

# **Laporan Manual Book Frontend-User PWBS-TA-Kopikap**

## **Sabda Adjie Saputra**

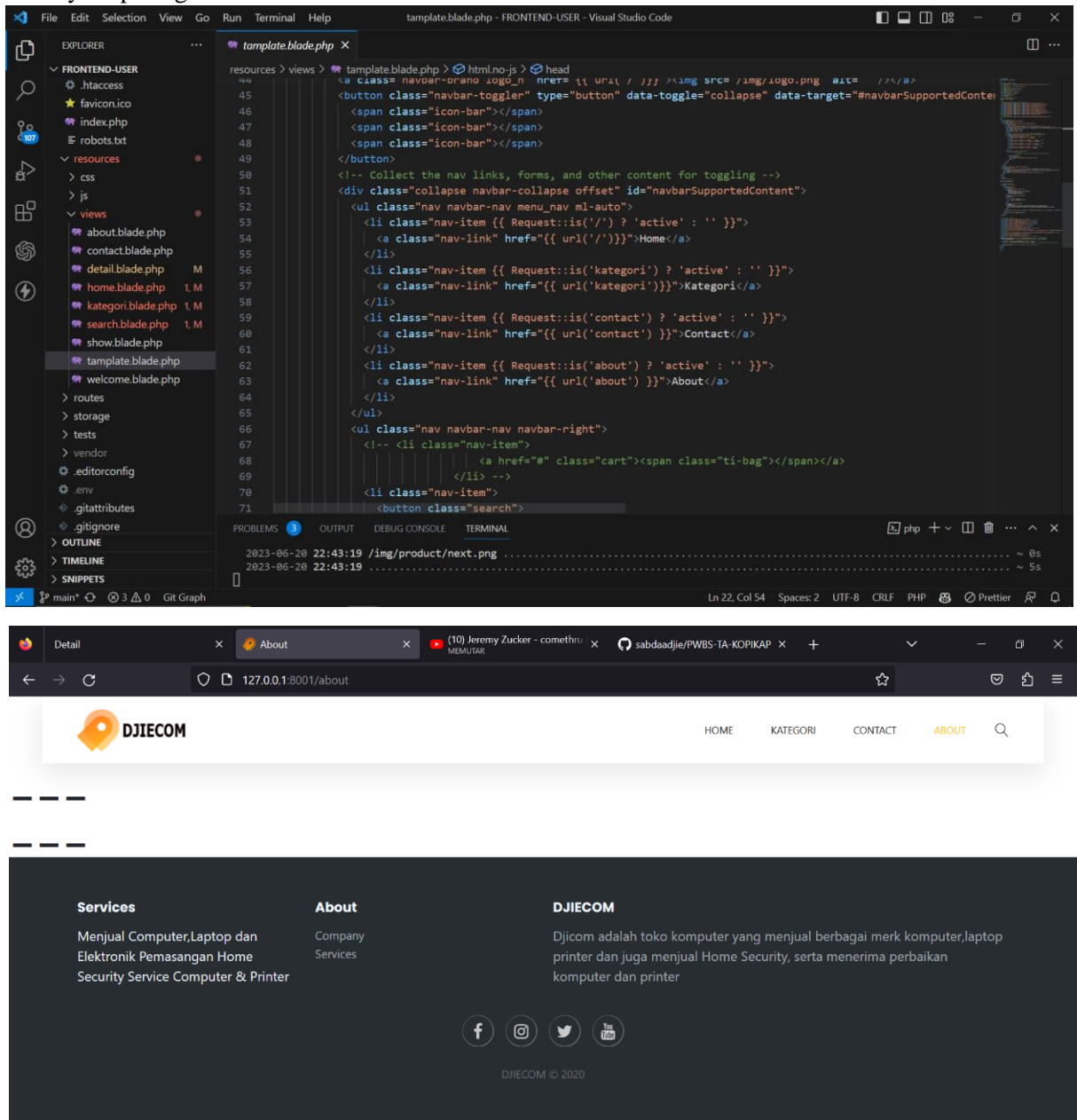
### **Penjelasan singkat tentang aplikasi.**

Website katalog toko komputer menjadi sarana yang penting dalam era digital saat ini untuk memudahkan pengguna dalam mencari dan membeli produk-produk komputer. Tujuan dari penelitian ini adalah untuk mengembangkan sebuah website katalog toko komputer dengan nama DJIECOM yang menyediakan informasi terkini mengenai produk-produk komputer yang ditawarkan oleh toko tersebut.

Hasil dari pengembangan website ini diharapkan dapat meningkatkan aksesibilitas dan kenyamanan pengguna dalam mencari dan membeli produk komputer. Dengan adanya website katalog toko komputer DJIECOM, diharapkan dapat membantu pengguna dalam mengambil keputusan pembelian yang lebih baik dan efisien.

Saya sebagai frontend user bertugas untuk membuat tampilan untuk user.

1. Hal pertama yang saya lakukan adalah membuat **template.blade.php** untuk Header dan Footer agar bisa digunakan sebagai template untuk page lainnya. Untuk Codingan dan hasilnya seperti gambar di bawah



2. Lalu saya menyetting routes->web.php agar ketika di klik pada menu navigasi header dapat berpindah halaman.

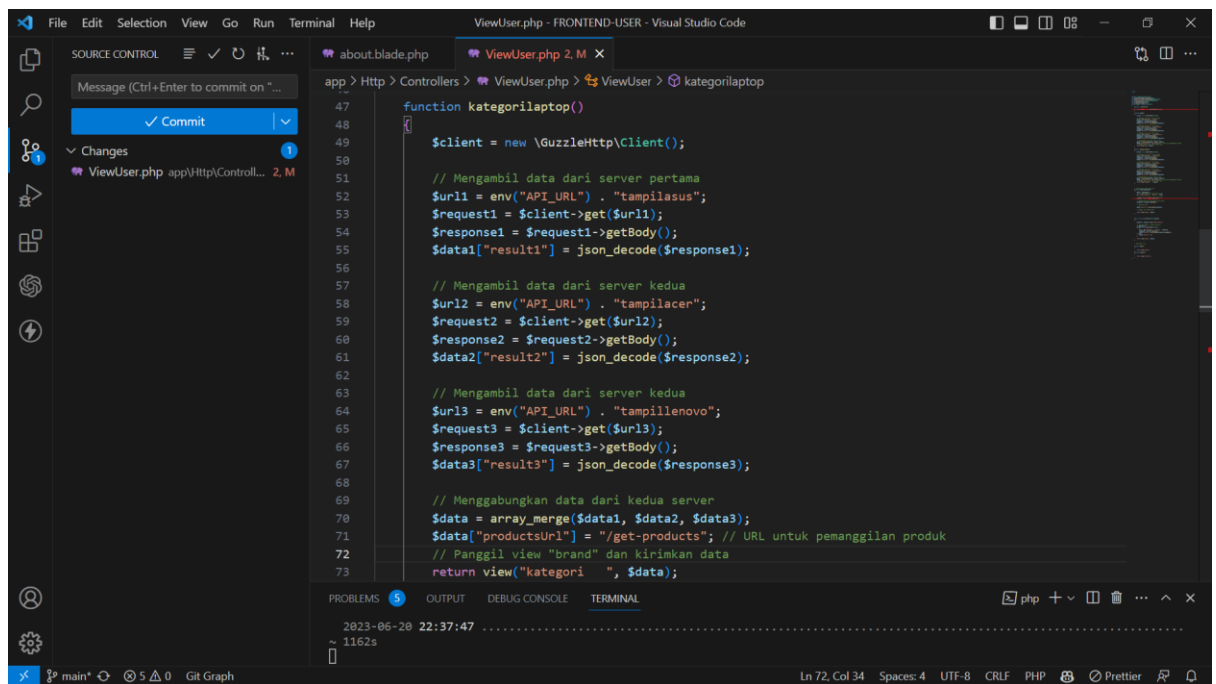
```
routes > web.php
17
18 Route::get('/', [ViewUser::class, 'brand']);
19
20 // Route untuk kategori
21 Route::get('kategori', [ViewUser::class, 'kategoriLaptop']);
22
23 // Route untuk brand
24 Route::get('brand', [ViewUser::class, 'brand']);
25
26 // Route untuk about
27 Route::get('about', [ViewUser::class, 'about']);
28
29 // Route untuk contact
30 Route::get('contact', [ViewUser::class, 'contact']);
31
32 //route untuk detail data user
33
34 Route::get('detail/{parameter}', [ViewUser::class, 'detail'])->name('detail/{parameter}');
35
36 Route::get('search', [ViewUser::class, 'search'])->name('search');
```

3. Kemudian saya melakukan pemanggilan api di Http->Controller agar page yang akan di buat dapat melakukan pemanggilan data dari api yang telah di buat oleh backend. Disini saya beri nama **ViewUser.php**.

