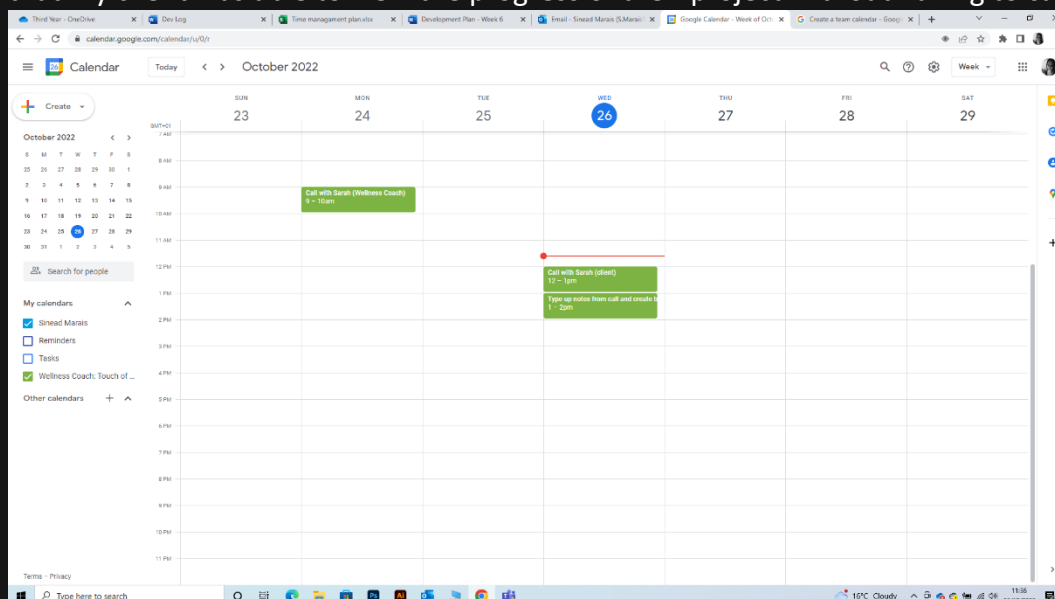


## Development Log/Bibliography:

### The beginning:

- To start this project I read through the handbook to understand my module aims and learning outcomes. I explored what my options were in terms of the clients who I may be able to provide services for and looked to see what different types of placements and freelance projects are available. There were a few options organised as client projects through university and someone reached out to my social media account where I showcase my projects from my time at university. However, these were either not right for me or the client took too long to get back to me.
- It was then when I found Sarah, the owner of 'Touch of Grace'. I arranged a call with her to fully understand the scope of her project. From there I began the project with continuous calls throughout project.
- I found a web source which provided guidance on how to create a website with packages and a payment functionality, this was using a Laravel Framework.
- Research into Laravel was undertaken (<https://laravel.com/>), I had used Laravel before so it is nice to be able to develop my knowledge further.
- Next, I looked at the client's (Sarah) existing branding and marketing to see how I could further improve this and create a wireframe of the website that I can show to receive feedback.
- Throughout my studies I have created an Excel Workbook to manage my projects and time, however I did not feel as though this was very professional or the best method to communicate online with my client so I created a shared Google Calendar. This meant that my client was able to view the progress of their project without having to call me.



As well as this, I decided to create a rough plan for the project as a flow diagram to keep myself accountable and on track.

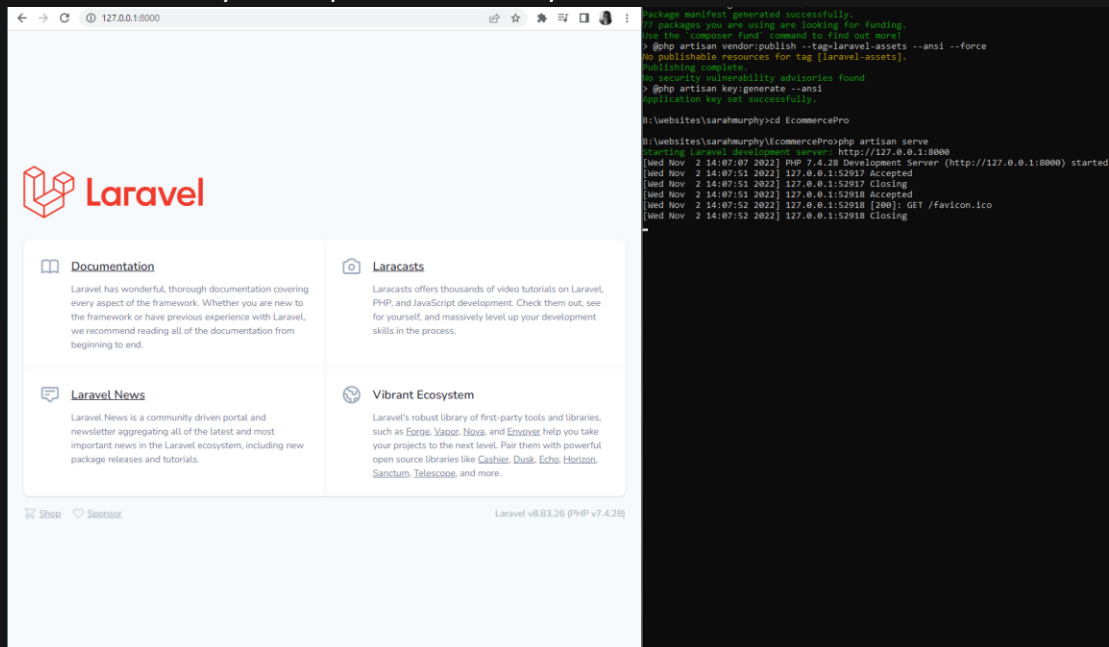
## The Set up:

