

COMP710/2001 – Laravel Assignment Food Ordering

ASSIGNMENT COVER SHEET

Submission Date	7 May 2020	
Due Date	7 May 2020	
Student Name & ID	<u>Name</u> Jithu Saji Kanichattu	<u>ID</u> 18477280
Case Study	Online Food Takeaway	

This assignment is my own work:

Your name: Jithu Saji Kanichattu

Case Study: Online Food Takeaway(Coffee Agora)

Signature: NR

COMP710/2001 – Laravel Assignment Food Ordering

Table of Contents

Introduction.....	3
PHP Artisan Commands.....	3
Design Your Storyboard for your Web Applications.....	3
ERD Diagram.....	4
Screenshots.....	5
Screenshot of Home Page.....	5
Registration Form.....	5
Login Page.....	6
Screenshot of Single Menu Page	6
Screenshot of Adding to Cart.....	7
Screenshot of Cart Page.....	7
Screenshots of create categories using php artisan.....	8
Screenshots of migration of products & carts table.....	8
Code.....	9
model/controller/route.....	9
views.....	11
routes.....	12
MVC Model vs COMP606.....	13

COMP710/2001 – Laravel Assignment Food Ordering

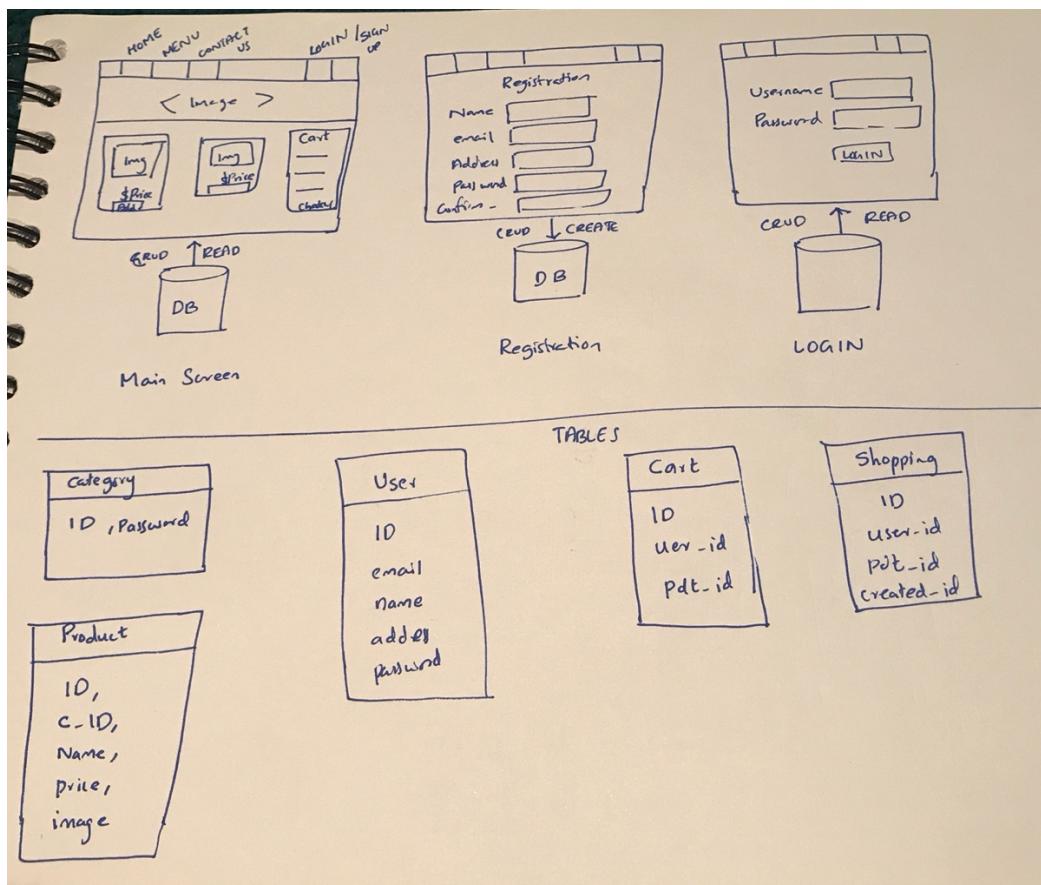
INTRODUCTION

Coffee Agora is a coffee shop with variety of special cuisines of their own and shopping is much easier for the customer via online.

