

PROJECT : UMBUL BERKAH LAUNDRY

Disusun Oleh :

Fikri Yoma Rosyidan

Aristo Riza Muhammad

Deskripsi Project

Dalam Project ini, kami memiliki stakeholder berupa usaha laundry bernama “Umbul Berkah”. Di sini kami akan membuat website yang dapat membantu pemilik, karyawan laundry maupun customer dalam bertransaksi jasa laundry. Website ini nantinya akan dapat menyimpan data profil akun (admin, karyawan, customer), menambah karyawan, mencatat pemesanan laundry pada bagian karyawan, menambah customer pada bagian karyawan, dan memantau riwayat dan status laundry pada bagian customer.

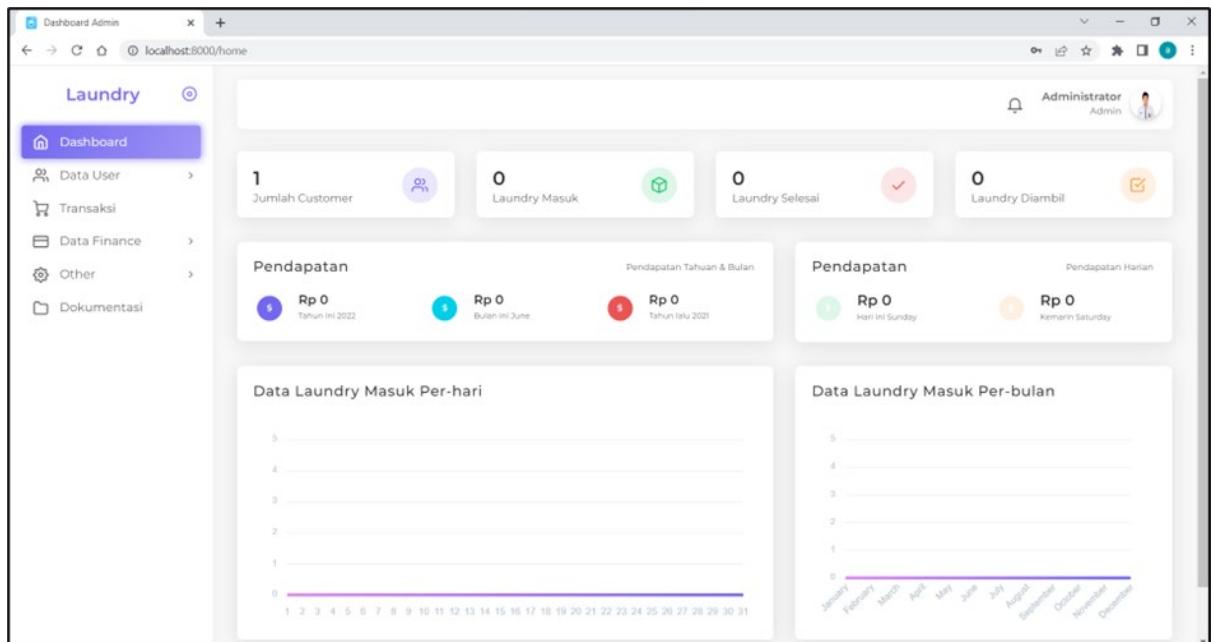
Alamat domain project

<https://umbulberkah.si20.site/>

Penjelasan Aplikasi

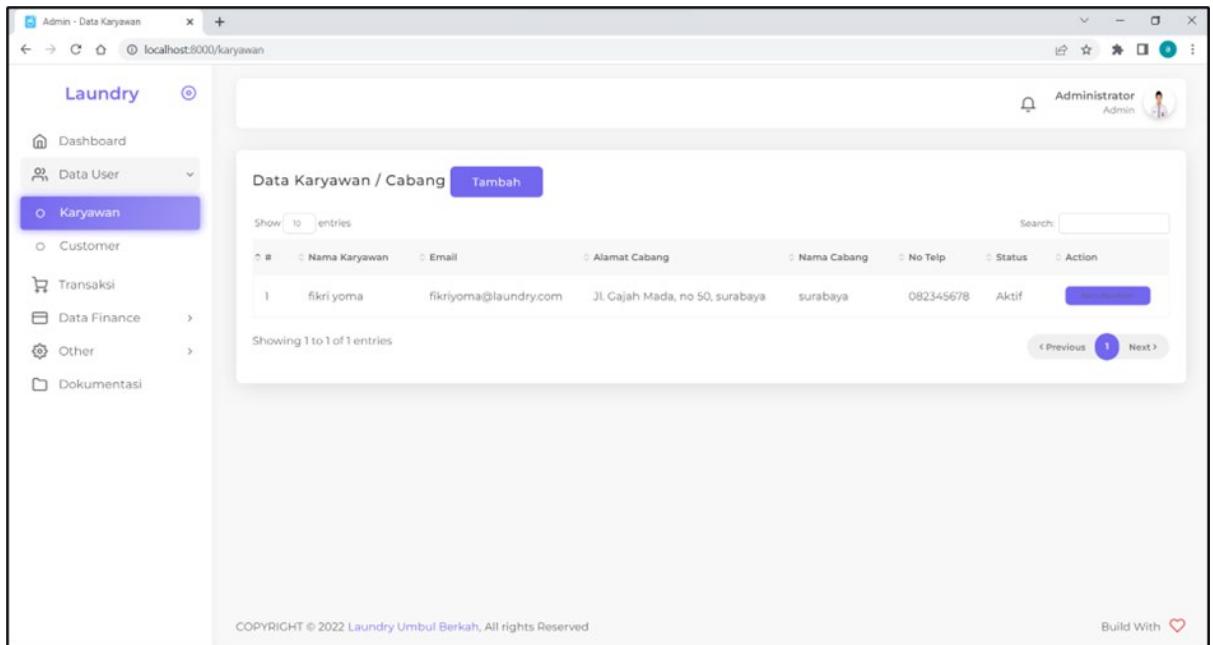
Pada website, terdapat tiga aktor yang berperan dalam menjalankannya yaitu admin, karyawan, dan customer.

