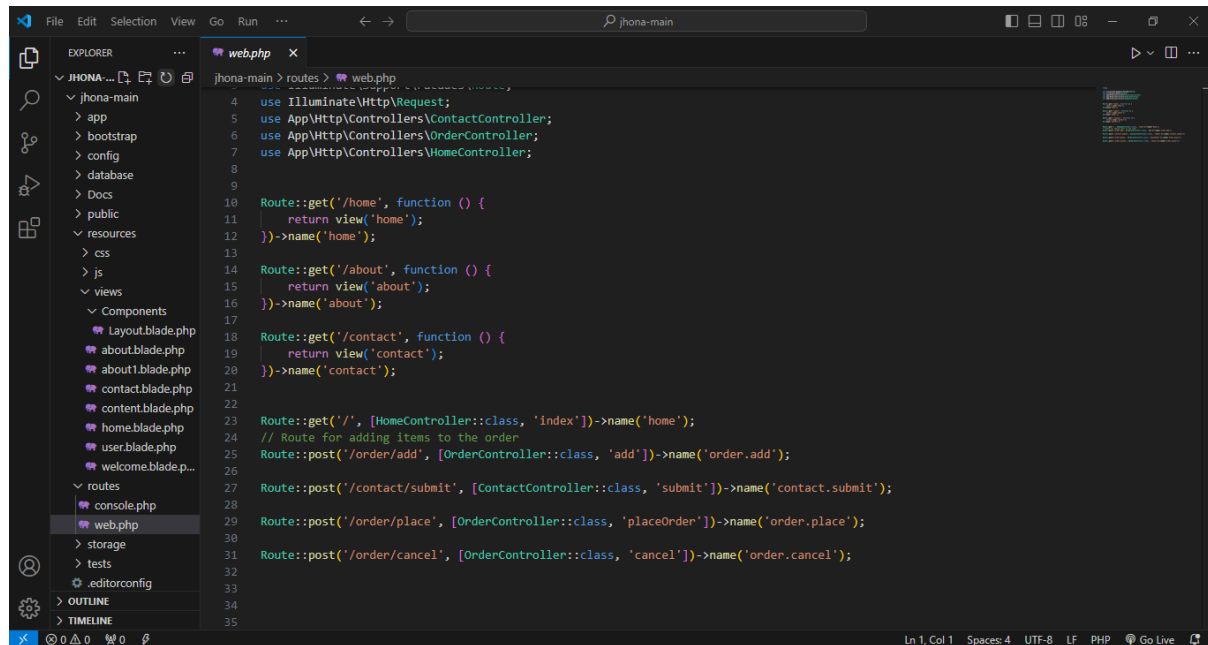


KIM ANTONETTE BRON

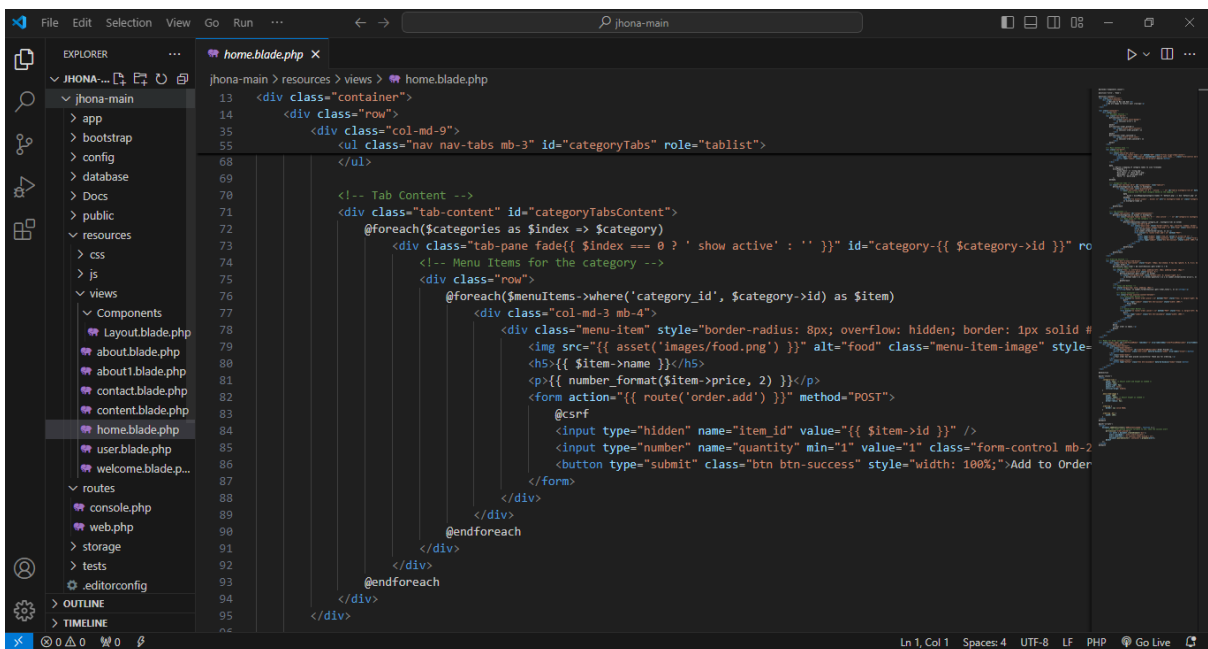
BSIT 3C

WEB DEVELOPMENT



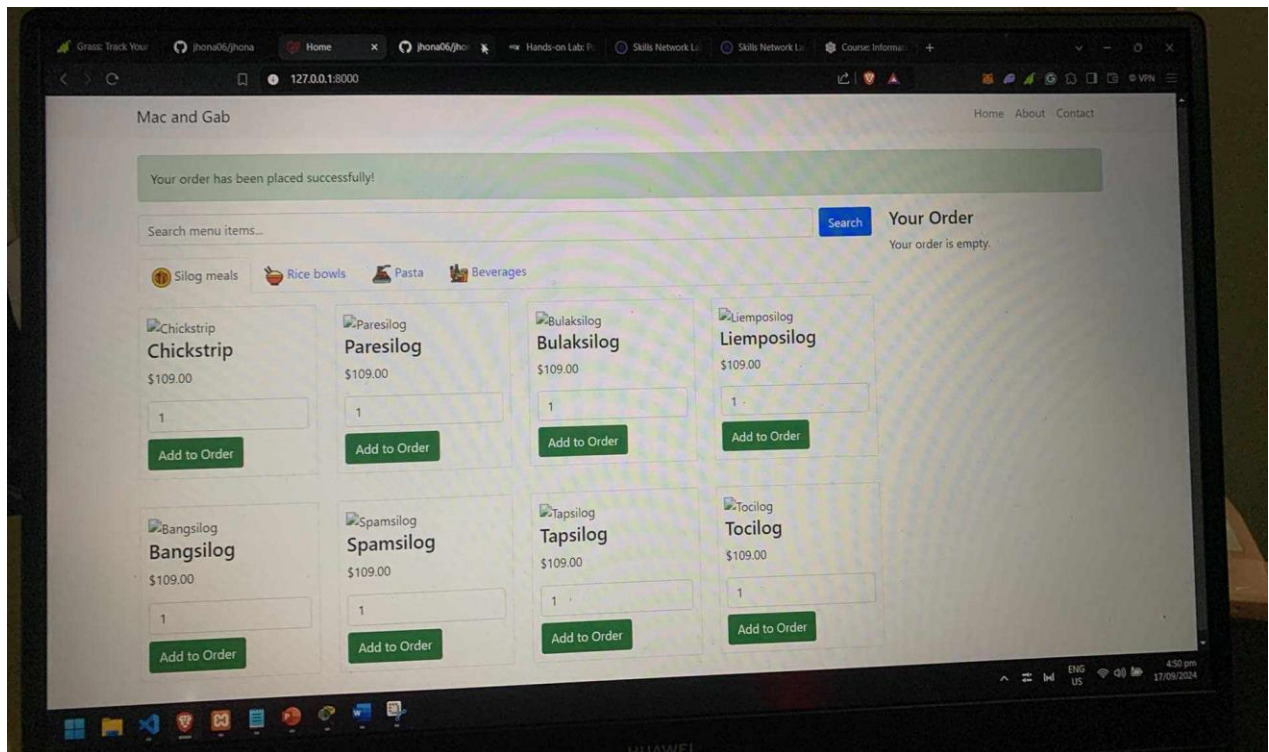
The screenshot shows the Visual Studio Code editor with a project named 'jhona-main'. The Explorer sidebar on the left shows the file structure, with 'resources > routes > web.php' selected. The main editor displays the contents of 'web.php', which defines several routes for the application. The routes include a home page, an about page, a contact page, and routes for adding, placing, and canceling orders. The code is as follows:

```
1 use Illuminate\Http\Request;
2 use App\Http\Controllers\ContactController;
3 use App\Http\Controllers\OrderController;
4 use App\Http\Controllers\HomeController;
5
6 Route::get('/home', function () {
7     return view('home');
8 })->name('home');
9
10 Route::get('/about', function () {
11     return view('about');
12 })->name('about');
13
14 Route::get('/contact', function () {
15     return view('contact');
16 })->name('contact');
17
18 Route::get('/', [HomeController::class, 'index'])->name('home');
19 // Route for adding items to the order
20 Route::post('/order/add', [OrderController::class, 'add'])->name('order.add');
21
22 Route::post('/contact/submit', [ContactController::class, 'submit'])->name('contact.submit');
23
24 Route::post('/order/place', [OrderController::class, 'placeOrder'])->name('order.place');
25
26 Route::post('/order/cancel', [OrderController::class, 'cancel'])->name('order.cancel');
```