Pada gambar ini saya memanggil 3 data dari api yang akan di return ke home

```
app > Http > Controllers > ViewUser.php > ViewUser > brand
18 function brand()
19
20 $client = new \GuzzleHttp\Client();
21
22 // Mengambil data dari server pertama
23 $url1 = env('API_URL') . 'tampil';
24 $request1 = $client->get($url1);
25 $response1 = $request1->getBody();
26 $data1['result1'] = json_decode($response1);
27
28 // Mengambil data dari server kedua
29 $url2 = env('API_URL') . 'tampilLaptop';
30 $request2 = $client->get($url2);
31 $response2 = $request2->getBody();
32 $data2['result2'] = json_decode($response2);
33
34 // Mengambil data dari server kedua
35 $url3 = env('API_URL') . 'tampilmonitor';
36 $request3 = $client->get($url3);
37 $response3 = $request3->getBody();
38 $data3['result3'] = json_decode($response3);
39
40 // Menggabungkan data dari kedua server
41 $data = array_merge($data1, $data2, $data3);
42 $data['productsUrl'] = '/get-products'; // URL untuk pemanggilan produk
43 // Panggil view 'brand' dan kirimkan data
44 return view('home', $data);
```

Pada gambar ini saya memanggil 3 data lagi yang akan di jadikan untuk menampilkan data sesuai kategori merk, pada codingan ini akan mereturn ke tampilan Kategori.blade.php



```
function kategoriLaptop()
{
    $client = new \GuzzleHttp\Client();

    // Mengambil data dari server pertama
    $url1 = env("API_URL") . "tampilasus";
    $request1 = $client->get($url1);
    $response1 = $request1->getBody();
    $data1["result1"] = json_decode($response1);

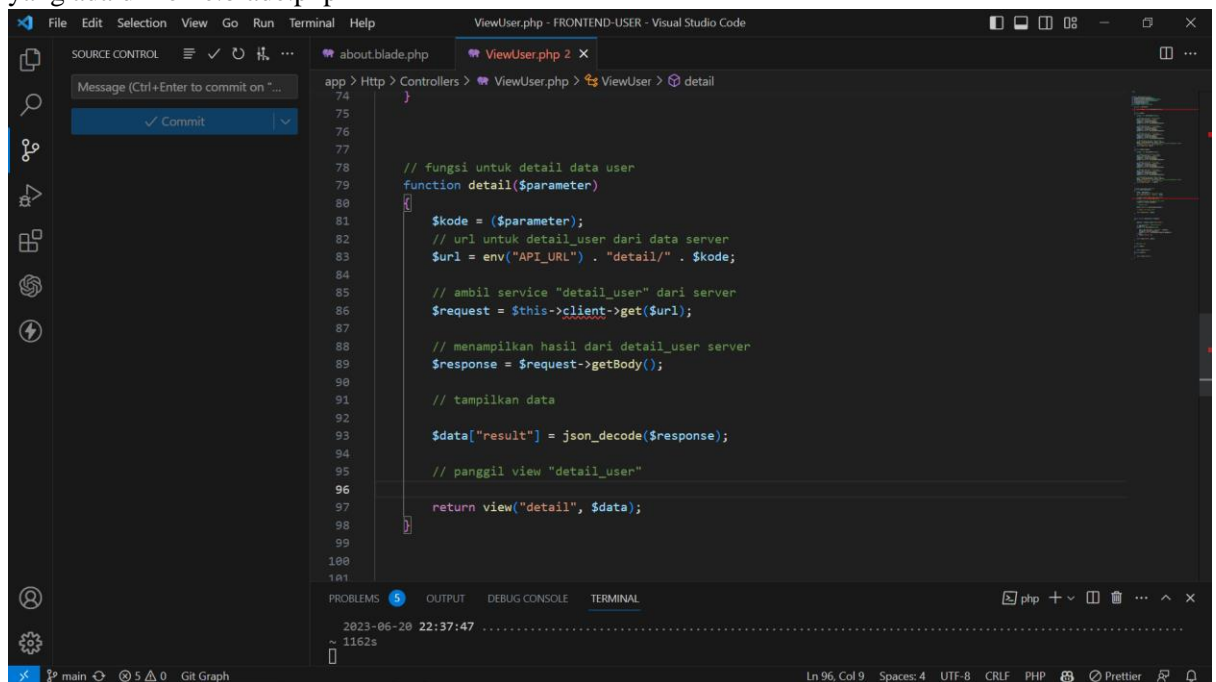
    // Mengambil data dari server kedua
    $url2 = env("API_URL") . "tampilacer";
    $request2 = $client->get($url2);
    $response2 = $request2->getBody();
    $data2["result2"] = json_decode($response2);

    // Mengambil data dari server kedua
    $url3 = env("API_URL") . "tampillenovo";
    $request3 = $client->get($url3);
    $response3 = $request3->getBody();
    $data3["result3"] = json_decode($response3);

    // Menggabungkan data dari kedua server
    $data = array_merge($data1, $data2, $data3);
    $data["productsUrl"] = "/get-products"; // URL untuk pemanggilan produk

    // Panggil view "brand" dan kirimkan data
    return view("kategori", $data);
}
```

Pada gambar di bawah ini saya melakukan pemanggilan detail data yang akan di return ke tampilan detail.blade.php, pada tampilan ini akan menampilkan detail data dari tampilan data yang ada di home.blade.php



```
function detail($parameter)
{
    $kode = ($parameter);
    // url untuk detail_user dari data server
    $url = env("API_URL") . "detail/" . $kode;

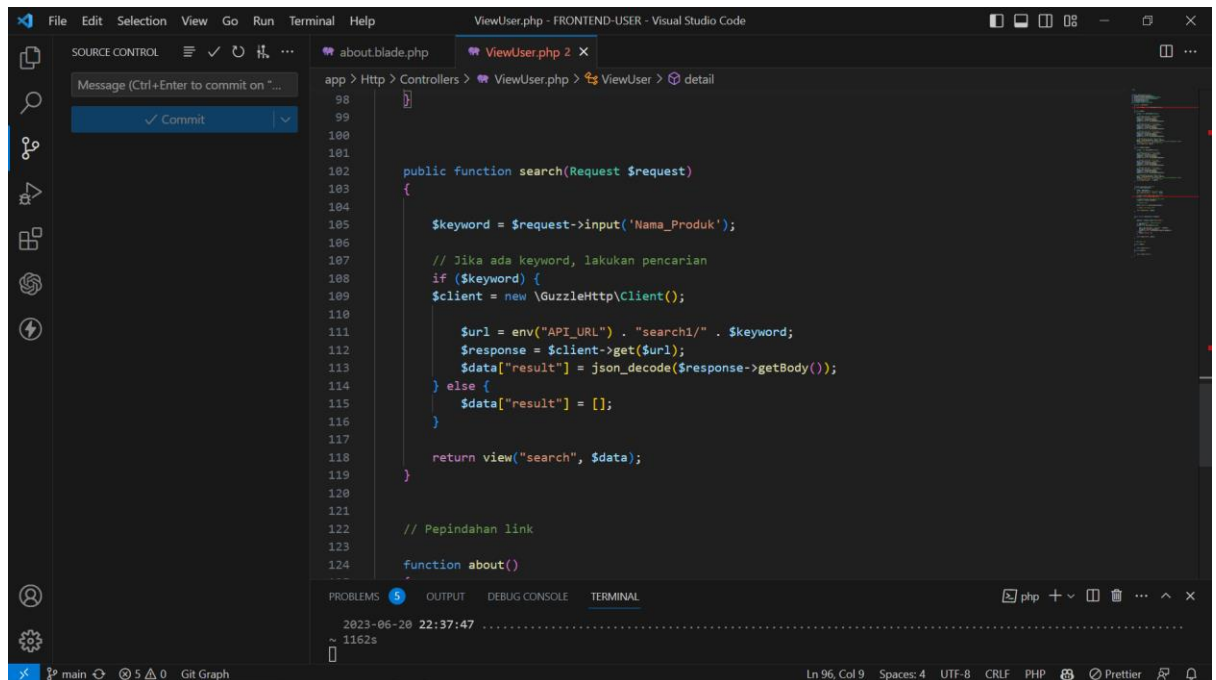
    // ambil service "detail_user" dari server
    $request = $this->client->get($url);

    // menampilkan hasil dari detail_user server
    $response = $request->getBody();

    // tampilkan data
    $data["result"] = json_decode($response);

    // panggil view "detail_user"
    return view("detail", $data);
}
```