- Admin



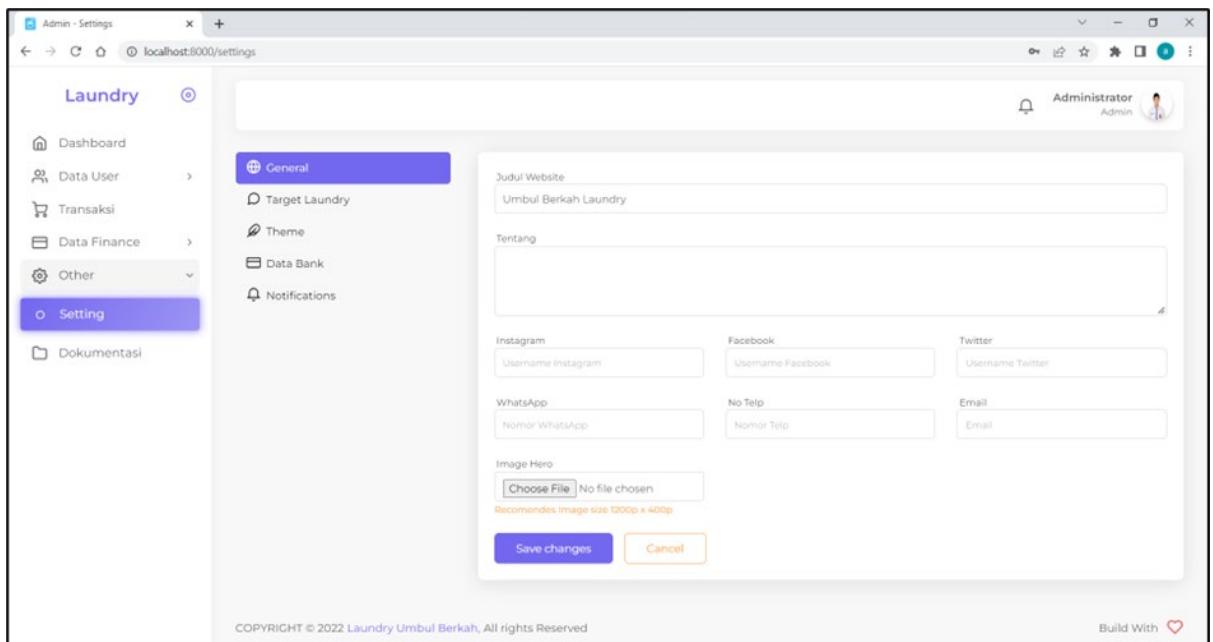
Gambar 1. Halaman Dashboard Admin

Peran admin lebih banyak bila dibandingkan dengan karyawan atau customer. Pada halaman dashboard terlihat admin dapat melihat jumlah customer, jumlah laundry masuk, selesai, hingga laundry yang telah diambil oleh customer. Admin juga dapat melihat pendapatan laundry mulai dari per hari, bulan, hingga tahun. Pendapatan tersebut kemudian diproyeksikan dalam bentuk nominal dan statistik.



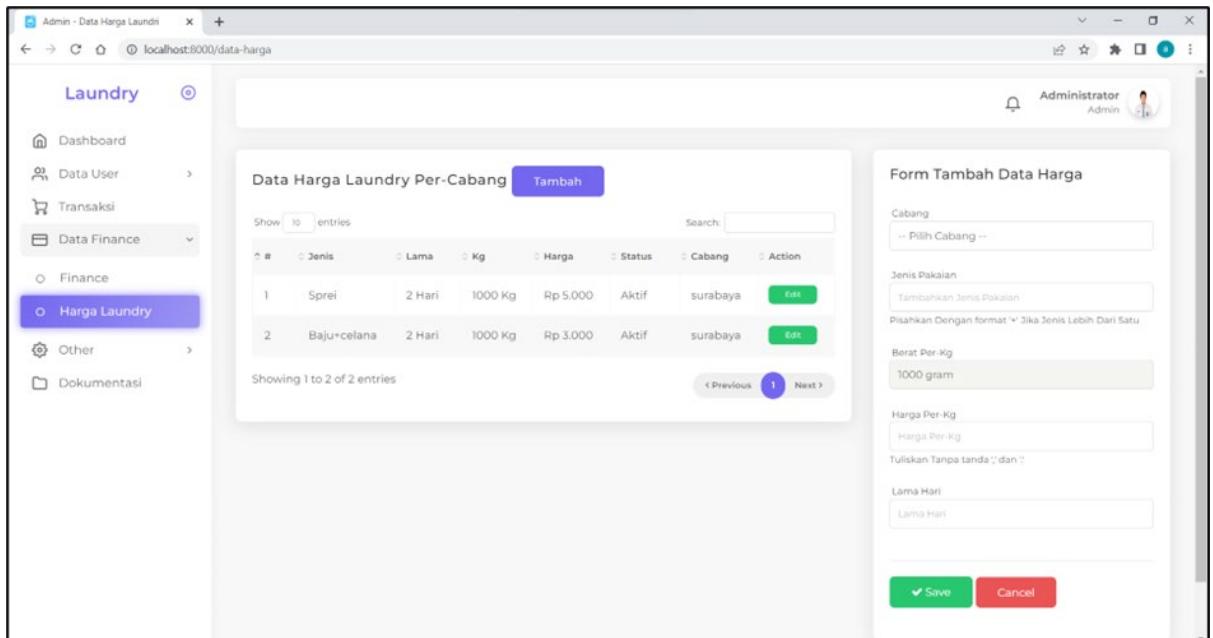
Gambar 2. Halaman Karyawan Pada Admin

Admin dapat menambahkan karyawan dan menonaktifkan karyawan tersebut, tetapi tidak dapat menghapus karyawan untuk arsip karyawan. Pada bagian customer, admin tidak dapat menambahkan maupun menghapus melainkan hanya melihat detail info customer. Yang dapat menambahkan atau menghapus customer adalah peran aktor karyawan nantinya.



