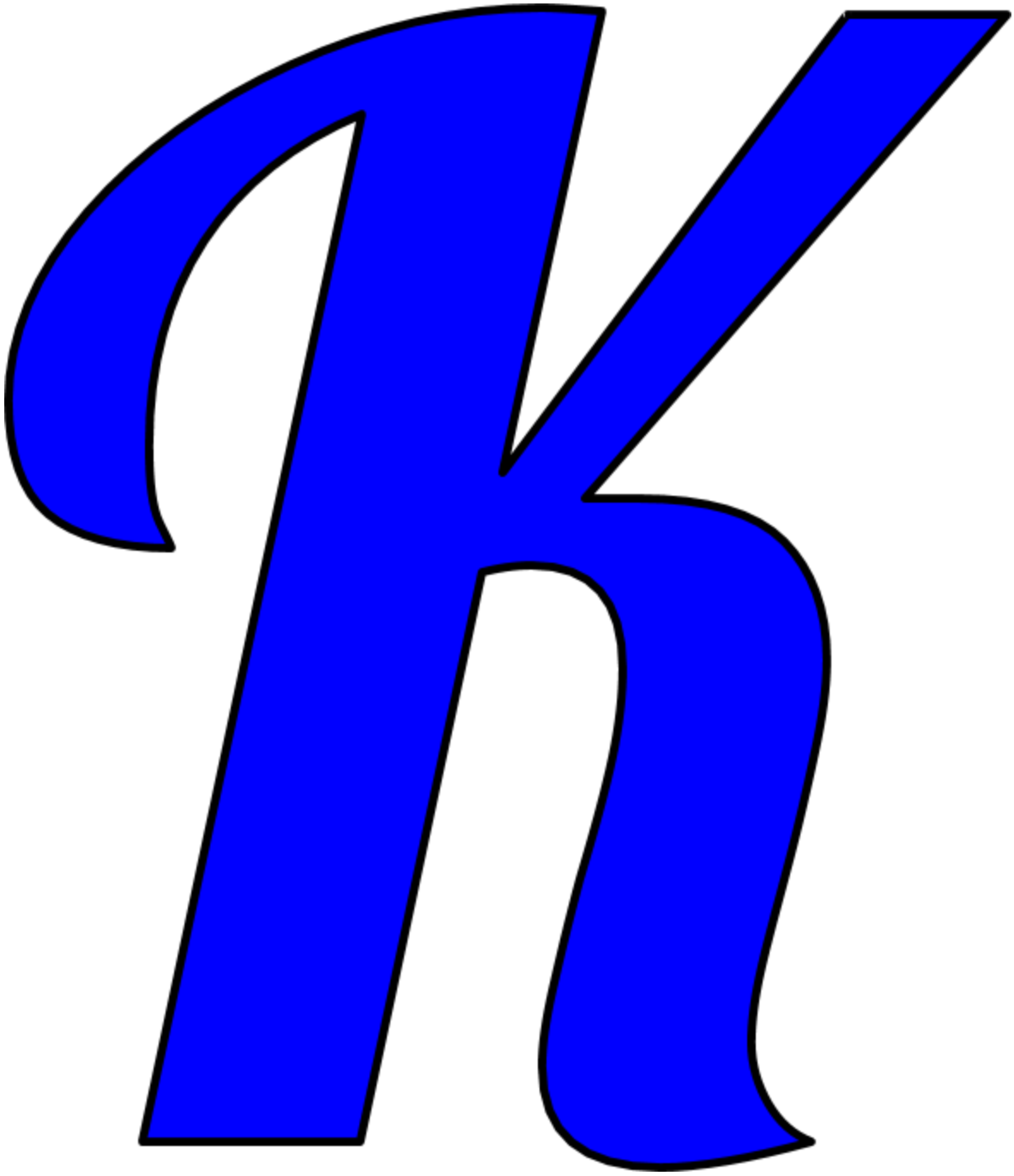


Deliverable 2

By K

Alex

Logan

Peter

Karan

Carter

Introduction

Our team decided that we wanted to focus on the “viewing projects” aspect of the User stories. Our program will store and fetch all submitted projects from a database and display them for the user. Our program will also feature a tag system, where users can add various tags to different projects and search for specific tags. For example, if a user wants to view desks they can simply search for the “desk” tag. We felt this was more efficient than our previous idea of requiring users to navigate through categorized windows to find what they wanted. However, our project still maintains the “I just want to browse projects” idea by initially displaying all submitted projects. These will be sorted to some degree by their tags. Users can “window shop” through projects, while also being able to search for specific tags.

