

## **Deliverable 3 - MVP Project Planner**

Minimum Viable Product

Cody Dukes, Nathan Grimsey, Maple Gunn

itsmobnt@gmail.com | ngrimsey@uw.edu | gunnra@uw.edu

<https://github.com/TCSS360Group2Fall2023/Golden-Group-2-Repository>

School of Engineering and Technology, University of Washington Tacoma

TCSS360: Software Development And Quality Assurance Techniques

Professor Jeffrey Weiss

December 15th, 2023

## Contents

Introduction.....	3
Requirements addressed.....	4
How to run the application.....	9
Contributions.....	13
Tests.....	16
Source catalog.....	19
Emergency contacts.....	24

## **Introduction**

MPP is an app with a scope to keep track of tools, budgets, and individual projects. With the time given in the quarter, we invested a fair amount into making a solid Graphical User Interface (GUI) for this project along with a simpler backend. It was important to have an intuitive user interface as it'd help test our models. We started with a Model/View design pattern, and transitioned to a Model/View/Test pattern for the final deliverable. We ended up achieving many of the goals we sought after during our design phase. The final product can add tools, materials, expenses and logs to a project. It can store multiple projects, which can be searched for from the main menu. Tools and materials can be stored and saved from one project only to be imported into a different project. There is a darkmode, settings that can be saved/exported, project sharing, and budgeting. All of these elements are backed by a high quality GUI.

### Requirements Addressed

Yes	US01 - As a DIY enthusiast, I'd like to start a project that contains my budget.
Yes	US01.1 - As a DIY enthusiast, I'd like to be able to store my tools/materials in a project.
Yes	US01.2 - As a DIY enthusiast, I'd like to be able to log my progress.
Yes	US01.3 - As a DIY enthusiast, I'd like to be able to keep track of purchases I've made.
Yes	US02 - As a project manager with a lot of projects, I'd like the ability to use keywords in a search instead of having to memorize entire folder names.
Yes	US03 - As a DIY enthusiast, I'd like to be able to store my purchased tools outside my projects, and reference them inside my projects as needed without having to add them to my budget.
Somewhat, the GUI tends to be fairly intuitive in some areas, but still lacking in others. Saving expenses, for instance, could be more	US04 - As a DIY enthusiast, I'd like a tool that makes the learning experience of planning and budgeting easier and more

intuitive.	accessible.
No, other than Materials and Tools being available outside Projects, there is not a way to directly reference another project or import specific elements from one project to another.	US05 - As a DIY enthusiast, I want to be able to reference a previous project in a new project so I can use floor plans and material estimates in my new projects.
Yes, mostly. The default mode is “administrator mode”, there is no lower level of permission. This was a limitation of time, but would be high priority if development were to continue.	US06 - As a project manager who wants easy access to backend features, I’d like to be able to access maintenance mode without a password.
No. The model is in place to allow storing images with tools, but the GUI hasn’t gotten to the point of adding and displaying images.	US07 - As a DIY enthusiast, I’d like a way to keep all warranties and receipts for my tools in one place so I can easily access them later.
Yes, MVP Project Planner could certainly accomplish that task.	US08 - As a stay-at-home mom, I'd like a way to keep track of what I'm buying to build a new garden.
Yes, expenses are recorded and total cost and budget are easily displayed. Budget visualization through 2d graphics is not implemented, but the model class BudgetCalc has been written. The GUI just needs to display that information in an intuitive way.	US09 - As a DIY enthusiast, I’d like a way to keep records of the budget so that everyone can stay on the same page.
Half. Budget is done, but there are no ways to schedule tasks, other than perhaps by making a log dated in the future.	US10 - As a project manager I'd like to keep track of our budget and deadlines.
No, this would be where images attached to	US11 - As a project manager, I’d like to be

Projects would be helpful, but the GUI is not currently developed enough to this user story.	able to easily keep track of floor plans and renovations to property.
Yes, the logs feature would be suited well to this.	US12 - As a project manager I'd like to journal progress as we build a new apartment complex.
Yes, settings are easily exportable, and projects, materials, and tool are all saved to files that can be easily transferred for import/export.	US13 - As a project manager I need to be able to export data to share with my other departments.
Yes, though it would perhaps be a bit unintuitive. Given that this program is a jar and not an exe, it would provide a higher burden of entry to new users to share content with. Additionally, there is not an easy way to share data other than to export it (which requires installing the program), or to show someone in person.	US14 - As a stay-at-home mom, I want to create project ideas and share them with others.
Yes	US15 - As a stay-at-home mom, I want to keep track of our home purchases.
Yes, theming is available.	US16 - As a stay-at-home mom I'd like to have the option for the UI to be darker for my poor eyes.
No, the application is not nearly developed enough to consider building it for another platform. It would take much more work before this program was Android ready.	US17 - As a project manager who doesn't always have access to a desktop, I'd like to be able to use the app on my phone.