Gambar 3. Settings Pada Admin

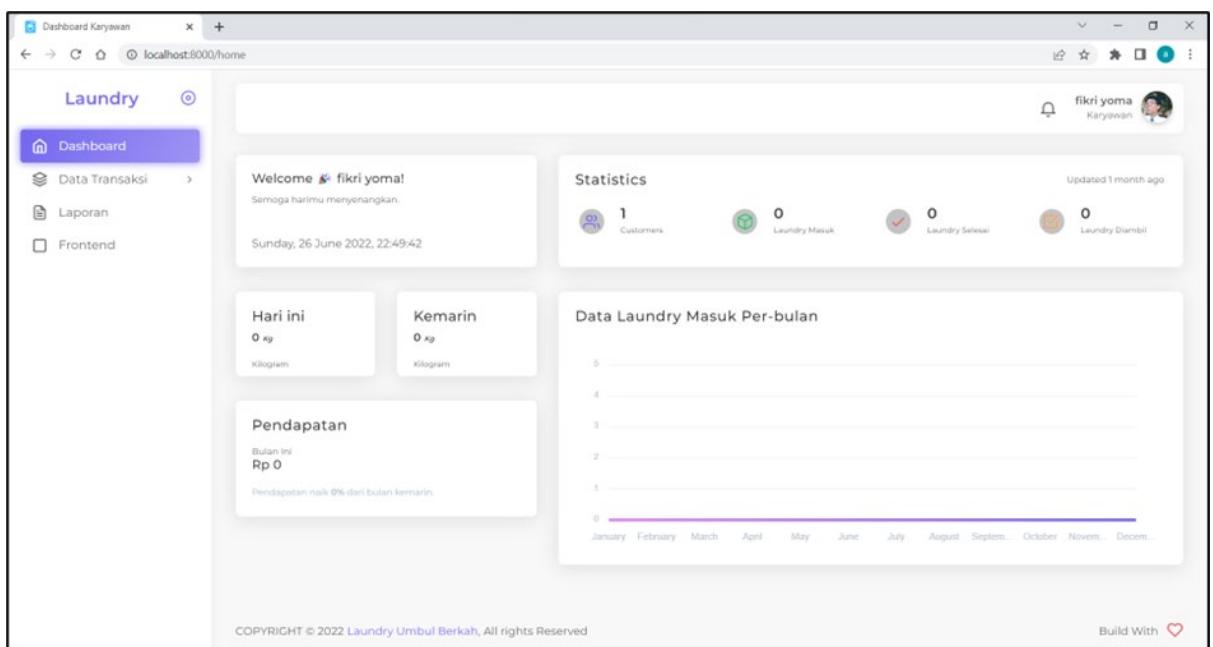
Admin juga dapat mengatur tampilan general pada website laundry, bahkan mengganti namanya. Admin juga dapat menentukan target laundry dan menambahkan data rekening bank yang tersimpan.



Gambar 4. Halaman Tambah Harga

Setelah menambahkan data rekening bank, pada data finance admin dapat menambahkan data harga pada sebuah cabang laundry untuk menjadi acuan perhitungan laundry tiap cabang tersebut.

- Karyawan



Gambar 5. Halaman Dashboard Karyawan

Pada halaman dashboard, karyawan dapat melihat jumlah customer, jumlah laundry masuk, selesai, hingga laundry yang telah diambil oleh customer. Lalu karyawan dapat melihat jumlah laundry hari ini dan kemarin dengan satuan kilogram. Karyawan

juga dapat melihat pendapatan laundry per bulan untuk membantu karyawan dalam mencapai target. Terdapat juga statistik laundry yang masuk per bulannya.

The screenshot shows a web-based application for laundry management. The left sidebar has a purple navigation bar with options: Dashboard, Data Transaksi (selected), Order Masuk, Tambah Order (highlighted in purple), Data Customer, Laporan, and Frontend. The main content area is titled 'Form Tambah Data Order' with a red '+ Customer Baru' button. It contains fields for: Nama Customer (dropdown, empty), No Transaksi (text input, '680622022'), Jenis Pembayaran (dropdown, empty), Status Pembayaran (dropdown, empty); Berat Pakalan (text input, empty), Pilih Pakalan (dropdown, empty), Hari (text input, empty), Disc (text input, '0'); Total Berat Pakalan (text input, empty), Lama Hari (dropdown, empty), Total Harga (text input, empty), Total Disc (text input, empty), and Total (text input, empty). At the bottom are 'Tambah' and 'Reset' buttons. The footer includes copyright information and a 'Build With ❤️' link.