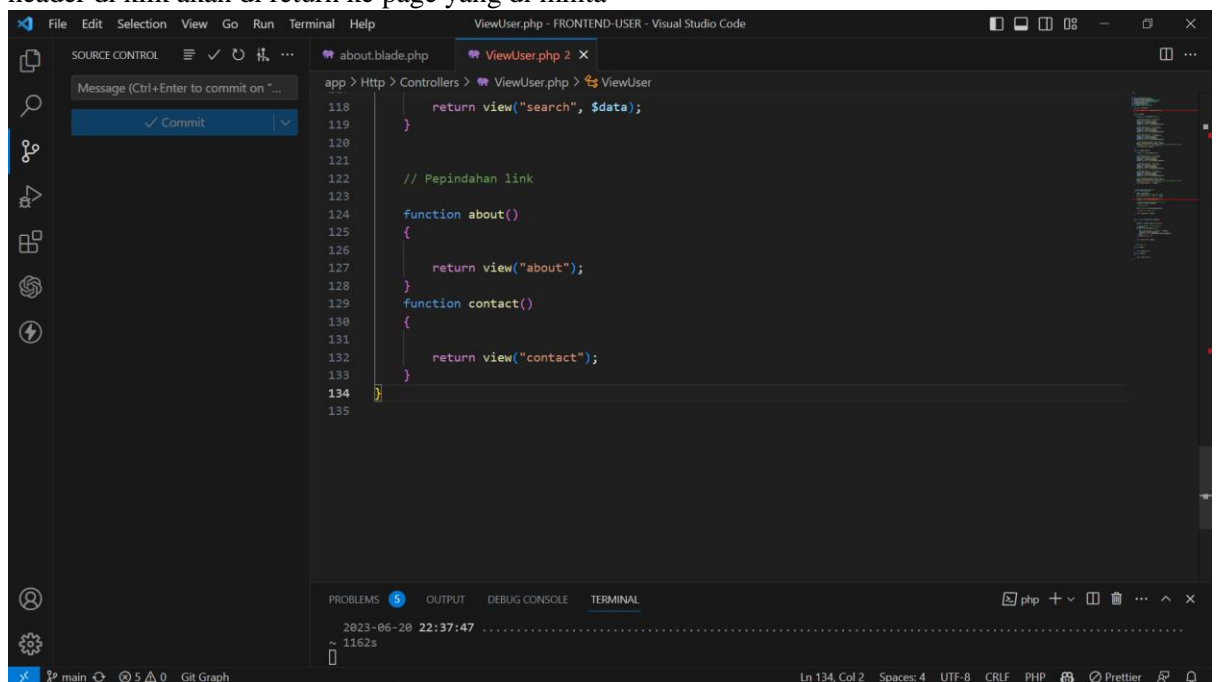
Pada gambar di bawah ini berfungsi untuk mencari data yang ada di database lalu di return ke tampilan search.blade.php



The screenshot shows the Visual Studio Code editor with the file `ViewUser.php` open. The code is in the `Controllers > ViewUser.php` directory. The `search` function is defined, which takes a `Request $request` as input. It extracts the `Nama_Produk` from the request input. If a keyword is provided, it uses the `GuzzleHttpClient` to send a GET request to the `API_URL` with the keyword. The response is decoded as JSON and stored in `$data["result"]`. If no keyword is provided, `$data["result"]` is set to an empty array. The function then returns the view `search` with the `$data` array. Below the function, there is a comment `// Pepindahan link` and the start of the `about` function.

```
98
99
100
101
102 public function search(Request $request)
103 {
104
105     $keyword = $request->input('Nama_Produk');
106
107     // Jika ada keyword, lakukan pencarian
108     if ($keyword) {
109         $client = new \GuzzleHttpClient();
110
111         $url = env("API_URL") . "search/" . $keyword;
112         $response = $client->get($url);
113         $data["result"] = json_decode($response->getBody());
114     } else {
115         $data["result"] = [];
116     }
117
118     return view("search", $data);
119 }
120
121
122 // Pepindahan link
123
124 function about()
125 ...
```

Pada gambar di bawah ini berfungsi hanya sebagai fungsi ketika tulisan contact dan abiut di header di klik akan di return ke page yang di minta



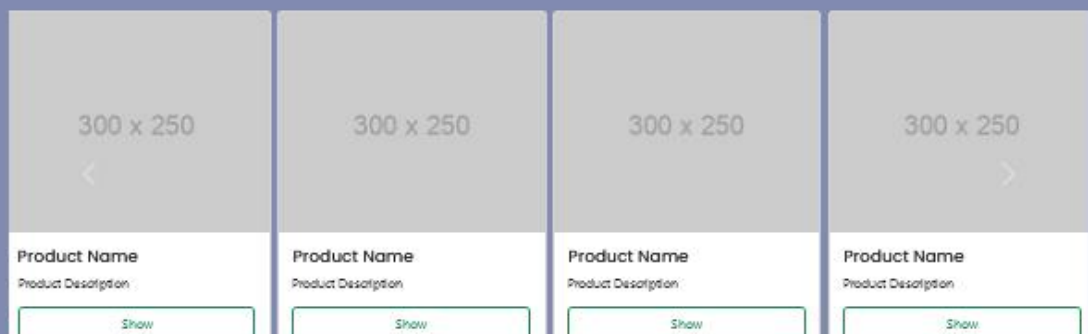
The screenshot shows the Visual Studio Code editor with the file `ViewUser.php` open. The code is in the `Controllers > ViewUser.php` directory. The `about` function is defined, which returns the view `about`. Below it, the `contact` function is defined, which returns the view `contact`.

```
118     return view("search", $data);
119 }
120
121
122 // Pepindahan link
123
124 function about()
125 {
126
127     return view("about");
128 }
129 function contact()
130 {
131
132     return view("contact");
133 }
134
135
```

4. Selanjutnya saya membuat tampilan [Home.blade.php](#) yang sudah terintegrasi dengan pemanggilan data dari api backend untuk tampilannya seperti gambar di bawah.



