

INFS3202/7202 – Web Information Systems

Lecture Week 3: MVC – Controllers and Views

Dr Aneesha Bakharia (Senior Lecturer, EECS)
Email: a.bakharia1@uq.edu.au

Contents

-
- 01 The Model View Controller (MVC) Architecture

 - 02 Convention vs Configuration

 - 03 Installing CodeIgniter 4

 - 04 Introduction to CodeIgniter 4 – Routes, Controllers and Views

 - 05 Base Templates for Layout

 - 06 UI Design with CSS Frameworks

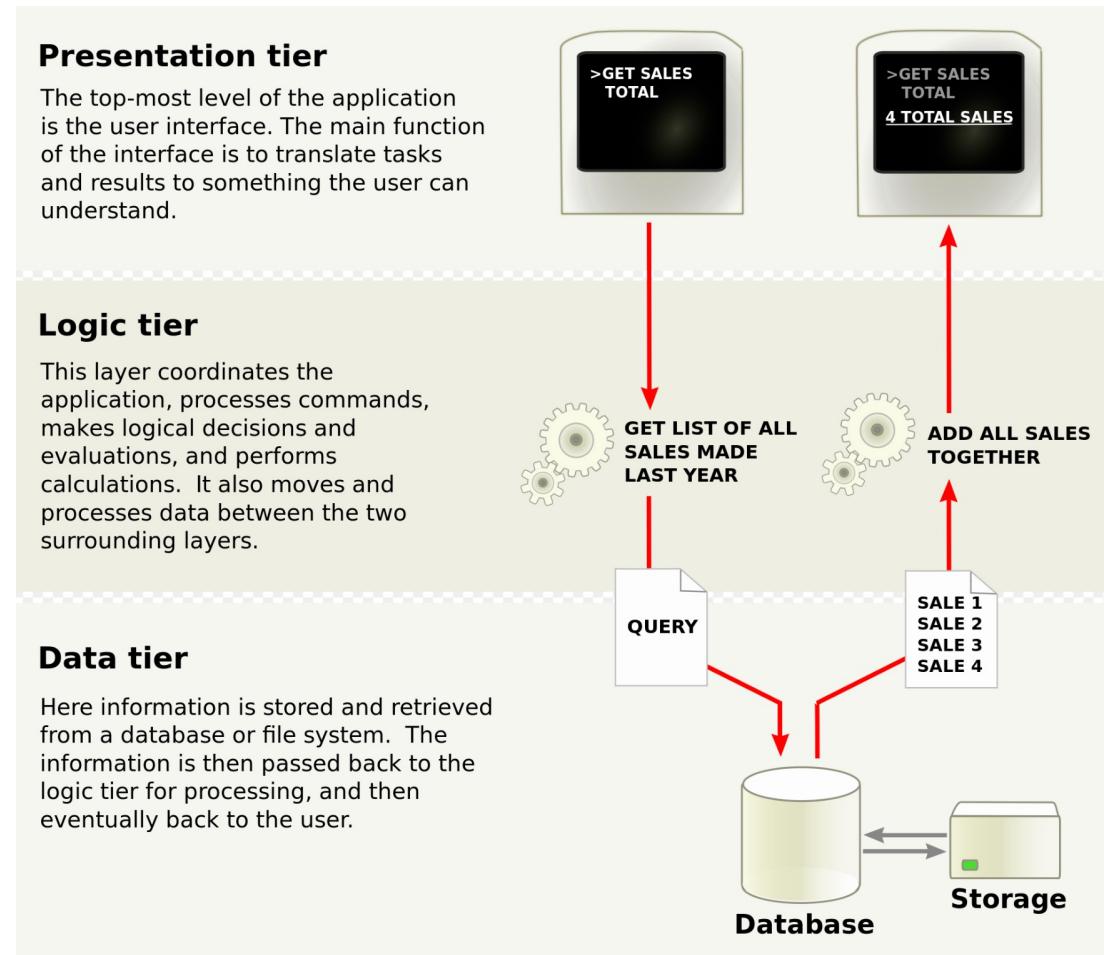
 - 07 Using GenAI (Part 1)

-

The MVC Architecture

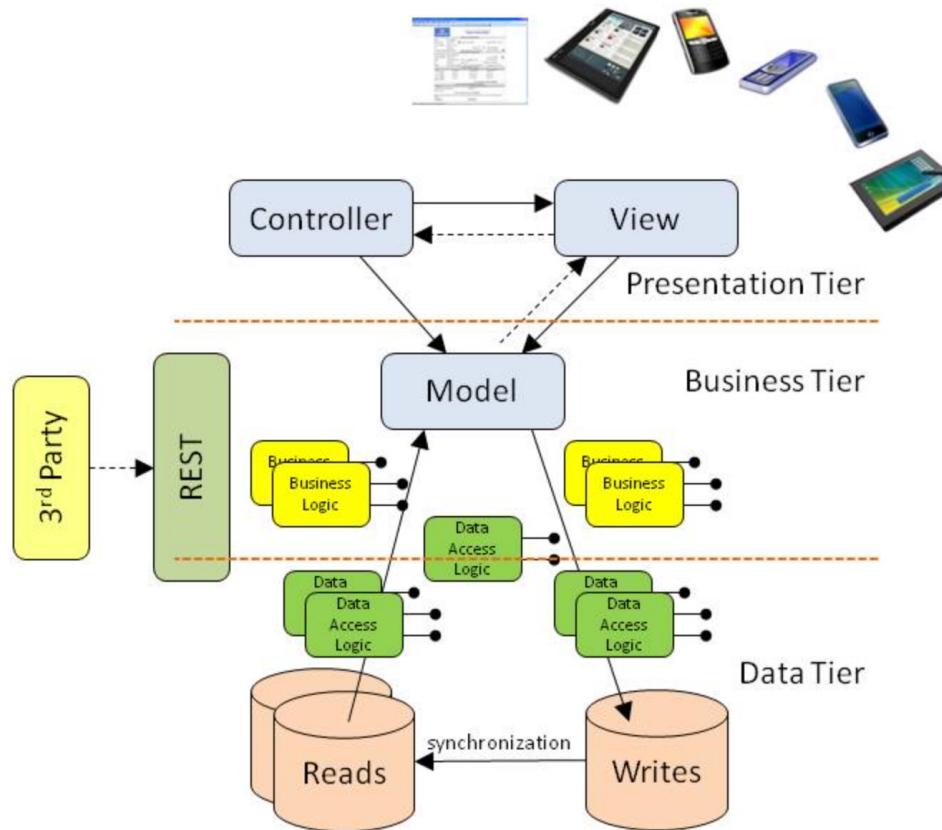
3-Layer Architecture

- The 3-Layer Architecture is a software design pattern that divides applications into three main categories or layers:
 - Presentation
 - Business Logic (or Application Layer)
 - Data Layer
- This architecture aims to separate the user interface, the business logic, and the data storage, which promotes a clean separation of concerns.



https://en.wikipedia.org/wiki/Multitier_architecture

Model View Controller (MVC)



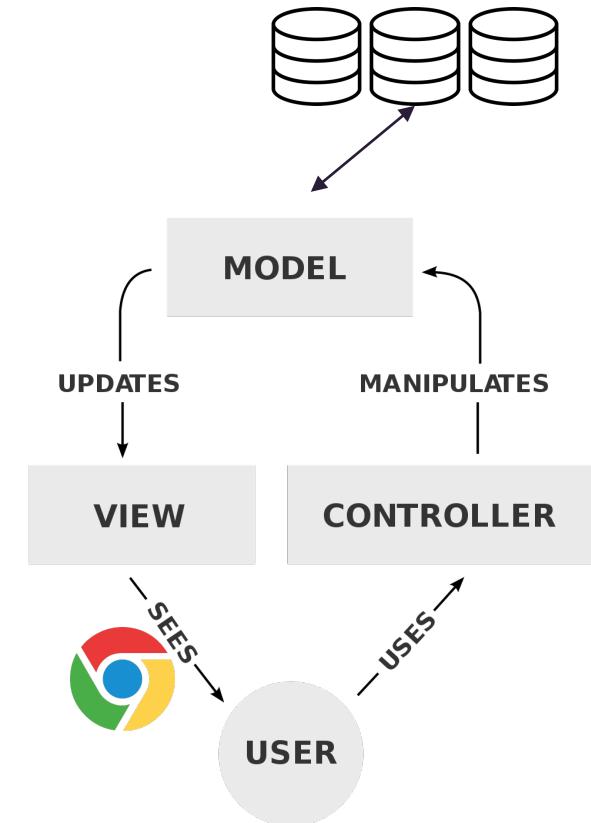
The MVC pattern is a specialized form of 3-layer architecture, making it highly relevant to modern web development practices

Image from https://en.wikiversity.org/wiki/Three-Tier_Architecture#/media/File:MVC3tierArchitecture.jpg

Components of MVC

Model–view–controller (MVC) - a software architectural pattern, that separates an application's data, user interface, and control flow into three components:

- **Model:**
 - Represents the **data**.
 - The model is often connected to a **database** and is responsible for **retrieving** and **storing** data.
- **View:**
 - Represents the **user interface** of an application,
 - is responsible for **displaying data and getting user input**.
 - HTML, CSS, and JavaScript.
- **Controller:**
 - Manages the **flow of data** between the **model** and the **view**,
 - is responsible for **processing user input** and **directing the application's behaviour**.
 - includes code that implements **business rules** and other functionality.



Benefits of MVC

- **Simplifies Complex Applications:**

By separating the application into three components, developers can focus on one aspect without affecting others.

- **Facilitates Parallel Development:**

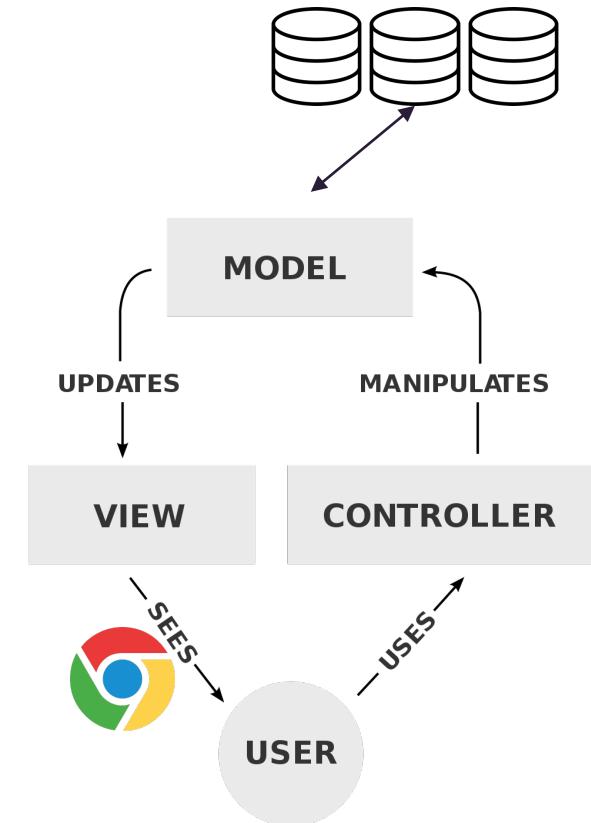
Different developers can work on the Model, View, and Controller simultaneously.

- **Easy to Modify or Scale:**

Changes to the business logic or user interface can be made with minimal impact on the other components.

- **Supports Reusability:**

Components can be reused across different parts of the application or in different applications.



Convention vs Configuration

Convention vs Configuration

Convention vs. Configuration is a design philosophy that:

- impacts how software frameworks are structured (i.e. where files are located)
- how they expect developers to interact with them
- It's a trade-off between the ease of following predefined paths and files (convention) or the flexibility to define everything explicitly (configuration).
- Important advice:
 - Web frameworks (CodeIgniter, Django, Ruby on Rails, Flask, Next.js) come with a predefined folder structure
 - Learn what the structure is and where files go
 - This will improve your productivity!

Introduction to CodeIgniter 4

[Discuss](#) [Contribute](#) [Learn](#)[Download](#)

CodeIgniter⁴

The small framework with powerful features

CodeIgniter is a powerful PHP framework with a very small footprint, built for developers who need a simple and elegant toolkit to create full-featured web applications.

[Learn more](#)

 Star 5,055  Fork 1,838

<https://www.codeigniter.com/>

Why CodeIgniter?



Framework with a small footprint

CodeIgniter 4 is a 1.1MB download, plus 1.6MB for the user guide (epub).



Exceptional performance

CodeIgniter consistently outperforms most of its competitors.



Simple solutions over complexity

CodeIgniter encourages MVC, but does not force it on you.



Strong Security

We take security seriously, with built-in protection against CSRF and XSS attacks. Version 4 adds context-sensitive escaping and CSP



Clear documentation

The User Guide contains an introduction, tutorial, a number of "how to" guides, and then reference documentation for the components that make up the framework.



Nearly zero configuration

Almost everything is set in CodeIgniter. Just connect your database!

<https://www.codeigniter.com/>

Why Choose CodeIgniter

- **Simplicity and Efficiency:**

Ideal for developers who need a straightforward and rapid development process without the bulkiness of larger frameworks.

- **Flexibility:**

Unlike some frameworks that enforce strict project environments, CodeIgniter provides the flexibility to organize files and structure the application as desired.

- **Documentation & Support:**

Excellent documentation and community support make it accessible for beginners and efficient for experienced developers.

Great First Framework to Learn with many transferrable concepts for other MVC frameworks.

Installing CodeIgniter 4

Installing CodeIgniter – Linux Server

- Week 3 Lab 2 includes step-by-step instructions for installation on your UQCloud Zone (i.e., a Linux server)
- Main steps
 - Install composer (a PHP package manager)
 - Add the php-intl (internationalisation) extension
 - Update Nginx config to run your CodeIgniter project from /var/www/yourproject folder. Restart Nginx.
 - Run the following composer command to create a CodeIgniter project
 - `composer create-project codeigniter4/appstarter /var/www/yourproject`
 - This is a Composer command that sets up a new project from an existing package. This command downloads the package, installs it in a new directory, installs all its dependencies, and initializes the project's autoloader.