Gambar 6. Halaman Tambah Order

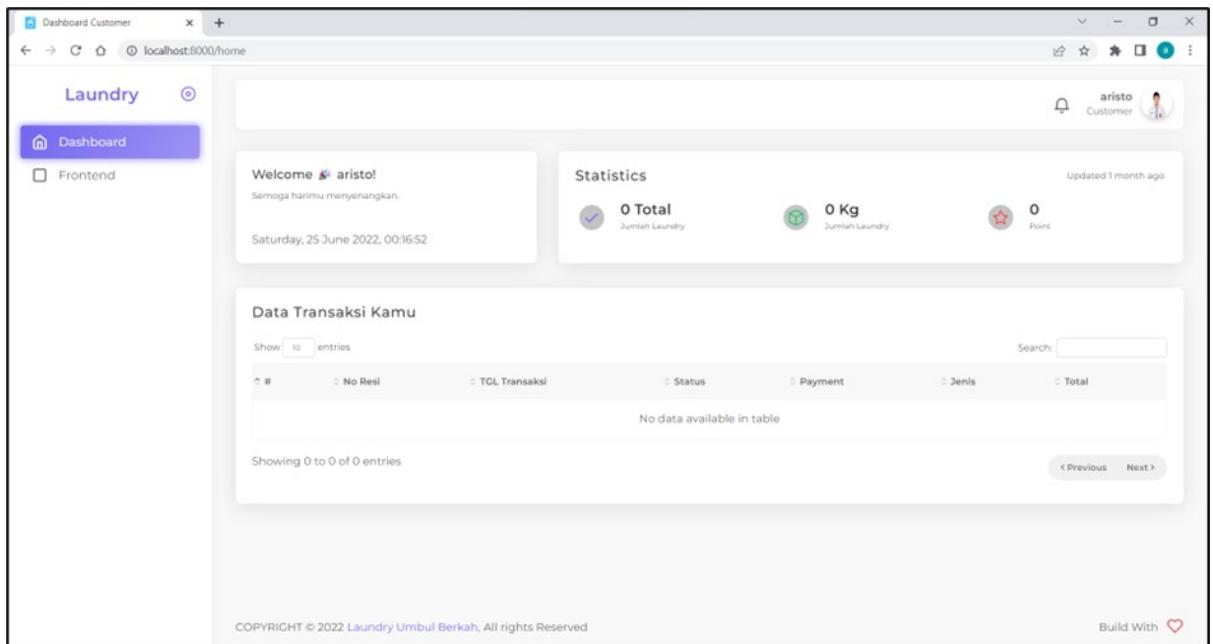
Peran memasukkan order hanya dapat dilakukan oleh karyawan. Diasumsikan pelanggan sedang berada di cabang laundry sehingga karyawanlah yang menginputkan data orderan. Pelanggan dapat menelepon atau mengirim email ke laundry apabila tidak dapat datang ke cabang laundry, sehingga karyawan yang akan menginputkannya. Orderan yang telah masuk akan tercatat di halaman Order Masuk.

The screenshot shows a web-based application for laundry management. The left sidebar has a purple navigation bar with options: Dashboard, Data Transaksi (selected), Order Masuk, Tambah Order, Data Customer (highlighted in purple), Laporan, and Frontend. The main content area is titled 'Tambah Customer'. It shows a table with one entry: Row 1, Name: aristo, Email: aristo@gmail.com, Address: Jl. Gajah Mada no. 15, Phone: 6282367589087. Action buttons for Detail, Delete, and Edit are shown. Below the table, it says 'Showing 1 to 1 of 1 entries'. The footer includes copyright information and a 'Build With ❤️' link.

Gambar 7. Halaman Tambah Customer

Untuk dapat memasukkan order maka customer harus telah terdaftar terlebih dahulu dan yang dapat menginputkannya adalah karyawan. Tombol “+Customer Baru” pada halaman Tambah Order akan mengarahkan karyawan ke halaman Data Customer atau dapat melalui navigasi yang ada pada sebelah kiri. Di sini akan ditampilkan data customer yang telah terdaftar dan karyawan dapat menambahkan customer baru, menghapus, mengedit, ataupun melihat detail info customer.