## Produk Terbaru

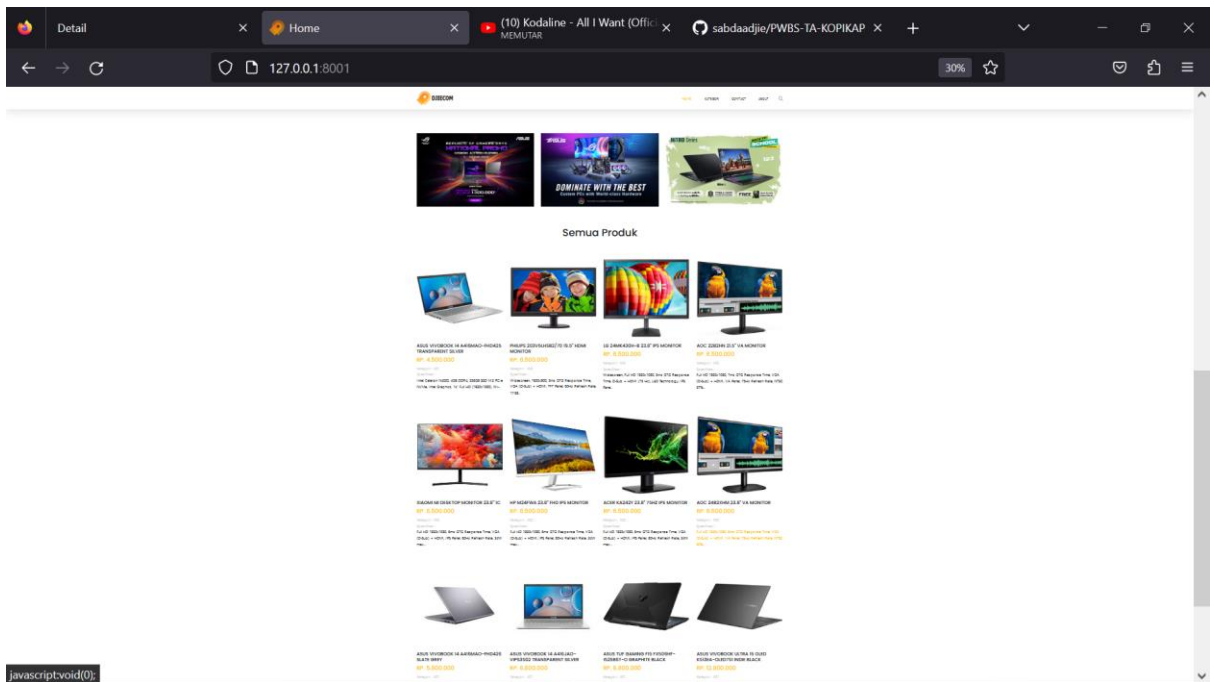


## MONITOR

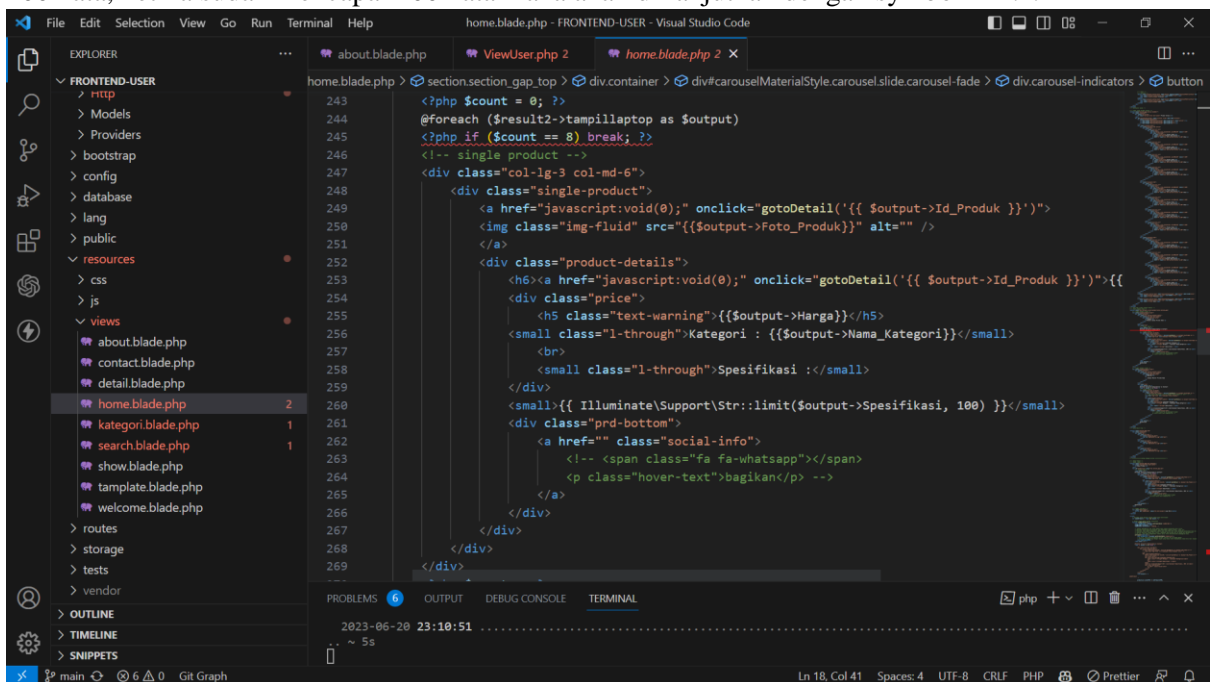
Temukan Monitor Pilihan Kamu



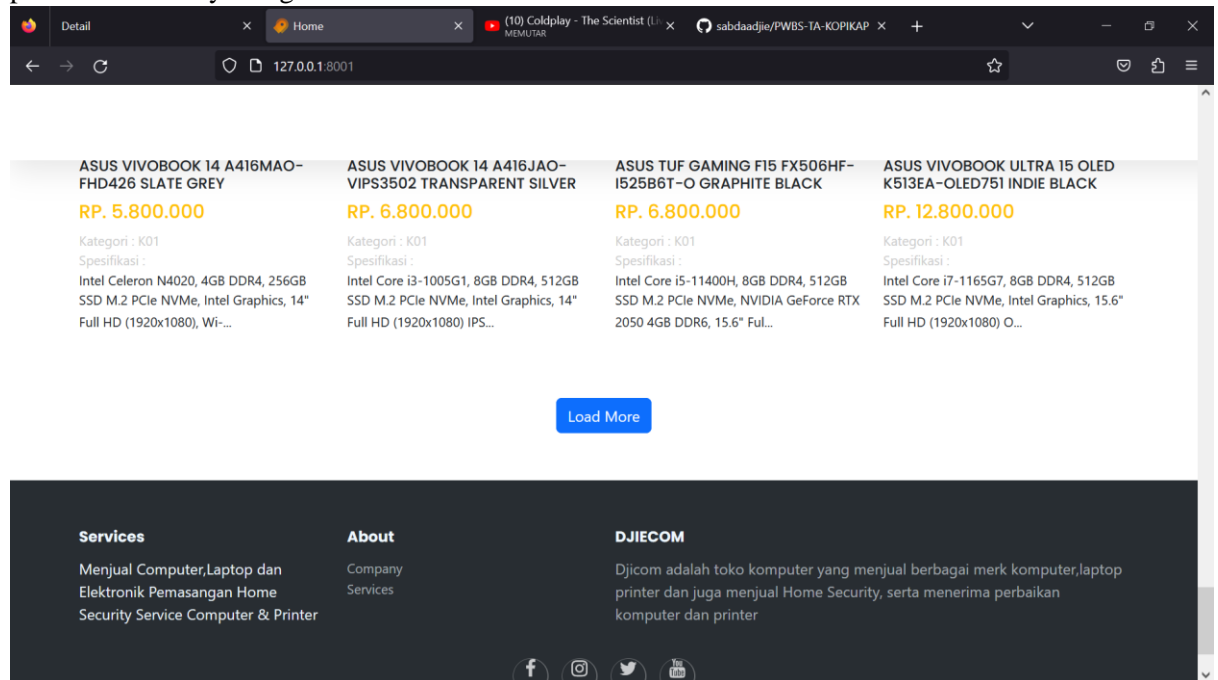




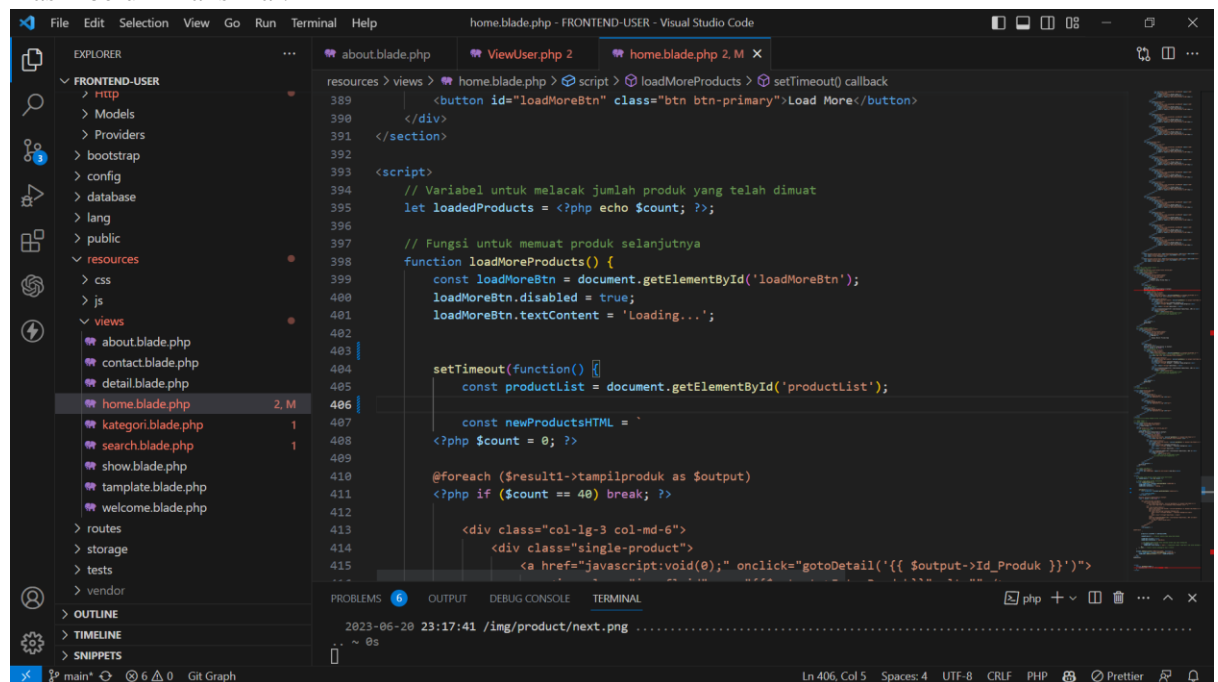
Pada gambar di bawah ini saya melakukan pemanggilan api tampillaptop yang di looping Ketika data sudah mencapai 8 maka data tidak di tampilkan lagi. Dan pada pemanggilan data spesifikasi saya melakukan limit pada kata yang ada di database spesifikasi yaitu sebanyak 100 kata, ketika sudah mencapai 100 kata maka akan di lanjutkan dengan symbol “....”.