Installing CodeIgniter – Locally on a Mac

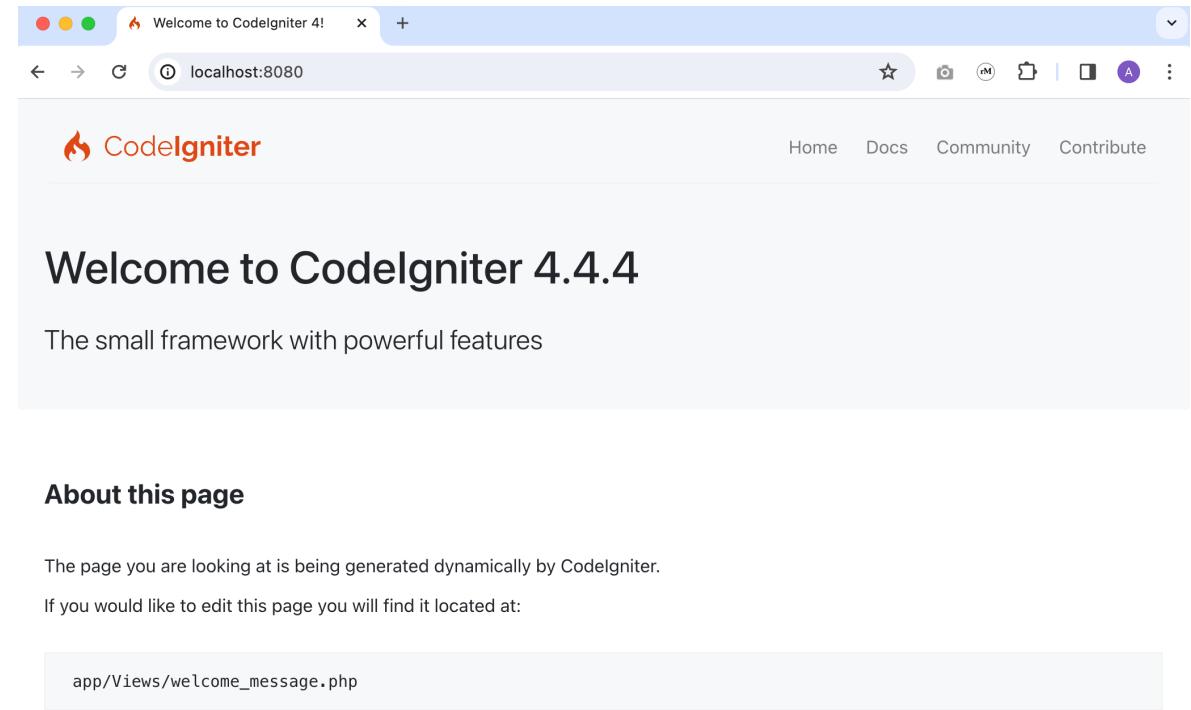
- Main steps
 - Update PHP using brew
 - `brew install php`
 - `brew services start php`
 - Download and install Composer
 - Download from <https://getcomposer.org/download/>
 - `sudo mv composer.phar /usr/local/bin/composer`
 - Run the following composer command to create a CodeIgniter project
 - `composer create-project codeigniter4/appstarter yourproject`
 - `php spark serve`
 - Go to `http://localhost:8080`

CodeIgniter - Sucessfull Installation

```
[uqabakh1@uqabakh1-9893 codeignitertest % php spark serve

CodeIgniter v4.4.4 Command Line Tool - Server Time: 2024-03-02 08:57:19 UTC+00:00

CodeIgniter development server started on http://localhost:8080
Press Control-C to stop.
[Sat Mar  2 18:57:19 2024] PHP 8.3.1 Development Server (http://localhost:8080) started
[Sat Mar  2 18:57:43 2024] 127.0.0.1:52950 Accepted
[Sat Mar  2 18:57:43 2024] 127.0.0.1:52950 Closing
[Sat Mar  2 18:57:43 2024] 127.0.0.1:52952 Accepted
[Sat Mar  2 18:57:44 2024] 127.0.0.1:52952 [200]: GET /favicon.ico
[Sat Mar  2 18:57:44 2024] 127.0.0.1:52952 Closing
```



<http://localhost:8080>

Installing CodeIgniter – Locally on a Windows

- Main steps
 - **Install XAMPP**
 - <https://www.apachefriends.org/>
 - This will give you a PHP and MySQL database
 - Open C:\xampp\php\php.ini and uncomment (remove ;):
extension=php_openssl.dll
extension=php_curl.dll
 - **Download and Install Composer**
 - <https://getcomposer.org/download/>
 - **Run the following composer command to create a CodeIgniter project**
 - composer create-project codeigniter4/appstarter yourproject
 - php spark serve
 - Go to <http://localhost:8080>

Note: You can also serve from htdocs in XAMPP using the Apache Web server or use spark serve.

CodeIgniter – What is spark

- In CodeIgniter 4, "spark" is a command-line tool that facilitates the development and management of CodeIgniter applications.
 - Running database migrations and seeders to manage database schema and initial data setup
 - Starting a development web server
 - Generating boilerplate code for new controllers, models, and other components.
 - Listing routes, displaying current environment settings, and more, providing insights into the application configuration

```
uqabakh1@uqabakh1-9893 codeigniter test % php spark
CodeIgniter v4.4.4 Command Line Tool - Server Time: 2024-03-02 09:04:45 UTC+00:00

Cache
  cache:clear      Clears the current system caches.
  cache:info       Shows file cache information in the current system.

CodeIgniter
  env             Retrieves the current environment, or set a new one.
  filter:check    Check filters for a route.
  help            Displays basic usage information.
  list             Lists the available commands.
  namespaces      Verifies your namespaces are setup correctly.
  publish          Discovers and executes all predefined Publisher classes.
  routes           Displays all routes.
  serve            Launches the CodeIgniter PHP-Development Server.

Database
  db:create        Create a new database schema.
  db:seed           Runs the specified seeder to populate known data into the database.
  db:table          Retrieves information on the selected table.
  migrate          Locates and runs all new migrations against the database.
  migrate:refresh  Does a rollback followed by a latest to refresh the current state of the database.
  migrate:rollback Runs the "down" method for all migrations in the last batch.
  migrate:status   Displays a list of all migrations and whether they've been run or not.

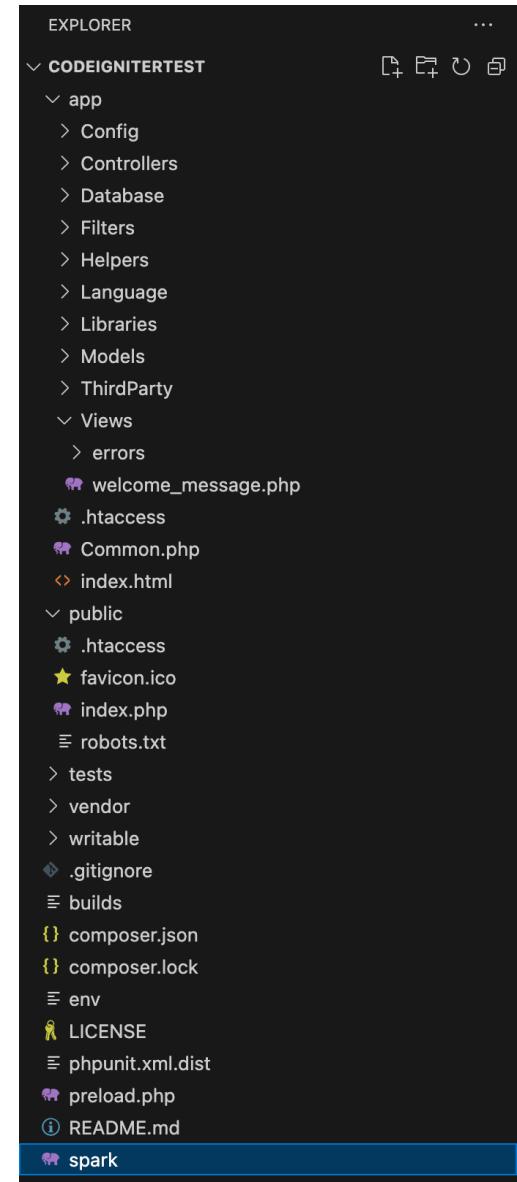
Encryption
  key:generate     Generates a new encryption key and writes it in an '.env' file.

Generators
  make:cell         Generates a new Controlled Cell file and its view.
  make:command     Generates a new spark command.
  make:config       Generates a new config file.
  make:controller  Generates a new controller file.
  make:entity      Generates a new entity file.
  make:filter       Generates a new filter file.
  make:migration   Generates a new migration file.
  make:model        Generates a new model file.
  make:scaffold    Generates a complete set of scaffold files.
  make:seeder       Generates a new seeder file.
  make:validation  Generates a new validation file.
  migrate:create   [DEPRECATED] Creates a new migration file. Please use "make:migration" instead.
  session:migration [DEPRECATED] Generates the migration file for database sessions, Please use "make:migration --session" instead.

Housekeeping
  debugbar:clear   Clears all debugbar JSON files.
  logs:clear       Clears all log files.
```

A CodeIgniter Project – Main Folders

- **app/:**
The heart of your CodeIgniter application, containing the models, views, controllers, and configurations.
It's where the majority of your development work will take place.
 - **Controllers/:**
Contains your application's controllers that handle requests, execute business logic, and call views.
 - **Models/:**
Houses the models that represent and work with your application's data.
 - **Views/:**
Where your HTML files go. These files are responsible for rendering the user interface.
 - **Config/:**
Contains configuration files, where you can customize the way CodeIgniter works and behaves.
- **public/:**
The document root of your application. It contains the index.php file, which is the entry point for all requests entering your application, and assets like stylesheets, JavaScript files, and images.



Create a .env file

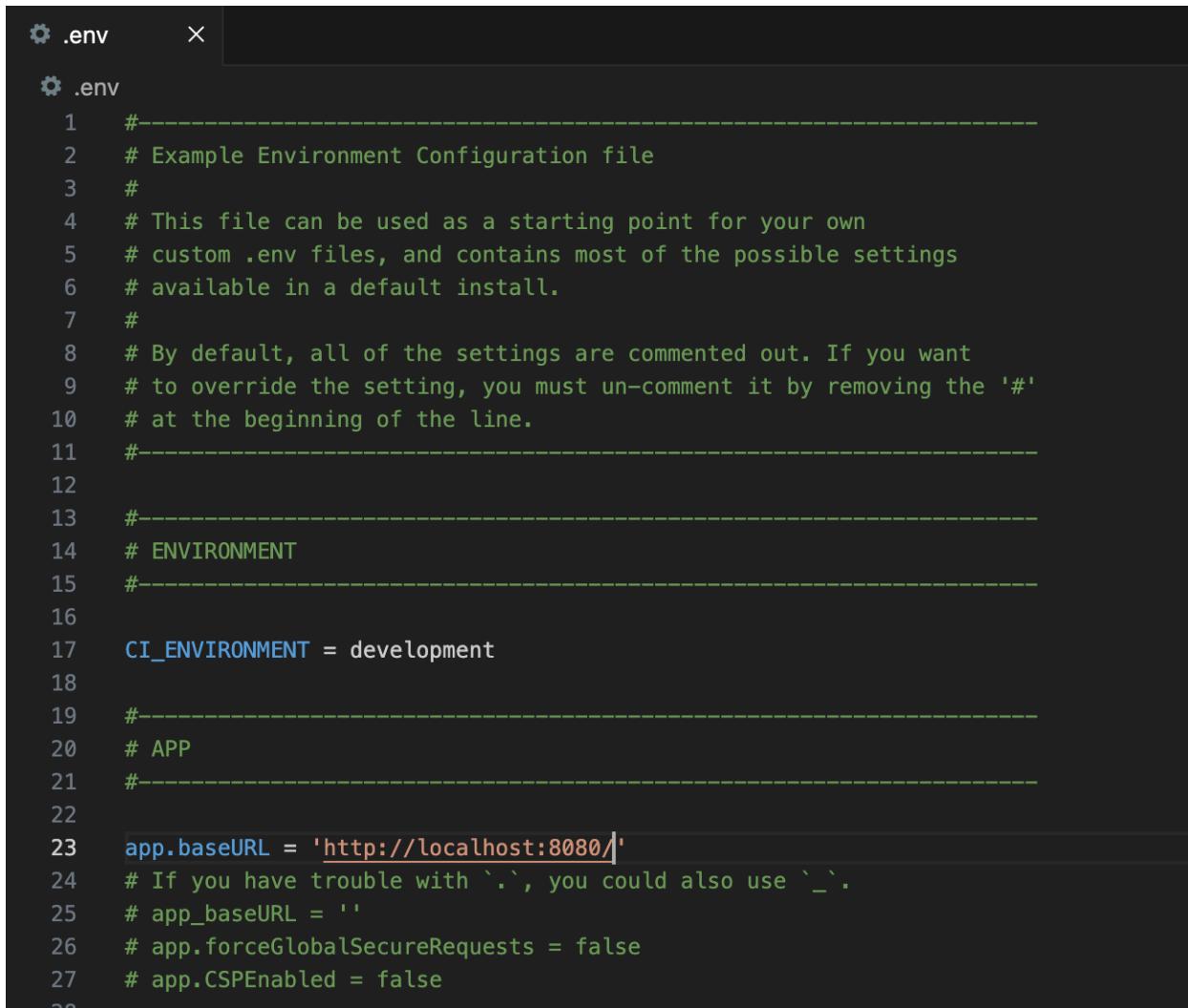
- Copy env to .env

- Set

```
CI_ENVIRONMENT = development  
app.baseURL = localhost or UQCloud Zone URL
```

- Development mode displays:

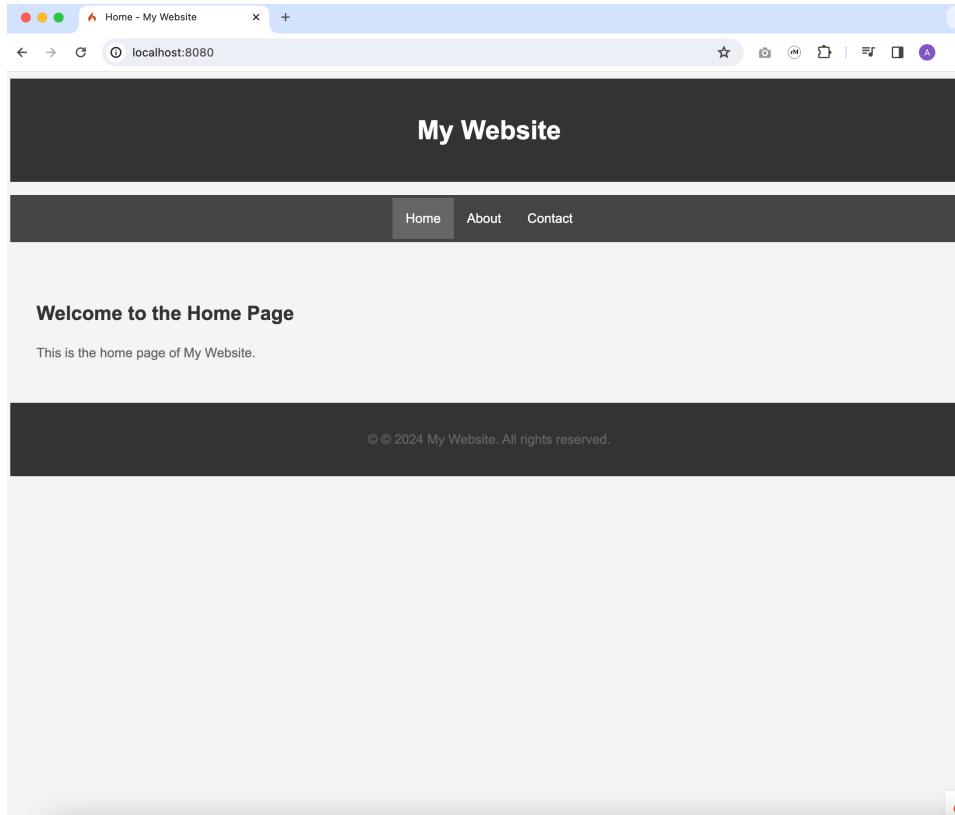
- additional debugging information when an error occurs
- Gives you the CodeIgniter debug toolbar



```
 1  #-----  
 2  # Example Environment Configuration file  
 3  #  
 4  # This file can be used as a starting point for your own  
 5  # custom .env files, and contains most of the possible settings  
 6  # available in a default install.  
 7  #  
 8  # By default, all of the settings are commented out. If you want  
 9  # to override the setting, you must un-comment it by removing the '#'  
10 # at the beginning of the line.  
11 #-----  
12 #-----  
13 #-----  
14 # ENVIRONMENT  
15 #-----  
16  
17 CI_ENVIRONMENT = development  
18 #-----  
19 #-----  
20 # APP  
21 #-----  
22  
23 app.baseURL = 'http://localhost:8080/'  
24 # If you have trouble with `.` , you could also use `_`.  
25 # app_baseURL = ''  
26 # app.forceGlobalSecureRequests = false  
27 # app.CSPEnabled = false  
28
```

