Introduction

A service that backs up all WP posts

1. Post title
2. Post content
3. Post date

What we will be building - Laravel App

1. User registration
2. Sign in
3. Basic Auth
4. Store post from WP
5. List Posts

What we will be building - WordPress plugin

1. Connect WP plugin to Laravel app
2. On post save push data to Laravel app

Notes
This is not a talk on api security (further reading - https://laravel.com/docs/5.7/passport)
This is not a workshop on OOP, TDD, CI or any other acronyms

Laravel App

We will assume you have configured Homestead to be accessible at http://your-app.test

Login to the Vagrant box

vagrant up

vagrant ssh

cd /vagrant/

Create basic authentication

php artisan make:auth

php artisan migrate

Visit http://your-app.test/register

Create posts table, Model and Controller

php artisan make:migration create_posts_table
Php artisan make:model Post

Or

php artisan make:model Post --migration

```
$table->string('post_id')->nullable();
$table->string('post_title')->nullable();
$table->longText('post_content')->nullable();
$table->dateTime('post_date')->nullable();
$table->integer('user_id')->nullable();
```

php artisan migrate

php artisan make:controller PostsController --resource

Add posts route to routes/web.php

```
Route::get('/posts', 'PostsController@index')->name('posts');
```

http://your-app.test/posts

Create posts api endpoint in routes/api.php

```
Route::middleware('api')->get('/posts', 'PostsController@apiIndex');
```

http://your-app.test/api/posts

Add very basic api token based auth

php artisan make:migration add_token_to_users_table

Add api_token field to migration

```
$table->string('api_token', 60)->nullable();
```

```
$table->dropColumn('api_token');
```

Updated Register Process to set API key

```
    /**
     * Create an api_token based on the registered users email
     *
     * @param $email
     *
     * @return bool|string
     */
    private function generateApiToken($email){
        $alpha_numeric = preg_replace("/[^A-Za-z0-9 ]/", '', bcrypt($email.str_random(60)));
        $api_token = substr($alpha_numeric, 0, 60);
        return $api_token;
    }
```

Add api_token to the User fillable variable

php artisan migrate:refresh - to clear out the database

Show the api token to the logged in user

HomeController

```
use Illuminate\Support\Facades\Auth;
```

Update the index method

```
    public function index()
    {
        $user = Auth::user();
        return view('home', compact('user'));
    }
```

Update the home view

```
                <div class="card-body">
                    Your API token is {{$user->api_token}}
                </div>
```

Update the API route to require authentication

```
Route::middleware('auth:api')->get('/posts', 'PostsController@apiIndex');
```

Test updated api in Postman with api_token being sent in the request

WordPress plugin

Add the settings page to store the users api token

https://codex.wordpress.org/Settings_API#Adding_Setting_Fields

Add the hook to save the posts data

https://developer.wordpress.org/reference/hooks/save_post/

```
add_action( 'save_post', 'sbj_save_post', 11, 2 );
add_action( 'post_updated', 'sbj_save_post', 11, 2 );
```

Add a function to push the data to our app

https://developer.wordpress.org/reference/functions/wp_remote_post/

```
$app_response = wp_remote_post(
        $api_url,
        array(
                'timeout' => 45,
                'body'    => $post_body,
        )
);
```

Laravel App

Setup posts POST API endpoint and Controller method

```
Route::middleware('auth:api')->post('/posts', 'PostsController@apiStore');
```

Update Post Model with fillables

```
protected $fillable = [
    ‘user_id’,
    'post_id',
    'post_title',
    'post_content'
  ];
```

Use Post Model in Posts Controller

use App\Post;
use Illuminate\Support\Facades\Auth;

Update apiStore method to a) check for an Authed user and b) create a Post

Doing it this way adds the post twice, because both actions trigger

So we should check if the Post exists OR create a PATCH endpoint

I prefer the check post exists method, personal choice

Don’t forget the user_id

Set up posts index to a show list of posts

Update Posts index method to get all Posts and display posts view

Create Posts index view

https://laravel.com/docs/5.7/blade