Table of Contents

1. Rationale Summary
2. Class Diagram
3. User SSD
4. System Startup SSD

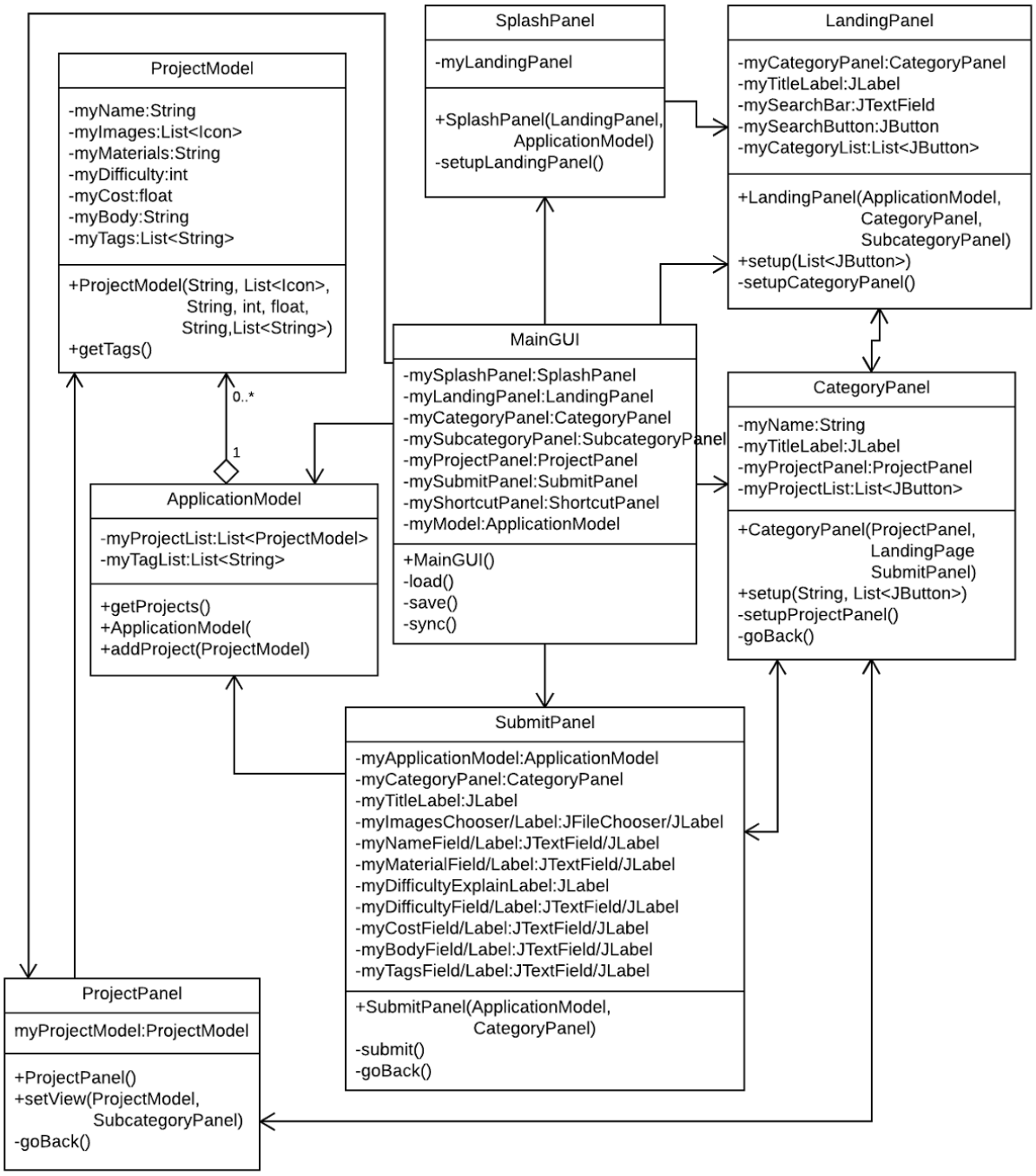
Rationale summary

Using a database instead of a cloud storage: The first important decision we took as a team was if we are going to keep our application offline or online. After evaluating the pros and cons of both the options we decided to go ahead with a database. One of the main reasons we decided to go ahead with this approach was because one of our user story is “we want be able to use this application offline”.

Search Functionality: Important functionality of our application is our search feature. We had a lot of discussion on how our search functionality would work. We had a lot of options for this as well, as the projects could be searched by title, difficulty, or by keywords. For the this specific functionality we decided to go ahead with tags. We have added a field called tags to our projects and different projects could be searched by searching for a specific tag.