- After reminding myself of Laravel, I downloaded composer and made sure that I still have, Node.js and XAMPP installed.
  - MAMP: local server environment
  - XAMPP: open-source, cross-platform web server solution stack package. As well as other uses, it interprets scripts written in PHP.
- Next, I installed Laravel using the CMD.

```
Microsoft Windows [Version 10.0.19044.2130]
(c) Microsoft Corporation. All rights reserved.

C:\Users\smara>
C:\Users\smara>composer create-project laravel/laravel example-app
Creating a "laravel/laravel" project at "./example-app"
Info from https://repo.packagist.org: #StandWithUkraine
Cannot use laravel/laravel's latest version v9.3.10 as it requires php ^8.0.2 which is not satisfied by your platform.
Installing laravel/laravel (v8.6.12)
 - Downloading laravel/laravel (v8.6.12)
 - Installing laravel/laravel (v8.6.12): Extracting archive
Created project in C:\Users\smara\example-app
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
Updating dependencies
Lock file operations: 106 installs, 0 updates, 0 removals
 - Locking asm89/stack-cors (v2.1.1)
 - Locking brick/math (0.9.3)
 - Locking dflydev/dot-access-data (v3.0.2)
 - Locking doctrine/inflector (2.0.6)
 - Locking doctrine/instantiator (1.4.1)
 - Locking doctrine/lexer (1.2.3)
 - Locking dragonmantank/cron-expression (v3.3.2)
 - Locking egulias/email-validator (2.1.25)
 - Locking facade/flare-client-php (1.10.0)
 - Locking facade/ignition (2.17.6)
 - Locking facade/ignition-contracts (1.0.2)
 - Locking fakerphp/faker (v1.20.0)
 - Locking filp/whoops (2.14.5)
 - Locking fruitcake/laravel-cors (v2.2.0)
 - Locking graham-campbell/result-type (v1.1.0)
 - Locking guzzlehttp/guzzle (7.5.0)
 - Locking guzzlehttp/promises (1.5.2)
 - Locking guzzlehttp/psr7 (2.4.3)
 - Locking hamcrest/hamcrest-php (v2.0.1)
 - Locking laravel/framework (v8.83.26)
 - Locking laravel/sail (v1.16.2)
 - Locking laravel/sanctum (v2.15.1)
 - Locking laravel/serializable-closure (v1.2.2)
 - Locking laravel/tinker (v2.7.2)
 - Locking league/commonmark (2.3.6)
 - Locking league/config (v1.1.1)
 - Locking league/flysystem (1.1.10)
 - Locking league/mime-type-detection (1.11.0)
 - Locking mockery/mockery (1.5.1)
 - Locking monolog/monolog (2.8.0)
 - Locking myclabs/deep-copy (1.11.0)
 - Locking nesbot/carbon (2.62.1)
 - Locking nette/schema (v1.2.2)
 - Locking nette/utils (v3.2.8)
 - Locking nikic/php-parser (v4.15.1)
 - Locking nunomaduro/collision (v5.11.0)
 - Locking opis/closure (3.6.3)
 - Locking phar-io/manifest (2.0.3)
 - Locking phar-io/version (3.2.1)
 - Locking phpoption/phpoption (1.9.0)
 - Locking phpunit/php-code-coverage (9.2.18)
 - Locking phpunit/php-file-iterator (3.0.6)
 - Locking phpunit/php-invoker (3.1.1)
 - Locking phpunit/php-text-template (2.0.4)
 - Locking phpunit/php-timer (5.0.3)
```

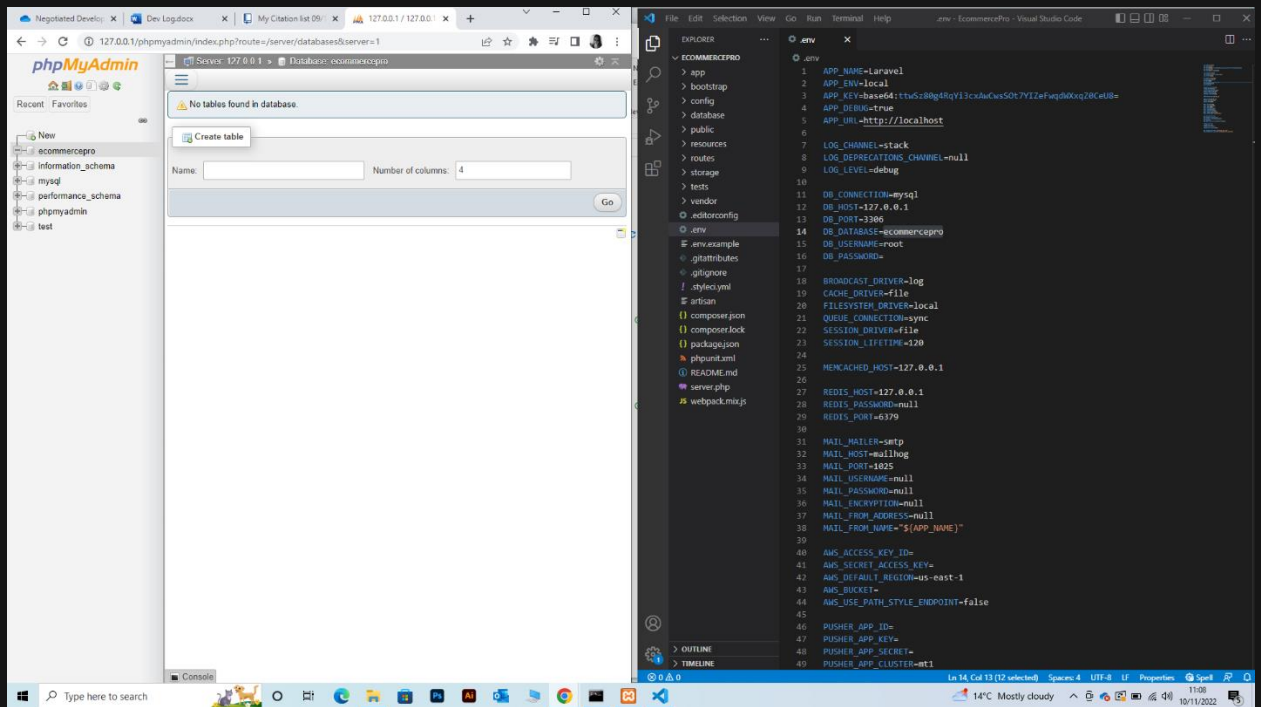
- Then I started my development server as you can see here.



- Whilst I started the build of the website, I was in communication with the client via email and we scheduled a call to finalise the mockup and discuss the type of imagery she would like. Once she had decided she sent them via email.

#### Week 8:

- This week I setup the database using phpMyAdmin. I did this by changing the name of the database in the '.env' file in the code editor and made this correlate with the new database created in phpMyAdmin. (database\_Setup.png)