No, this was entirely due to time constraints. This would be a high priority issue in the future of development.	US18 - As a stay-at-home mom who can be a bit clumsy when it comes to my data, I'd like protection from unauthorized deletion.
Yes	US19 - As a user I want to enter settings such as my first name and email address
Yes	US20 - As a user I want to export settings for synchronization to other devices
Yes	US21 - As a user I want to import settings to synchronize with other devices
Yes	US22 - As a user I want to export and import other information in the application
Yes	USX01: As a user I want to setup the app so it will have my first name and email address
Yes	USX02: As a user I want to see an About screen for the app so I can see the version number
Yes	TRX01: The version number cannot be hard coded in the UI – The version number must be requested by the UI code from a constant property of an object. The code for the object may change from version to version, the UI code must not.
Yes	BR01 - No passwords required when accessing the “maintenance mode”.
Yes	NF01. Other than features that involve

	refreshing or uploading data, all aspects of the application must be able operate offline, meaning there is no connection to a network or the Internet.
Yes	NF02. The system does not require secure user authentication. Where an identity is needed, a person signs in when the application starts using a unique name and or PIN.
Yes	NF03. The application executes from a runnable jar file on a single Windows machine as a desktop application with only a single user at a time.
Yes	NF04. Persistent data will be stored in one or more files on the local file system.
Yes	NF05. The programming will be carried out using current versions of Eclipse or equivalent IDE, Java, and JUnit.
Yes	NF06. Source code will be managed in a GitHub repository.
Yes	NF07. All third-party components and libraries must be approved in advance and properly cited. Third party executable components must be packaged with the application and cannot depend on prior installation or configuration.



## How to run the application

Full run-through of features via a script:

- Run MVP-Project-Planner.jar.
- To start a project, click New Project. To add tools or materials, click the upper left menu and navigate to tools or materials.
- In tools/Materials, you can create a permanent item that can be referenced in future projects.
- If you choose to start a project, enter a name and budget. Optionally you may also enter a description.
- Click save.
- Click add expense to create an empty expense.
- Enter data into the expense fields, then click update to save the expense information.
- Click save in the lower left corner to save the project, as well as the expense you've added.
- Click add Material to add a material.
- If you added a Material from the main menu, it can be accessed here and added as an expense. Alternatively, you can create a new Material.
- Add as an expense.
- Click save.
- Click on the upper left menu.
- Click Project Tools.

- If you added a Tool from the main menu, it can be accessed here and added as a Tool.

Alternatively, you can create a new Tool.

- Click create new tool.
- Fill the fields as desired.
- Toggle it as an expense, and then click save. This allows you to easily add a tool as an expense or remove it as an expense, since tools are typically one-time purchases.
- Check to see that both materials and tools are in Expenses.
- Click the upper left menu.
- Click Logs
- Click add Log
- Make sure the date is valid (format is MM/DD/YYYY), add a name, and a description if you desire.
- Click update.
- Click save.
- Close program.
- Reopen program.
- Open the project to see that logs and expenses have persisted.
- You may add and delete as many expenses, tools, and logs you want in a project, just be sure to save!
- Go back to the main menu.
- Create as many projects, tools, and/or materials as you like. The pages they are displayed on are scrollable, so don't worry about overfilling them!
- Then go back to the main menu again.

- Type any part of a project title to search for that project (type in the search bar and hit enter to search).
- Searching works in the Tools and Materials screens too.
- Click upper left menu
- Click settings.
- Input name/email.
- Enable/disable dark mode as desired.
- Save.
- Export settings.
- Go to the About section to see your information.
- Go back to settings and make a new profile.
- Save
- Check about.
- Go back to settings.
- Import old settings.
- Profile and all settings are now updated to the ones you imported.
- Check about.
- Go to projects.
- Click project.
- Go to details.
- Edit name and budget.