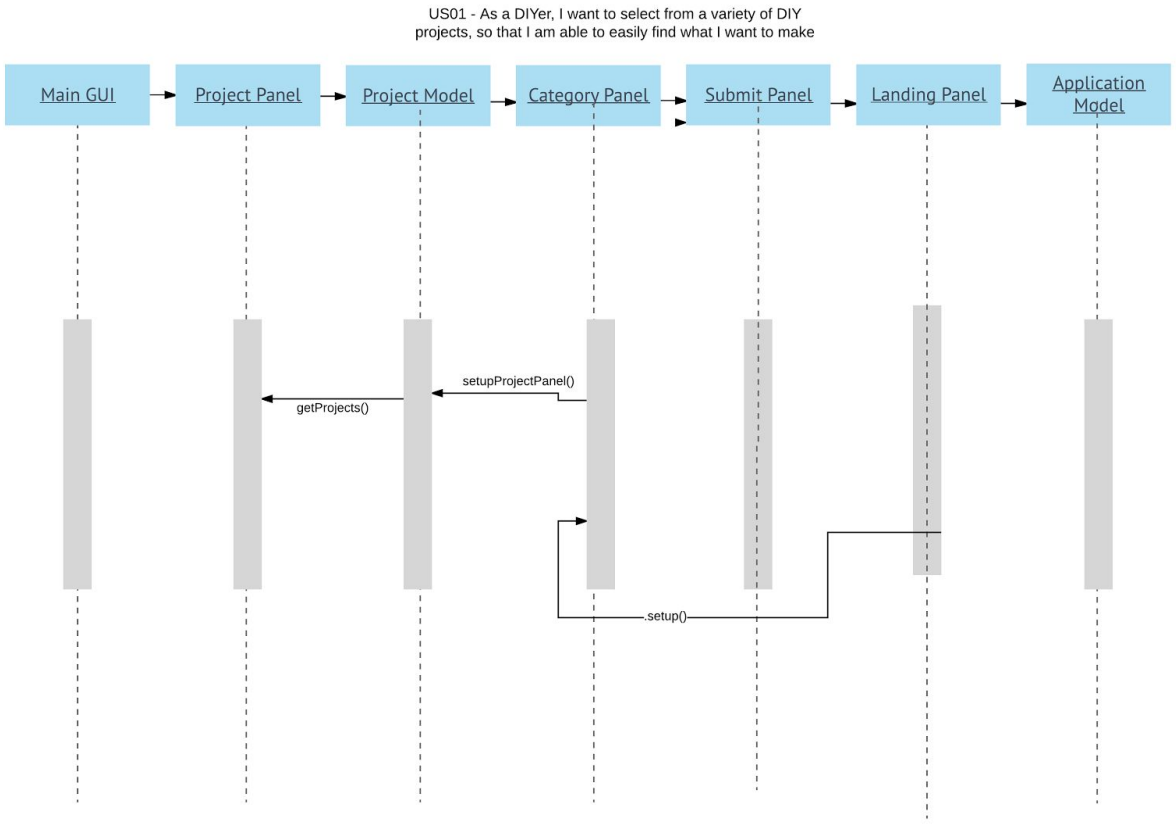
UI Convention: The UI is an integral part of this project. And, it was important for us to discuss this at an early stage as both our front end and back end would need to sync with each other. We decided to go ahead with a data-driven approach. By data-driven we mean that we are going to use JPanels of data and keep switching them instead of adding fields to our JFrame. We thought this was a good way to decrease the dependencies in our code.