Pada gambar di bawah ini saya membuat button “Load More” berfungsi untuk menampilkan produk lebih banyak lagi.

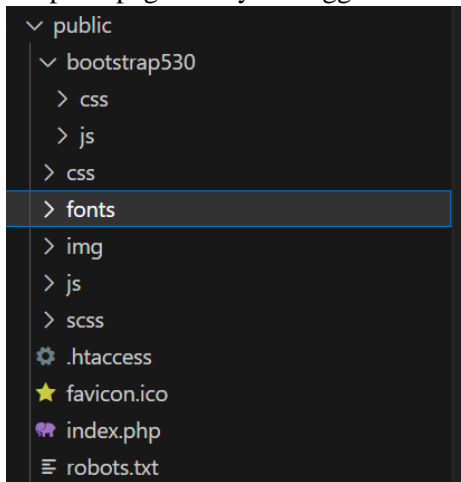


Dan pada gambar di bawah ini adalah code untuk fungsi button “Load More” akan tetapi masih belum maksimal.



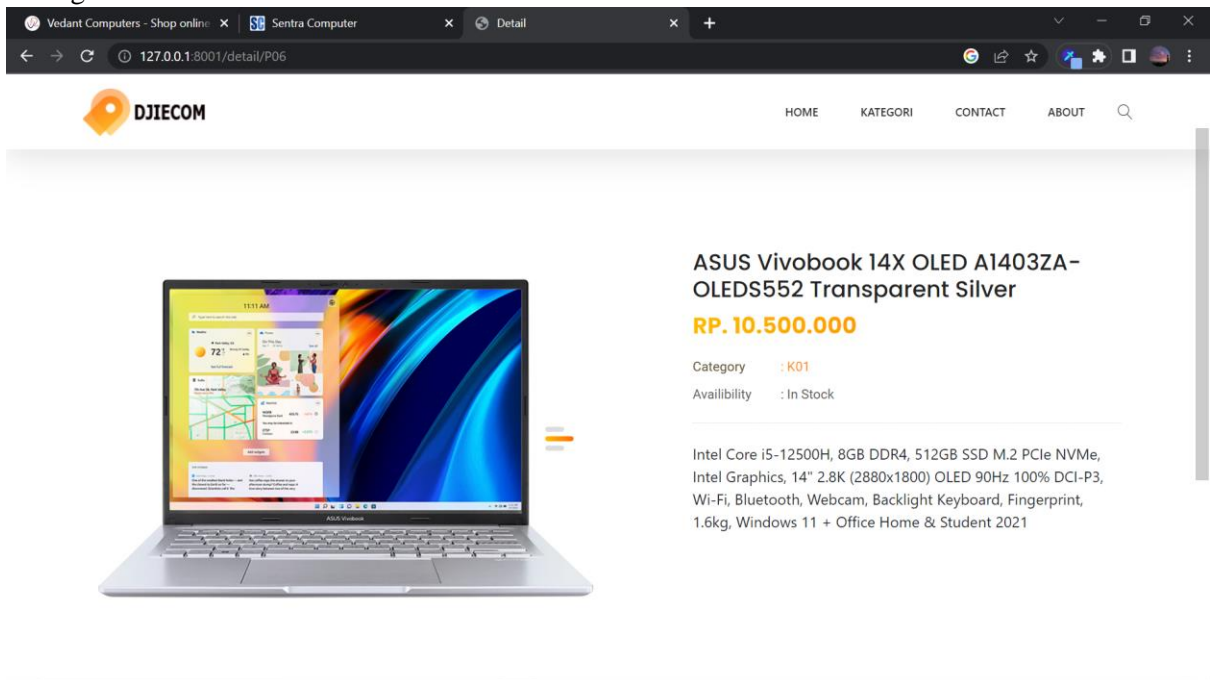


Pada gambar di bawah ini saya melakukan custom pada tampilan home dan juga untuk tampilan page lainnya menggunakan bootstrap dan css



##### 5. Selanjutnya saya membuat Detail.blade.php

Pada gambar di bawah ini adalah hasil detail dari data



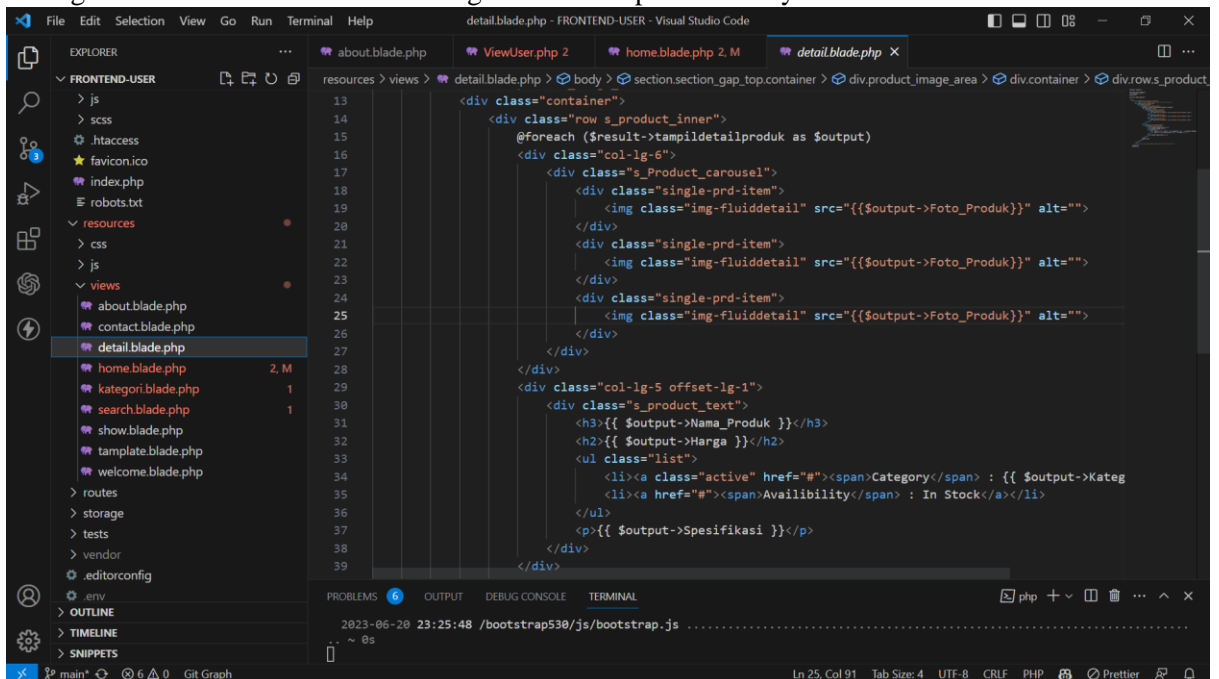
Paada codingan di bawah ini adalah codingan dari [home.blade.php](#) agar ketika produk di klik akan melakukan redirect ke tampilan detail.blade.php

```

<div class="col-lg-3 col-md-6">
  <div class="single-product">
    <a href="javascript:void(0);" onclick="gotoDetail('{{ $output->Id_Produk }}')">
      
    </a>
    
    <div class="product-details">
      <h6><a href="javascript:void(0);" onclick="gotoDetail('{{ $output->Id_Produk }}')">{{ $output->Nama_Produk }}</a></h6>
      <div class="price">
        <h5 class="text-warning">{{ $output->Harga }}</h5>
        <small class="1-through">Kategori : {{ $output->Nama_Kategori }}</small>
        <br>
        <small class="1-through">Spesifikasi :</small>
      </div>
    </div>
  </div>
</div>

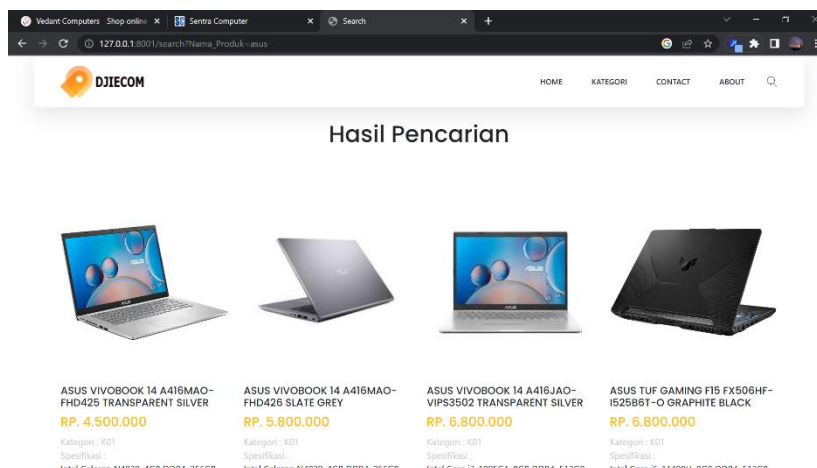
```