The screenshot shows the Visual Studio Code editor with the same 'jhona-main' project. The Explorer sidebar shows 'resources > views > home.blade.php' selected. The main editor displays the Blade template for the home page. The code uses Laravel's Blade syntax to create a container, a row, and a tabbed interface for categories. It includes a loop for categories and a loop for menu items within each category. The code is as follows:

```
13 <div class="container">
14     <div class="row">
15         <div class="col-md-9">
16             <ul class="nav nav-tabs mb-3" id="categoryTabs" role="tablist">
17                 <!-- Tab Content -->
18                 <div class="tab-content" id="categoryTabsContent">
19                     @foreach($categories as $index => $category)
20                         <div class="tab-pane fade{{ $index == 0 ? ' show active' : '' }}" id="category-{{ $category->id }}" role="tabpanel">
21                             <!-- Menu Items for the category -->
22                             <div class="row">
23                                 @foreach($menuItems->where('category_id', $category->id) as $item)
24                                     <div class="col-md-3 mb-4">
25                                         <div class="menu-item" style="border-radius: 8px; overflow: hidden; border: 1px solid #ccc;">
26                                             
27                                             <h5>{{ $item->name }}</h5>
28                                             <p>{{ number_format($item->price, 2) }}</p>
29                                             <form action="{{ route('order.add') }}" method="POST">
30                                                 @csrf
31                                                 <input type="hidden" name="item_id" value="{{ $item->id }}" />
32                                                 <input type="number" name="quantity" min="1" value="1" class="form-control mb-2"/>
33                                                 <button type="submit" class="btn btn-success" style="width: 100%;>Add to Order
34                                             </form>
35                                         </div>
36                                     </div>
37                                 @endforeach
38                             </div>
39                         </div>
40                     @endforeach
41                 </div>
42             </div>
43         </div>
44     </div>
45 </div>
```



Q1: Purpose of the layout file and its usage.

The layout file functions as our main container, while the other three Blade files act as content sections or components that we can insert into the layout.

Q2: How each view file extends the layout and inserts specific content.

Each Blade view file holds reusable content that we can easily insert as needed. This approach maintains a clear separation of concerns, making it easier for us to manage and troubleshoot the code.

Q3: Routing setup and how it serves the views.

In the web.php file, we added a route for the Layout Blade file, allowing us to integrate it into the project. The components are dynamically rendered within the layout.

Q4: Challenges faced and how they were resolved.

We faced a small challenge where each operating system needed specific dependencies. To resolve this, we installed the necessary dependencies for each device accordingly.

Q5: Difference between `{{ $slot }}` and `@yield`

`{{ $slot }}` is used as a placeholder for reusable components that can be rendered according to our needs, while `@yield` defines where child views or components will be injected into the parent layout.