Class Diagrams



User Story SSD

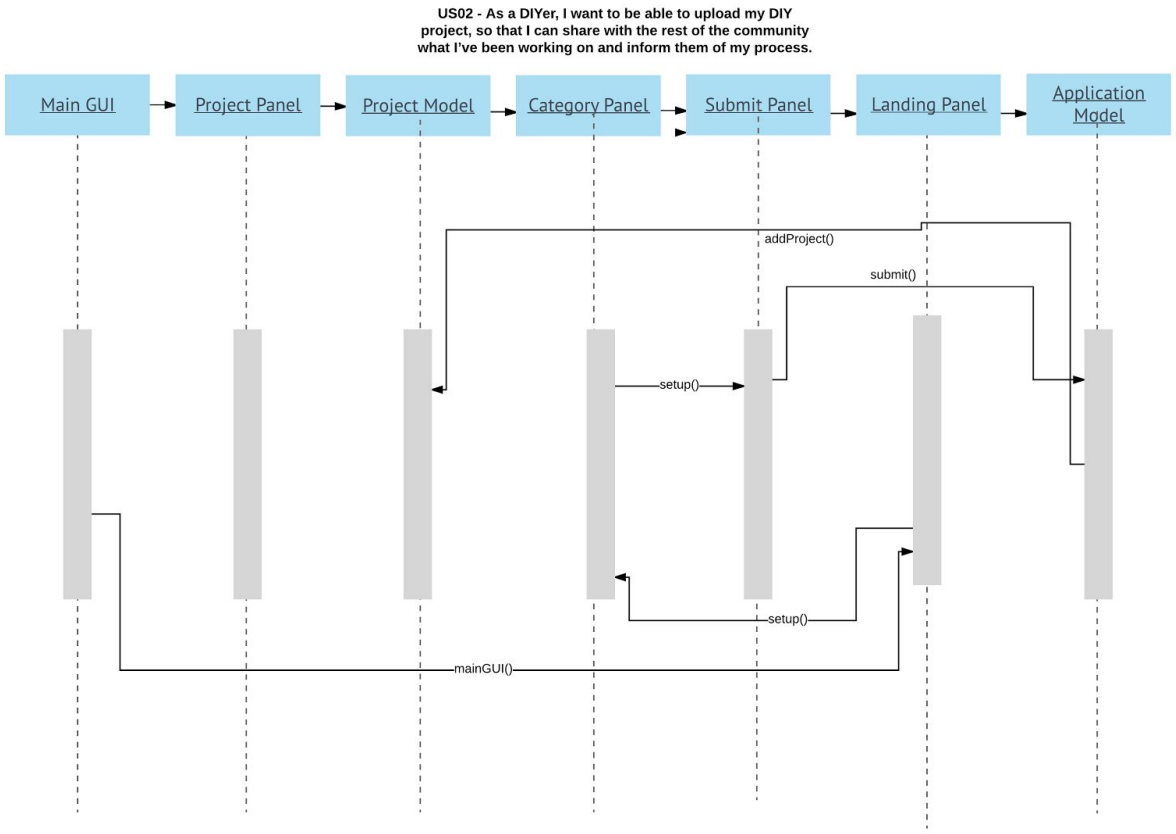
“US01 - As a DIYer, I want to select from a variety of DIY projects, so that I am able to easily find what I want to make.”



The Landing Page will set up the Category Panel. The Category Panel will then setup the Project Model, which will fetch all of the projects and display them in a grid layout for the user to interact with.

User Story SSD

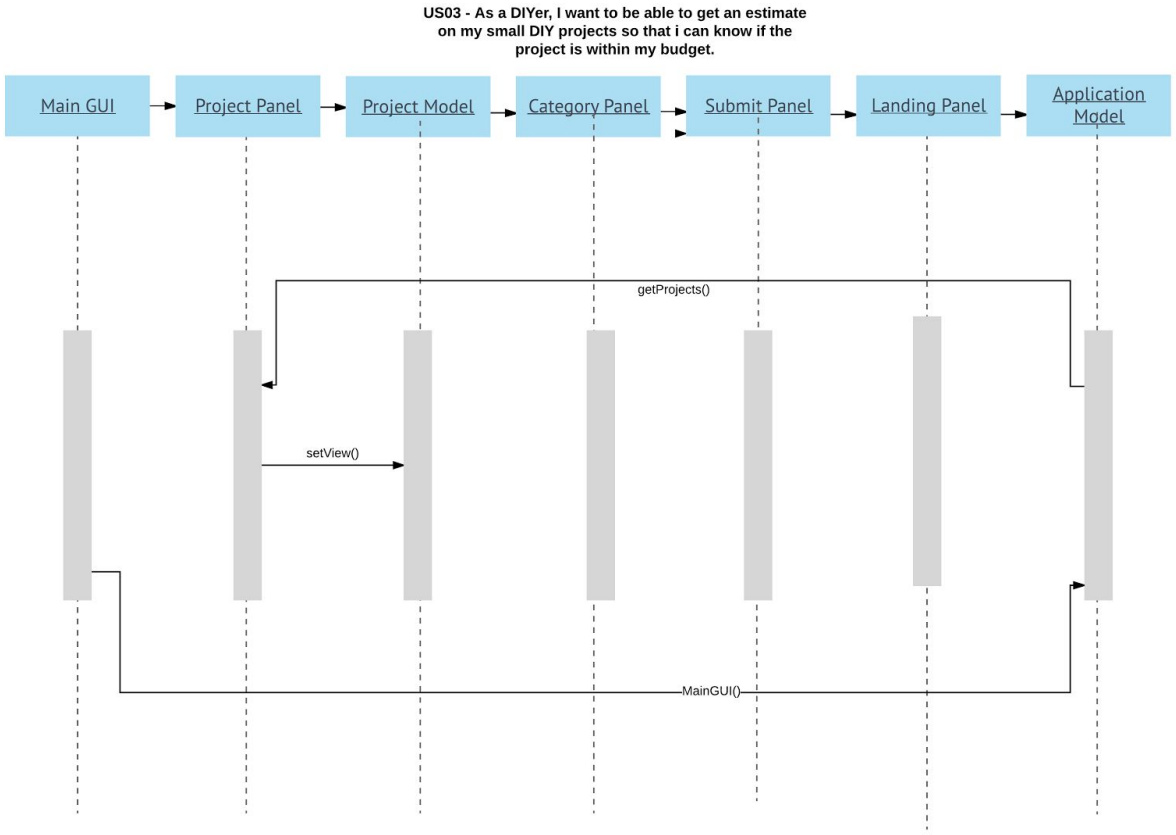
“US02 - As a DIYer, I want to be able to upload my DIY project, so that I can share with the rest of the community what I’ve been working on and inform them of my process.”