- I also created Login section for my client. To do this the first thing that I did was install a Laravel Package called Jetstream. I installed and ran the package using 'npm install' and 'npm run dev'.

```

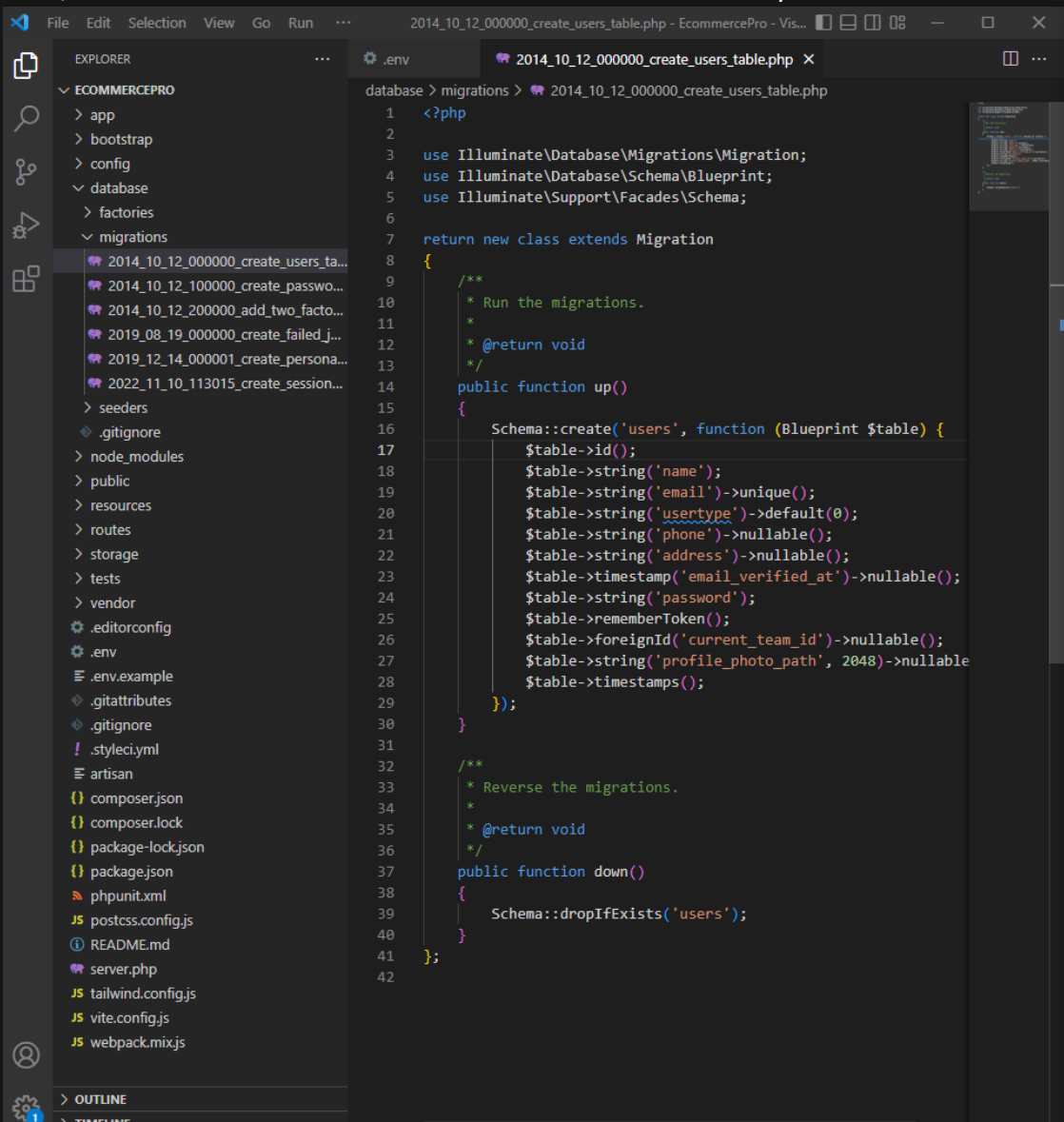
Select C:\Windows\System32\cmd.exe - composer require laravel/jetstream
Microsoft Windows [Version 10.0.19044.2130]
(c) Microsoft Corporation. All rights reserved.

B:\websites\sarahmurry\ecommercepro>composer require laravel/jetstream
Info from https://repo.packagist.org: #StandWithUkraine
Cannot use laravel/jetstream's latest version v2.12.5 as it requires php ^8.0.2 which is not satisfied by your platform.

Using version ^2.9 for laravel/jetstream
./composer.json has been updated
Running composer update laravel/jetstream
Loading composer repositories with package information
Updating dependencies
Lock file operations: 9 installs, 0 updates, 0 removals
  - Locking bacon/bacon-qr-code (2.0.7)
  - Locking dasprid/enum (1.0.3)
  - Locking jaybizzle/crawler-detector (v1.2.112)
  - Locking jenssegers/agent (v2.6.4)
  - Locking laravel/fortify (v1.13.7)
  - Locking laravel/jetstream (v2.9.0)
  - Locking mobiledetect/mobiledetectlib (2.8.41)
  - Locking paragonie/constant-time-encoding (v2.6.3)
  - Locking pragmarx/google2fa (v8.0.1)
Writing lock file
Installing dependencies from lock file (including require-dev)
Package operations: 9 installs, 0 updates, 0 removals
  - Downloading dasprid/enum (1.0.3)
  - Downloading bacon/bacon-qr-code (2.0.7)
  - Downloading jaybizzle/crawler-detector (v1.2.112)
  - Downloading paragonie/constant-time-encoding (v2.6.3)
  - Downloading pragmarx/google2fa (v8.0.1)
  - Downloading laravel/fortify (v1.13.7)
  - Downloading mobiledetect/mobiledetectlib (2.8.41)
  - Downloading jenssegers/agent (v2.6.4)
  - Downloading laravel/jetstream (v2.9.0)
  - Installing dasprid/enum (1.0.3): Extracting archive
  - Installing bacon/bacon-qr-code (2.0.7): Extracting archive
  - Installing jaybizzle/crawler-detector (v1.2.112): Extracting archive
  - Installing paragonie/constant-time-encoding (v2.6.3): Extracting archive
  - Installing pragmarx/google2fa (v8.0.1): Extracting archive
  - Installing mobiledetect/mobiledetectlib (2.8.41): Extracting archive
  - Installing jenssegers/agent (v2.6.4): Extracting archive
  - Installing laravel/jetstream (v2.9.0): Extracting archive
Package suggestions were added by new dependencies, use 'composer suggest' to see details.
Package swiftmailer/swiftmailer is abandoned, you should avoid using it. Use symfony/mailer instead.
Generating optimized autoload files

```

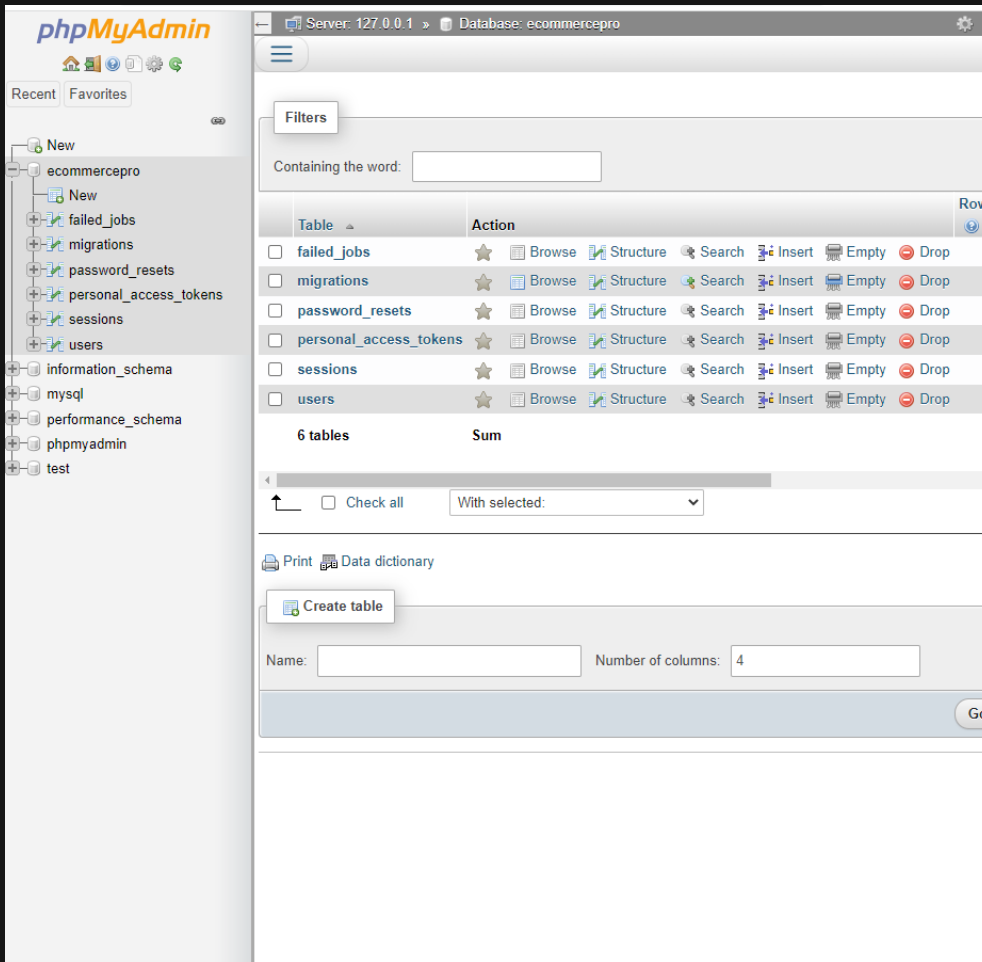
- Next, I created user column fields in the database inside the my text editor.