PHP ARTISAN COMMANDS

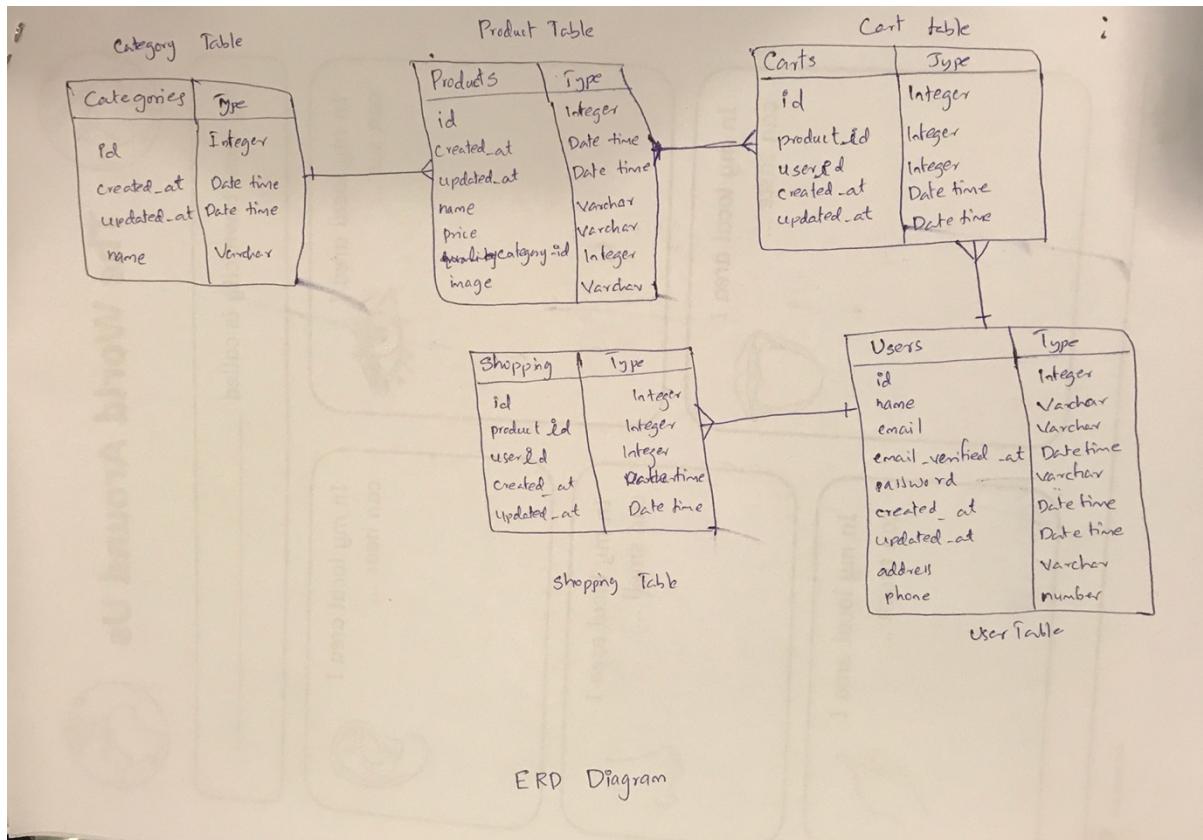
1. php artisan make: model Category -mc (Model creation)
2. php artisan make: model Product -mc
3. php artisan make: model Cart -mc