The CodeIgniter Debug Toolbar

- Click the CodeIgniter logo in bottom right of browser window
- Can then enable features e.g. see Views and Routes matched



1 APPPATH/Views/base_template.php
2 APPPATH/Views/home.php

Routes

MATCHED ROUTE

Directory: \App\Controllers\Home
Controller: Home
Method: index
Params: 0 / 0

DEFINED ROUTES

Method	Route	Handler
GET	/	\App\Controllers\Home::index
GET	about	\App\Controllers\Home::about
GET	contact	\App\Controllers\Home::contact
GET	simplecss	\App\Controllers\Home::simplecss

39.5 ms 1.568 MB Views 2 Files 124 Routes 5 Events 1 History 1 Vars 4.4.4

Codelgniter 4 - Routes, Controllers and Views

How is the CI 4.4 Welcome Message displayed?

http://localhost:8080

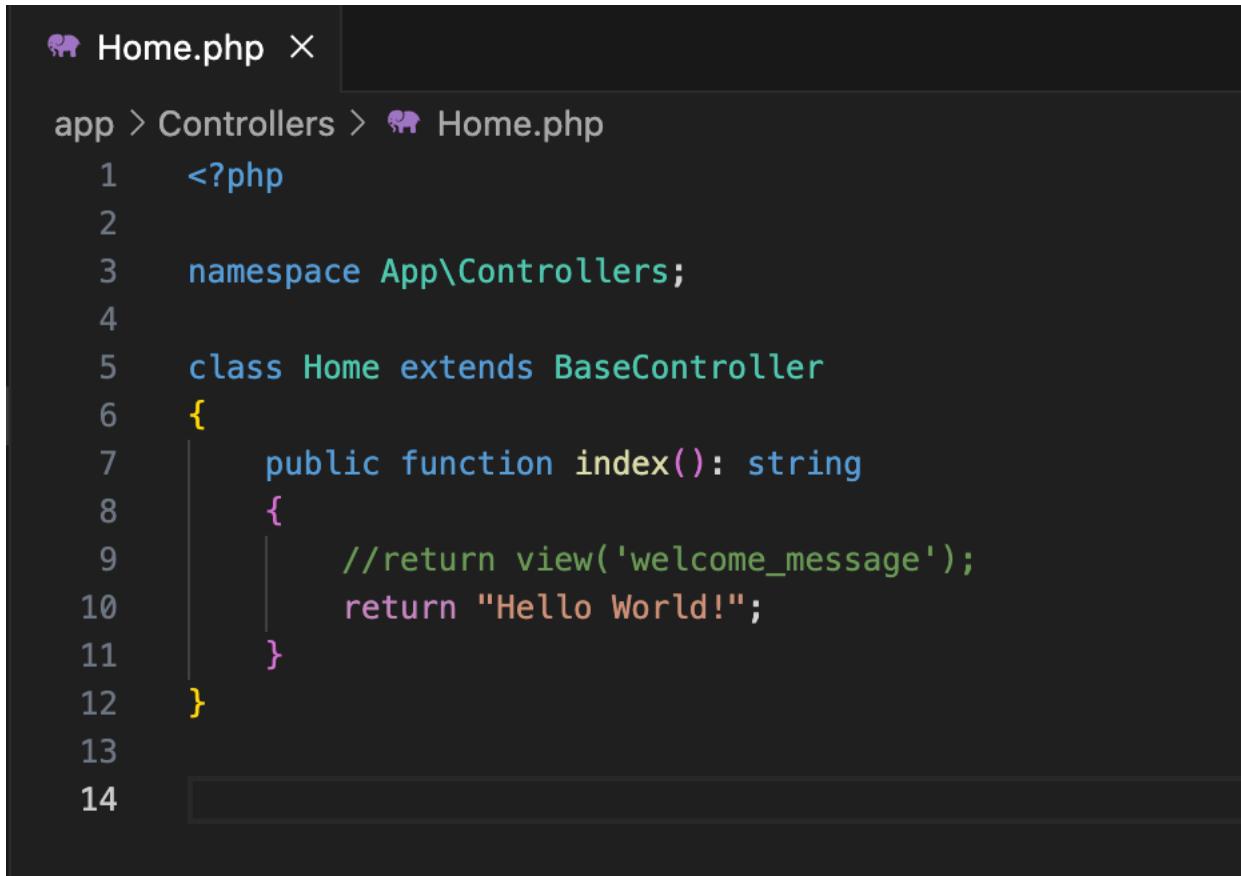
The screenshot shows a code editor interface with a dark theme. On the left is the Explorer sidebar, which lists the project structure under 'CODEIGNITERTEST'. It includes 'app' (Config, Controllers, BaseController.php, Home.php), 'Database', 'Filters', 'Helpers', 'Language', 'Libraries', 'Models', 'ThirdParty', 'Views' (errors, welcome_message.php). The 'Home.php' file is selected in the Explorer. On the right is the main editor area showing 'Home.php' code:

```
<?php  
namespace App\Controllers;  
class Home extends BaseController  
{  
    public function index(): string  
    {  
        return view('welcome_message');  
    }  
}
```

A white arrow points from the 'welcome_message.php' entry in the Explorer to the 'view()' call in the 'index()' method of 'Home.php'. Another white arrow points from the 'welcome_message.php' entry in the Explorer to the 'welcome_message.php' file in the editor.

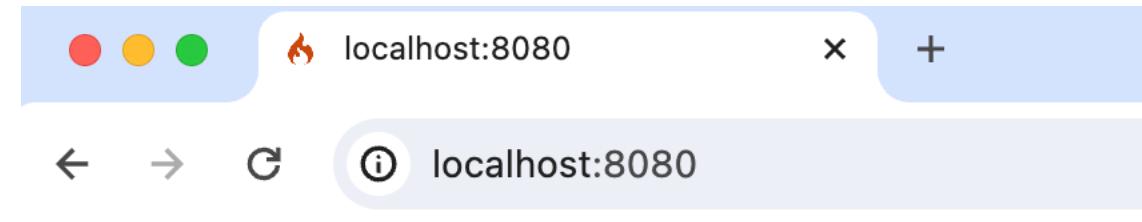
- / path runs the index() method
- the view function specifies the .PHP file to run
- The .PHP file can contain HTML and embedded PHP
- Welcome_message.php contains the “Welcome to CI 4.4 message”

CodeIgniter Hello World - String



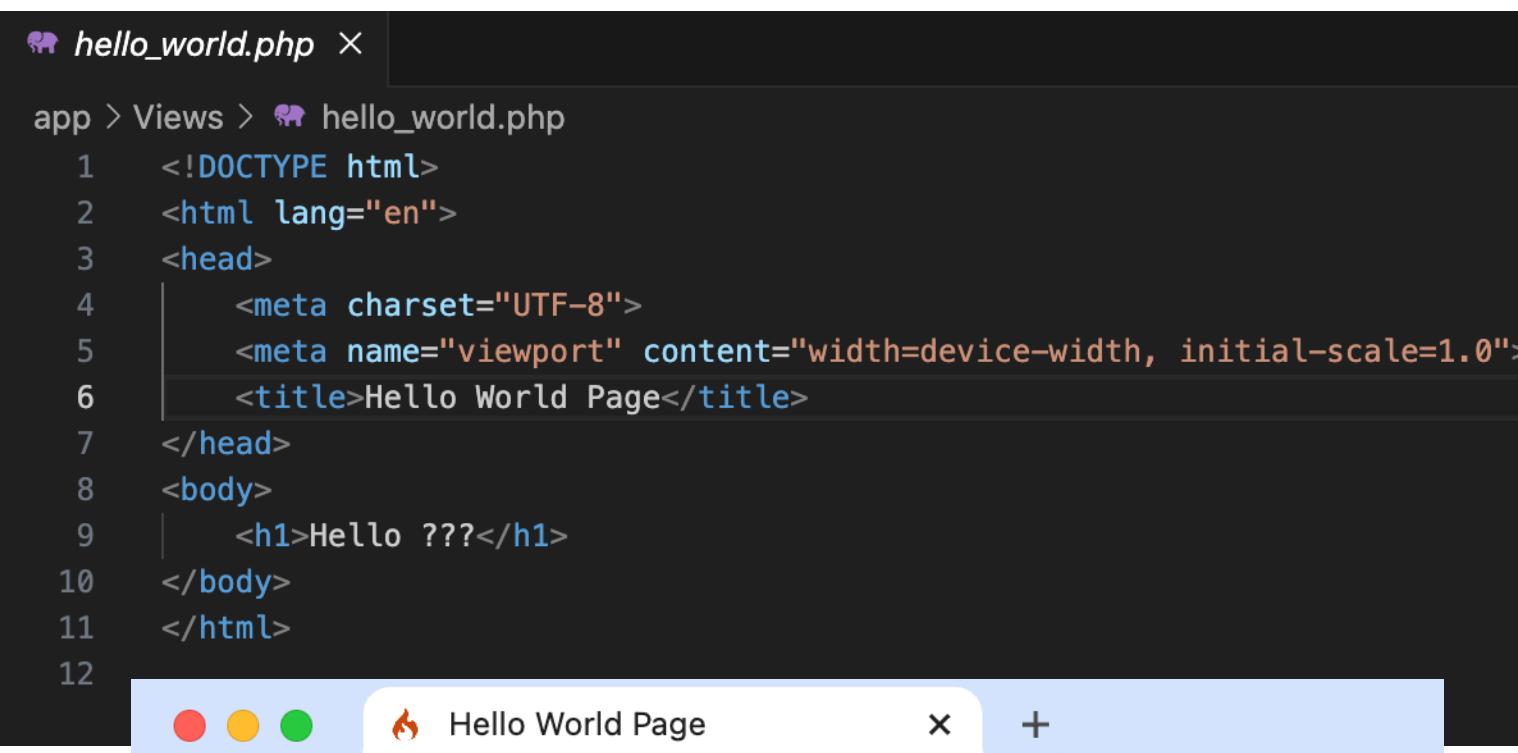
Home.php

```
app > Controllers > Home.php
1  <?php
2
3  namespace App\Controllers;
4
5  class Home extends BaseController
6  {
7      public function index(): string
8      {
9          //return view('welcome_message');
10         return "Hello World!";
11     }
12 }
13
14 
```



- We can just return a string

CodeIgniter Hello World - View



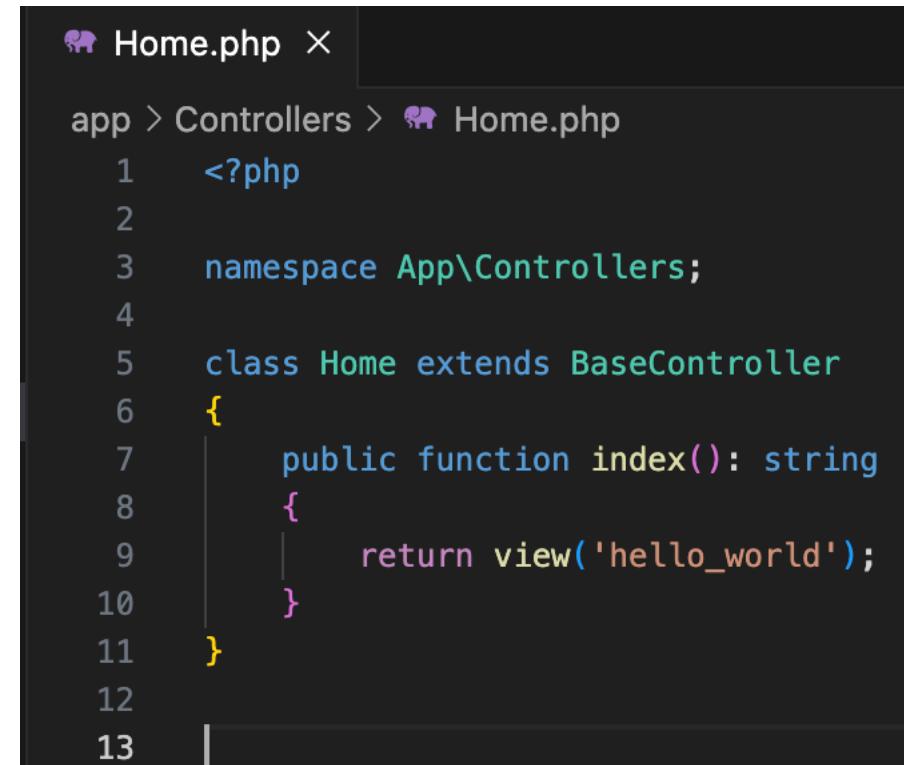
File: `hello_world.php`

app > Views > `hello_world.php`

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Hello World Page</title>
7  </head>
8  <body>
9      <h1>Hello ???</h1>
10 </body>
11 </html>
```

Output preview:

Flame icon: Hello World Page



File: `Home.php`

app > Controllers > `Home.php`