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with folders like 'app', 'bootstrap', 'config', 'database', 'factories', 'migrations', 'seeders', and files like '.gitignore', 'composer.json', 'package-lock.json', 'phpunit.xml', 'postcss.config.js', 'README.md', 'server.php', 'tailwind.config.js', 'vite.config.js', and 'webpack.mix.js'. The code editor shows a PHP migration file named '2014\_10\_12\_000000\_create\_users\_table.php'. The code is as follows:

```
1 <?php
2
3 use Illuminate\Database\Migrations\Migration;
4 use Illuminate\Database\Schema\Blueprint;
5 use Illuminate\Support\Facades\Schema;
6
7 return new class extends Migration
8 {
9     /**
10      * Run the migrations.
11      *
12      * @return void
13      */
14     public function up()
15     {
16         Schema::create('users', function (Blueprint $table) {
17             $table->id();
18             $table->string('name');
19             $table->string('email')->unique();
20             $table->string('usertype')->default(0);
21             $table->string('phone')->nullable();
22             $table->string('address')->nullable();
23             $table->timestamp('email_verified_at')->nullable();
24             $table->string('password');
25             $table->rememberToken();
26             $table->foreignId('current_team_id')->nullable();
27             $table->string('profile_photo_path', 2048)->nullable();
28             $table->timestamps();
29         });
30     }
31
32     /**
33      * Reverse the migrations.
34      *
35      * @return void
36      */
37     public function down()
38     {
39         Schema::dropIfExists('users');
40     }
41 };
42
```

- Then I used the command 'php artisan migrate', by doing this it migrate all of the tables to our database.



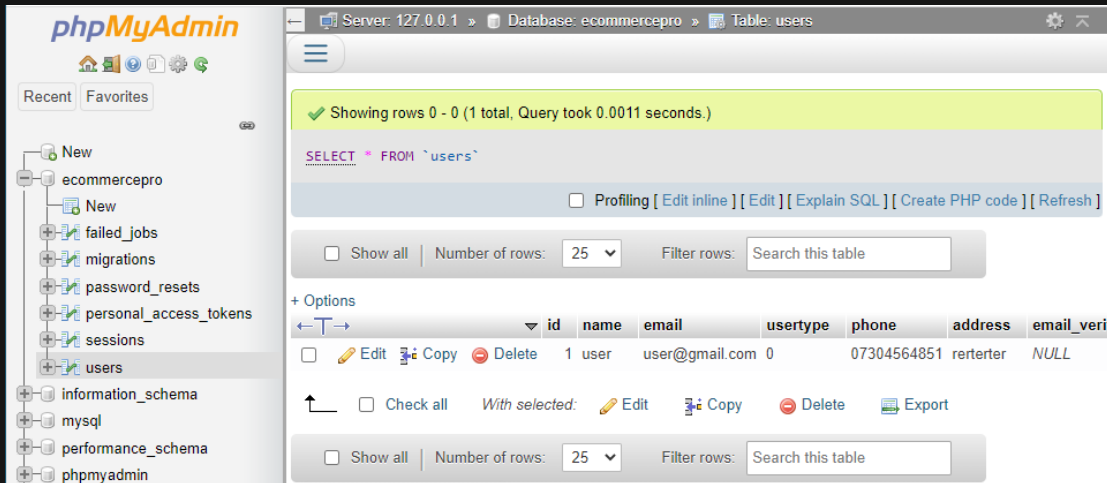
- From doing the above and with the install of Jetstream, I added a 'Login' and 'Register' option on the frontend of my development server!

The screenshot shows the Laravel website homepage. At the top left is the Laravel logo. Below it are four main sections:
 

- Documentation**: Laravel has wonderful, thorough documentation covering every aspect of the framework. Whether you are new to the framework or have previous experience with Laravel, we recommend reading all of the documentation from beginning to end.
- Laracasts**: Laracasts offers thousands of video tutorials on Laravel, PHP, and JavaScript development. Check them out, see for yourself, and massively level up your development skills in the process.
- Laravel News**: Laravel News is a community driven portal and newsletter aggregating all of the latest and most important news in the Laravel ecosystem, including new package releases and tutorials.
- Vibrant Ecosystem**: Laravel's robust library of first-party tools and libraries, such as [Forge](#), [Vapor](#), [Nova](#), and [Envoyer](#) help you take your projects to the next level. Pair them with powerful open source libraries like [Cashier](#), [Dusk](#), [Echo](#), [Horizon](#), [Sanctum](#), [Telescope](#), and more.

 At the bottom left, there are links for 'Shop' and 'Sponsor'. At the bottom right, the version information 'Laravel v8.83.26 (PHP v7.4.28)' is displayed.

- To modify the fields on the register page, I used the 'register.blade.php' and the 'user.php' files. I copied the existing fields such as 'email' and I re-created ones for both 'password' and 'address'.
- By doing so, it has allowed me to create a test account as you can see below.



- I added this Ecommerce Project to Git Hub, it is an industry stand and I have done this with all of my projects whilst at university. I can now work collaboratively with my lecturer so when I have scheduled 1-2-1 calls because I have run into an issue, he is able to view my code and help me correct the error.
- I can see a CSS error where I did not link the CSS file correctly.



- I added another user under the name 'admin', I then changed the user type to 1 in the PHP my admin making it the admin login. (user\_admin.png). This allows me to access the admin dashboard.

The screenshot shows the phpMyAdmin interface with the following details:

- Header:** Server: 127.0.0.1 » Database: ecommercepro » Table: users
- Left Panel:** A tree view showing the database structure. The 'ecommercepro' database is expanded, and the 'users' table is selected.
- SQL Query:** An UPDATE statement is shown: `UPDATE `users` SET `usertype` = '1', `email_verified_at` = NULL, `two_factor_confirmed_at` = NULL WHERE `users`.`id` = 2;`
- Message:** A green banner indicates "1 row affected."
- Query Results:** A SELECT query is shown: `SELECT * FROM `users``. The results table contains two rows:

	id	name	email	usertype	phone	address	email_
<input type="checkbox"/>	1	user	user@gmail.com	0	07304564851	rerterter	NULL
<input type="checkbox"/>	2	admin	admin@gmail.com	1	01234567897	asdsadas	NULL
- Options:** Includes a search filter, "Number of rows" set to 25, and "Sort by key" set to None.
- Query results operations:** Includes buttons for Print, Copy to clipboard, Export, Display chart, and Create view.
- Bookmarking:** A section for "Bookmark this SQL query" with a label input field and a checkbox for "Let every user access this bookmark".