Design your Storyboard for your Web Application



COMP710/2001 – Laravel Assignment Food Ordering

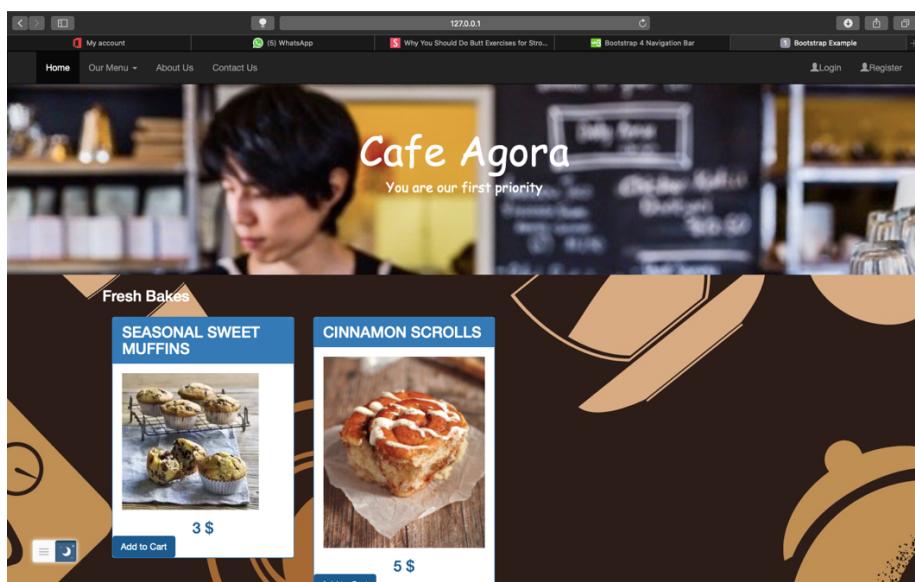
ERD Diagram



COMP710/2001 – Laravel Assignment Food Ordering

Screenshots

Screenshot of Home Page

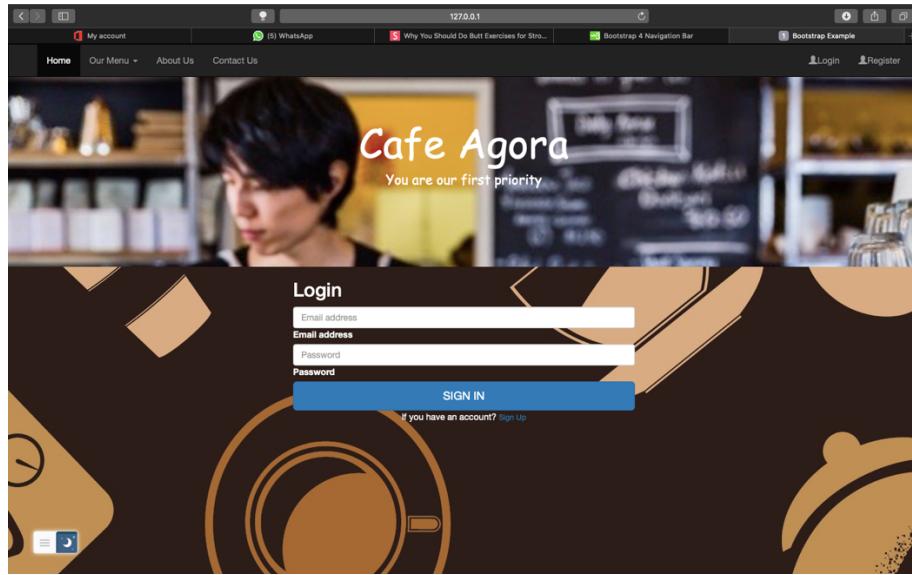


Registration Form

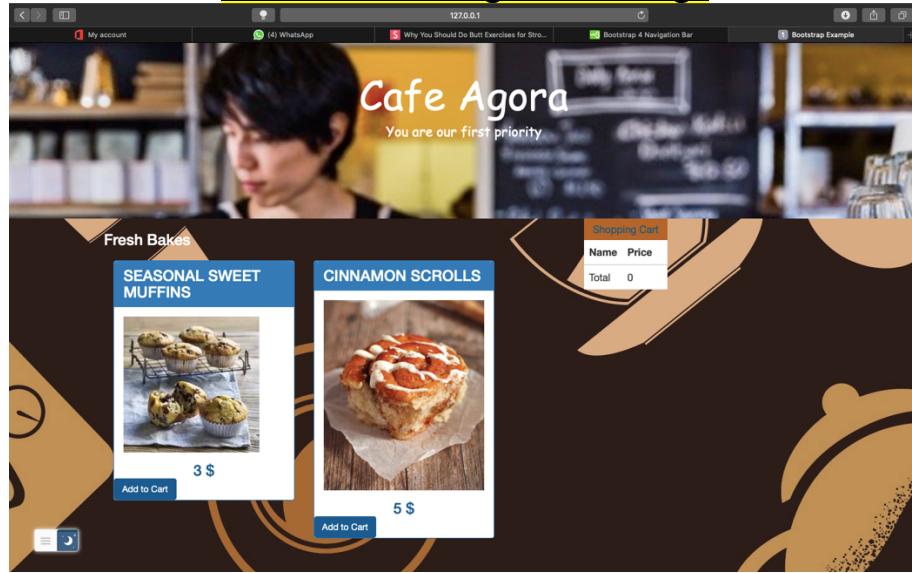
A screenshot of a web browser showing the registration form for the cafe. The form is titled "Register" and contains fields for "Name", "Email address", "Address", "Phone No.", and "Password". Each field has a corresponding input box. Below the form is a blue "SIGN UP" button. At the bottom of the page, there's a link "If you have an account? [Sign in](#)". The background of the page shows a blurred image of a person in a cafe setting.