```
1  <?php
2
3  namespace App\Controllers;
4
5  class Home extends BaseController
6  {
7      public function index(): string
8      {
9          return view('hello_world');
10     }
11 }
12
13 |
```

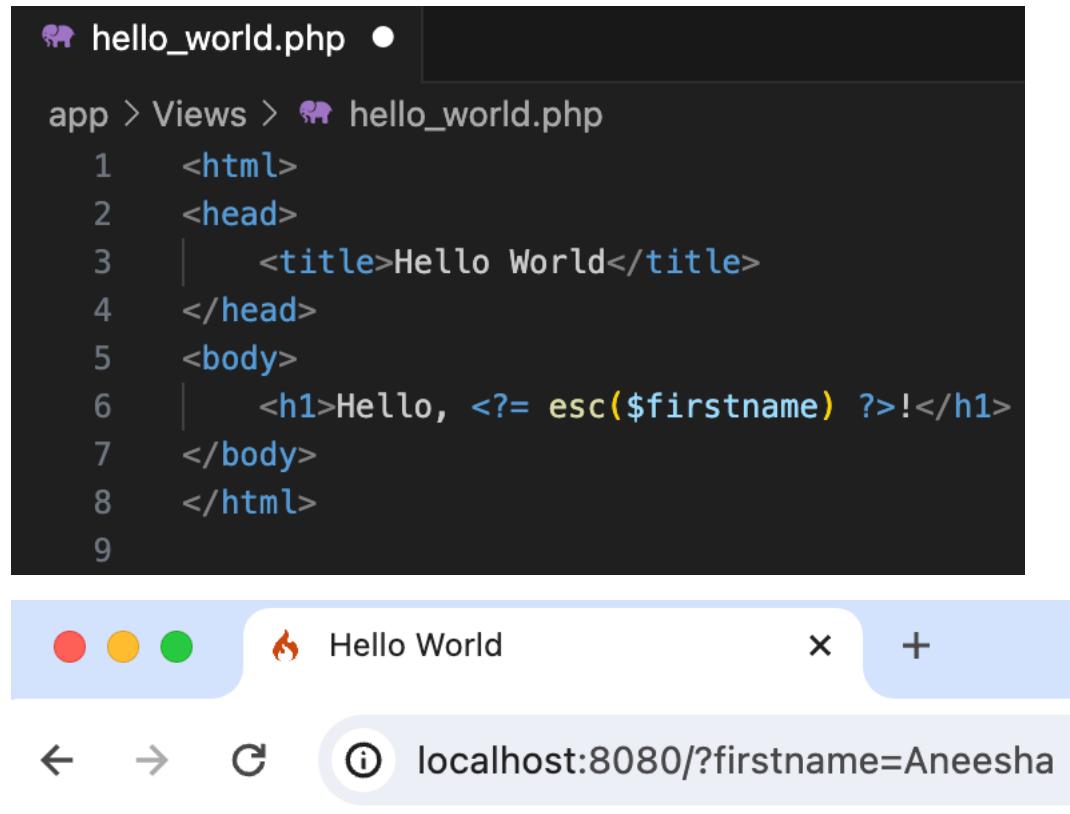
← → ⌂ ⓘ localhost:8080

Hello ???

Note:

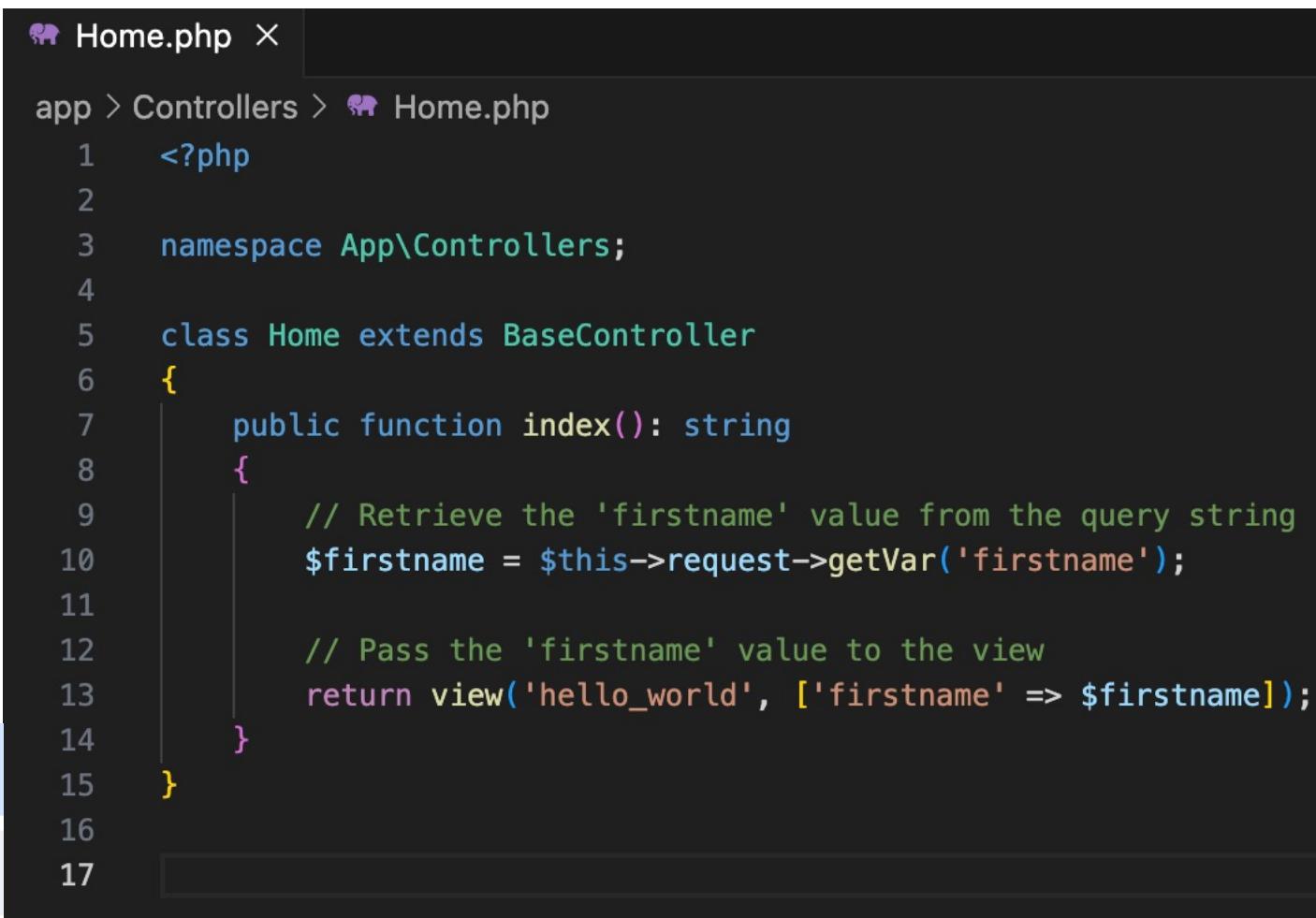
- We don't need to specify the folder with the view files!
- This is a convention.

CodeIgniter Hello World – Pass data to View



The screenshot shows a browser window with the title "Hello World". The address bar indicates the URL is "localhost:8080/?firstname=Aneesha". The page content displays the text "Hello, Aneesha!".

```
❶ <html>
❷   <head>
❸     <title>Hello World</title>
❹   </head>
❺   <body>
❻     <h1>Hello, <?= esc($firstname) ?>!</h1>
❼   </body>
➋ </html>
```



The screenshot shows a code editor with the file "Home.php" open. The code defines a controller named "Home" that extends "BaseController". It contains an "index" method that retrieves the "firstname" value from the query string, escapes it using the "esc" function, and then passes it to the view "hello_world".

```
❶ <?php
❷
❸ namespace App\Controllers;
❹
❺ class Home extends BaseController
❻ {
❽   public function index(): string
❾   {
❿     // Retrieve the 'firstname' value from the query string
❽     $firstname = $this->request->getVar('firstname');

❽     // Pass the 'firstname' value to the view
❽     return view('hello_world', ['firstname' => $firstname]);
❾   }
❿ }
```

Hello, Aneesha!

Note:

- Make sure to use the esc function to escape the output and prevent XSS attacks.

Adding a new Route

- Routes are specified in app/Config/Routes.php file
- Routes maps URL paths to a controller
- Let's add /goodbye
 - Add a route entry in Routes.php
 - Add a method to the Home controller

```
app > Config > 🐘 Routes.php
```

```
1  <?php
2
3  use CodeIgniter\Router\RouteCollection;
4
5  /**
6   * @var RouteCollection $routes
7   */
8  $routes->get('/', 'Home::index');
9
10 $routes->get('/', 'Home::goodbye');
```

```
app > Controllers > 🐘 Home.php
1  <?php
2
3  namespace App\Controllers;
4
5  class Home extends BaseController
6  {
7      public function index(): string
8      {
9          // Retrieve the 'firstname' value from the query string
10         $firstname = $this->request->getVar('firstname');
11
12         // Pass the 'firstname' value to the view
13         return view('hello_world', ['firstname' => $firstname]);
14     }
15
16     public function goodbye(): string
17     {
18         return "Goodbye!";
19     }
20 }
21
```

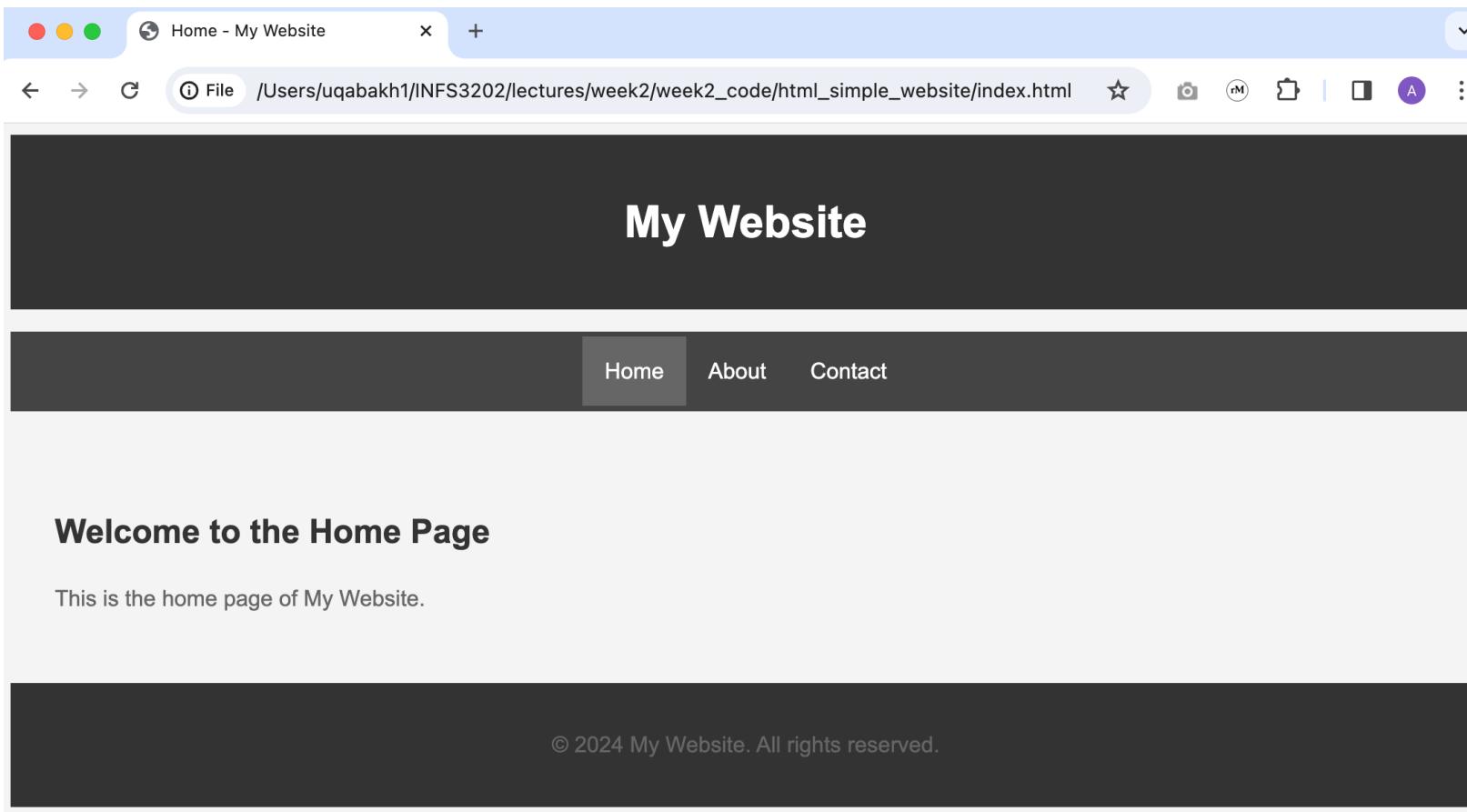
Base Templates for Layouts

`week3_code/Cibasetemplate`

Note: Only the changed files are included not the whole CI project

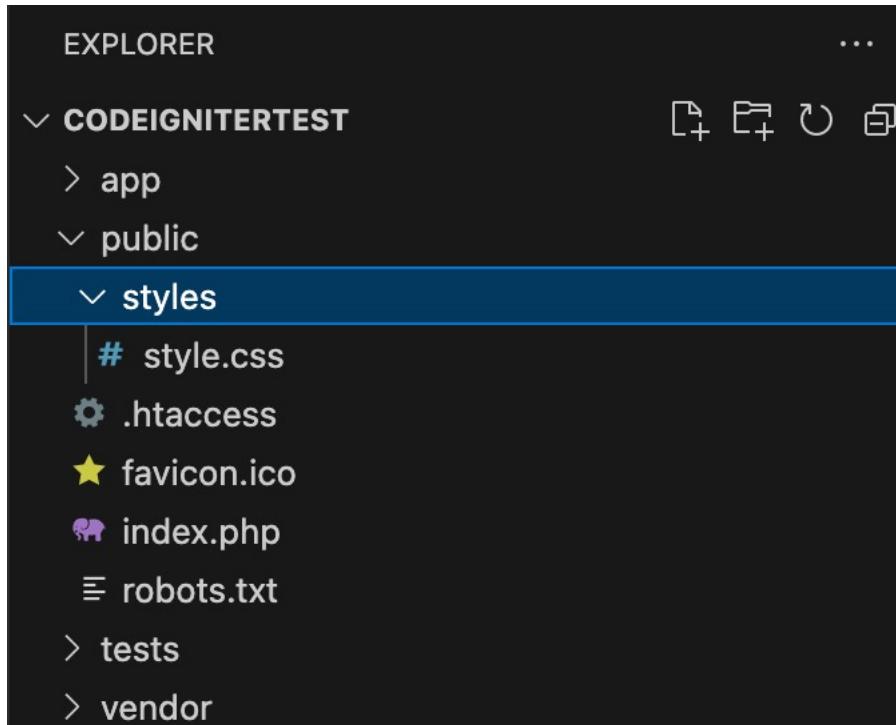
Convert Website to CodeIgniter

- In the Week 2 Lecture we converted this simple website to PHP
- Let's now make a better version using CodeIgniter



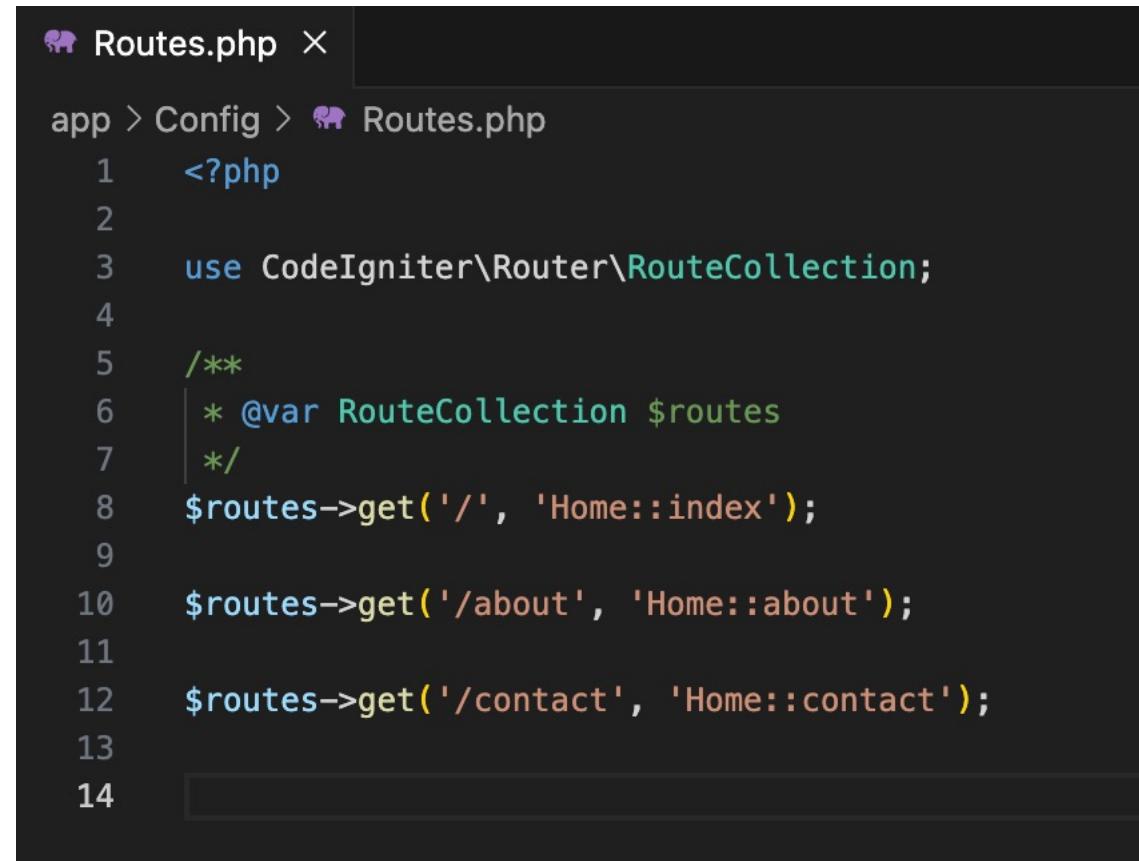
Serving Static files

- Static files eg .js or .css or image files (.jpg, .png) must be placed in the public folder
- For our example website
 - Make a new folder in public folder called styles
 - Copy style.css into the public/styles folder



Creating Routes

- / - Home
- /about – About
- /contact - Contact

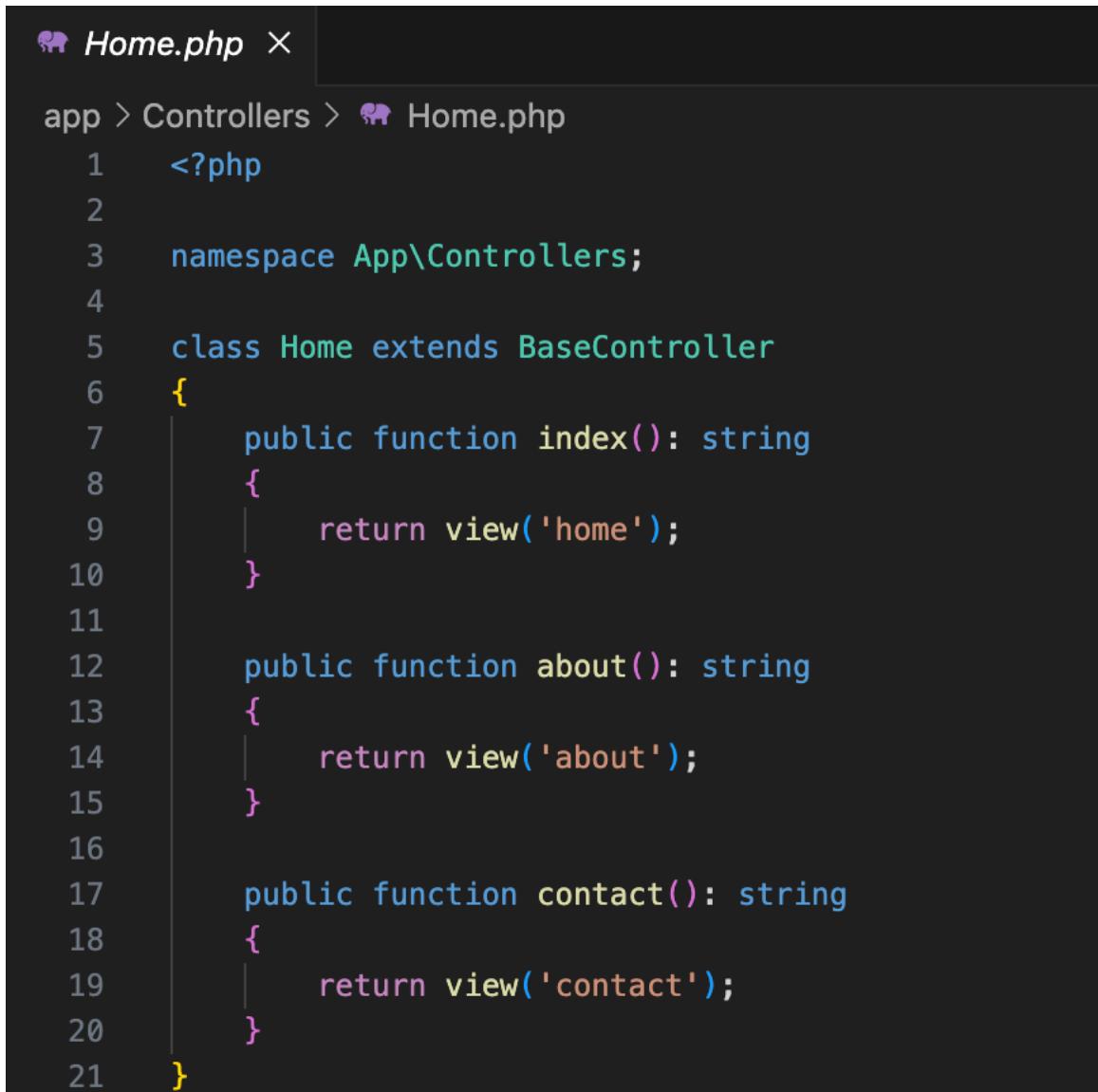


The image shows a screenshot of a code editor with a dark theme. The file being edited is named 'Routes.php' located in the 'Config' folder of an 'app' directory. The code itself is written in PHP and defines three routes using the CodeIgniter Router's RouteCollection class.