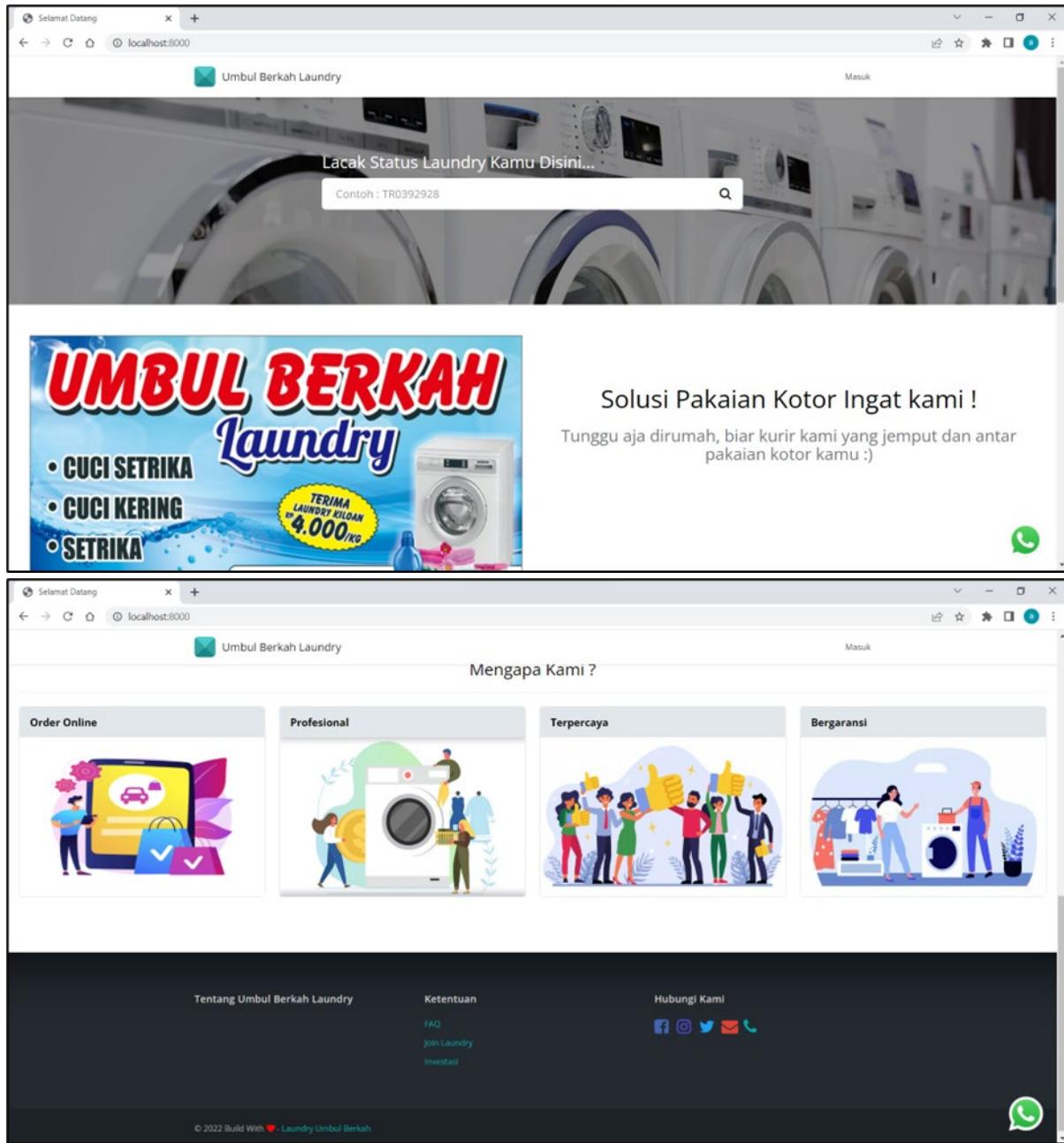
- Customer



Gambar 6. Halaman Dashboard Customer

Di sini customer hanya dapat melihat perkembangan status laundry dan riwayat transaksi laundry. Customer juga dapat melihat point yang telah didapatkan.

Halaman Landing Page



Gambar 7. Halaman Landing Page

Pada Landing Page, seperti tampak pada gambar terdapat header, banner, konten, dan juga footer. Pada header terdapat logo dan tulisan “Umbul Berkah Laundry” yang juga berfungsi sebagai button yang apabila di klik akan mengarahkan ke Landing Page. Sebelah kanannya ada button “Masuk” untuk menuju halaman login. Berikut adalah kode frontend untuk header

```

1 <div id="header" class="header navbar navbar-default navbar-fixed-top">
2   <!-- begin container -->
3   <div class="container">
4     <!-- begin navbar-header -->
5     <div class="navbar-header">
6       <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target="#header-navbar">
7         <span class="icon-bar"></span>
8         <span class="icon-bar"></span>
9         <span class="icon-bar"></span>
10        </button>
11        <a href="{{url('/')}}" class="navbar-brand">
12          <span class="navbar-logo"></span>
13          <span class="brand-text">
14            | ${setpage != NULL ? $setpage->judul : 'Judul Disini'}
15          </span>
16        </a>
17      </div>
18    <!-- end navbar-header -->
19    <!-- begin #header-navbar -->
20    <div class="collapse navbar-collapse" id="header-navbar">
21      <ul class="nav navbar-nav navbar-right">
22        @auth
23        <li> <a href="{{url('/home')}}">Welcome, {{Auth::user()->name}}</a> </li>
24        @else
25        <li><a href="{{route('login')}}">Masuk</a></li>
26        @endauth
27      </ul>
28    </div>
29    <!-- end #header-navbar -->
30  </div>
31  <!-- end container -->
32 </div>

```

Setelah header, ada banner yang juga terdapat kolom pencarian untuk melacak status laundry. Customer tidak perlu login, langsung ketikan nomor transaksi, sistem akan mencariakan status laundry customer. Berikut kode frontend pada banner

```

1  {{-- Banner --}}
2   <div class="bg-cover">
3     
4   </div>
5   <!-- end bg-cover -->
6   <!-- begin container -->
7   <div class="container">
8     <h3>Lacak Status Laundry Kamu Disini...</h3>
9     <div class="input-group m-b-20">
10       <input type="text" class="form-control input-lg" id="search_status" placeholder="Contoh : TR0392928" />
11       <span class="input-group-btn">
12         <button type="submit" class="btn btn-lg" id="search-btn"><i class="fa fa-search"></i></button>
13       </span>
14     </div>
15     @include('frontend.modal')
16   </div>
17 {{-- End Header --}}

```

Untuk selanjutnya akan diproses dengan menggunakan script berikut

```

26 @section('scripts')
27 <script type="text/javascript">
28 $(document).on('click', '.search-btn', function(e){
29   _curr_val = $('#search_status').val();
30   $('#search_status').val(_curr_val + $(this).html());
31 });
32
33 $(document).on('click', '#search-btn', function (e) {
34   var search_status = $("#search_status").val();
35   $.get('pencarian-laundry', {'_token': $('meta[name=csrf-token]').attr('content'), search_status:search_status}, function(resp){
36     if (resp != 0) {
37       $(".modal_status").show();
38       $("#customer").html(resp.customer);
39       $("#tgl_transaksi").html(resp.tgl_transaksi);
40       $("#status_order").html(resp.status_order);
41     }else{
42       swal({html: "No Invoice Tidak Terdaftar!"})
43     }
44   });
45 });
46 function close_dlg(){
47   $(".modal_status").hide();
48   $('#search_status').val("");
49 }
50 </script>
51 @endsection