COMP710/2001 – Laravel Assignment Food Ordering

Login Page



Screenshot of Single Menu Page



COMP710/2001 – Laravel Assignment Food Ordering

Screenshot of Adding to Cart

A screenshot of a food ordering website. On the left, there's a menu section titled "Non-Vegetarian" with four items: "VIETNAMESE PORK BELLY SALAD" (20 \$), "SPANISH THREE EGG OMELETTE" (17 \$), "HUEVOS RANCHEROS (COWBOY EGGS)" (20 \$), and "AGORA EGGS BENNY(FREE RANGE BACON)" (22 \$). Each item has an "Add to Cart" button. On the right, a "Shopping Cart" sidebar shows a list of items with quantities and "Remove" buttons:

Name	Price
CINNAMON SCROLLS	5
SEASONAL SWEET MUFFINS	3
SEASONAL SWEET MUFFINS	3
SEASONAL SWEET MUFFINS	3
CINNAMON SCROLLS	5
SEASONAL SWEET MUFFINS	3
CINNAMON SCROLLS	5

Screenshot of Cart Page

A screenshot of a food ordering website. At the top, it says "cate Agora" and "You are our first priority". Below, there's a "Fresh Bakes" section with "SEASONAL SWEET MUFFINS" (3 \$) and "CINNAMON SCROLLS" (5 \$), each with an "Add to Cart" button. On the right, a "Shopping Cart" sidebar shows a list of items with quantities and "Remove" buttons, plus a "Checkout" button at the bottom:

Name	Price
CINNAMON SCROLLS	5
HUEVOS RANCHEROS (COWBOY TOFU)	20
AGORA EGGS BENNY(ROASTED PORTABELLO)	22
AGORA EGGS BENNY(ROASTED PORTABELLO)	22
BABY BENNY(MUSHROOM)	13
Total	82

COMP710/2001 – Laravel Assignment Food Ordering

Screenshots of create categories using php artisan

```
jithusaji@Jithus-MacBook-Air cafe % php artisan make:migration create_categories_table --create categories
Created Migration: 2020_05_10_183223_create_categories_table
jithusaji@Jithus-MacBook-Air cafe %
```

Screenshots of migration of products, carts and shopping table

```
[jithusaji@Jithus-MacBook-Air cafe % php artisan make:migration create_categories_table --create categories
Created Migration: 2020_05_10_183223_create_categories_table
[jithusaji@Jithus-MacBook-Air cafe % php artisan make:migration create_products_table --create products
Created Migration: 2020_05_10_183359_create_products_table
[jithusaji@Jithus-MacBook-Air cafe % php artisan make:migration create_carts_table --create carts
Created Migration: 2020_05_10_183434_create_carts_table
[[[[A
[jithusaji@Jithus-MacBook-Air cafe % php artisan make:migration create_shopping_table --create shopping
Created Migration: 2020_05_10_183507_create_shopping_table
```