```
<?php  
use CodeIgniter\Router\RouteCollection;  
  
/**  
 * @var RouteCollection $routes  
 */  
$routes->get('/', 'Home::index');  
$routes->get('/about', 'Home::about');  
$routes->get('/contact', 'Home::contact');
```

Creating the Controller

- / - Home
- /about – About
- /contact - Contact



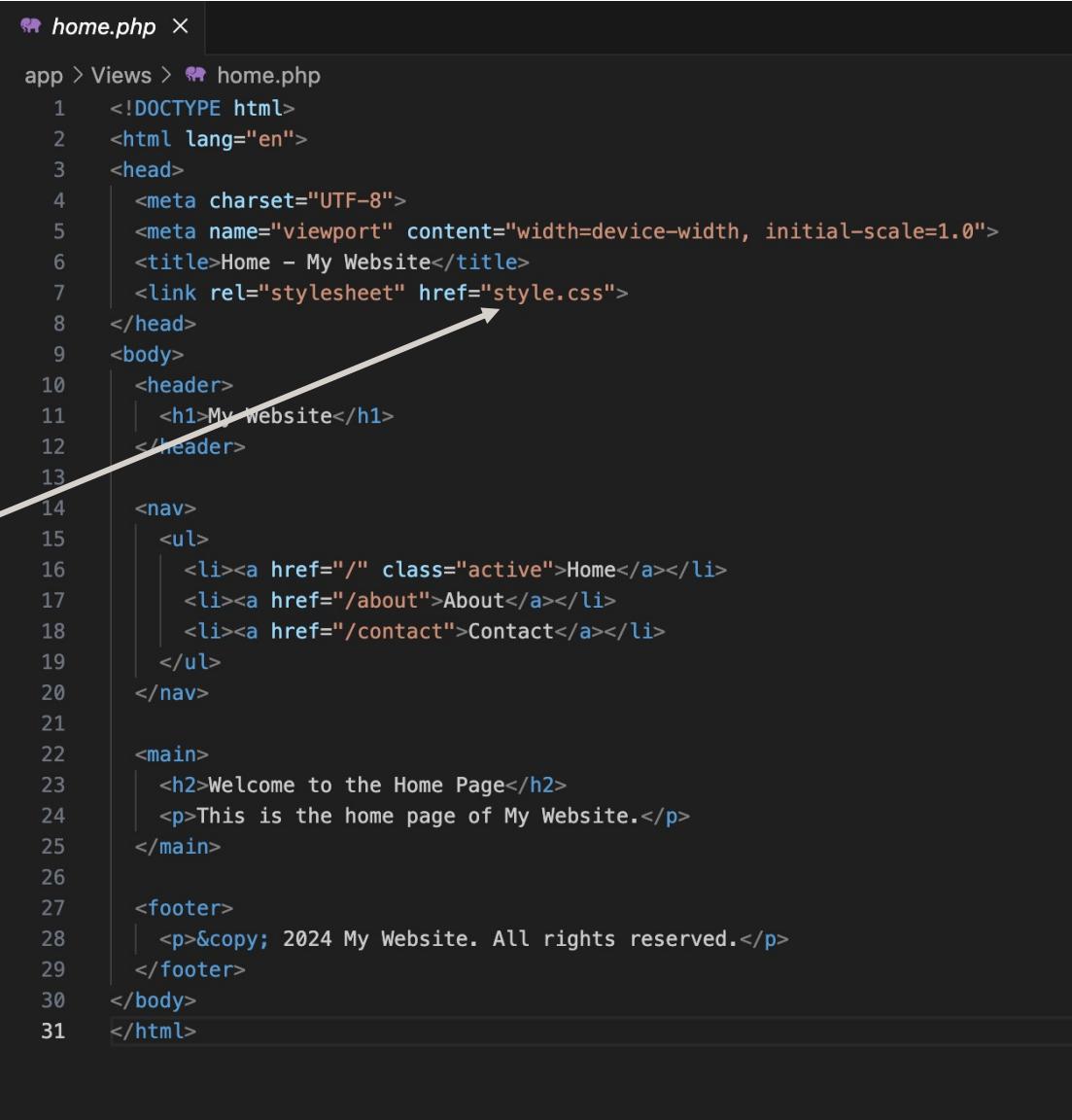
The screenshot shows a code editor window with the file `Home.php` open. The file is located in the `app > Controllers` directory. The code defines a controller class `Home` that extends `BaseController`. It contains three methods: `index()`, `about()`, and `contact()`, each returning a view with the corresponding name.

```
<?php  
namespace App\Controllers;  
  
class Home extends BaseController  
{  
    public function index(): string  
    {  
        return view('home');  
    }  
  
    public function about(): string  
    {  
        return view('about');  
    }  
  
    public function contact(): string  
    {  
        return view('contact');  
    }  
}
```

Creating the View

- / - Home

How can we get the path
to the public folder?



```
home.php x
app > Views > home.php
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |<meta charset="UTF-8">
5  |<meta name="viewport" content="width=device-width, initial-scale=1.0">
6  |<title>Home - My Website</title>
7  |<link rel="stylesheet" href="style.css">
8  </head>
9  <body>
10 |<header>
11 | |<h1>My Website</h1>
12 |</header>
13
14 <nav>
15 |<ul>
16 | |<li><a href="/" class="active">Home</a></li>
17 | |<li><a href="/about">About</a></li>
18 | |<li><a href="/contact">Contact</a></li>
19 |</ul>
20 </nav>
21
22 <main>
23 |<h2>Welcome to the Home Page</h2>
24 |<p>This is the home page of My Website.</p>
25 </main>
26
27 <footer>
28 |<p>&copy; 2024 My Website. All rights reserved.</p>
29 </footer>
30 </body>
31 </html>
```

Loading the URL Helper

app > Controllers >  Home.php

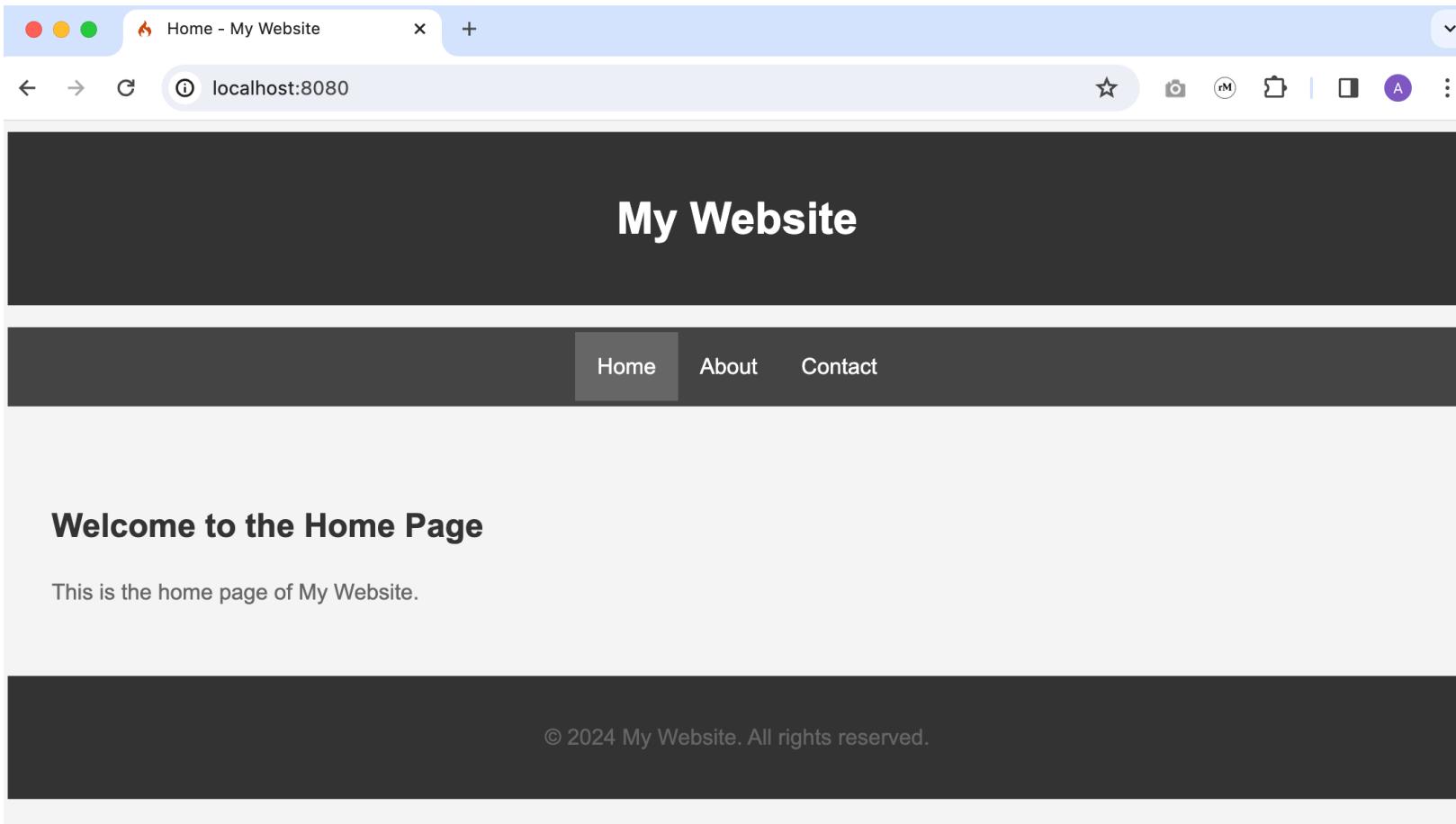
```
1  <?php
2
3  namespace App\Controllers;
4
5  class Home extends BaseController
6  {
7
8      public function __construct()
9      {
10         // Load the URL helper
11         helper('url');
12     }
13
14     public function index(): string
15     {
16         return view('home');
17     }
18
19     public function about(): string
20     {
21         return view('about');
22     }
23
24     public function contact(): string
25     {
26         return view('contact');
27     }
28 }
```

app > Views >  home.php

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Home - My Website</title>
7      <link rel="stylesheet" href="= base_url('styles/style.css'); ?&gt;"&gt;
8
9  &lt;/head&gt;
10 &lt;body&gt;
11     &lt;header&gt;
12         &lt;h1&gt;My Website&lt;/h1&gt;
13     &lt;/header&gt;
14
15     &lt;nav&gt;
16         &lt;ul&gt;
17             &lt;li&gt;&lt;a href="/" class="active"&gt;Home&lt;/a&gt;&lt;/li&gt;
18             &lt;li&gt;&lt;a href="/about"&gt;About&lt;/a&gt;&lt;/li&gt;
19             &lt;li&gt;&lt;a href="/contact"&gt;Contact&lt;/a&gt;&lt;/li&gt;
20         &lt;/ul&gt;
21     &lt;/nav&gt;
22
23     &lt;main&gt;
24         &lt;h2&gt;Welcome to the Home Page&lt;/h2&gt;
25         &lt;p&gt;This is the home page of My Website.&lt;/p&gt;
26     &lt;/main&gt;
27
28     &lt;footer&gt;
29         &lt;p&gt;&amp;copy; 2024 My Website. All rights reserved.&lt;/p&gt;
30     &lt;/footer&gt;
31
32 &lt;/body&gt;
33 &lt;/html&gt;</pre
```

Use the `base_url()` function to get the path to the public folder.