```

Dan akan ditampilkan pada window status. Berikut kode frontend dari window status

```

1  <div class="modal_status">
2      <div class="modal_window">
3          {{-- <div class="title">Hasil Pencarian</div> --}}
4          <div class="row">
5              <div class="col-lg-5">
6                  <p class="text">Customer</p>
7                  <p class="font" id="customer"></p>
8              </div>
9              <div class="col-lg-3">
10                 <p class="text">Tgl Transaksi</p>
11                 <p class="font" id="tgl_transaksi"></p>
12             </div>
13             <div class="col-lg-3">
14                 <p class="text">Status</p>
15                 <p class="font" id="status_order"></p>
16             </div>
17         </div>
18         <br />
19         <button class="btn btn-danger btn-block" onclick="close_dlgs()">Close</button>
20     </div>
21 </div>

```

Setelah banner, kami memasukkan dua konten. Yang pertama adalah gambar desain promosi Laundry beserta kalimat iklan dan ikon – ikon yang menunjukkan kelebihan Laundry umbul Berkah. Masih terdapat banyak ruang untuk selanjutnya dapat dikembangkan lebih lanjut untuk penambahan konten. Berikut kode frontend untuk konten pertama dan kedua

```

1  <template>
2      <div class="col-12" style="margin-bottom: 35% !important">
3          <div class="panel panel-forum">
4              <div class="justify-content-start">
5                  <div class="col-md-6">
6                      
7                  </div>
8                  <div class="col-md-6" style="max-height:400px; min-height:400px;">
9                      <div class="" style="margin-top: 20%">
10                         <h1 class="text-center" style="color: black">
11                             Solusi Pakaian Kotor Ingat kami !
12                         </h1>
13                         <h3 class="text-center">
14                             Tunggu aja dirumah, biar kurir kami yang jemput dan antar pakaian kotor kamu : )
15                         </h3>
16                     </div>
17                 </div>
18             </div>
19         </div>
20     </div>
21 </template>

```

```

1 <template>
2   <div>
3     <h3 class="text-center" style="color: black;">Mengapa Kami ?</h3>
4     <hr>
5     <div class="row">
6       <div class="col-md-3 promo">
7         <div class="panel panel-forum">
8           <div class="panel-heading">
9             <h4 class="panel-title" style="font-weight:bold; color:black; font-size:15px">
10               Order Online
11             </h4>
12           </div>
13           
14         </div>
15       </div>
16
17       <div class="col-md-3 shadow-lg">
18         <div class="panel panel-forum tips">
19           <div class="panel-heading">
20             <h4 class="panel-title" style="font-weight:bold; color:black; font-size:15px">
21               Profesional
22             </h4>
23           </div>
24           
25         </div>
26       </div>
27
28       <div class="col-md-3 shadow-lg">
29         <div class="panel panel-forum tips">
30           <div class="panel-heading">
31             <h4 class="panel-title" style="font-weight:bold; color:black; font-size:15px">
32               Terpercaya
33             </h4>
34           </div>
35           
36         </div>
37       </div>

```

Dan yang terakhir ada footer, berisikan info – info tentang Laundry Umbul Berkah, ketentuan seperti FAQ, Join Laundry, dan investasi, alamat kontak seperti media sosial dan telepon, dan ada copyright. Berikut kode frontend untuk bagian footer

```

1 <div id="footer" class="footer">
2   <!-- begin container -->
3   <div class="container">
4     <!-- begin row -->
5     <div class="row">
6       <!-- begin col-4 -->
7       <div class="col-xl-4 col-lg-4 col-12">
8         <!-- begin section-container -->
9         <div class="section-container">
10           <h4>Tentang Umbul Berkah Laundry</h4>
11           <p>
12             {$setpage != NULL ? $setpage->tentang : 'Tentang belum disini'}
13           </p>
14         </div>
15         <!-- end section-container -->
16       </div>
17       <!-- end col-4 -->
18       <!-- begin col-4 -->
19       <div class="col-xl-4 col-lg-4 col-12">
20         <!-- begin section-container -->
21         <div class="section-container">
22           <h4>Ketentuan</h4>
23           <ul class="latest-post">
24             <li>
25               <a href="">FAQ</a>
26             </li>
27
28             <li>
29               <a href="">Join Laundry</a>
30             </li>
31
32             <li>
33               <a href="">Investasi</a>
34             </li>
35           </ul>
36         </div>
37         <!-- end section-container -->

```

```

39      </div>
40      <!-- begin col-4 -->
41      <div class="col-xl-4 col-lg-4 col-12">
42          <!-- begin section-container -->
43          <div class="section-container">
44              <h4>Hubungi Kami</h4>
45              <ul class="new-user">
46                  <li>
47                      <a href="https://Facebook.com/{{setpage->facebook ?? ''}}" target="_blank">
48                          <i class="fa fa-facebook fa-2x" style="color: #4267B2"></i>
49                      </a>
50                  </li>
51                  <li>
52                      <a href="https://Instagram.com/{{setpage->instagram ?? ''}}" target="_blank">
53                          <i class="fa fa-instagram fa-2x" style="color: #5B51D8"></i>
54                      </a>
55                  </li>
56                  <li>
57                      <a href="https://Twitter.com/{{setpage->twitter ?? ''}}" target="_blank">
58                          <i class="fa fa-twitter fa-2x" style="color: #1DA1F2"></i>
59                      </a>
60                  </li>
61                  <li>
62                      <a href="mailto:{{setpage->email ?? ''}}" target="_blank">
63                          <i class="fa fa-envelope fa-2x" style="color: #DB4437"></i>
64                      </a>
65                  </li>
66                  <li>
67                      <a href="tel:{{setpage->no_telp ?? ''}}" target="_blank">
68                          <i class="fa fa-phone fa-2x"></i>
69                      </a>
70                  </li>
71              </ul>
72          </div>
73          <!-- end section-container -->
74      </div>
75      <!-- end col-4 -->
76  </div>