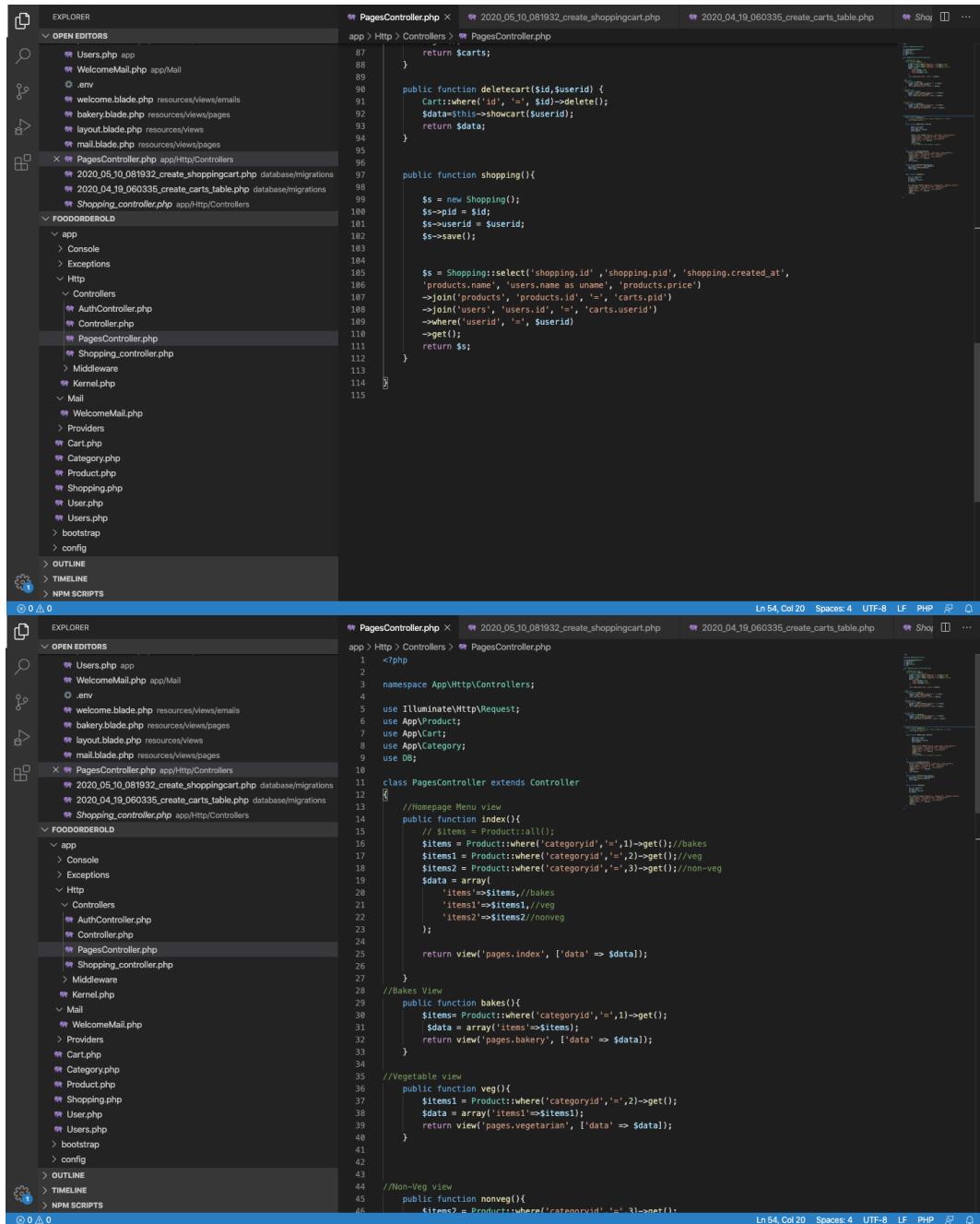
```
[jithusaji@Jithus-MacBook-Air cafe % php artisan migrate
Migration table created successfully.
Migrating: 2014_10_12_000000_create_users_table
Migrated: 2014_10_12_000000_create_users_table (0.05 seconds)
Migrating: 2019_08_19_000000_create_failed_jobs_table
Migrated: 2019_08_19_000000_create_failed_jobs_table (0.02 seconds)
Migrating: 2020_05_10_183223_create_categories_table
Migrated: 2020_05_10_183223_create_categories_table (0.02 seconds)
Migrating: 2020_05_10_183359_create_products_table
Migrated: 2020_05_10_183359_create_products_table (0.02 seconds)
Migrating: 2020_05_10_183434_create_carts_table
Migrated: 2020_05_10_183434_create_carts_table (0.01 seconds)
Migrating: 2020_05_10_183507_create_shopping_table
Migrated: 2020_05_10_183507_create_shopping_table (0.02 seconds)
jithusaji@Jithus-MacBook-Air cafe %
```

COMP710/2001 – Laravel Assignment Food Ordering

Code

model/controller/route

Shopping Controller



The image shows two side-by-side code editors, both displaying the same file: `PagesController.php`. The file is located at `app/Http/Controllers`.

Top Editor Content:

```
    return $carts;
}

public function deletecart($id,$userid) {
    Cart::where('id', '=', $id)->delete();
    $data=$this->showcart($userid);
    return $data;
}

public function shopping(){
    $s = new Shopping();
    $s->id = $id;
    $s->userid = $userid;
    $s->save();

    $s = Shopping::select('shopping.id','shopping.pid','shopping.created_at',
    'products.name','users.name as uname','products.price')
    ->join('products','products.id','=','carts.pid')
    ->join('users','users.id','=',$carts.userid')
    ->where('userid','=',$userid)
    ->get();
    return $s;
}
```

Bottom Editor Content:

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Product;
use App\Cart;
use App\Category;
use DB;

class PagesController extends Controller
{
    //Homepage Menu view
    public function index(){
        // $items = Product::all();
        $items = Product::where('categoryid','=',1)->get(); //bakes
        $item1 = Product::where('categoryid','=',2)->get(); //veg
        $item2 = Product::where('categoryid','=',3)->get(); //non-veg
        $data = array(
            'items'=>$items,
            'item1'=>$item1,
            'item2'=>$item2
        );
        return view('pages.index', ['data' => $data]);
    }