Displaying the Home page



... but we want to share the header and footer across the 3 pages

Creating a Base View Template

Create section using
renderSection() for each section that
will change

```
app > Views > base_template.php
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |  <meta charset="UTF-8">
5  |  <meta name="viewport" content="width=device-width, initial-scale=1.0">
6  |  <title>Home - My Website</title>
7  |  <link rel="stylesheet" href="= base_url('styles/style.css') ?&gt;"&gt;
8
9 &lt;/head&gt;
10 &lt;body&gt;
11  |  &lt;header&gt;
12  |  |  &lt;h1&gt;My Website&lt;/h1&gt;
13  |  &lt;/header&gt;
14
15  &lt;nav&gt;
16  |  &lt;ul&gt;
17  |  |  &lt;li&gt;&lt;a href="/" class="<?= service('request')-&gt;getUri()-&gt;getPath() == '' ? 'active' : '' ; ?&gt;"Home</a></li>
18  |  |  <li><a href="/about" class="= service('request')-&gt;getUri()-&gt;getPath() == 'about' ? 'active' : '' ; ?&gt;"About</a></li>
19  |  |  <li><a href="/contact" class="= service('request')-&gt;getUri()-&gt;getPath() == 'contact' ? 'active' : '' ; ?&gt;"Contact</a></li>
20  |  </ul>
21  </nav>
22
23
24  <main>
25  |  <?= $this->renderSection('content') ?> <!-- Placeholder for page content -->
26  </main>
27
28  <footer>
29  |  <p>&copy; &copy; <?= date('Y') ?> My Website. All rights reserved.</p>
30  </footer>
31 </body>
32 </html>
33
34
```

Using a Base template

🐘 home.php ×

```
app > Views > 🐘 home.php
1   <?= $this->extend('base_template') ?>
2   <?= $this->section('content') ?>
3
4   <h2>Welcome to the Home Page</h2>
5   <p>This is the home page of My Website.</p>
6
7   <?= $this->endSection() ?>
8   |
```

🐘 about.php ×

```
app > Views > 🐘 about.php
1   <?= $this->extend('base_template') ?>
2   <?= $this->section('content') ?>
3
4   <h2>About Us</h2>
5   <p>We are a passionate team working on amazing projects.</p>
6
7   <?= $this->endSection() ?>
8
9
```

🐘 contact.php ×

```
app > Views > 🐘 contact.php
1   <?= $this->extend('base_template') ?>
2   <?= $this->section('content') ?>
3
4   <h2>Contact Us</h2>
5   <p>Feel free to get in touch with us.</p>
6
7   <?= $this->endSection() ?>
```

Base Templates for Layouts Demo

[week3_code/Cibasetemplate](#)

Note: Only the changed files are included not the whole CI project

Loading Multiple Views

CodeIgniter will intelligently handle multiple calls to `$this->load->view()` from within a controller. This is an alternate to using base templates.

If more than one call happens they will be appended together.

```
1 <?php
2
3 class Page extends CI_Controller {
4
5     public function index()
6     {
7         $data['page_title'] = 'Your title';
8         $this->load->view('header');
9         $this->load->view('menu');
10        $this->load->view('content', $data);
11        $this->load->view('footer');
12    }
13
14 }
```

Data is passed from the **controller** to the **view** by way of an **array** or an **object** in the **second parameter** of the view loading method. Here is an example using an array:

Your view files can also be stored within sub-directories if you prefer that type of organization.

```
11         $this->load->view('footer');
12         $this->load->view('directory_name/file_name');
```

Dynamic Data

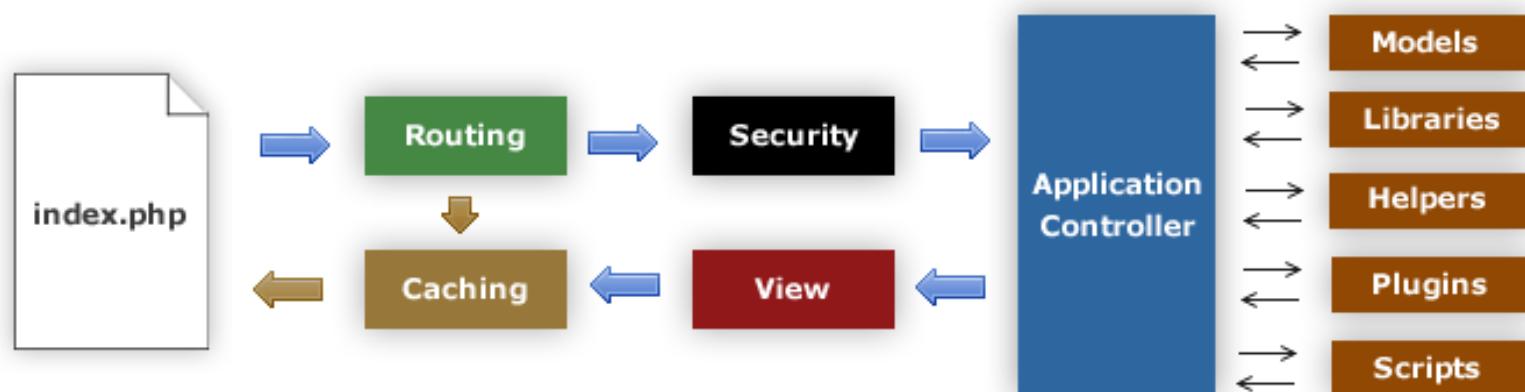
If your \$data has multi-dimensions (e.g. 2D array), you can create a loop in view code to display all data.

```
1 <?php
2 class Blog extends CI_Controller {
3
4     public function index()
5     {
6         $data[ 'todo_list' ] = array( 'Prac on Monday',
7             'Lecture on Thursday',
8             'Birthday Party on Saturday');
9
10        $data[ 'title' ] = "My Plan in Week 3";
11        $data[ 'heading' ] = "Todos:";
12
13        $this->load->view( 'myview', $data );
14    }
15 }
```

```
1 <html>
2 <head>
3 | | | <title><?php echo $title;?></title>
4 </head>
5 <body>
6 | | | <h1><?php echo $heading;?></h1>
7
8 | | | <h3>My Todo List</h3>
9
10 | | <ul>
11 | | | <?php foreach ($todo_list as $item):?>
12 | | | | <li><?php echo $item;?></li>
13 | | | <?php endforeach;?>
14 | | </ul>
15
16 | | </body>
17
18 | | </html>
```

CodeIgniter Workflow

- The index.php serves as the front controller, initializing the base resources needed to run CodeIgniter.
- The Router examines the HTTP request to determine what should be done with it.
- If a cache file exists, it is sent directly to the browser, bypassing the normal system execution.
- Security: Before the application controller is loaded, the HTTP request and any user-submitted data are filtered for security.
- The Controller loads the model, core libraries, helpers, and any other resources needed to process the specific request.
- The finalized View is rendered and then sent to the web browser to be seen. If caching is enabled, the view is cached first so that on subsequent requests it can be served.



UI Design with CSS Frameworks

Benefits of Using a CSS Framework

- **Consistent Design:**
 - Ensures a uniform look and feel across your website or application, promoting a cohesive user experience.
- **Rapid Development:**
 - Speeds up the development process by providing a ready-made set of CSS classes and components (buttons, forms, navigation bars, etc.).
- **Responsiveness:**
 - Most frameworks come with built-in support for responsive design, making it easier to create websites that work well on desktops, tablets, and mobiles without additional effort.
- **Accessibility:**
 - Many frameworks encourage accessibility standards, helping to create websites that are usable by people with a wide range of disabilities.
- **Cross-browser Compatibility:**
 - Takes care of common cross-browser compatibility issues, ensuring that your site functions and appears consistently across various browsers.
- **Customization:**
 - Allows for customization of the default styles to match your brand or design requirements, providing flexibility while maintaining the framework's benefits.
- **Community Support and Documentation:**
 - Benefits from extensive documentation and community support, making it easier to learn, troubleshoot issues, and stay updated with best practices.

Common CSS Frameworks

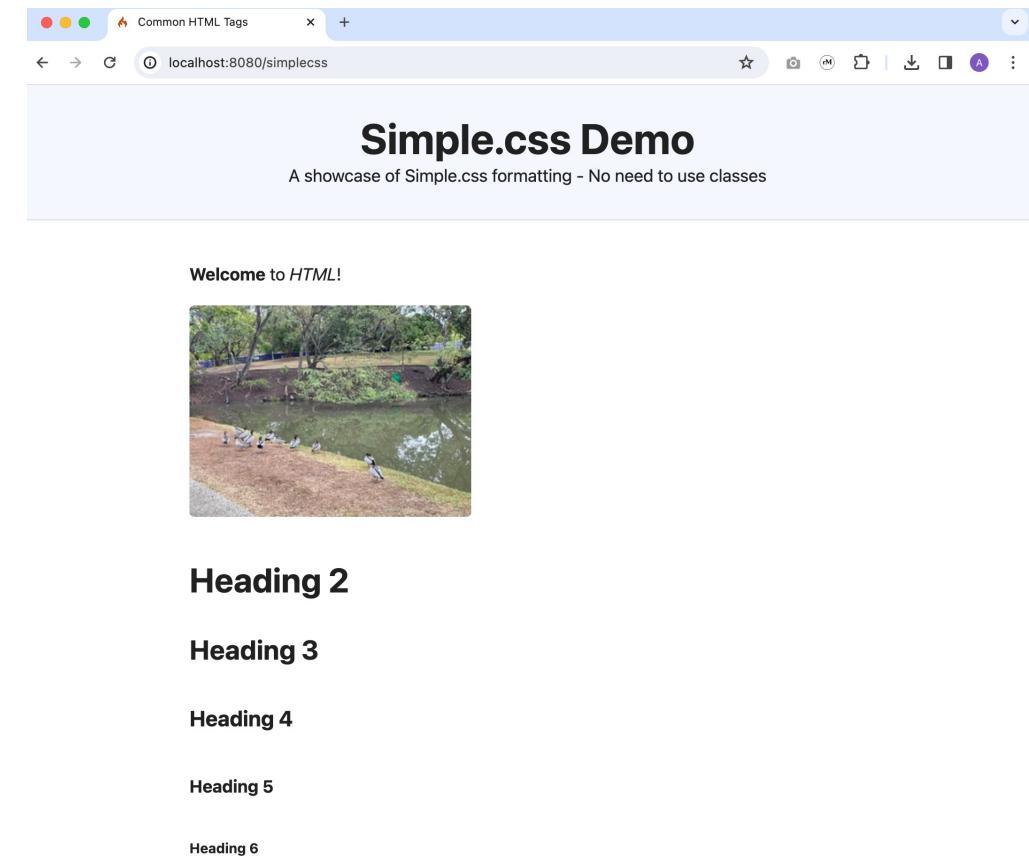
- **Classless Frameworks:**
 - Eg Simple.css <https://simplecss.org/>
- **Bootstrap:**
 - <https://getbootstrap.com/>
- **Tailwind CSS:**
 - <https://tailwindcss.com/>
- **Daisy UI:**
 - Component Library for Tailwind CSS
<https://daisyui.com/>

To use these frameworks you need to place the CSS files (and any associated js files) in the public directory of a CodeIgniter project.

Using Simple.css

- **Download from:**
 - <https://github.com/kevquirk/simple.css/blob/main/simple-v1.min.css>
- **Place css file in public/styles folder:**

```
simplecss.php X
app > Views > simplecss.php
1  <!DOCTYPE html>
2  <html>
3  <head>
4  | <title>Common HTML Tags</title>
5  | <link rel="stylesheet" href="= base_url('styles/simple-v1.min.css') ?&gt;"&gt;
6  &lt;/head&gt;
7  &lt;body&gt;
8  &lt;header&gt;
9  | &lt;h1&gt;Simple.css Demo&lt;/h1&gt;
10 | &lt;p&gt;A showcase of Simple.css formatting - No need to use classes&lt;/p&gt;
11 &lt;/header&gt;
12
13 &lt;main&gt;
14 | &lt;p&gt;&lt;b&gt;Welcome&lt;/b&gt; to &lt;i&gt;HTML&lt;/i&gt;!&lt;/p&gt;
15
16 &lt;img src="<?= base_url('images/uq_duck_meeting.jpg') ?&gt;" alt="A meeting of ducks"&gt;
17
18 &lt;h2&gt;Heading 2&lt;/h2&gt;
19 &lt;h3&gt;Heading 3&lt;/h3&gt;
20 &lt;h4&gt;Heading 4&lt;/h4&gt;
21 &lt;h5&gt;Heading 5&lt;/h5&gt;
22 &lt;h6&gt;Heading 6&lt;/h6&gt;
23
24 &lt;div&gt;
25 | &lt;h2&gt;Ordered List:&lt;/h2&gt;
26 | &lt;ol&gt;
27 | | &lt;li&gt;Item 1&lt;/li&gt;
28 | | &lt;li&gt;Item 2&lt;/li&gt;
29 | | &lt;li&gt;Item 3&lt;/li&gt;
30 | &lt;/ol&gt;
31 &lt;/div&gt;
32</pre
```



Using Bootstrap

- **Download Zip from :**
 - <https://github.com/twbs/bootstrap/releases/download/v5.3.3/bootstrap-5.3.3-dist.zip>
- **Place minified css and js in the public folder:**

The screenshot shows the Bootstrap documentation page at getbootstrap.com/docs/5.3/getting-started/download/. The page has a purple header with navigation links for Docs, Examples, Icons, Themes, Blog, a search bar, and social media links. A sidebar on the left lists 'Getting started' and 'Customize' sections. The main content area features a 'Download' section with a heading, a 'View on GitHub' button, and a 'Squarespace' advertisement. Below this is a 'Compiled CSS and JS' section with a 'Download' button.

