Asgn03: Passing values to views

# Objectives

* MSVSC (Microsoft Visual Studio Code) extensions
* Pass variable values to a view
* Pass array value to a view
* Pass database values to a view
* Use Git to save your project’s state
* Use Git to create a new branch

# Introduction

Continue working in your local test environment.

# MSVSC Extensions

A quick detour from Laravel to help improve efficiency as you write code. MSVSC (from now on just Code) has a lot of great extensions. Let’s start with an HTML boilerplate for Laravel Blade. Watch the screencast on the HTML extension.

## HTML:5

Install the **html:5 boilerplate** if not already installed.

Install any of the **lorum ipsum** extensions.

Optional – install any other extensions you may find helpful for this class or any other coding you happened to be working on.

# Pass variable values to a view

In the last assignment, we looked at passing an array to a view but what if we wanted to pass just a variable? Here is a code snippet from the [Laravel Docs site on creating Views](https://laravel.com/docs/5.4/views). In addition to your text, Laravel Docs is a great place to look up any information on Laravel.

## Create a fistName and LastName

* In the web.php route create two variables named firstName and lastName.
* Create a view named greeting.blade.php that displays the phrase “Welcome firstName, lastName”.

# Pass array values to a view

We did this in the previous lesson. Please refer to it if you need a refresher

## Birds

* In the web.php route create a new array named birds.
* List at least three birds in your array.
* Create a new view named birds.blade.php that displays the route.

# Set up the database environment

Create an empty database named test\_db if you haven’t already.

Open the .env file in the edior and change the following lines so they match your database login criteria.

DB\_CONNECTION=mysql

DB\_HOST=127.0.0.1

DB\_PORT=3306

DB\_DATABASE=homestead

DB\_USERNAME=homestead

DB\_PASSWORD=secret

Next open the config->database.php file and make the necessary changes to the section that looks like

'mysql' => [

'driver' => 'mysql',

'host' => env('DB\_HOST', '127.0.0.1'),

'port' => env('DB\_PORT', '3306'),

'database' => env('DB\_DATABASE', 'forge'),

'username' => env('DB\_USERNAME', 'forge'),

'password' => env('DB\_PASSWORD', ''),

'unix\_socket' => env('DB\_SOCKET', ''),

'charset' => 'utf8mb4',

'collation' => 'utf8mb4\_unicode\_ci',

'prefix' => '',

'strict' => true,

'engine' => null,

],

# Test with a migration

Now that you have configured the database environment, you should be able to run a migration and create the tables in your project.

Open Git Bash

Navigate to your project (this assumes your project is in a folder named test)

### Windows

$ cd c:/code/test

### Mac

$ cd ~/code/test

Run the command

$ php artisan migrate

# Pass values from a table to a view

Open the database->migrations folder in your editor and examine the two existing files.

create\_users\_table

reset\_existing\_password\_table.

These files are automatically generated when you create a Laravel project. Now they are just code so we need to run an artisan command to generate the code into an actual table.

Open GitBash

$ php artisan migrate

Migration table created successfully.

Migrating: 2014\_10\_12\_000000\_create\_users\_table

Migrated: 2014\_10\_12\_000000\_create\_users\_table

Migrating: 2014\_10\_12\_100000\_create\_password\_resets\_table

Migrated: 2014\_10\_12\_100000\_create\_password\_resets\_table

Check out the result in MySQL command line or Workbench. Mac users should check out SQLPro as an alternative.

You should see three tables

migrations

password\_resets

users

## Add a table for bird data

Now that you know how to create and migrate tables, do one of your own. At the Bash command line

Navigate to your project. You can test to see where you are by using

$ pwd

If it doesn’t show c:/code/test (Windows) or ~/code/test then run the commands as you did earlier in this exercise to navigate to the appropriate directory.

Before we create a new table, take a moment to use the php artisan help function to see what is available.

$ php artisan help make:migration

Notice the option

--create[=CREATE] The table to be created.

We can use that option with the following to create a table within our schema. Try the following to create a new birds table.

$ php artisan make:migration create\_birds\_table --create=birds

Here is [link with additional information on creating tables from Laravel’s site](https://laravel.com/docs/5.4/migrations#creating-tables)

At this point we have only created the schema within Laravel. Let’s check it out before running the migration. Navigate to **database->migrations->date\_crteate\_birds\_table.** It should look like

It should look like

<?php

use Illuminate\Support\Facades\Schema;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Database\Migrations\Migration;

**class** CreateBirdsTable **extends** Migration

{

*/\*\**

*\* Run the migrations.*

*\**

*\* @return void*

*\*/*

**public** **function** up()

{

Schema::create('birds', **function** (**Blueprint** $table) {

$table->increments('id');

$table->timestamps();

});

}

*/\*\**

*\* Reverse the migrations.*

*\**

*\* @return void*

*\*/*

**public** **function** down()

{

Schema::dropIfExists('birds');

}

}

There are two methods generated for us, the **up** and **down** methods. Use the **users** table as an example to generate code for the birds table. Place the code in the **up** method. For now, let’s keep it easy and just add the common name (species) of the bird.

