Gleason Reels

State of Completion and Guide

Sect. 1) Current State

Sect. 2) Directories and getting started with Laravel

Sect. 3) Crash Course on VB

Sect. 4) Important Files/MVC basics/Source code overview

**Section 1**

**Disclaimer – I wrote this for anyone, whether they have MVC/Laravel/PHP/MySQL experience or not. That being said, I’m trying to be thorough without being absurdly verbose.**

**THIS PROJECT IS IN LARAVEL 4.1.31 – Version 4.2 documentation is very accurate for it.**

Reel Portion: mostly completed, some work needed on the actual quotes being generated (needs to look a little cleaner and include a few more pieces)

Festoon: Many of the main subroutines have been translated to functions in PHP for the back end. Front end is almost complete minus some methods that auto populate search data.

Powertrack: I have set up the skeleton for this section, but have not implemented any of the back end.

Other misc. sections have been started, some have been completed. The primary focus would be to get the three main programs up and running as the misc. sections are low priority.

There are auxiliary programs such as the conversion calculator that may not be needed anymore as this is an online tool and not a desktop application anymore. I did not get around to implementing this. This is likely also low priority.

**Section 2**

Laravel is a framework for PHP that introduces MVC. You will need to host it locally to test/debug/develop on it. Obviously, do not develop on the live directory, simply transfer your code there when it’s complete. Section 3 will cover that.

**TOOLS I RECOMMEND:**

First of all, you need to download some sort of web server stack. I recommend **XAMPP** or WAMP. It’s also a good idea to download **MySQL workbench**, the database is completely done in MySQL. **PHPStorm** was my personal IDE of choice along with **Atom** as my text editor (it’s clunky but it has a nice directory search that beats having to use GREP). Of course, you’ll also want a copy of the Gleason Program to reference as well as the source code files. Those are supplied on the **Repo here:** [**https://bitbucket.org/caigleason/gleason-reels**](https://bitbucket.org/caigleason/gleason-reels) . Laravel can be implemented in many ways but it’s well documented online better than I can do here. **Composer** is a dependency manager for Laravel. It’s really nice because if you are looking for third party tools or something - for example, I use a Debug bar- you just have to add it to the dependency list and, assuming everything is correct it should install from the repo you listed. You can list them in the composer.json file.

You’ll need some way to convert XLS files to CSV. You can just use Excel for that. Then you have to convert it to a sql file to insert into the database. Several have gone through this process already, but I’m certain there are many more that need to be converted. I usually just used an online csv to mysql converter, though you can import csv into mysql natively.

**STARTING THE SERVER:**

You’ll first need to start up XAMPP, then you’ll have to start the Apache server and the MySQL server. Next, go to the root directory (where the artisan file is stored), Shift+right click to get the “open command window here” from the list. Click it. Then enter “php artisan serve” to start the Laravel server. From here you can now view server GET info for resources. Enter localhost:8000 into your address bar for your browser. Select a link on the side panel. You should now be able to see the sign in page.

**Section 3**

**Stuff VB Does**

In the Gleason source code, you’ll run into things, specifically numbers, possibly strings with a character at the end such as %,!,#,@,&… These are Type Characters. They enforce a type because VB is a strongly typed language. Since this is being re-written in PHP you won’t have to worry about that because PHP is dynamic/weakly typed.

|  |  |  |
| --- | --- | --- |
| **Identifier type character** | **Data type** | **Example** |
| **%** | **Integer** | Dim L% |
| **&** | **Long** | Dim M& |
| **@** | **Decimal** | Const W@ = 37.5 |
| **!** | **Single** | Dim Q! |
| **#** | **Double** | Dim X# |
| **$** | **String** | Dim V$ = "Secret" |

For Loops are another example of stuff VB does differently.

For index As Integer = 1 To 5

Debug.Write(index.ToString & " ")

Next

Debug.WriteLine("")

' Output: 1 2 3 4 5

Clearly, this looks radically different than many popular C-style languages. One thing that confused me was if 1 to 5 included 5 or not. Be aware that it is.

**Gleason is an especially bad offender of using global variables everywhere. That being said, looking up variables with Atom is a nice easy process.**

**Also, unless you really want to go through the trouble, keep the goto statements until you’re done. You’ll waste a lot of time trying to re-engineer it with little to no comments on what anything does.**

**Section 4 – Important Files**

There are some files that are very crucial to getting your application to run.

Basically the site is comprised of 3 types of files: the Models, which represent a table in the database and what values are associated with them, as well as how they relate to other tables.. The Views, which are generally the frontend where you’ll work with Jquery/javascript/css/html. It is the client-facing portion. The controllers – this is where the meat of the program is stored, they generate output based on input routing which will be covered in this section.

**Database.php** in app/config – where you must set up DB credentials. Important that when you migrate changes to live you are pointing it to the live database.

**Routes.php** in app – This is where laravel processes and directs POST/GET requests. For brief example, sending a get request for results like it does when you submit your query for reel

Route::post('/reel/results', 'ReelController@showResults');

Will direct it to the ReelController in app/controllers and direct it to the method showResults().

Another thing to note is that you can access form data with Input::get(‘field’);

**ReelController.php** -- This file serves as a good template for all other main controllers like FestoonController and PowerTrakController. In it, there is the main algorithm for selecting viable reels for cables/hoses, as well as the PDF generation tool included in the project for generating reports.

**Models** in app/models – Each of these represents a database entity.

Laravel uses an includes/extends relationship with views. That means, if you want to extend a layout you have to declare it like this:

**@extends('layouts.master')**

// will extend the master layout, including everything that the master layout links to..

**@include('modal')**

// includes it as a resource, meaning you can access it and use it if you want to create a modal window

**@section('content')**

// declares what will be put into the content section

/\*html and client form stuff here\*/

**@stop**

**SOURCE CODE**

Reel source code is of course located in the Reel folder of the gleasprogvb6, as are all the programs (festoon in fest, power trak in trak..)

In each of those folders, there are a MAININP.FRM file which builds the interface for entering data for the search criteria. The REELMOD file is the backend logic. This \*MOD naming convention is the same for FESTOON and TRAK. **In Laravel terms, the .FRM files are the views** **and the \*MOD.BAS are the controllers.**

The Data folder holds the xls files that have to be converted and put into the MySQL database. As of right now, I believe I have migrated all the necessary files for Festoon (And Reel)

//ends the content section or any “section”

These will all make more sense if you look through the master.blade.php file in layouts folder. Basically, the Content section is where the meat of the front end will be, the left side navigation bar and header bar thing are statically there (at least ideally, you could make a more dynamic layout if it makes sense)

**Using composer**

To use composer for dependencies you can either add to the “require” in the composer json and run **php artisan optimize** from command line or just simply run

**composer require** *[dependancy]*

Then just run the php artisan serve command and it should update and include the dependancies.

If you run into an issue using either method, check the app.php file in app/config/ directory. For the providers array, it’ll have to have the path for the service provider and the alias.

**Other Points**

-I’ve added the sql dump to the repo. It’s the Dump20160512 file.

-PDF creation and forms are currently made in the reelController file.

-The conversion calculator logic is in Tools folder in the Conversi.frm file if it’s a requirement.