Docs Examples Icons Themes Blog

Search

View on GitHub

On this page

Compiled CSS and JS

Source files Examples CDN via jsDelivr Alternative CDNs Package managers npm yarn RubyGems Composer NuGet

Getting started

- Introduction
- Download**
- Contents
- Browsers & devices
- JavaScript
- Webpack
- Parcel
- Vite
- Accessibility
- RFS
- RTL
- Contribute

Customize

- Overview
- Sass
- Options
- Color
- Color modes
- Components
- CSS variables

Download

Download Bootstrap to get the compiled CSS and JavaScript, source code, or include it with your favorite package managers like npm, RubyGems, and more.

Build the best looking website for any business with Squarespace.
ads via Carbon

Compiled CSS and JS

Download ready-to-use compiled code for **Bootstrap v5.3.3** to easily drop into your project, which includes:

- Compiled and minified CSS bundles (see [CSS files comparison](#))
- Compiled and minified JavaScript plugins (see [JS files comparison](#))

This doesn't include documentation, source files, or any optional JavaScript dependencies like Popper.

Download

Using Bootstrap - Components

- View Docs
- You will be able to see the HTML and styles that you can re-use
- Once you include the CSS and JS in your HTML, you can start using the styles
- Bootstrap uses classes
- Review all the available components

The screenshot shows a web browser displaying the Bootstrap v5.3 documentation for buttons. The URL in the address bar is `getbootstrap.com/docs/5.3/components/buttons/`. The page has a purple header with the Bootstrap logo, navigation links for Docs, Examples, Icons, Themes, and Blog, a search bar, and social media links. On the left, there's a sidebar with links to Overview, Form control, Select, Checks & radios, Range, Input group, Floating labels, Layout, Validation, and a Components section listing Accordion, Alerts, Badge, Breadcrumb, Buttons (which is highlighted), Button group, Card, Carousel, Close button, Collapse, Dropdowns, List group, Modal, Navbar, and Navs & tabs. The main content area features a heading 'Buttons' with a sub-copy about using Bootstrap's custom button styles. Below this is a 'Base class' section with a description, a 'Base class' example, and its corresponding HTML code. A sidebar on the right lists additional topics under 'Base class' and 'CSS'.

On this page

- Base class
- Variants
- Disable text wrapping
- Button tags
- Outline buttons
- Sizes
- Disabled state
- Link functionality caveat
- Block buttons
- Button plugin
- Toggle states
- Methods

CSS

- Variables
- Sass variables
- Sass mixins
- Sass loops

Base class

Bootstrap has a base `.btn` class that sets up basic styles such as padding and content alignment. By default, `.btn` controls have a transparent border and background color, and lack any explicit focus and hover styles.

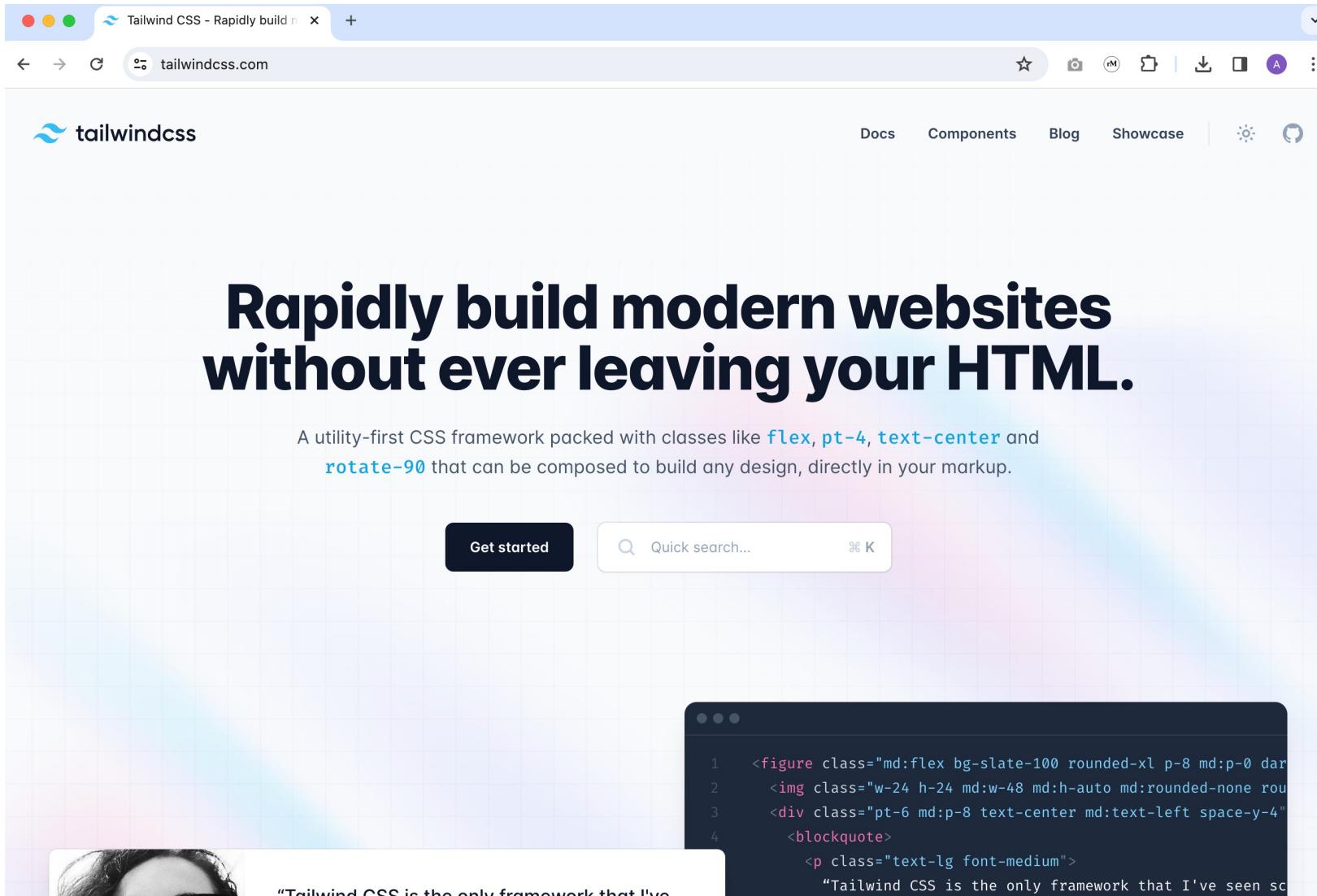
Base class

HTML

```
<button type="button" class="btn">Base class</button>
```

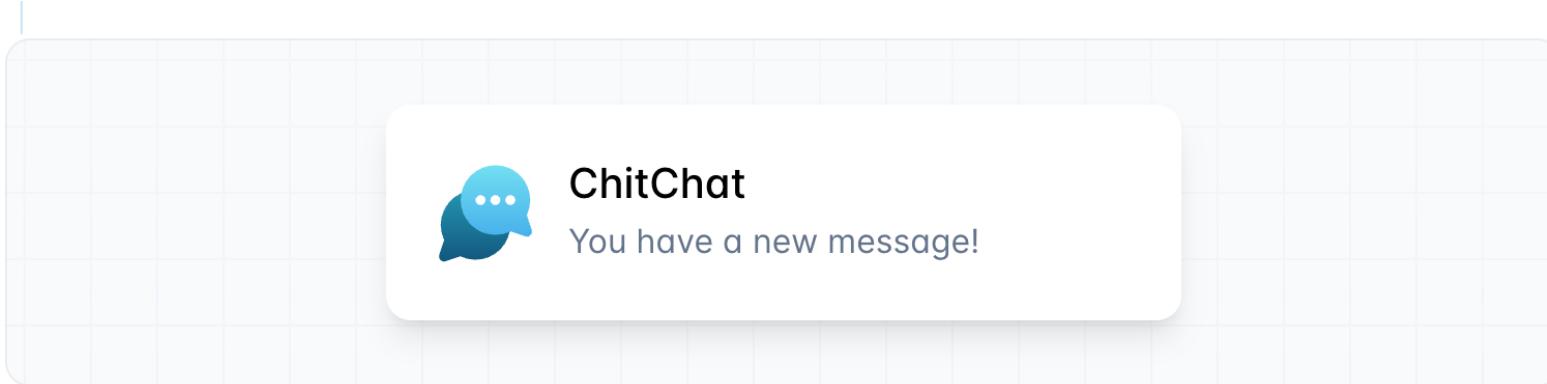
Tailwind CSS

- Tailwind is a set of utility CSS classes that you can mix and match all within HTML



What do utility classes look like?

- From <https://tailwindcss.com/docs/utility-first>



```
<div class="p-6 max-w-sm mx-auto bg-white rounded-lg shadow-lg flex items-center space-between">
  <div class="shrink-0">
    
  </div>
  <div>
    <div class="text-xl font-medium text-black">ChitChat</div>
    <p class="text-slate-500">You have a new message!</p>
  </div>
</div>
```

Tailwind Playground

- [Playground: `https://play.tailwindcss.com/`](https://play.tailwindcss.com/)
- Also see Tailwind Cheatsheets: <https://tailwindcomponents.com/cheatsheet/>

The screenshot shows the Tailwind Play interface. On the left, there's a code editor with syntax highlighting for Tailwind CSS. The code is a template for a landing page. On the right, there's a preview area showing the rendered Tailwind CSS output.

```
1 1!--
2 2Welcome to Tailwind Play, the official Tailwind CSS playground!
3 3
4 4Everything here works just like it does when you're running Tailwind
5 5locally
6 6with a real build pipeline. You can customize your config file, use
7 7features
8 8like `@apply`, or even add third-party plugins.
9 9
10 10Feel free to play with this example if you're just learning, or trash it
11 11and
12 12start from scratch if you know enough to be dangerous. Have fun!
13 13-->
14 14<div class="relative flex min-h-screen flex-col justify-center
15 15overflow-hidden bg-gray-50 py-6 sm:py-12">
16 16
18 18<div class="absolute inset-0 bg-[url(/img/grid.svg)] bg-center
19 19[mask-image:linear-gradient(180deg,white,rgba(255,255,255,0))]"></div>
20 20<div class="relative bg-white px-6 pt-10 pb-8 shadow-xl ring-1
21 21ring-gray-900/5 sm:mx-auto sm:max-w-lg sm:rounded-lg sm:px-10">
22 22<div class="mx-auto max-w-md">
23 23
24 24<div class="divide-y divide-gray-300/50">
```

tailwind PLAY

An advanced online playground for Tailwind CSS, including support for things like:

- ✓ Customizing your `tailwind.config.js` file
- ✓ Extracting classes with `@apply`
- ✓ Code completion with instant preview

Perfect for learning how the framework works, prototyping a new idea, or creating a demo to share online.

Want to dig deeper into Tailwind?
[Read the docs →](#)

Using Tailwind CSS

- **Install the CLI and follow the instructions:**
 - <https://tailwindcss.com/docs/installation>
- Tailwind needs to watch your Views as you build them and dynamically create the output.css file based on the classes you are using.

Get started with Tailwind CSS

Tailwind CSS works by scanning all of your HTML files, JavaScript components, and any other templates for class names, generating the corresponding styles and then writing them to a static CSS file.

It's fast, flexible, and reliable — with zero-runtime.

Installation

[Tailwind CLI](#) [Using PostCSS](#) [Framework Guides](#) [Play CDN](#)

The simplest and fastest way to get up and running with Tailwind CSS from scratch is with the Tailwind CLI tool. The CLI is also available as a [standalone executable](#) if you want to use it without installing Node.js.

The screenshot shows the Tailwind CSS documentation page. At the top, there's a navigation bar with links to 'Tailwind CLI', 'Using PostCSS', 'Framework Guides', and 'Play CDN'. Below this, a section titled 'Get started with Tailwind CSS' explains how Tailwind CSS works by scanning files for class names and generating CSS. A sub-section 'Installation' is highlighted. To the right, there are two terminal windows. The top window shows the command to install the Tailwind CLI via npm: `npm install -D tailwindcss` and `npx tailwindcss init`. The bottom window shows a code editor with the contents of a `tailwind.config.js` file, which includes configuration for template paths.

```
tailwind.config.js
/** @type {import('tailwindcss').Config} */
module.exports = {
  content: ['./src/**/*.{html,js}'],
}
```

Using Tailwind CSS

- **Install the CLI and follow the instructions:**

- <https://tailwindcss.com/docs/installation>

A screenshot of a forum post from forum.codeigniter.com. The post is by a user named **encodedigital**, a Junior Member, posted on 09-08-2023 at 04:03 PM. The post has 6 comments. The user's profile includes a placeholder profile picture, a yellow double-star rating, and stats: Posts: 11, Threads: 4, Joined: Oct 2022, Reputation: 0.

The post content starts with a message: "Hi, I am successfully using Tailwind CSS in my CodeIgniter 4 project. If anyone still searching it, then I followed below steps:

Step 1: In the CodeIgniter project folder, run below commands to install node package and create blank **`tailwind.config.js`**

Code:

```
npm install -D tailwindcss
npx tailwindcss init
```

Step 2: Update **`tailwind.config.js`** as below:

Code:

```
/** @type {import('tailwindcss').Config} */
module.exports = {
  content: ['./app/Views/**/*.*'],
  theme: {
    extend: {},
  },
  plugins: [],
}
```

Step 3: Create input.css with below code. You can create it anywhere. I created at app > Views > css > input.css

Code:

```
@tailwind base;
@tailwind components;
@tailwind utilities;
```

Step 4: On terminal in CodeIgniter project folder, run below command. If you using different path for input.css and output css (mine styles.css) then change path accordingly.

Code:

```
npx tailwindcss -i ./app/Views/css/input.css -o ./public/assets/css/styles.css --watch
```

Step 5: Now just include out css file (mine styles.css) in your php files. With Step 4 command, it will keep running, so if you add any tailwind css class in php file, it will automatically add that to output css file.
Hope this helps!

From <https://forum.codeigniter.com/showthread.php?tid=86279>

Daisy UI

- Components built with Tailwind utility classes

The screenshot shows the daisyUI website homepage. At the top, there's a navigation bar with a search bar and links for 'Components' and 'Theme'. Below the header, there's a large call-to-action section with the text 'The most popular component library for Tailwind CSS' and a subtext explaining how it adds component class names to Tailwind CSS. It also highlights 'Pure CSS. No JS dependency' and 'Works on all frameworks'. On the right side, there's a visual representation of the 'Design system' featuring various UI components like checkboxes and radios in different sizes and colors, along with a color palette for semantic colors.

daisyUI — Tailwind CSS Components

daisyui.com

Components Theme

4.7.2

Search...

npm i -D daisyui daisyUI 4 upgrade guide

The most popular component library for Tailwind CSS

daisyUI adds component class names to Tailwind CSS so you can make beautiful websites faster than ever.

See components

How to use? →

Daisy UI – Components

The screenshot shows the Daisy UI components documentation page for the Alert component. The URL in the browser is daisyui.com/components/alert/.

The left sidebar lists various UI components: Breadcrumbs, Bottom navigation, Link, Menu, Navbar, Pagination, Steps, Tab (updated), Feedback, Alert (selected), Loading, Progress, Radial progress, Skeleton (new), Toast, Tooltip, Data input, Checkbox, File Input, Radio, Range, and Rating.