Pada gambar di bawah ini adalah codingan untuk tampilan detailnya

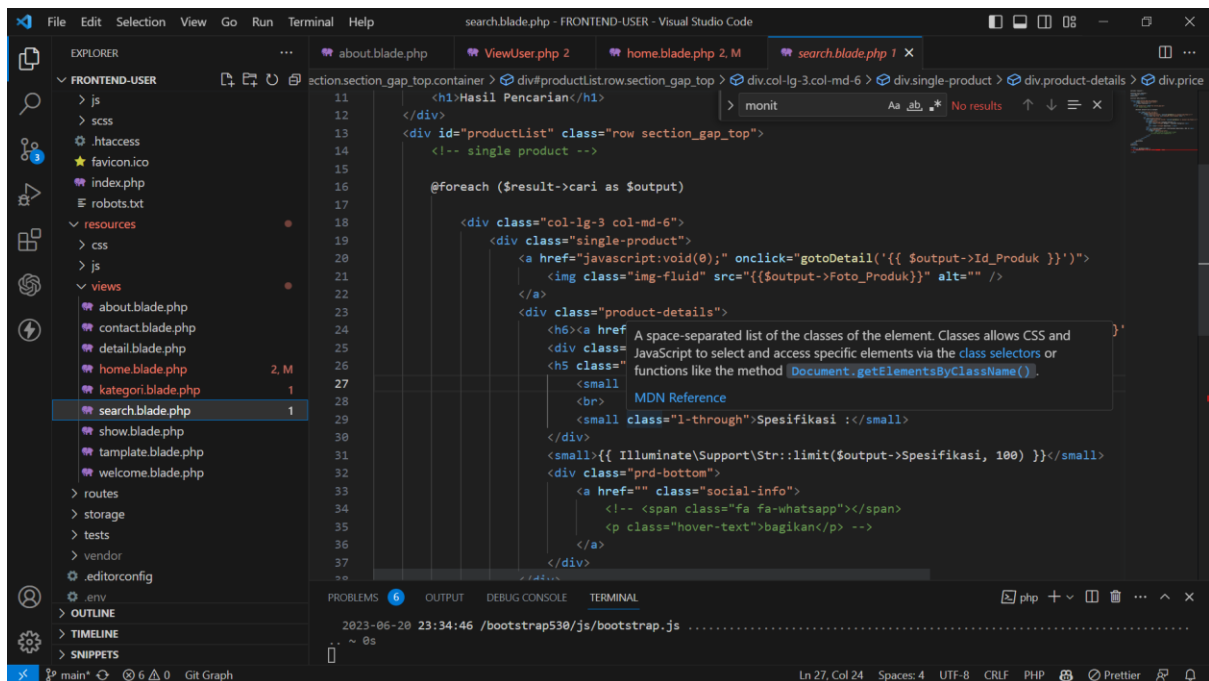


6. Selanjutnya saya membuat tampilan search product

Pada gambar di bawah ini adalah hasil dari pencarian product asus

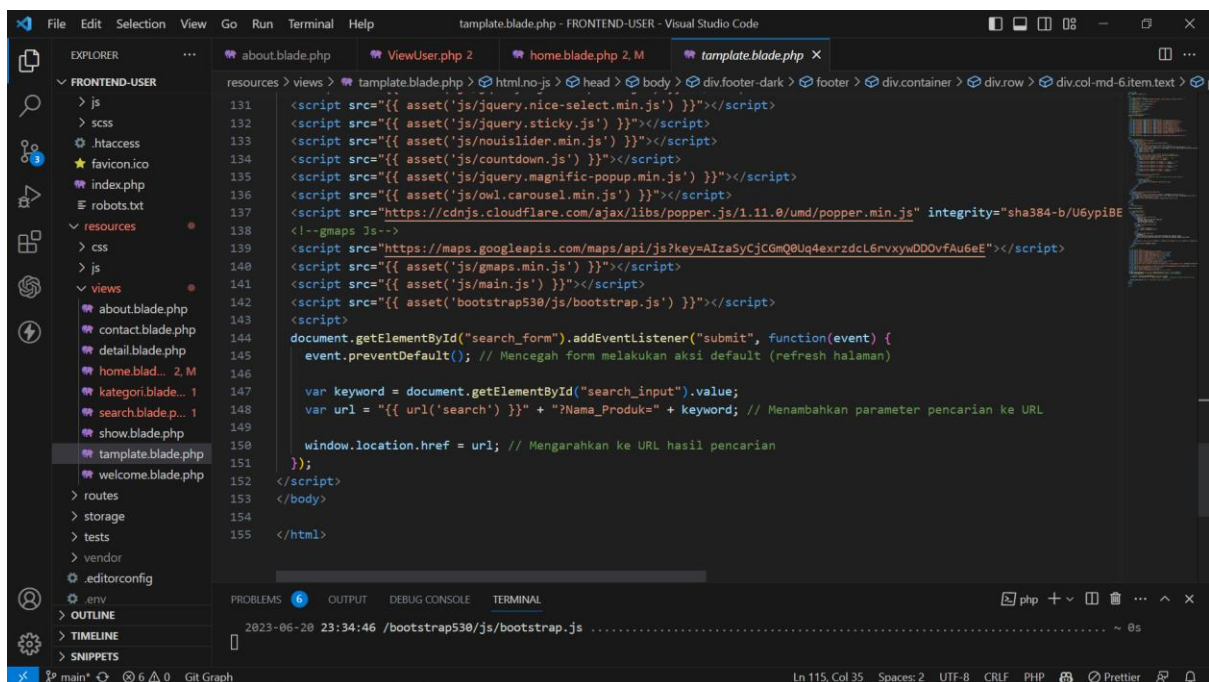


Gambar di bawah ini adalah codingan untuk tampilan search.blade.php



```
11 <h1>Hasil Pencarian</h1>
12 </div>
13 <div id="productList" class="row section_gap_top">
14 <!-- single product -->
15
16 @foreach ($result->cari as $output)
17
18 <div class="col-lg-3 col-md-6">
19 <div class="single-product">
20 <a href="javascript:void(0);" onclick="gotoDetail('{{ $output->Id_Produk }}')">
21 
22 </a>
23 <div class="product-details">
24 <h6><a href=
25 <div class=
26 <h5 class=
27 <small
28 <br>
29 MDN Reference
30 <small class="1-through">Spesifikasi </small>
31 </div>
32 <small>{{ Illuminate\Support\Str::limit($output->Spesifikasi, 100) }}</small>
33 <div class="prd-bottom">
34 <a href="" class="social-info">
35 <!-- <span class="fa fa-whatsapp"></span>
36 <p class="hover-text">bagikan</p> -->
37 </a>
38 </div>
39 </div>
```

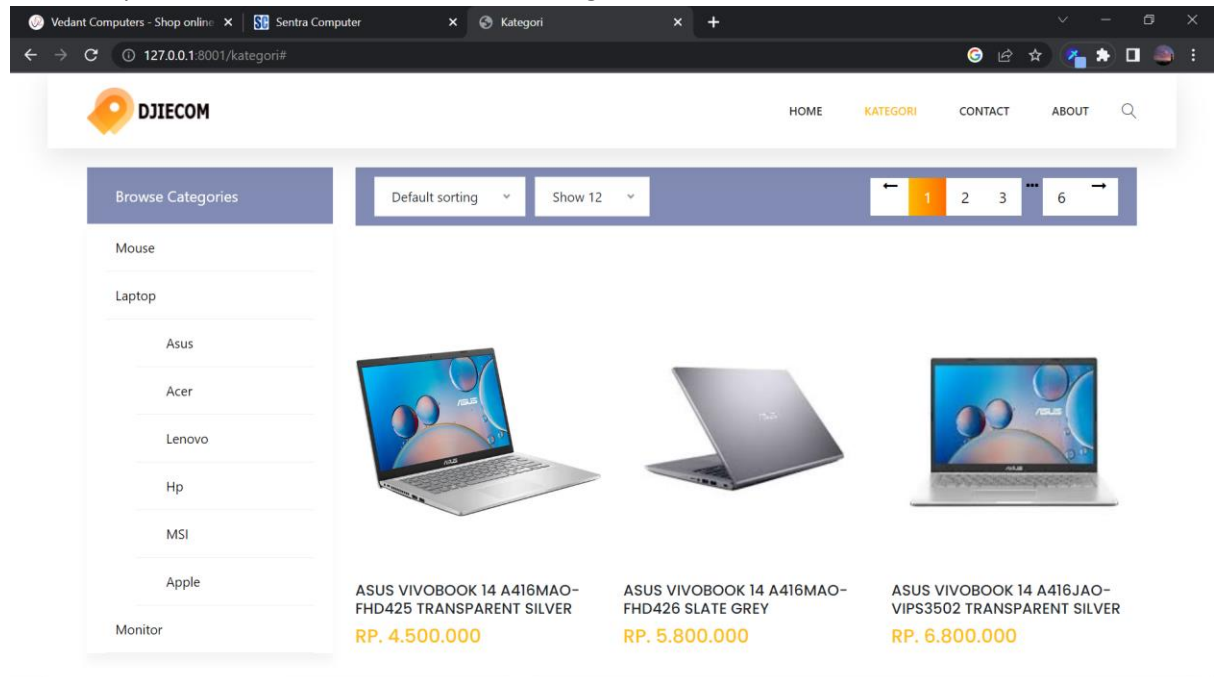
Pada tampilan dibawah ini adalah fungsi untuk pencarian