- Then I went into the 'RouteServiceProvider.php' file and changed the direction to 'redirect' instead of 'dashboard'. I did this by creating a route in the 'web.php' file.

```
routes > web.php
1  <?php
2
3  use Illuminate\Support\Facades\Route;
4  use App\Http\Controllers\HomeController;
5
6  /*
7  |-----|
8  | Web Routes
9  |-----|
10 |
11 | Here is where you can register web routes for your application. These
12 | routes are loaded by the RouteServiceProvider within a group which
13 | contains the "web" middleware group. Now create something great!
14 |
15 */
16
17 Route::get('/', function () {
18     return view('welcome');
19 });
20
21 Route::middleware([
22     'auth:sanctum',
23     config('jetstream.auth_session'),
24     'verified'
25 ]->group(function () {
26     Route::get('/dashboard', function () {
27         return view('dashboard');
28     }->name('dashboard'));
29
30     route::get('/redirect',[HomeController::class,'redirect']);
31 });
32
```

- Once this was done I added a 'HomeController' Route but because I did not yet have a home controller I created one in the command line.
- After the HomeController.php file was created I wrote a function for the 'redirect' command inside that file. This means that when the user is trying to login in, the function redirect will check which user type it is. The way that user type is defined is from the PHP my Admin that I changed previously.

```

pp > Http > Controllers > HomeController.php
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class HomeController extends Controller
8  {
9      public function redirect()
10     {
11         $usertype=Auth::user()->usertype;
12     }
13 }
14

```

- The code below shows the function and if statement that define the admin user as 1 in the code rather than PHP my Admin and the if statement say if it is a user with the value of 1, then return to the admin dashboard area. Then the else statement says, for any other user type, return to the regular dashboard area.

```

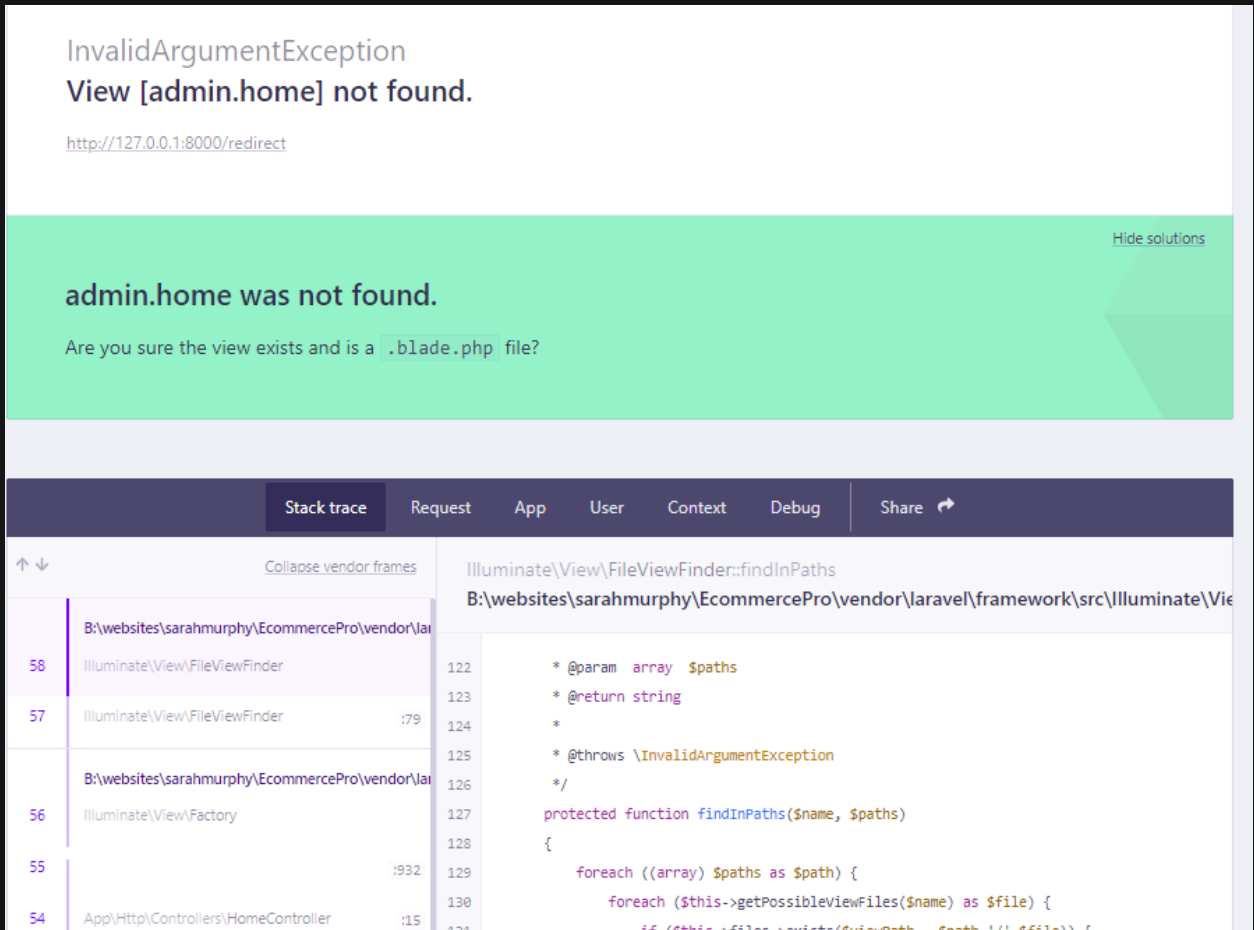
app > Http > Controllers > HomeController.php
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6  use Illuminate\Support\Facades\Auth;
7
8  class HomeController extends Controller
9  {
10     public function redirect()
11     {
12         $usertype=Auth::user()->usertype;
13         if($usertype=='1')
14         {
15             return view('admin.home');
16         }
17         else
18         {
19             return view('dashboard');
20         }
21     }
22 }
23

```

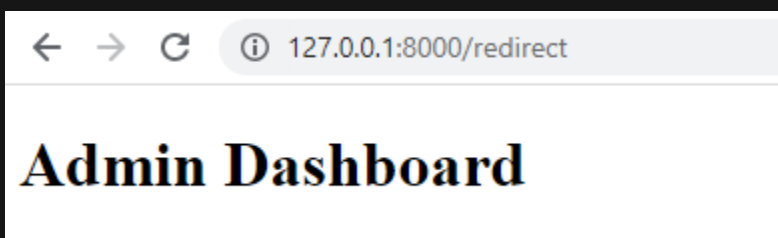
- This means that now when I login with the user login, it tasks me to the user dashboard area.

### Setting up the Dashboards:

- However, when I try login as an admin, I receive an error saying that the page is not found.



- To fix this I created an admin.home file with a H1 title so there is content in it to register that the Admin Dashboard exists.
- The screenshot below shows the redirect function being called from 'web.php'.



- Next I created a 'logout' section from the Admin dashboard, to do this copied this existing code '<x-app-layout></x-app-layout>' into the admin.home page just created. The 'app layout' code is responsible for creating a logout option.

- Whilst I was here I added in a temporary logo and navigation links to check that it worked, as well as to give the client an first glimpse of what the website may look like.