## Contributions

Nathan Grimsey (Para-lyzed) - All GUI/frontend for app, lead design.

Lines of code - Additions: 6347 Deletions: 2068

Maple Gunn (MapleSugarstone) - Backend for projects, expenses, materials/tools, search, tests.

Lines of code - Additions: 913 Deletions: 35

Cody Dukes (MrMemori) - Backend for importing/exporting, user settings, logs, tests.

Lines of code - Additions: 1295 Deletions: 148

(Note: GitHub insights tab shows repackaging as additions and deletions, so code lines were counted by manually summing pull requests)

Person	Class
Nathan Grimsey - coding Maple Gunn - coding	About
Cody Dukes - coding	BudgetCalc
Cody Dukes - coding Nathan Grimsey - editing	DataIO
Maple Gunn - coding	Expense
Cody Dukes - coding	Log

Nathan Grimsey - coding Maple Gunn - coding (Main.searchFiles())	Main
Maple Gunn- coding	Material
Nathan Grimsey - coding Maple Gunn - coding	Profile
Maple Gunn - coding Nathan Grimsey - editing	Project
Maple Gunn - coding	Tool
Cody Dukes - coding Nathan Grimsey - editing	UserSettings
Cody Dukes - coding	AboutTest
Cody Dukes - coding Nathan Grimsey - editing	DataIOTest
Maple Gunn - coding	ExpenseTest
None	MainTest
Maple Gunn - coding	MaterialTest
Cody Dukes - coding	ProfileTest
Maple Gunn - coding	ProjectTest
Cody Dukes - coding	ToolTest
Maple Gunn - coding	UserSettingsTest
Nathan Grimsey - coding	AboutScreen

Nathan Grimsey - coding	BaseFrame
Nathan Grimsey - coding	BaseScreen
Nathan Grimsey - coding	BaseSelectorScreen
Nathan Grimsey - coding	CustomButton
Nathan Grimsey - coding	CustomScrollBar
Nathan Grimsey - coding	CustomTextField
Nathan Grimsey - coding	EditMaterialScreen
Nathan Grimsey - coding	EditToolScreen
Nathan Grimsey - coding	MaterialSelectScreen
Nathan Grimsey - coding	Menu
Nathan Grimsey - coding	NewMaterialScreen
Nathan Grimsey - coding	NewProjectScreen
Nathan Grimsey - coding	NewScreen
Nathan Grimsey - coding	NewToolScreen
Nathan Grimsey - coding	ProjectDetailsScreen
Nathan Grimsey - coding	ProjectEntryRow
Nathan Grimsey - coding	ProjectExpensePanel
Nathan Grimsey - coding	ProjectExpenseScreen
Nathan Grimsey - coding	ProjectLogPanel
Nathan Grimsey - coding	ProjectLogScreen

Nathan Grimsey - coding	ProjectMaterialSelectScreen
Nathan Grimsey - coding	ProjectSecondaryPanelTemplate
Nathan Grimsey - coding	ProjectSelectScreen
Nathan Grimsey - coding	ProjectToolPanel
Nathan Grimsey - coding	ProjectToolScreen
Nathan Grimsey - coding	ProjectToolSelectScreen
Nathan Grimsey - coding	SettingScreen
Nathan Grimsey - coding	ToolSelectScreen

## Tests

✓	✓ AboutTest (tests)	50 ms
✓	testAboutGetOwner()	35 ms
✓	testAboutGetVersion()	1 ms
✓	testAboutGetContributors()	1 ms
✓	testAboutGetOwnerString(')	10 ms
✓	testAboutUpdateProfile1()	2 ms
✓	testAboutUpdateProfile2()	1 ms

✓	✓ DataIOTest (tests)	233 ms
✓	testLoadTool()	193 ms
✓	testLoadMaterial()	7 ms
✓	testLoadProject()	13 ms
✓	testSaveTool()	7 ms
✓	testSaveProject()	8 ms
✓	testSaveMaterial()	5 ms

✓	✓ ExpenseTest (tests)	26 ms
✓	testExpensePrice()	24 ms
✓	testExpenseName()	2 ms



✓	MaterialTest (tests)	35 ms
✓	testMaterialName()	33 ms
✓	testMaterialCategory()	1 ms
✓	testMaterialPrice()	1 ms