    //Bakes View
    public function bakes(){
        $items = Product::where('categoryid','=',1)->get();
        $data = array('items'=>$items);
        return view('pages.bakery', ['data' => $data]);
    }

    //Vegetable view
    public function veg(){
        $item1 = Product::where('categoryid','=',2)->get();
        $data = array('item1'=>$item1);
        return view('pages.vegetarian', ['data' => $data]);
    }

    //Non-Veg view
    public function nonveg(){
        $item2 = Product::where('categoryid','=',3)->get();
    }
}
```

COMP710/2001 – Laravel Assignment Food Ordering

The screenshot shows a code editor with the file `PagesController.php` open. The code is a PHP script with several functions:

```
43 //Non-Veg view
44 public function nonveg(){
45     $item2 = Product::where('categoryid','>','3')->get();
46     $data = array($item2 =>$item2);
47     return view('pages.nonvegetarian', ['data' => $data]);
48 }
49
50 //all menu products
51 public function products(){
52     // $items = DB::table('items')->where('categoryid','1')->get();
53     return view('products');
54 }
55
56 public function addCart($id, $userid){
57
58     $cart = new Cart();
59     $cart->pid = $id;
60     $cart->userid = $userid;
61     $cart->save();
62
63     $scarts = Cart::select('carts.id', 'carts.pid', 'carts.created_at',
64     'products.name', 'users.name as uname', 'products.price')
65     ->join('products', 'products.id', '=', 'carts.pid')
66     ->join('users', 'users.id', '=', 'carts.userid')
67     ->where('userid', '=', $userid)
68     ->get();
69     return $scarts;
70
71     // return Redirect::to('showcart/'.$userid);
72 }
73
74
75 public function showCart($userid){
76     $scarts = Cart::select('carts.id', 'carts.pid', 'carts.created_at',
77     'products.name' , 'users.name as uname', 'products.price')
78     ->join('products', 'products.id', '=', 'carts.pid')
79     ->join('users', 'users.id', '=', 'carts.userid')
80     ->where('userid', '=', $userid)
81     ->get();
82     return $scarts;
83 }
84
85
86
87
88
89
90
91 }
```

Ln 54, Col 20 Spaces: 4 UTF-8 LF PHP

Model

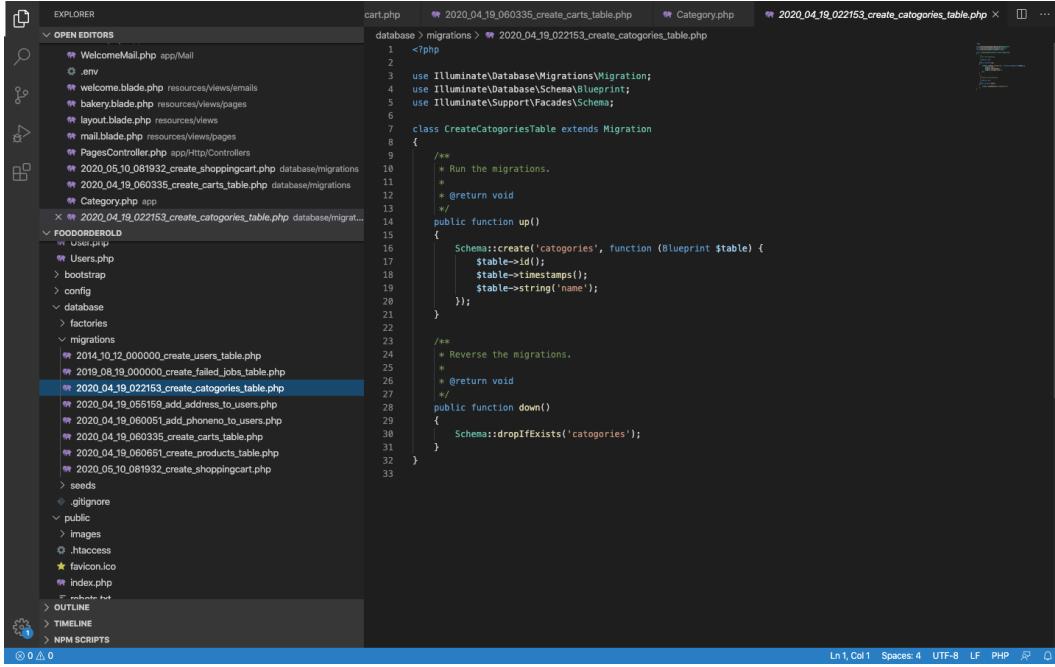
The screenshot shows a code editor with the file `Category.php` open. The code is a PHP script defining a model class:

```
1 <?php
2
3 namespace App;
4 use Illuminate\Database\Eloquent\Model;
5
6 class Category extends Model
7 {
8
9 }
```

Ln 11, Col 1 Spaces: 4 UTF-8 LF PHP

COMP710/2001 – Laravel Assignment Food Ordering

Category Migration



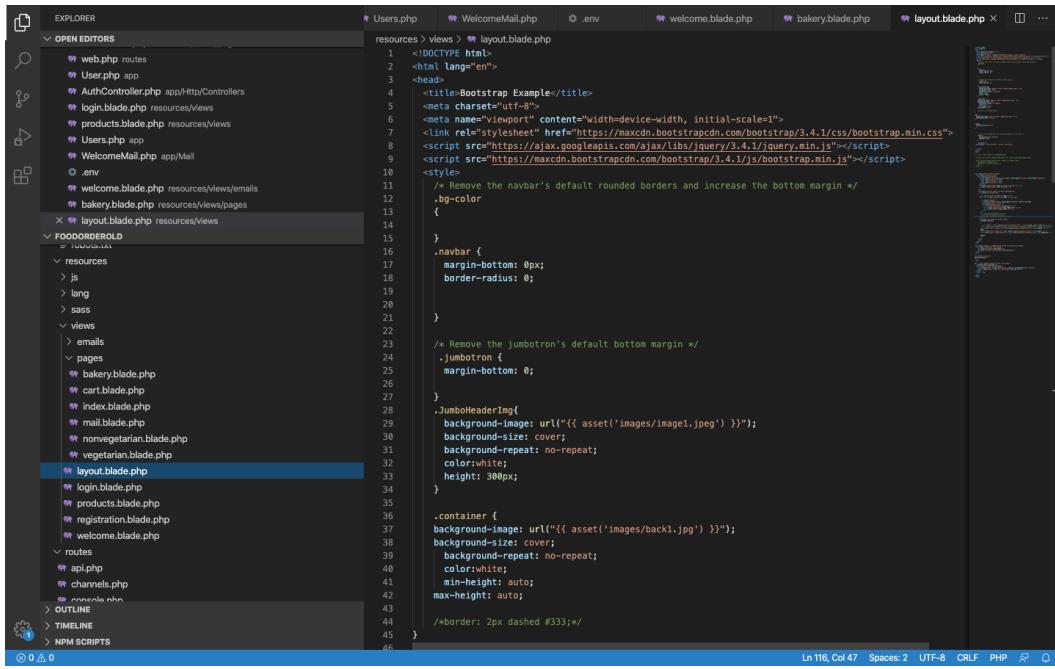
```
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateCategoriesTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('categories', function (Blueprint $table) {
            $table->id();
            $table->timestamps();
            $table->string('name');
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('categories');
    }
}
```

Views



```
<!DOCTYPE html
  <html lang="en">
  <head>
    <title>Bootstrap Example</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
  <style>
    /* Remove the navbar's default rounded borders and increase the bottom margin */
    .bg-color
    {
    }

    .navbar {
      margin-bottom: 0px;
      border-radius: 0;
    }

    /* Remove the jumbotron's default bottom margin */
    .jumbotron {
      margin-bottom: 0;
    }