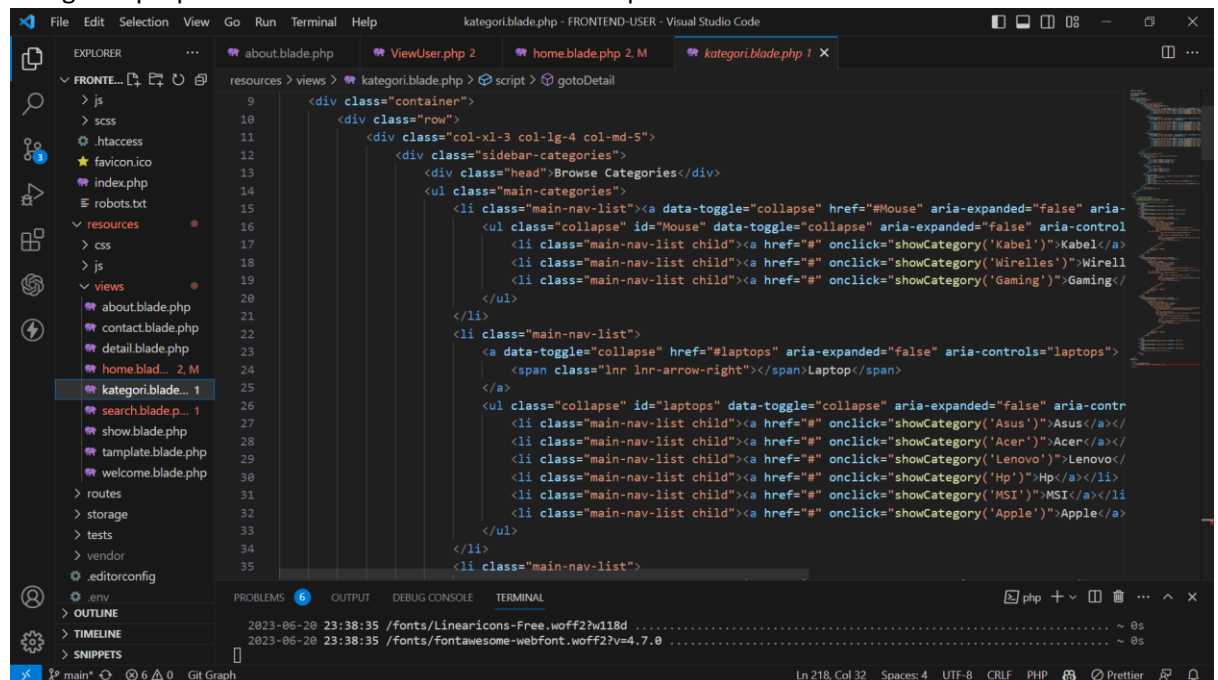
```
131 <script src="{{ asset('js/jquery.nice-select.min.js') }}"></script>
132 <script src="{{ asset('js/jquery.sticky.js') }}"></script>
133 <script src="{{ asset('js/nouislider.min.js') }}"></script>
134 <script src="{{ asset('js/countdown.js') }}"></script>
135 <script src="{{ asset('js/jquery.magnific-popup.min.js') }}"></script>
136 <script src="{{ asset('js/owl.carousel.min.js') }}"></script>
137 <script src="https://cdn.jsdelivr.net/npm/popper.js/1.11.0/umd/popper.min.js" integrity="sha384-b/U6ypiBE
138 <!-- gmaps js -->
139 <script src="https://maps.googleapis.com/maps/api/js?key=AIzaSyCjCGmQ0U4exrxzdcL6rvxywDD0vFAu6eE"></script>
140 <script src="{{ asset('js/gmaps.min.js') }}"></script>
141 <script src="{{ asset('js/main.js') }}"></script>
142 <script src="{{ asset('bootstrap530/js/bootstrap.js') }}"></script>
143 <script>
144 document.getElementById("search_form").addEventListener("submit", function(event) {
145 event.preventDefault(); // Mencegah form melakukan aksi default (refresh halaman)
146
147 var keyword = document.getElementById("search_input").value;
148 var url = "{{ url('search') }}" + "?Nama_Produk=" + keyword; // Menambahkan parameter pencarian ke URL
149
150 window.location.href = url; // Mengarahkan ke URL hasil pencarian
151 });
152 </script>
153 </body>
154 </html>
```

## 7. Selanjutnya saya membuat tampilan Kategori.blade.php

Pada tampilan dibawah ini adalah hasil dari kategori Asus



Pada gambar dibawah ini adalah codingan untuk meampilkan produk per kategori, ketika kategori laptop-> asus di clik maka data akan di tampilkan



Visual Studio Code interface showing the file explorer on the left with the project structure. The Explorer pane shows the following structure:

- FRONTEND
  - resources
    - views
      - kategori.blade.php (selected)

The main editor displays the content of `kategori.blade.php` with the following PHP code:

```
<script>
function showCategory(category) {
    // Menghapus konten kategori sebelumnya
    document.getElementById("categoryContent").innerHTML = "";

    // Menampilkan konten kategori sesuai yang dipilih
    switch (category) {
        case "Kabel":
            document.getElementById("categoryContent").innerHTML = ``;
            break;
        case "Wireless":
            document.getElementById("categoryContent").innerHTML = "Konten Wireless";
            break;
        case "Gaming":
            document.getElementById("categoryContent").innerHTML = "Konten Gaming";
            break;
        case "Asus":
            document.getElementById("categoryContent").innerHTML = `
                <section class="latest-product-area pb-40 category-list">
                    <div class="row">
                        @php $count = 0; @endphp
                        @foreach ($result1->tampilas as $output)
                            @php if ($count == 40) break; @endphp
                            <div class="col-lg-4 col-md-6">
                                <div class="single-product">
                                    
                                </div>
                            </div>
                        @endforeach
                    </div>
                </section>
            `;
            break;
    }
}
```

The bottom panel shows the TERMINAL with the following output:

```
2023-06-20 23:38:35 /fonts/Linearicons-Free.woff2?v118d ..... ~ 0s
2023-06-20 23:38:35 /fonts/fontawesome-webfont.woff2?v4.7.0 ..... ~ 0s
```

Visual Studio Code interface showing the file explorer on the left with the project structure. The Explorer pane shows the following structure:

- FRONTEND
  - resources
    - views
      - kategori.blade.php (selected)

The main editor displays the content of `kategori.blade.php` with the following PHP code:

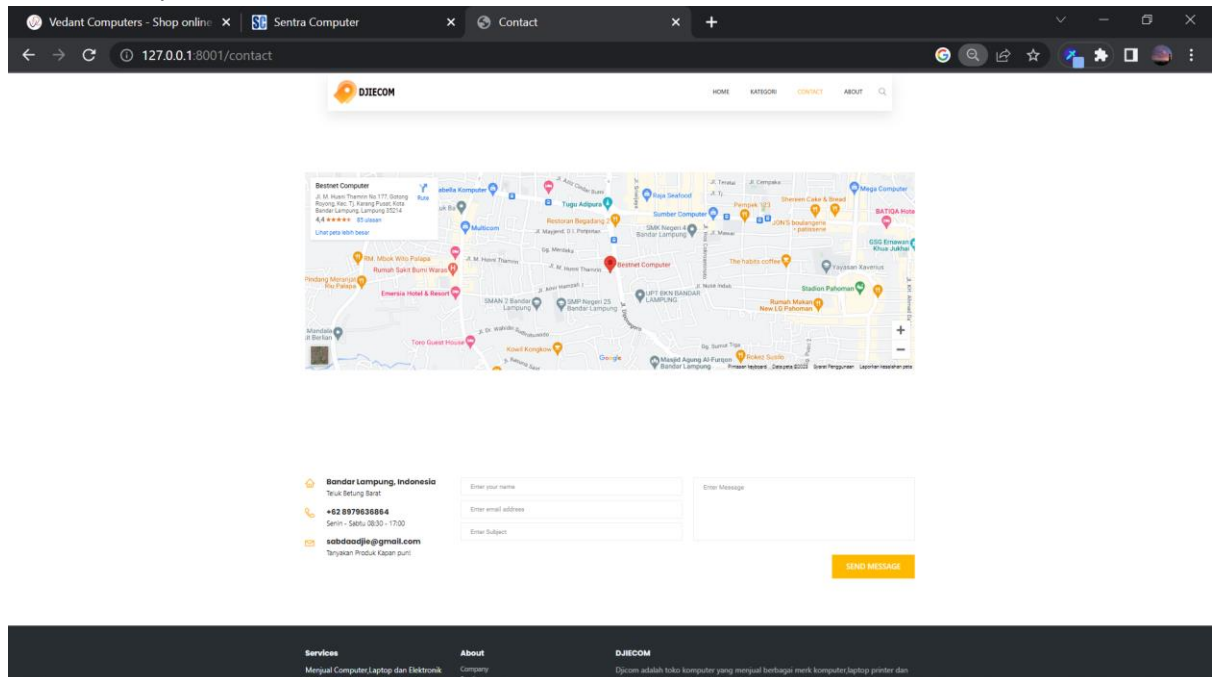
```
break;
case "Asus":
    document.getElementById("categoryContent").innerHTML = `
        <section class="latest-product-area pb-40 category-list">
            <div class="row">
                @php $count = 0; @endphp
                @foreach ($result1->tampilas as $output)
                    @php if ($count == 40) break; @endphp
                    <div class="col-lg-4 col-md-6">
                        <div class="single-product">
                            
                            <div class="product-details">
                                <h6><a href="javascript:void(0);" onclick="gotoDetail('{{ $output->Id_Produk
                                <div class="price">
                                    <h5 class="text-warning">{{ $output->Harga }}</h5>
                                    <br>
                                    <small class="1-through">Spesifikasi :</small>
                                </div>
                                <small>{{ Illuminate\Support\Str::limit($output->Spesifikasi, 100) }}</small>
                                <div class="prd-bottom">
                                    <a href="" class="social-info">
                                        </a>
                                    </div>
                                </div>
                            </div>
                        </div>
                    @endforeach
                </div>
            </section>
        `;
    break;
```

The bottom panel shows the TERMINAL with the following output:

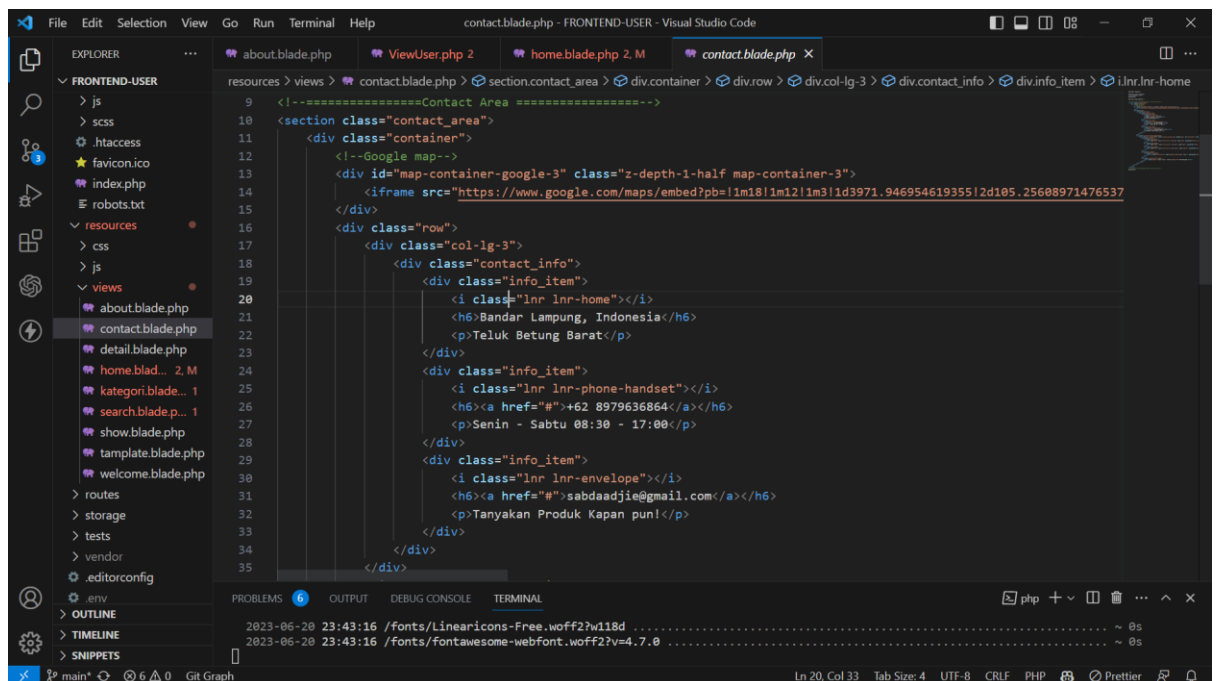
```
2023-06-20 23:40:51 /js/main.js ..... ~ 0s
2023-06-20 23:40:51 /bootstrap530/js/bootstrap.js ..... ~ 0s
```

## 8. Selanjutnya saya membuat Tampilan Contact

Untuk hasilnya,



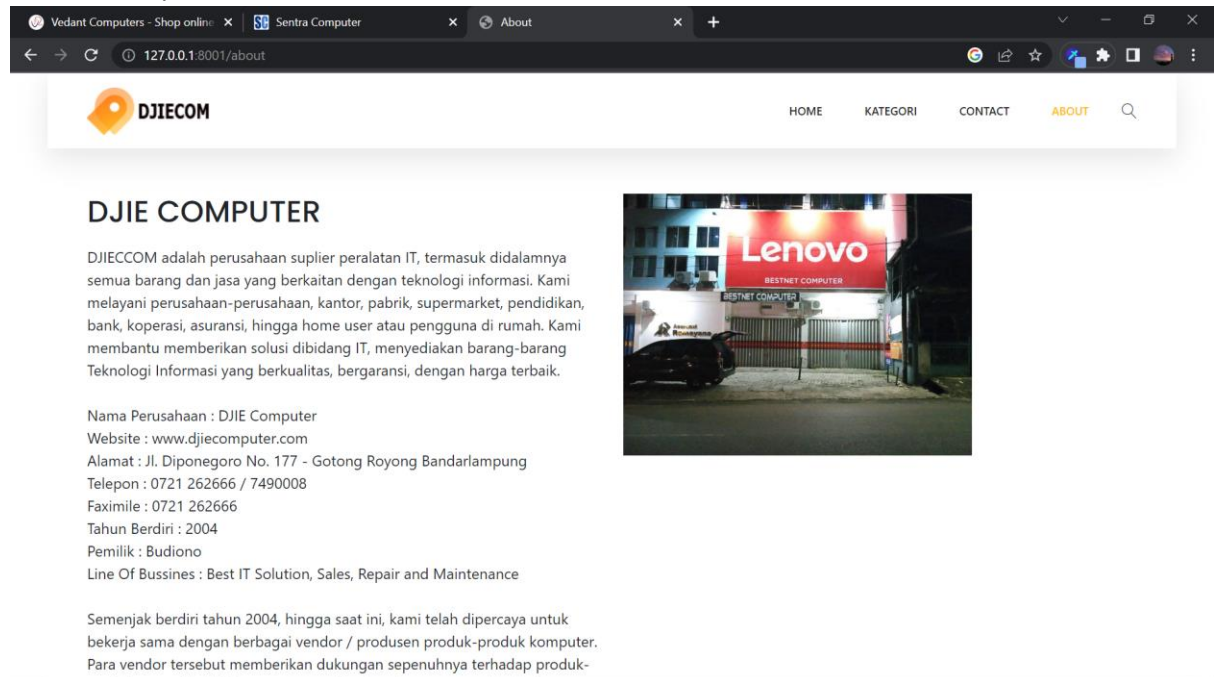
Pada gambar dibawah ini adalah codingan untuk contact.blade.php disini saya menambahkan peta agar pelanggan yang melihat bisa langsung mengetahui lokasi took





## 9. Selanjutnya saya membuat tampilan about.blade.php

Untuk hasilnya



Untuk codingannya