✓	ProfileTest (tests)	28 ms
✓	testProfileEmail()	24 ms
✓	testProfileEqualsTrue()	1 ms
✓	testProfileEqualsFalse()	1 ms
✓	testProfileName()	1 ms
✓	testSetProfile()	1 ms

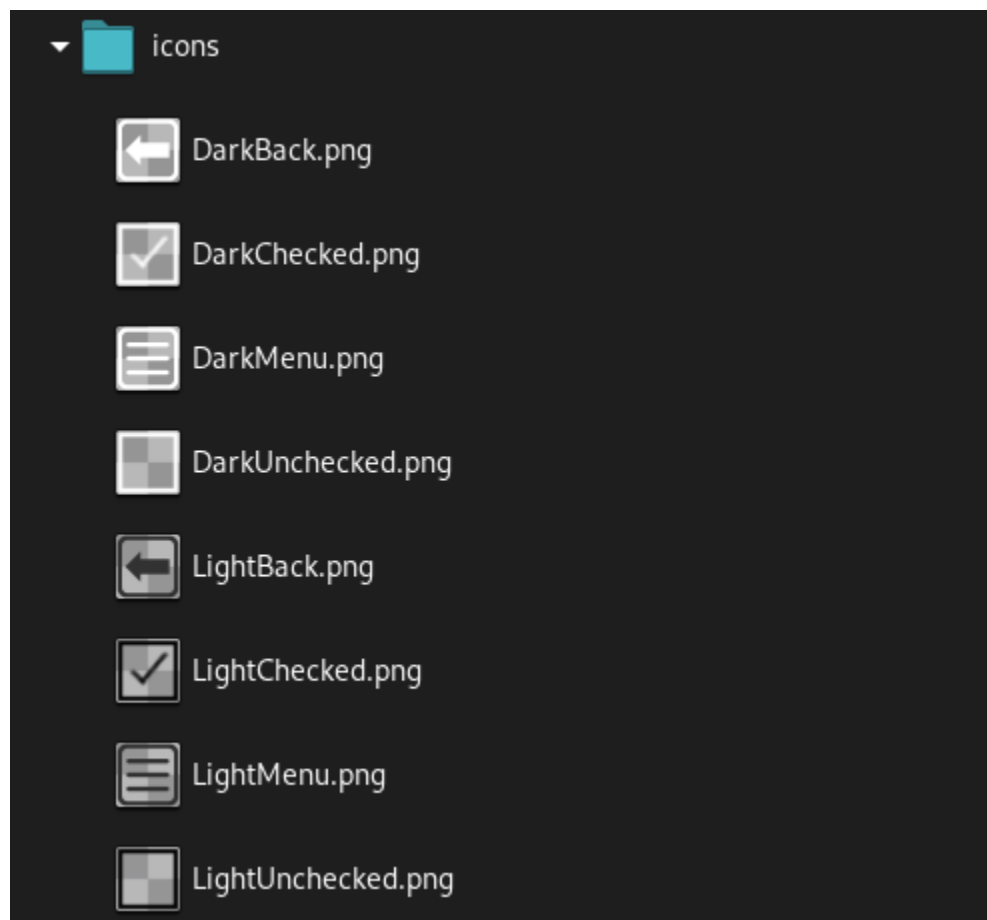
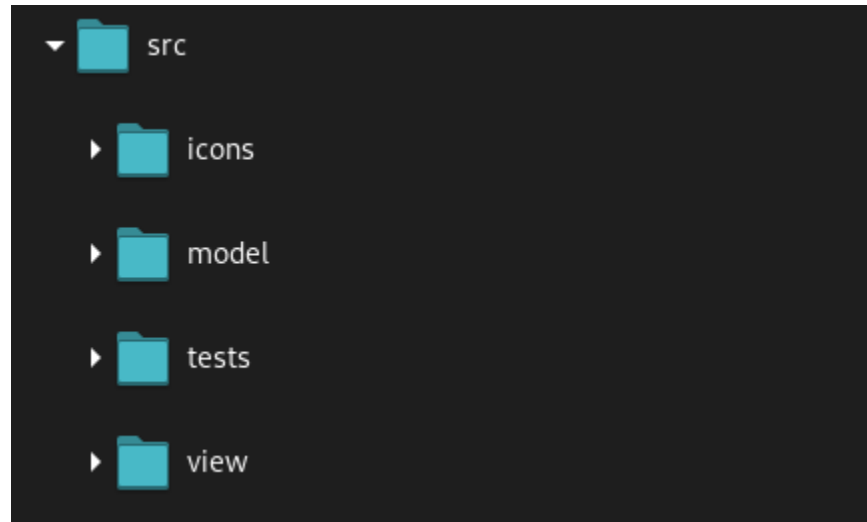
✓	ProjectTest (tests)	35 ms
✓	testProjectName()	34 ms
✓	testBudget()	1 ms