The Main Gui will launch the Landing Panel. From there the Category panel will be set up. When the user wishes to upload a project, the Category Panel will set up the Submit Panel. The Submit Panel will send the information to the Application Model, which will then save the project in the Project Model.

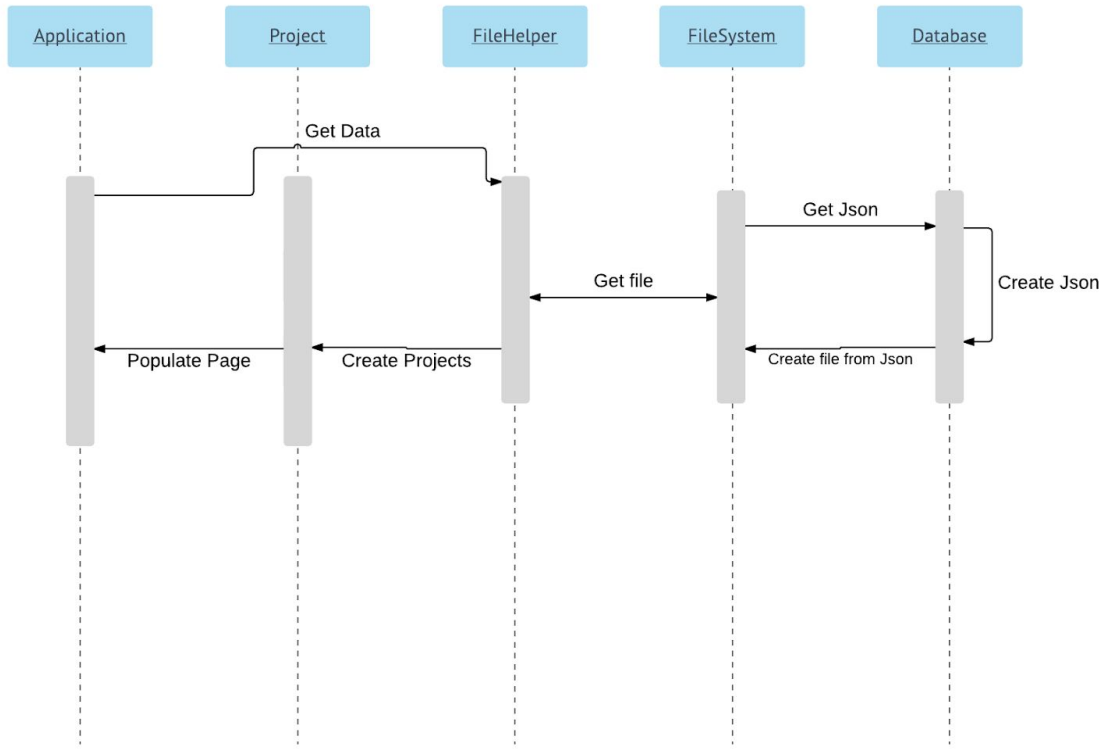
User Story SSD

“US03 - As a DIYer, I want to be able to get an estimate on my small DIY projects so that i can know if the project is within my budget.**”**



When the user starts the program the main gui will be displayed. Information will be retrieved from the Application Model and sent to the Project panel. From there the the Project Panel will populate the gui with projects. Included in each project is a cost estimate provided by the user who submitted the project.

System Startup SSD



When a user starts the application, the application first looks for projects files in the current directory. If files exists, project objects are created and populated with the data from the files assisted by the file helper. From here. the project objects will be used to populate any page that calls them. Future iterations of this project hope to also call a data base for any project files. If the database has any files that don't exist in the current directory the file helper will query the database for the new data.