will be present here. If you want to export the project to a text file, click the *export* button. This is the entire walkthrough of the features in our application.

E. Contributions

Backend and data modeling - Eric Hoover

GUI Controller - Minh Pham, Eric Hoover, Curran Seam

GUI Components - Curran Seam, Sharanjit Singh, Minh Pham

JUnit Testing - Sharanjit Singh, Curran Seam, Minh Pham, Eric Hoover

F. Tests

Merge branch 'master' of https://github...

 Curran Seam committed an hour ago

added date on scrollablepane

 Curran Seam committed an hour ago

Commented and dated the code

 aiinori committed 2 hours ago

fixed removeFieldfromItem test case

 Curran Seam committed 19 hours ago

Test

 Sharanjit Singh committed 19 hours ago

a

 Curran Seam committed 19 hours ago

Testing

 Sharanjit Singh committed 19 hours ago

Update ProjectControllerTest.java

 Curran Seam committed 20 hours ago

Testing

 Sharanjit Singh committed 20 hours ago

Another testing

 Sharanjit Singh committed 21 hours ago

▼  ProjectControllerTest [Runner: JUnit 5] (5.516 s)

 testaddItemnull (1.101 s)

 testaddToField (0.290 s)


 testgetLocationnull (0.311 s)

 testremoveItem (0.239 s)


 testaddItemnull1 (0.271 s)

 testaddFieldFromItemnull (0.252 s)


 testremoveFieldFromItemnull (0.267 s)

 testremoveItemNull (0.245 s)

 testremoveFieldFromItem (0.280 s)

 testsetName (0.251 s)

 testProjectDescriptionnull (0.225 s)

 testProjectDescription (0.270 s)

added all my comments and completed ...

 Curran Seam committed a day ago

adding email setting

 Curran Seam committed a day ago

Testing

 Sharanjit Singh committed 2 days ago

testing

 Sharanjit Singh committed 2 days ago

Testing fixes

 Vlarrick committed 2 days ago

Testing

 Sharanjit Singh committed 2 days ago

Backend cleanup, Controller Refactor

 Vlarrick committed 6 days ago

Post Demo cleanup Step #1

 Vlarrick committed 6 days ago

After demo

 aiinori committed Jun 6, 2019

House keeping edit

 aiinori committed Jun 6, 2019

▼  AboutTest [Runner: JUnit 5] (0.001 s)

 testGetAbout() (0.001 s)

▼  SchemaTest [Runner: JUnit 5] (0.103 s)

 correctGetChildAndGetParent() (0.062 s)

G. Source catalogs

The Java API: <https://docs.oracle.com/javase/10/docs/api/overview-summary.html>

Previous assignment in TCSS 305 for GUI Controller.

Previous assignment in TCSS 305 for JUnit Testing.

H. Emergency Contacts

Team Member Name	Email	PhoneNumber
Curran Seam	cseam@uw.edu	(206) - 403 - 6916
Eric Thomas Hoover	erich34@uw.edu	(253) - 709 - 7203
Sharanjit Singh	Singhs22@uw.edu	(253) - 258 - 9606
Minh Pham	minhp317@uw.edu	(253) - 457 - 8776