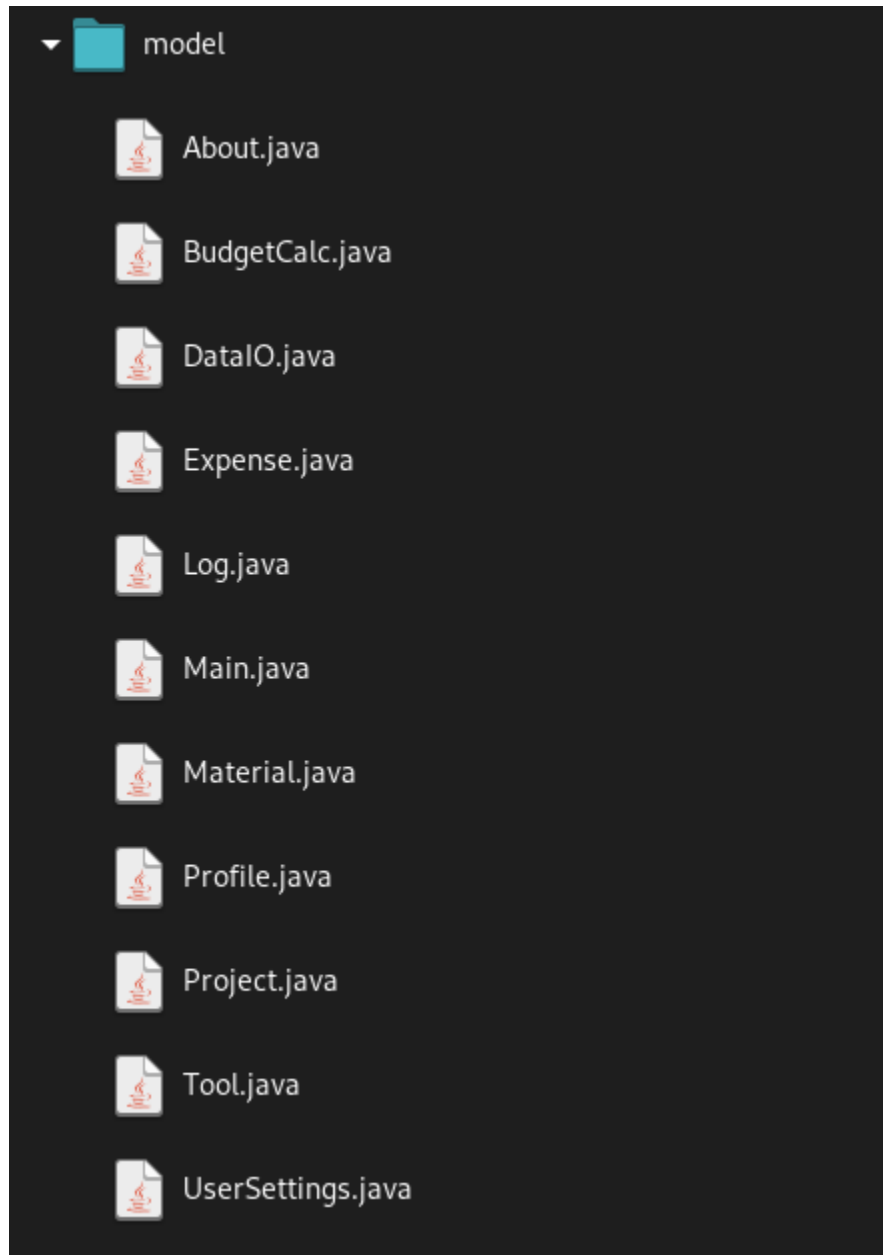
✓	ToolTest (tests)	31 ms
✓	testToolName()	30 ms
✓	testToolPrice()	1 ms