#### Week 11:

- I had been experiencing a consistant CSS error, I followed the advice suggested from Stack Overflow to correct this. (Source: <https://stackoverflow.com/questions/73180945/jetstream-css-and-js-not-working-and-showing-viteresources-css-app-css-re>).
- I used a template from a website (<https://themewagon.com/themes/famms-free-responsive-bootstrap-4-e-commerce-website-template/>) which gave me template files to use. By doing so, I now have created a template HTML page to use in Laravel. I used the code from the template and created a my own template called 'userpage.blade.php'.
- To check that this was working I changed the route to the home page, calling the 'index' rather than the standard 'Laravel Welcome'.

```
route::get('/redirect',[HomeController::class,'index']);

Route::middleware([
    'auth:sanctum',
    config('jetstream.auth_session'),
    'verified'
])->group(function () {
    Route::get('/dashboard', function () {
        return view('dashboard');
    })->name('dashboard');

    route::get('/redirect',[HomeController::class,'redirect']);
});
```

- Next, I created a function in the 'homecontroller' folder which calls the view to my template page 'userpage.blade.php'.
- There was an error with the styling being pulled across (template\_css\_error.png), the reason which this was happening was because although there were stylesheets being referenced it was not the one in the correct directory. Instead, they now read 'home/css/...
- I also did this where the 'script' tags are to make sure that the files from the home folder were being pulled. To avoid having to do this with every image, I moved the 'image folder' outside of the 'home folder'.

- Within the 'userpage.blade.php' there was lots of lines of code, making it hard to read through it all. To make this easier for myself as a developer, I created components in 'blade' files.
  - I sectioned off the header section by creating a 'header.blade.php' file. The I referenced that new page with the header details with this line of code '@include("home.header")'.
  - I duplicated that process for other parts of the page that I wanted to create as components.
- Next I edited the HTML code in the header.blade.php page created to make it more custom. I added a login and register button using a Bootstrap class.

```
<li class="nav-item">
  | <a class="btn btn-primary" href="contact.html">Login</a>
</li>

<li class="nav-item">
  | <a class="btn btn-success" href="contact.html">Register</a>
</li>
```

- In order to get my Login and Register page's links to work I needed to change the href. I used the code that I know works for the 'welcome.blade.php' file (href="{{ route('login') }}" / ref="{{ route('register') }}"). This means that a user is now able to use the login and register options.
- Next, I wanted the 'login' button to disappear when a user is logged in and a 'logout' button to appear. I did this by redirecting to the 'userpage' and no longer the 'dashboard' page.

```
public function redirect()
{
    $usertype=Auth::user()->usertype;
    if($usertype=='1')
    {
        return view('admin.home');
    }
    else
    {
        return view('home.userpage');
    }
}
```

- However, at this point I have not yet created a 'logout' option. To do this I created an 'auth' condition in the header component as well as, as if statement. It is saying that if

the user is logged in 'show logout button' and if it is anything else 'show login/register options'.

```
@if (Route::has('login'))

@auth

<li class="nav-item">
| <a class="btn btn-primary" id="login_css" href="{{ route('login') }}">Logout</a>
</li>

@else

<li class="nav-item">
| <a class="btn btn-primary" id="login_css" href="{{ route('login') }}">Login</a>
</li>

<li class="nav-item">
| <a class="btn btn-success" href="{{ route('register') }}">Register</a>
</li>

@endauth
@endif
```

#### Creating the Admin Dashboard:

- Next I used a free template to build out the admin dashboard. (<https://technext.github.io/corona-free-dark-bootstrap-admin-template/index.html>). I downloaded it and then copied the files into my project's files.
- Just like before, I made sure that the correct directories were being referenced and I separated the HTML template into components to make it easier to develop.
- Although the template comes with an 'log out' option, it does not work. I reused the code '<x-app-layout></x-app-layout>' to create the logout button. Minus the slight styling issue, a user is now able to log in and out as an admin.

#### Admin Dashboard/ Database:

- Next I am laying out and designing a category page for the Admin Dashboard, this means that I am creating options an 'admin' would see when they're logged in which is different to the 'User's dashboard'. I changed one of the existing elements on the 'sidebar' to 'Category' and to make this link take the user to the desired page I gave it a route using this code '<a class="nav-link" href="{{url('view\_category')}}"' . Then I actually created the route in the 'web.php' file.

```
// Admin controller instead of home
route::get('view_category', [AdminController::class, 'view_category']);
```

- And I made an 'Admin Controller' using the CMD (cmd\_admin\_controller.png), this then auto created a file in my project in 'app/Http/AdminController.php'. I linked this to the 'web.php' file using this 'use App\Http\Controllers\AdminController;' and finally, I created a function to return the user to the 'admin category' section.

```
class AdminController extends Controller
{
    public function view_category ()
    {
        return view('admin.category');
    }
}
```

- To get the Admin Dashboard view in the 'Category page' I copied the code from the 'home.blade.php' file showing php's '@include' statements. Once the data was being shown I removed the body section.
- This empty section is where I added some input boxes for a form. I created the form using HTML and it was styled with CSS.

```
<div class="main-panel">
  <div class="content-wrapper">
    <div class="div_center">
      <h2 class="h2_font">Add Category</h2>
      <form>
        <input class="input_color" type="text" name="name" placeholder="Write Category name">
        <input type="submit" class="btn btn-primary" name="submit" value="Add Category">
      </form>
    </div>
  </div>
</div>
```

- By doing this, it means that I can add a category to my database in phpMyAdmin. However, I first needed to create a category table. This was done using the CMD. (.png). A new file was then created in Models/database/migrations. This is where I was able to create rows and columns in the table in the phpMyAdmin database.

Table	Action	Rows		
		Count	Type	Collation
<input type="checkbox"/> categories	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode
<input type="checkbox"/> failed_jobs	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode
<input type="checkbox"/> migrations	★ Browse Structure Search Insert Empty Drop	7	InnoDB	utf8mb4_unicode
<input type="checkbox"/> password_resets	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode
<input type="checkbox"/> personal_access_tokens	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode
<input type="checkbox"/> sessions	★ Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_unicode
<input type="checkbox"/> users	★ Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_unicode
<b>7 tables</b>	<b>Sum</b>	<b>10</b>	<b>InnoDB</b>	<b>utf8mb4_general</b>

- I created an 'add category' link in the form, and then I created the route for the url in the 'web.php' file. The same steps were followed as when creating the 'view category' field.
  - It is important to note that because I was posting data, that my function included '(Request \$request)'.
  - Finally I created a message that was sent from the Admin Controller to tell the user that they have successfully added a new category ((success\_message.png). To view the code that did this, see this image (add\_message1.png) and this one (add\_message2.png).