The main content area displays the Alert component. It includes a preview section showing a light gray alert box with rounded corners containing the text "12 unread messages. Tap to see." and a small info icon. Below the preview are tabs for Preview, HTML, and JSX. The HTML tab is selected, showing the following code:

```
<div role="alert" class="alert">
  <svg xmlns="http://www.w3.org/2000/svg" fill="none" viewBox="0 0 24 24" class="stroke-info">
    <span>12 unread messages. Tap to see.</span>
  </div>
```

On the right side, there is a sidebar with the Daisy UI logo and version 4.7.2, a search bar, and links for Components and Theme. It also features a "Sponsors" section with a Dynamic Process logo and a "Become a sponsor" button. A sidebar advertisement for GitLab DevSecOps Platform is present, along with an "ADS VIA CARBO" note.

Daisy UI – Installation

- Add as a plugin for Tailwind in tailwind.config.js

The screenshot shows a web browser with three tabs open: "Install daisyUI as a Tailwind CSS plugin", "daisyUI: CDN + Tailwind CSS", and "Codeigniter 4 w/ Tailwind CSS". The main content is titled "Install daisyUI as a Tailwind CSS plugin". It provides instructions for installation via "Tailwind Plugin" or "Use CDN". It also notes the requirement for Node.js and Tailwind CSS. Step 1 shows the command "npm i -D daisyui@latest" and step 2 shows the addition of "daisyui" to a tailwind.config.js file. A sidebar on the left lists various components like Docs, Components, and Data display. A sidebar on the right lists sponsors like Dynamic Process and Auth0 by Okta.

How to install daisyUI as a Tailwind CSS plugin?

Tailwind Plugin Use CDN

You need [Node.js](#) and [Tailwind CSS](#) installed.

1. Install daisyUI:
`npm i -D daisyui@latest`
2. Then add daisyUI to your `tailwind.config.js` files:
`module.exports = {
 ...
 plugins: [require("daisyui")],
}`

daisyUI example repositories

See example setup of daisyUI and Tailwind CSS on different frameworks and build tools.

INFS3202/7202 Project and Design Document

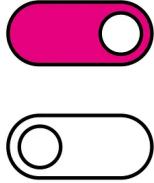
- You can use any CSS Framework and any components
- You are not allowed to use Admin templates or purchase Templates
- Review components and assemble each page in your Web App
- You will need to do this to create the HTML mockups for your Design Document Assessment Item
- Your HTML mockups will then become Views in your final Project Assessment Item
- In the Week 5 Lab you will get experience in using a CSS Framework

Using GenAI (Part 1)

Types of GenAI Products

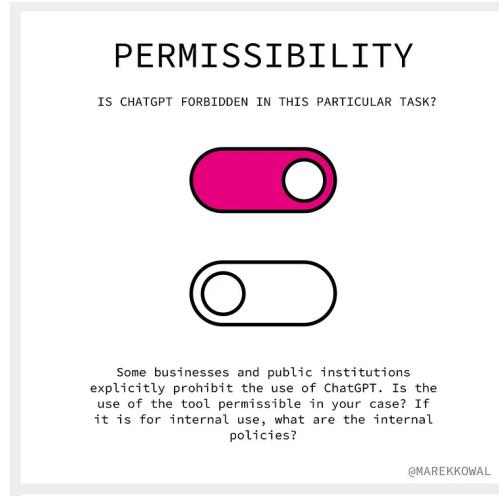
- Open AI ChatGPT – Chatbot
<https://chat.openai.com/>
 - Most off a free version after sign-up (sometimes for a short period of time).
- Google Gemini - Chatbot
<https://gemini.google.com/app>
 - There is no need to subscribe or pay for any GenAI to complete this course.
- Microsoft Bing Chat/CoPilot - Chatbot
<https://www.bing.com/chat>
 - There is no need to subscribe or pay for any GenAI to complete this course.
- Github CoPilot – AI Coding Assistant in VS Code
<https://github.com/features/copilot>
 - There is no need to subscribe or pay for any GenAI to complete this course.

Should I use GenAI?

PERMISSIBILITY	SUITABILITY
<p>IS CHATGPT FORBIDDEN IN THIS PARTICULAR TASK?</p>  <p>Some businesses and public institutions explicitly prohibit the use of ChatGPT. Is the use of the tool permissible in your case? If it is for internal use, what are the internal policies?</p> <p>@MAREKKOWAL</p>	<p>IS CHATGPT THE RIGHT TOOL FOR THE JOB?</p>  <p>Are you using ChatGPT just because it's easy? Are there other tools that are more suitable for the task? For instance, ChatGPT is not well suited for tasks requiring performing mathematical calculations.</p> <p>@MAREKKOWAL</p>
<p>PRIVACY</p> <p>WILL YOUR PROMPTS CONTAIN CONFIDENTIAL INFORMATION?</p>  <p>Large language models like ChatGPT retain the prompts. In some cases, these prompts may be used to further train the system and impact its future responses. Did you ensure that you're not using any sensitive or confidential information in your prompts?</p> <p>@MAREKKOWAL</p>	<p>ACCOUNTABILITY</p> <p>IS THERE A PROCESS FOR HUMAN APPROVAL OF THE CONTENT?</p>  <p>How much human input and accountability is there? Is there someone reviewing or modifying the outputs before they're shared or is the content generated autonomously with no explicit human review or approval?</p> <p>@MAREKKOWAL</p>

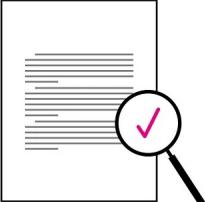
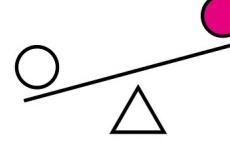
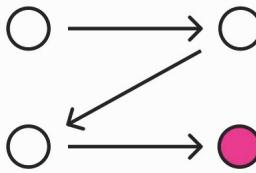
Should I use ChatGPT?

Should I use GenAI?



- At University
 - Courses and assessment items have different requirements
- At Work
 - Projects will have different Intellectual Property and Privacy requirements

What to keep in mind when using GenAI?

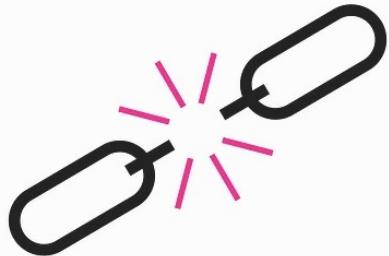
TRUTHFULNESS IS THE CONTENT FACTUALLY CORRECT?  <p>ChatGPT does not have a notion of truth. Instead, it relies on associations between concepts. It does not know anything about the world past 2021. Are you checking the content generated for factual errors?</p> <p>@MAREKKOWAL</p>	LACK OF BIAS IS THE CONTENT ETHICAL AND FREE OF BIAS?  <p>ChatGPT was trained on text written by humans. While its creators try to remove any biases, unethical or offensive language, it might still produce such content. Are you actively trying to debias and balance the content?</p> <p>@MAREKKOWAL</p>
COMMON SENSE DOES THE CONTENT OVERALL MAKE SENSE?  <p>ChatGPT can generate fluent and coherent text, but the logic might be missing. Are you checking the generated content for consistency?</p> <p>@MAREKKOWAL</p>	TRANSPARENCY DID YOU ACKNOWLEDGE THE USE OF CHATGPT?  <p>Just like with other forms of automation, it is appropriate, and sometimes required by law, to disclose the level of contribution of an algorithm. Make sure you are transparent about how you used ChatGPT in your work.</p> <p>@MAREKKOWAL</p>
What to keep in mind when using ChatGPT?	

<https://marekkowal.substack.com/p/chatgptchecks>

Ongoing Considerations

SELF-RELIANCE

DO YOU HAVE A BACKUP PLAN IF CHATGPT WAS NOT AVAILABLE?



ChatGPT is just a tool. Sometimes it might become unavailable. Are you able to maintain business continuity in case you are temporarily unable to use the algorithm or when you decide to stop using it completely?

@MAREKKOWAL

ALERTNESS

ARE YOU ALWAYS ON THE LOOKOUT FOR SURPRISING OUTCOMES?



ChatGPT occasionally produces very surprising content. It can work well for long periods and then, suddenly, create undesirable outputs. How are you maintaining your vigilance?

@MAREKKOWAL

Ongoing considerations.

Ongoing Considerations



- What if you don't have access, or it is not allowed where you work or on a specific project?
- At University, GenAI can be allowed in Assessments but you may still have an exam?
- As a University trained professional in IT or Computer Science you need a higher level of skills:
 - Design
 - Implementation
 - Debugging

Example Prompt – Act as a Tutor

- **Prompt**
Please act as an experienced Web Developer and Tutor.
You specialise in CodeIgniter 4.
I will ask you about concepts and you will explain them in simple language and give examples.
- This is called Persona prompting.

ChatGPT 3.5 ▾

A You
Please act as an experienced Web Developer and Tutor. You specialise in CodeIgniter 4. I will ask you about concepts and you will explain them in simple language and give examples.

ChatGPT
Of course! I'd be happy to help. What would you like to know about CodeIgniter 4?

A You
What is Routing?

ChatGPT
Routing in web development refers to the process of directing incoming web requests to the appropriate controller or handler based on the URL. In simpler terms, it's like a roadmap for your web application that tells it which code to run when a user visits a certain URL.

In CodeIgniter 4, routing is configured in the `app/Config/Routes.php` file. Here's a simple explanation of how it works:

1. **Defining Routes:** You specify URL patterns and associate them with specific controllers and methods. For example:

```
php
$route->get('blog', 'Blog::index');
```

Example Prompt – Be Specific

- ## Prompt

Please act as an experienced Web Developer and CodeIgniter 4 Developer. Convert this HTML into a base template and make Routes, a single Controller called WebsiteController and Views for the home, about and contact pages that use the base template.

```

include HTML

```

- ## Include your specific technology.



ChatGPT

To achieve this in CodeIgniter 4, you would follow these steps:

1. **Create a Base Template:** This will include the common structure like header, navigation, footer, etc.
2. **Set Up Routes:** Define routes for the home, about, and contact pages.
3. **Create Controller:** Create a single controller, `WebsiteController`, to handle requests for these pages.
4. **Create Views:** Develop views for the home, about, and contact pages that extend the base template.

Let's go through each step:

1. Create Base Template (`app/Views/templates/base.php`)

```
php Copy code
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title><?= $title ?></title>
    <link rel="stylesheet" href="<?= base_url('style.css') ?>">
</head>
<body>
    <header>
        <h1>My Website</h1>
    </header>
```

Prompting - Read, Test, Debug Cycles

- Need ability to read code and see what is incorrect!

Or

- Rapidly test in a project
- Continue chatting to fix issues
- Repeat

The View should be home
which then loads the base
template!

- Important to know when to write
yourself!



```
php
<?php

namespace App\Controllers;

use CodeIgniter\Controller;

class WebsiteController extends Controller
{
    public function home()
    {
        $data = [
            'title' => 'Home - My Website',
            'page' => 'home'
        ];

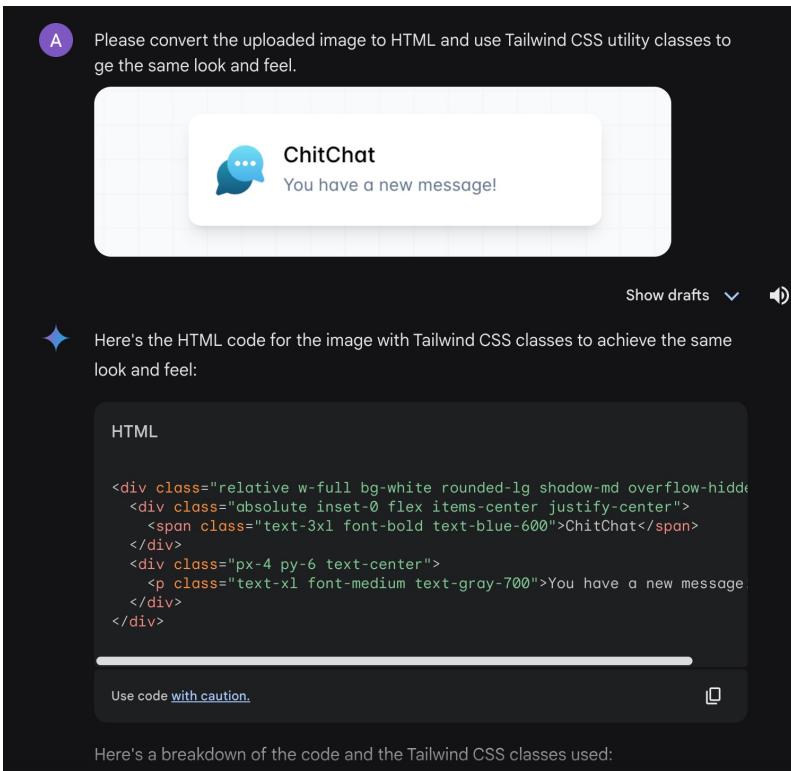
        echo view('templates/base', $data);
    }

    public function about()
    {
        $data = [
            'title' => 'About - My Website',
            'page' => 'about'
        ];

        echo view('templates/base', $data);
    }
}
```

Example Prompt – Multimodal Support

- **Prompt**
Please convert the uploaded image to HTML and use Tailwind CSS utility classes to get the same look and feel.
- **You can upload images, even hand drawn UI's**

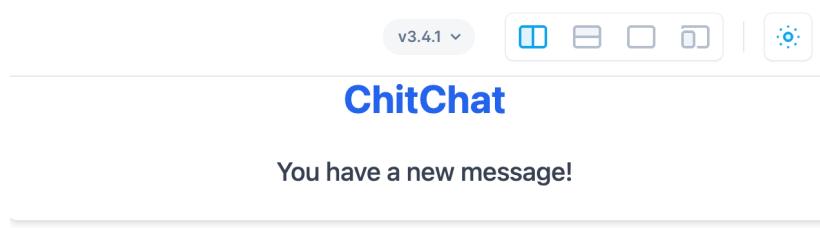


A screenshot of the TailwindPlay interface. The top navigation bar shows "play.tailwindcss.com", the Tailwind logo, and a "Share" button. On the right, there are icons for camera, file, and other tools. The main area has tabs for "HTML", "CSS", and "Config". The "HTML" tab displays the following code:

```
1 <div class="relative w-full bg-white rounded-lg shadow-md overflow-hidden">
  2   <div class="absolute inset-0 flex items-center justify-center">
  3     <span class="text-3xl font-bold text-blue-600">ChitChat</span>
  4   </div>
  5   <div class="px-4 py-6 text-center">
  6     <p class="text-xl font-medium text-gray-700">You have a new message!</p>
  7   </div>
  8 </div>
  9
 10
```

To the right, a preview window shows a dark-themed mobile application with a white notification card containing the text "ChitChat" and "You have a new message!".

- Run in TailwindPlay, get new screengrab, paste in and ask for a fix



In Using GenAI (Part 2)

- Advance Prompting & Running Local LLM models

Week 3: Todo

- Weekly RiPPL E task is due in Week 3 on Friday at 3pm
- Complete Week 3 Lab 2
- Start your Design Document Assessment Item



CREATE CHANGE

Q&A



CREATE CHANGE

Thank you