```

Kami juga menambahkan ikon whatsapp khusus terletak pada pojok kanan bawah halaman landing page dan akan mengikuti ketika halaman di scroll. Ikon tersebut juga apabila di klik akan mengarahkan ke menu aplikasi chat whatsapp. Berikut kode untuk penambahan ikon whatsapp

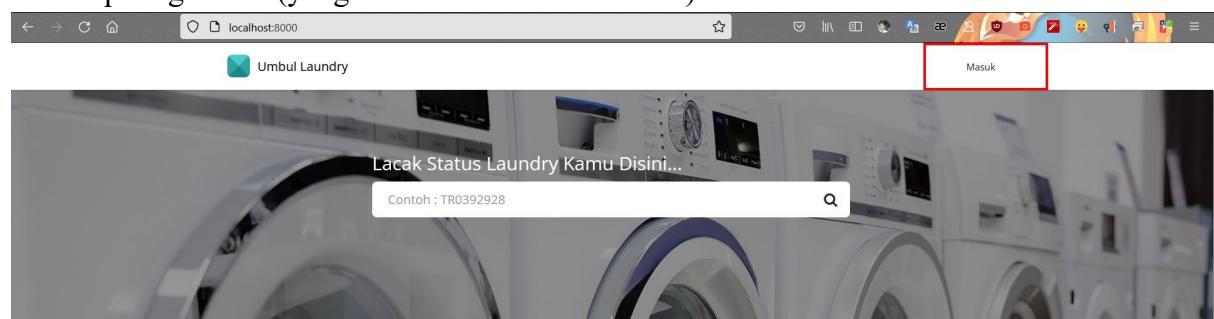
```

{-- Whatsapp Button Start--}
<a href="https://wa.me/{{setpage->whatsapp ?? ''}}" target="blank">
| 
</a>
{-- End: Whatsapp Button --}
@endsection

```

Halaman Login

untuk memulai login, terdapat button untuk masuk ke halaman login yakni di bagian header paling kanan (yang diberi kotak warna merah)



Solusi Pakaian Kotor Ingat kami !

Tunggu aja dirumah, biar kurir kami yang jemput dan antar pakaian kotor kamu :)



Kode yang digunakan untuk melakukan route login adalah sebagai berikut :

```

19  |     <!-- begin #header-navbar -->
20  |     <div class="collapse navbar-collapse" id="header-navbar">
21  |         <ul class="nav navbar-nav navbar-right">
22  |             @auth
23  |                 <li><a href="{{url('/home')}}">Welcome, {{Auth::user()->name}}</a> </li>
24  |             @else
25  |                 <li><a href="{{route('login')}}">Masuk</a></li>
26  |             @endauth
27  |         </ul>
28  |     </div>
29  |     <!-- end #header-navbar -->

```

Masuk

ketika mengklik “Masuk” maka halaman akan direferensikan ke link login

Ada beberapa bagian yang di generate oleh laravel, pada bagian route, akan ada route login yang sepaket di dalamnya ada route untuk login, logout, register, dan lain lain. yaitu fungsi **Auth::routes();** Sehingga kode yang kita gunakan dalam route web.php adalah satu fungsi saja yakni

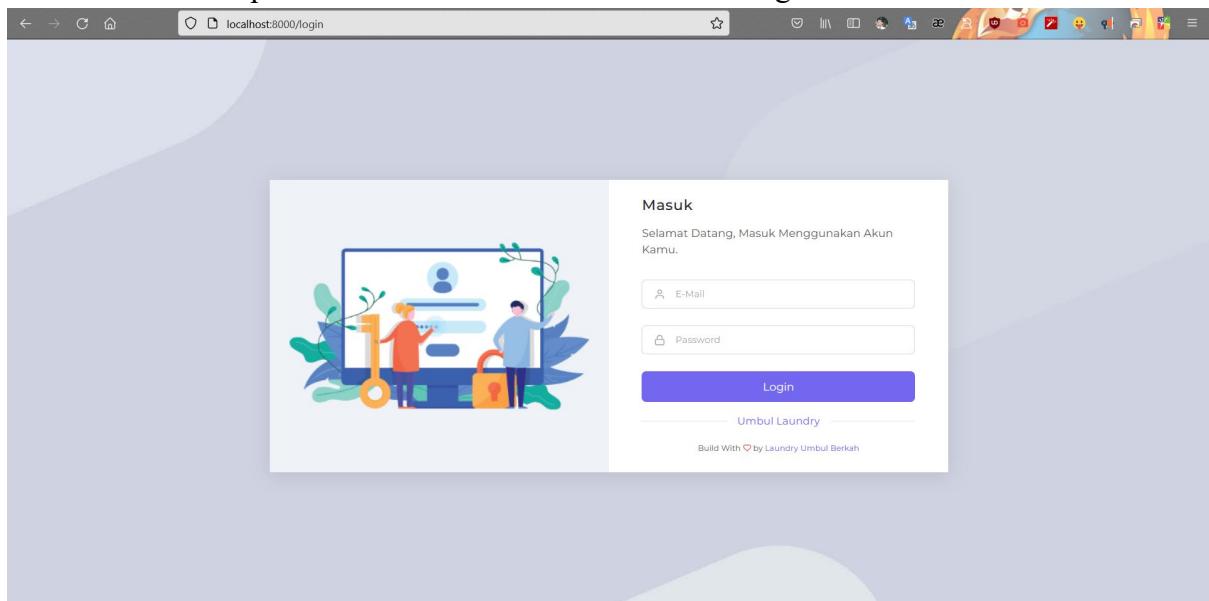
```

Auth::routes([
    'register' => false,
]);

```

‘register’ => false berfungsi untuk menonaktifkan pendaftaran untuk pengguna baru sehingga tidak ada formulir pendaftaran pada web yang kami buat ini