#### Showing category data in Admin Database:

- Initially I created a table using HTML and styled it with CSS to make it look like a table, this is where the 'category' data will be displayed
- The code below shows how this was done.

```

use Illuminate\Http\Request;

//Because the command below had already been used, it mean that all that was needed was the
// variable below to pull across data from the database on to the table frontend.
use App\Models\Category;

class AdminController extends Controller
{
    public function view_category ()
    {
        // this variable
        $data=category::all();

        // this gets
        return view('admin.category'compact('data'));
    }

    public function add_category(Request $request)
    {
        $data=new category;

        $data-> category_name=$request->category;

        $data->save();

        return redirect()->back()->with('message','Category Added Successfully');
    }
}

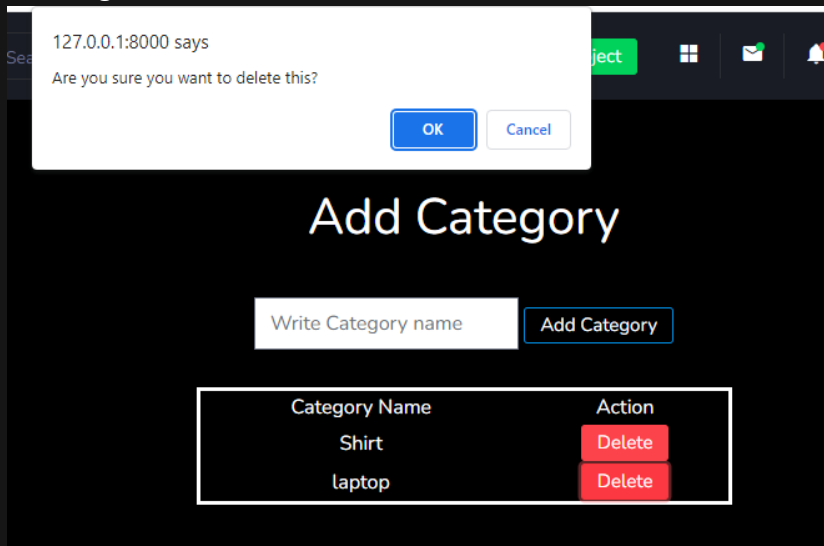
```



- I am pulling the data from the category table using 'use App\Models\Category;', then the data is being stored in the variable '\$data' and sending it to the 'admin. Category' view. To catch the data a foreach loop was created as seen here.

```
@foreach($data as $data)
<tr>
<td>{{ $data->category_name }}</td>
<td>
<a class="btn btn-danger" href="">Delete</a>
</td>
</tr>
@endforeach
```

- I created an 'Are you sure?' question incase user accidentally deletes a category by creating a browser button with PHP's 'onclick' function.



#### Creating 'Product' section to the side bar and linking to database:

- I did this by editing an existing Bootstrap elements in the HTML, in the 'sidebar.blade.php'. It was structured as a form.
- Repeating what I did earlier, I then created a table for the products in the database, using phpMyAdmin and the CMD.

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.

B:\websites\sarahmurphy\EcommercePro>php artisan make:model Product -m
```

- I then created the table using php.

```
public function up()
{
    Schema::create('products', function (Blueprint $table) {
        $table->id();

        $table->string('title')->nullable();
        $table->string('description')->nullable();
        $table->string('image')->nullable();
        $table->string('category')->nullable();
        $table->string('quantity')->nullable();
        $table->string('price')->nullable();
        $table->string('discount_price')->nullable();

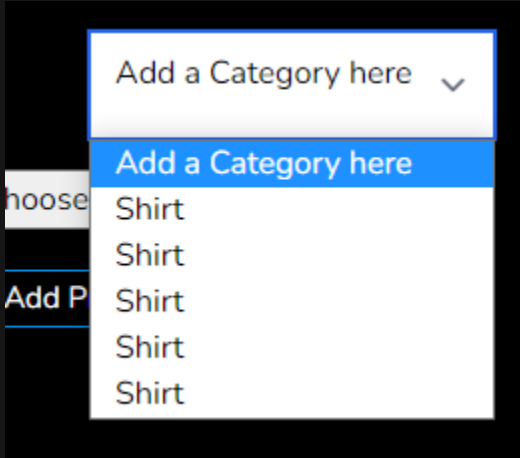
        $table->timestamps();
    });
}
```

Product table functionality:

- Next, I styled the form and added a required tag to ensure that the user must complete all fields to move forward. In addition, I added the tag 'enctype/form-data' which allows the user to upload files and the method used was 'POST' because I am uploading data to the database.
- It was important that when creating this route, I changed the method to 'POST' too. By using the 'Request \$request' when writing the function, it pulls the data from the form.
- Then I wanted to make sure that the data that had been added as a product category is accessible. To do this, I used the code seen in this screenshot.

```
public function view_product()
{
    $category=new category;
    return view('admin.product',compact('category'));
}
```

- This meant that by using the category model and the function written earlier, I can send the category data in the product.blade.php file. Using the 'option' tag, I created a foreach loop. This did pull across the category data, however it was just pulling across one.



- To solve this I added the 'category\_name' field from the database inside of the loop.

```
<select class="text_color" name="category" required="">
  <option value="" selected="">Add a Category here</option>

  @foreach($category as $category)

  <option>{{$category->category_name}}</option>

  @endforeach
</select>
```

- In order to receive all the data from the database then I pull across the database names and then requesting the loop.




```
public function add_product(Request $request)
{
    $product=new product;
    $product->title=$request->title;
    $product->description=$request->description;
    $product->price=$request->description;
    $product->quantity=$request->quantity;
    $product->discount_price=$request->discount_price;
    $product->category=$request->category;
```

- The following code explains how the image is stored in the image variable, then the image was given a unique name using the 'time' function. This means that every image will have a different name.

```
$image=$request->image;
$imagename=time().'.'.$image->getClientOriginalExtension();
$request->image->move('product',$imagename);
$product->image=$imagename;
```

- Next, I created a public folder called 'product' where the images are stored. I did encounter an error whilst trying to submit the form but I realised that it was because I had said 'post' rather than 'POST'.
- To create the link for the 'Show Product' tab in the sidebar I created a route in the 'web.php' file, I then created a 'Show Product' function in the 'Admin Controller' and then created a page to store the code. After creating, styling a table, I linked to the titles in the database.

## All Products

Product Title	Description	Quantity	Category	Price	Discount Price	Product Image
Shirt	Nice shirt	10	Shirt	2000	180	
Shirt	Nice shirt	10	Shirt	2000	180	
test	test	1	Shirt	10	1	

- In order to make the product table even more user friendly, I added an 'edit' and 'delete' column with a button using an 'anchor tag' and Bootstrap button classes.
- The code (delete\_table.png) shows how I linked the product id's in the table, meaning that the 'delete button' knows which product data to delete. Just like before, I then

created a route in the 'web.php' folder and a 'delete\_product' function in the 'admin.controller'.

```
public function delete_product($id)
{
    $product=product::find($id);
    $product->delete();
    return redirect()->back();
}
```

- This was repeated to create the 'update product' function, however I needed to create the page where to view was being returned to, so I created a 'update\_product.blade.php' file. The id is then caught using this function, the product is then stored in the '\$product' variable and then that specific data is being sent to the 'update\_product' page.

```
public function update_product($id)
{
    $product=product::find($id);
    return view('admin.update_product',compact(['product']));
}
```

- I used the code from the 'product.blade.php' page for the body and styling for the 'Update Product' page, however the CSS wasn't being found from the public folder so mentioned it using '<base href="/public">'.

#### Showing the product data frontend:

- Within a function in the 'home controller' I wrote '\$product=Product::all();' which means that all the product data will be shown on that home page as this is where the function is initially returning the view to.
- At this point, the data is being pulled but there is only fake content creating the products.
- I also create pagination to ensure that if the client wants to add lots of products, this will not over crowd the home page and the user will be able to scroll. I did have an issue viewing the pagination, but I realised that it was because I had only added three product and set it to paginate when there is more than three.