Add the following line after the line containing the increments

$table->text('species');

Now we are set to run the migration and update the database again

$ php artisan migrate

Migrating: 2017\_09\_11\_121844\_create\_birds\_table

Migrated: 2017\_09\_11\_121844\_create\_birds\_table

Now we are all set and have created a new table in our database. Again, you can check it out in MySQL.

* Leave the id field blank as it is auto-incremented
* Add three birds (any three will do)
* In Workbench use \func now() for the dates. In SequelPro use now()

Side note: If you run into an error when trying to migrate, try running the following command first

$ composer dump-autoload

# Display the birds from the table

## Create a new route

Create a new route in the web.php file by adding the following code to the bottom

Route::get('birds', **function**() {

$birds = DB::table('birds')->get();

return view('birds', compact('birds'));

});

The **get()** method returns all of the rows in the database. In this case it will return all of the birds.

## Create a new view

Create a new view named **birds.blade.php**. Remember that the **blade.php** part is necessary in order for Laravel to work.

Here are some hints for your birds view

You will need

@foreach($birds as $bird) // don’t forget the @endforeach

Show the list of birds in the database with the following line

{{ $bird->species }}

This is different than the array in that we need to specify the field name, species.

Tidy it up by making it an ordered list.

Finally, remember that you created a new blade so you will need to reference that in the URL with

test.dev/birds

Your output should look something similar to this

* Rufus-sided Towhee
* Eastern Bluebird
* Red Headed Woodpecker

# A little Git

If you got everything running then this is a good place to save your test project using version control. First check it out with git status just to check it out.

$ git status

On branch master

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git checkout -- <file>..." to discard changes in working directory)

modified: config/database.php

modified: routes/web.php

Untracked files:

(use "git add <file>..." to include in what will be committed)

database/migrations/2017\_09\_11\_121844\_create\_birds\_table.php

resources/views/birds.blade.php

resources/views/greeting.blade.php

resources/views/practice.blade.php

no changes added to commit (use "git add" and/or "git commit -a")

There is a lot of information here. A couple of items to notice are

* You are on the master branch (new feature branch coming soon!)
* Which files have been modified
* Which file are untracked (new files that we added)

Next we want to put the files in the staging area with the command

$ git add .

Check it out with git status again to see what has happened

$ git status

On branch master

Changes to be committed:

(use "git reset HEAD <file>..." to unstage)

modified: config/database.php

new file: database/migrations/2017\_09\_11\_121844\_create\_birds\_table.php

new file: resources/views/birds.blade.php

new file: resources/views/greeting.blade.php

new file: resources/views/practice.blade.php

modified: routes/web.php

Finally commit it with

$ git commit -m "completed the bird database. Next is additional blade emplating"

[master 66b6dad] completed the bird database. Next is additional blade emplating

6 files changed, 86 insertions(+), 2 deletions(-)

create mode 100644 database/migrations/2017\_09\_11\_121844\_create\_birds\_table.php

create mode 100644 resources/views/birds.blade.php

create mode 100644 resources/views/greeting.blade.php

create mode 100644 resources/views/practice.blade.php

Make sure you include a message with the –m “…”. We will go over in class what happens if you don’t.

If you want to see what git has been up to, then use the git log command. Here is what I have so far.

$ git log

commit 66b6dadb5b4ffc501d97135ba18f0f709b8c0559

Author: Charlie Wallin <charliekwallin@gmail.com>

Date: Mon Sep 11 09:42:31 2017 -0400

completed the bird database. Next is additional blade emplating

commit f7f694bc9d94bb6c61146706d6d437bdc7e9af00

Author: Charlie Wallin <charliekwallin@gmail.com>

Date: Thu Aug 31 15:47:58 2017 -0400

# Submitting Your Work

There is nothing to turn in but make sure you ***show your instructor*** that you have completed the following

* Added the Laravel Blade Snippet to Code (20 points)
* Created a view called greeting.blade.php that displays Hello, name (20 points)
* Created a view that shows your tasks array (20 points)
* Database updated with a few birds in it (20 points)
* Laravel displays the birds in test.dev/birds (20 points)