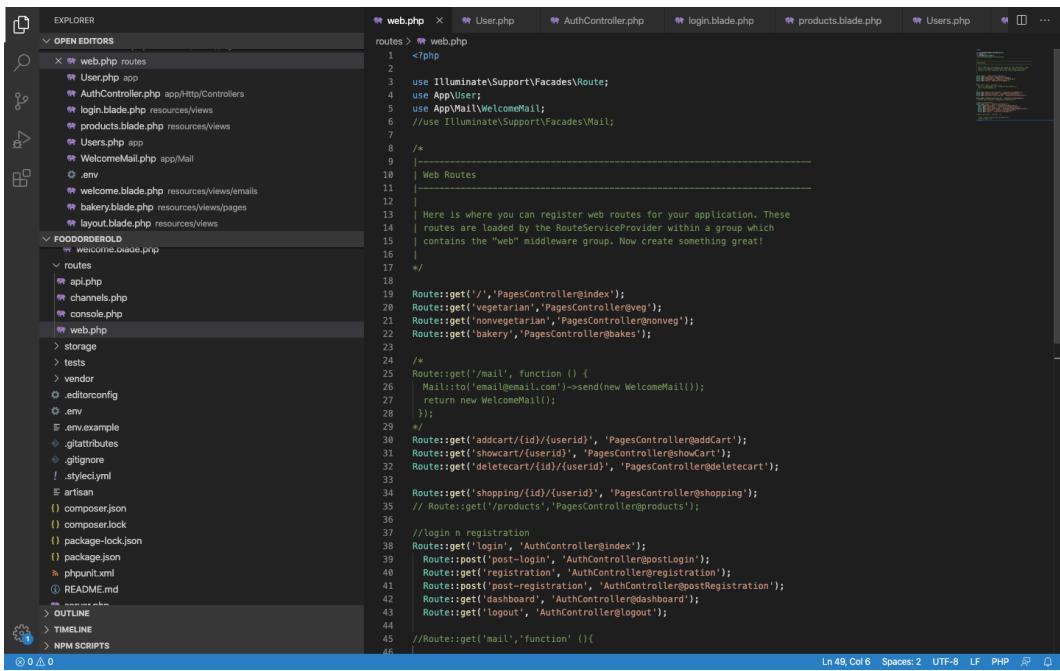
    .JumboHeaderImg{
      background-image: url('{{ asset('images/image1.jpeg') }}');
      background-size: cover;
      background-repeat: no-repeat;
      color:white;
      height: 300px;
    }

    .container {
      background-image: url('{{ asset('images/back1.jpg') }}');
      background-size: cover;
      background-repeat: no-repeat;
      color:white;
      min-height: auto;
      max-height: auto;
    }
  </style>

```

COMP710/2001 – Laravel Assignment Food Ordering

Routes



The screenshot shows a code editor interface with several tabs open. The active tab is 'routes' under 'FOODORDER01'. The code in the editor is the Laravel web routes configuration:

```
<?php  
use Illuminate\Support\Facades\Route;  
use App\User;  
use App\Mail\WelcomeMail;  
//use Illuminate\Support\Facades\Mail;  
  
Route::get('/', 'PagesController@index');  
Route::get('vegetarian', 'PagesController@veg');  
Route::get('nonvegetarian', 'PagesController@nonveg');  
Route::get('bakery', 'PagesController@bakes');  
  
Route::get('/mail', function () {  
    Mail::to('email@email.com')->send(new WelcomeMail());  
});  
  
Route::get('addcart/{userid}', 'PagesController@addCart');  
Route::get('showcart/{userid}', 'PagesController@showCart');  
Route::get('deletecart/{id}/{userid}', 'PagesController@deleteCart');  
  
Route::get('shopping/{id}/{userid}', 'PagesController@shopping');  
// Route::get('/products', 'PagesController@products');  
  
// login n registration  
Route::get('login', 'AuthController@index');  
Route::post('post-login', 'AuthController@postLogin');  
Route::get('registration', 'AuthController@registration');  
Route::post('post-registration', 'AuthController@postRegistration');  
Route::get('dashboard', 'AuthController@dashboard');  
Route::get('logout', 'AuthController@logout');  
  
//Route::get('mail', 'function' ()) {  
};
```

At the bottom of the editor, status information includes 'Ln 49, Col 6' and 'Spaces: 2'.

MVC MODEL VS COMP606

Model - The model component stores data and its related logic.

View - A View is that part of the application that represents the presentation of data.

Views are created by the data collected from the model data. A view requests the model to give information so that it resents the output presentation to the user.

Controller - The Controller is that part of the application that handles the user interaction. Controller send's commands to the model to update its state. The controller also sends commands to its associated view to change the view's presentation.

Model view controller is a framework application that's simpler and easier to understand. When working with databases classes for each object makes code reusable, easier to maintain & readable. But I think making single pages without MVC in comp606 were more flexible.

In Laravel, we do not need to include the database file in each page like we see in comp606. Here it is done by config the database in env file and just connect to the database. MVC is well organised and has a good structure for separation. It is well documented and is understood from the Laravel documentation website. In comp606 we need to create all html pages, CSS files and then include in all the pages. View contains little logic hardly using looping arrays ever. When it came down to implementing forms, the code was managed in the controller for database queries and validation with the views calling the functions & which all connects within the route. MVC also provides authentication and authorisation for the system.

The view shows how the data is displayed. Model is going to be handling data within the database migration & controller it change information in the model which works within the views telling it what to be displayed.