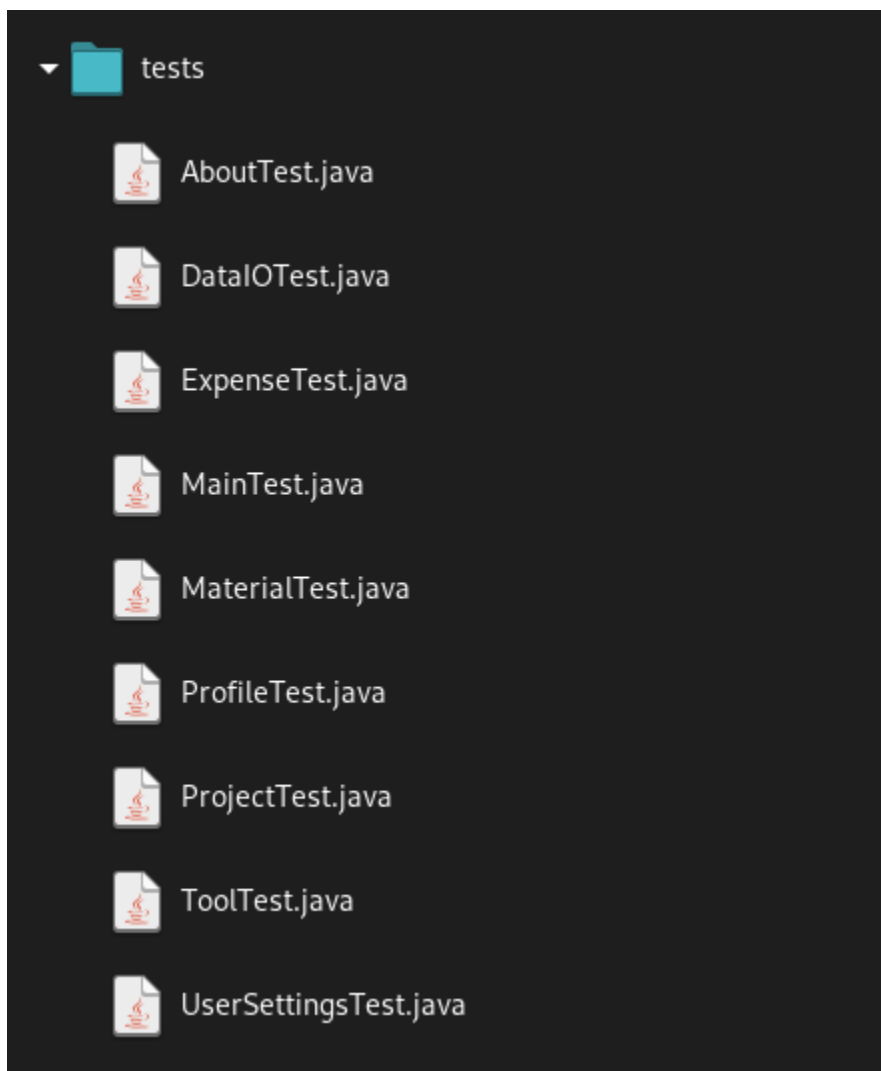
✓	UserSettingsTest (tests)	37 ms
✓	testProfile()	37 ms

