PRISM

What is Prism?

The Problem

The issue with many custom software projects is that too much time and money is spent building the individual components of the project from scratch as if each piece of the system is a completely unique idea.

To illustrate this point, let's compare your new custom software project to your dream home. You have some ideas on what you'd like: at least 4 bedrooms, high ceilings in the family room, a finished basement, etc.... What you're not thinking about is the configuration of pipes under the kitchen sink, the gauge of wiring behind the walls, or that you should ensure that the windows are made of glass so you can see through them. In fact, if you did design each of these things explicitly, you run the risk of unknowingly adding unique or incompatible options to the construction of your home when common options are readily and easily available.

It's a similar situation when creating custom software. We've found that our custom software projects often follow common patterns; eliminating the need to "reinvent the wheel" for many of the pieces of a new project. For instance, the login process for a web application has been widely standardized and there is usually no need to re-architect how this process operates. (e.g. Input your email and password and give the user a link to retrieve their forgotten password.) There may be minor variations based on explicit custom requirements, but this is only true in a small subset of new projects we encounter. Wouldn't it be great if there was an existing "login and security widget" that could be used to accelerate your software construction and make better use of your budget? What if widgets like this existed for the majority of the pieces of your application?

The Solution

Our solution to this problem is Prism. Prism is set of reusable and configurable software components (or 'modules' if we're being techie) that allow us to build the skeleton of your application very quickly and efficiently. Once that skeleton is complete, we then write additional custom code to flesh out the application to your exact specifications. This gives us the ability to put together a large part of your application quickly but still give the developers the flexibility to make the final product truly custom to your exact requirements. Some examples of these reusable modules include:

1. Contact Management
2. Workflow & Activity Management
3. File Management
4. Emailing Templates
5. Security & User Role Management
6. Payment Processing

This may sound similar to what is promised by other out-of-the-box or "configurable" systems, such as CMS, CRM, or ERP. Unfortunately, we've found that those platforms too often force you to follow a set of rules that are pre-defined and require you to adapt your process to theirs. Prism does not trap you into a finite set of options. Instead, we've specifically designed Prism to allow developers to expand the rules and processes to your exact business needs. Resulting, your new technology can be written quickly and efficiently to your exact specifications while taking advantage of the most common patterns we've encountered through a reusable framework. Of course, if a Prism module doesn't fit your needs, we simply don't use it, and we write a new module for you explicitly. This is the major difference of Prism compared to many out-of-the-box applications; Prism is built to enable your custom requirements first and foremost.

Let's Get More Technical

Prism is built in PHP on the Laravel framework and is easily managed through Composer. It utilizes a MySQL/Maria database and can be easily deployed via a standard LAMP stack. By including the Prism libraries in your software project, your project will include a set extendable database tables and REST web services to access pre-defined entities, logic, tools, and workflows.