```
</div>
</div>

@endforeach

{!!$product->appends(Request::all()->links())!!}

</div>
```

- This was the final error I met before submission, despite using Bootstrap classes throughout this project I saw the below error message.

InvalidArgumentException  
**View [bootstrap-5] not found. (View: B:\websites\sarahmurphy\EcommercePro\resources\views\home\product.blade.php)**  
<http://127.0.0.1:8000/>

[Hide solutions](#)

**bootstrap-5 was not found.**  
 Are you sure the view exists and is a `.blade.php` file?

Stack trace	Request	App	User	Context	Debug	Share
Illuminate\View\FileViewFinder::findInPaths B:\websites\sarahmurphy\EcommercePro\vendor\laravel\framework\src\Illuminate\View\FileViewFinder.php:137						
75	B:\websites\sarahmurphy\EcommercePro\vendor\laravel\framework\src\Illuminate\View\FileViewFinder.php:137					
74	Illuminate\View\FileViewFinder.php:92					
73	Illuminate\View\FileViewFinder.php:76					
B:\websites\sarahmurphy\EcommercePro\vendor\laravel\framework\src\Illuminate\View\Factory.php:137						
72	Illuminate\View\Factory.php:137					
71	Illuminate\Pagination\LengthAwarePaginator.php:93					
70	Illuminate\Pagination\LengthAwarePaginator.php:79					
69	Illuminate\Pagination\LengthAwarePaginator.php:63					
68	Illuminate\Filesystem\Filesystem.php:107					
67	Illuminate\Filesystem\Filesystem.php:108					

```

122     * @param array $paths
123     * @return string
124     *
125     * @throws \InvalidArgumentException
126     */
127     protected function findInPaths($name, $paths)
128     {
129         foreach ((array) $paths as $path) {
130             foreach ($this->getPossibleViewFiles($name) as $file) {
131                 if ($this->files->exists($viewPath = $path.'/'.$file)) {
132                     return $viewPath;
133                 }
134             }
135         }
136     }
137     throw new InvalidArgumentException("View [{$name}] not found.");
  
```

Overview:

- In terms of my development with programming languages, I feel as though I have really broadened my understanding with this project. My knowledge of the functions and way in which PHP is used to create a website/application is much clearer.

- Despite having solved the above error and using Bootstrap classes throughout this build, the last task I got to before submission was an error saying that Bootstrap could not be found.
- 

## **Bibliography:**

### **Dev Bibliography:**

*Browse thousands of Holistic images for design inspiration* (no date) *Dribbble.com*. Available at: <https://dribbble.com/search/holistic> (Accessed: December 12, 2022).

*Carbon footprint calculator* (no date) *Carbonindependent.org*. Available at: <https://www.carbonindependent.org/> (Accessed: December 15, 2022).

*Carbon footprint calculator* (no date) *Carbonfootprint.com*. Available at: <https://calculator.carbonfootprint.com/calculator.aspx?tab=6> (Accessed: December 15, 2022).

*Corona admin* (no date) *Github.io*. Available at: <https://technext.github.io/corona-free-dark-bootstrap-admin-template/index.html> (Accessed: January 5, 2023).

*ErrorException Trying to get property "title" of non-object* (View: *C:\DK\Practice\Laravel\example-app\resources\views\product.blade.php*) (no date) *Stack Overflow*. Available at: <https://stackoverflow.com/questions/72674975/errorexception-trying-to-get-property-title-of-non-object-view-c-dk-practic> (Accessed: December 13, 2022).

*How much office space do I need?* (2018) *Workspace.co.uk*. Workspace. Available at: <https://www.workspace.co.uk/content-hub/business-insight/how-much-office-space-do-i-need> (Accessed: December 15, 2022).

*"How to calculate your carbon footprint"* (2022) *Sustain.life*. Sustain.life, 18 November. Available at: [https://www.sustain.life/blog/calculate-carbon-footprint?utm\\_source=google&utm\\_medium=cpc&campaignid=17817971457&adgroupid=142233327027&utm\\_campaign=International+Blog+Dynamic&utm\\_term=&network=g&gclid=Cj0KCQjwqoibBhDUARIsAH2OpWhJo3V6UbuHhVXG7znWZXXKdrrx2KyY2\\_pi24r5HYIShHSQ93gZioYcaAi12EALw\\_wcB](https://www.sustain.life/blog/calculate-carbon-footprint?utm_source=google&utm_medium=cpc&campaignid=17817971457&adgroupid=142233327027&utm_campaign=International+Blog+Dynamic&utm_term=&network=g&gclid=Cj0KCQjwqoibBhDUARIsAH2OpWhJo3V6UbuHhVXG7znWZXXKdrrx2KyY2_pi24r5HYIShHSQ93gZioYcaAi12EALw_wcB) (Accessed: December 15, 2022).

*How to call PHP function on the click of a Button ?* (2019) *GeeksforGeeks*. Available at: <https://www.geeksforgeeks.org/how-to-call-php-function-on-the-click-of-a-button/> (Accessed: December 12, 2022).

*JetStream CSS and JS not working and showing @vite(['resources/css/app.css', 'resources/js/app.js'])* (no date) *Stack Overflow*. Available at: <https://stackoverflow.com/questions/73180945/jetstream-css-and-js-not-working-and-showing-viteresources-css-app-css-re> (Accessed: December 6, 2022).

*Laravel Call to a member function getClientOriginalExtension() on null* (no date) *Stack Overflow*. Available at: <https://stackoverflow.com/questions/37124454/laravel-call-to-a-member-function-getclientoriginalextension-on-null> (Accessed: December 13, 2022).

Mark, B. (2021) *Business energy costs: How much does the average office cost to run?*, *Thermatic Energy Services*. Available at: <https://thermaticenergy.co.uk/business-energy-costs/> (Accessed: December 15, 2022).

*PHP include and require* (2022) *W3schools.com*. Available at: [https://www.w3schools.com/php/php\\_includes.asp](https://www.w3schools.com/php/php_includes.asp) (Accessed: December 12, 2022).

Renouard, J. (2021) *How to host a website in 2022*, *Website Builder Expert*. Available at: <https://www.websitebuilderexpert.com/web-hosting/how-to-host-a-website/> (Accessed: December 12, 2022).

Tahir, O. (2022) *How to create a custom WordPress theme from scratch (2022)*, *WPForms*. Available at: <https://wpforms.com/how-to-create-a-custom-wordpress-theme/> (Accessed: December 12, 2022).

*Udemy.com*. Available at: <https://www.udemy.com/course/create-e-commerce-website-using-laravel-beginner-to-advance/learn/lecture/33055340?start=195#overview> (Accessed: December 6, 2022).

*UK: travel carbon footprint by transport mode 2022* (no date) *Statista*. Available at: <https://www.statista.com/statistics/1233337/carbon-footprint-of-travel-per-kilometer-by-mode-of-transport-uk/> (Accessed: December 15, 2022).

Way, J. (2019) *What is Laravel? Explain it like I'm five*, *DEV Community* 🇺🇸 🇬🇧. Available at: [https://dev.to/creativetim\\_official/what-is-laravel-explain-it-like-i-m-five-19eb](https://dev.to/creativetim_official/what-is-laravel-explain-it-like-i-m-five-19eb) (Accessed: December 15, 2022).

*What is the difference between a library and a dependency?* (no date) *Software Engineering Stack Exchange*. Available at: <https://softwareengineering.stackexchange.com/questions/408739/what-is-the-difference-between-a-library-and-a-dependency> (Accessed: December 15, 2022).