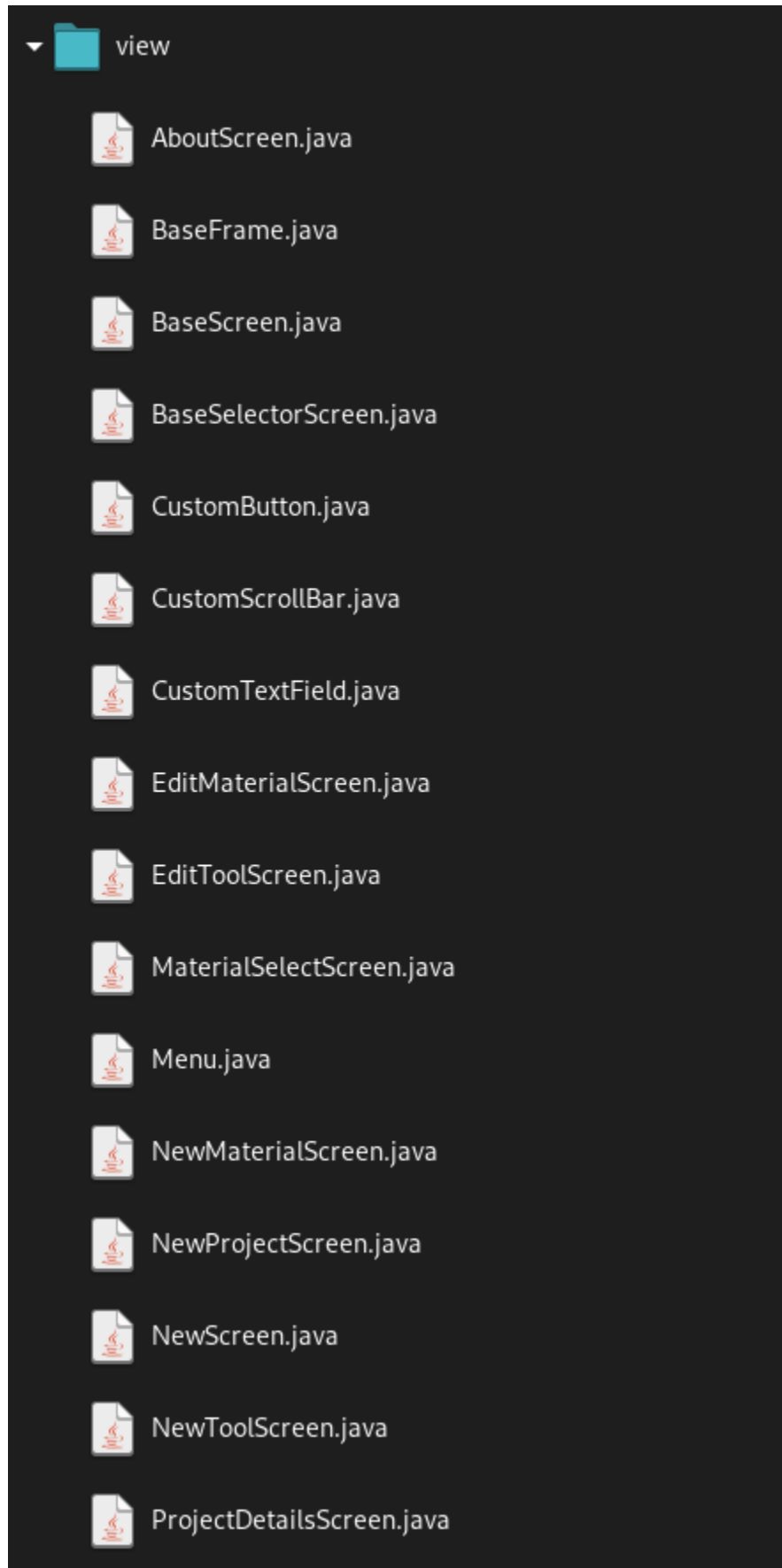
Testing was a weak point in this project. Since the vast majority of classes are GUI, they weren't feasible to test. All of the classes in the model package are good candidates to test, however. There were some struggles in figuring out how to test certain classes, which led to a couple sparse and incomplete test classes, and the struggle to finish the GUI in time took away resources that could have otherwise been used on testing. Overall, it is the biggest weakness of this program, though there have been many hours of manual testing.














## Source catalog









 ProjectEntryRow.java  
 ProjectExpensePanel.java  
 ProjectExpenseScreen.java  
 ProjectLogPanel.java  
 ProjectLogScreen.java  
 ProjectMaterialSelectScreen.java  
 ProjectSecondaryPanelTemplate.java  
 ProjectSelectScreen.java  
 ProjectToolPanel.java  
 ProjectToolScreen.java  
 ProjectToolSelectScreen.java  
 SettingScreen.java  
 ToolSelectScreen.java

**Emergency contacts**

Nathan Grimsey | [ngrimsey@uw.edu](mailto:ngrimsey@uw.edu) | [NathanCGrimsey@protonmail.com](mailto:NathanCGrimsey@protonmail.com)

Maple Gunn | [gunnra@uw.edu](mailto:gunnra@uw.edu)

Cody Dukes | [itsmobnt@gmail.com](mailto:itsmobnt@gmail.com)