Berikut adalah tampilan halaman user ketika melakukan login



Authentifikasi yang digunakan untuk login adalah E-Mail dan Password, berikut kode yang digunakan pada halaman ini

```

26 <form action="{{route('login')}}" method="POST">
27   @csrf
28   <fieldset class="form-label-group form-group position-relative has-icon-left">
29     <input type="email" name="email" class="form-control @error('email') is-invalid @enderror" id="email" placeholder="E-Mail" value=""
30     @error('email')
31       <span class="invalid-feedback text-danger" role="alert">
32         <strong>{{ $message }}</strong>
33       </span>
34     @enderror
35     <div class="form-control-position">
36       <i class="feather icon-user"></i>
37     </div>
38     <label for="email">E-Mail</label>
39   </fieldset>
40
41   <fieldset class="form-label-group position-relative has-icon-left">
42     <input type="password" name="password" class="form-control @error('password') is-invalid @enderror" id="user-password" placeholder=""
43     @error('password')
44       <span class="invalid-feedback text-danger" role="alert">
45         <strong>{{ $message }}</strong>
46       </span>
47     @enderror
48     <div class="form-control-position">
49       <i class="feather icon-lock"></i>
50     </div>
51     <label for="user-password">Password</label>
52   </fieldset>
53   <button type="submit" class="btn btn-primary float-right btn-inline btn-block">Login</button>
54 </form>

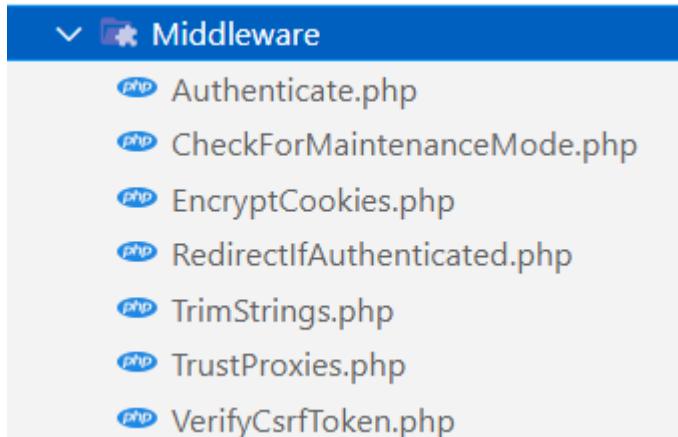
```

<form action="{{route('login')}}" method="POST">

form ini melakukan input terhadap email dan password yang setelah itu action form akan dikirim menggunakan method post ke route login untuk dilakukan validasi menggunakan fitur Middleware dari laravel

Integrasi login dengan halaman sesuai aktor

Setelah dilakukan submit email dan password maka form akan divalidasi masuk kedalam ranah Middleware. Middleware memiliki beberapa file yang berfungsi untuk melakukan autentifikasi user, berikut file yang ada dalam Middleware :



file yang perlu kita konfigurasi adalah file Authenticate.php dan RedirectIfAuthenticated.php menggunakan kode berikut

```

app > Http > Middleware >  Authenticate.php >  Authenticate
1  <?php
2
3  namespace App\Http\Middleware;
4
5  use Illuminate\Auth\Middleware\Authenticate as Middleware;
6
7  class Authenticate extends Middleware
8  {
9      /**
10      * Get the path the user should be redirected to when they are not authenticated.
11      *
12      * @param \Illuminate\Http\Request $request
13      * @return string
14      */
15      protected function redirectTo($request)
16      {
17          if (!$request->expectsJson()) {
18              return route('login');
19          }
20      }
21 }

```

kode ini adalah kode dalam file Authenticate.php yang berfungsi jika user tidak terautentifikasi maka user tersebut akan dikembalikan ke halaman login kembali.

```

app > Http > Middleware >  RedirectIfAuthenticated.php > ...
1  <?php
2
3  namespace App\Http\Middleware;
4
5  use Closure;
6  use Illuminate\Support\Facades\Auth;
7
8  class RedirectIfAuthenticated
9  {
10     /**
11      * Handle an incoming request.
12      *
13      * @param \Illuminate\Http\Request $request
14      * @param \Closure $next
15      * @param string|null $guard
16      * @return mixed
17      */
18     public function handle($request, Closure $next, $guard = null)
19     {
20         if (Auth::guard($guard)->check()) {
21             return redirect('/home');
22         }
23
24         return $next($request);
25     }
26 }

```

Dan kode ini adalah kode dalam file RedirectIfAuthenticated.php yang berfungsi jika user berhasil terautentifikasi maka user akan diteruskan atau di redirect ke halaman home. Nah setelah melewati tahap ini, tahap selanjutnya adalah menentukan role setiap user yang telah login melalui middleware route pada web.php

Agar middleware berfungsi secara global untuk ditugaskan kepada grup atau user individual maka kita perlu menambahkan kode pada file kernel.php dengan kode berikut :

```

53     protected $routeMiddleware = [
54         'auth' => \App\Http\Middleware\Authenticate::class,
55         'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,
56         'bindings' => \Illuminate\Routing\Middleware\SubstituteBindings::class,
57         'cache.headers' => \Illuminate\Http\Middleware\SetCacheHeaders::class,
58         'can' => \Illuminate\Auth\Middleware\Authorize::class,
59         'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,
60         'signed' => \Illuminate\Routing\Middleware\ValidateSignature::class,
61         'throttle' => \Illuminate\Routing\Middleware\ThrottleRequests::class,
62         'verified' => \Illuminate\Auth\Middleware\EnsureEmailIsVerified::class,
63         'role' => \Spatie\Permission\Middlewares\RoleMiddleware::class,
64         'permission' => \Spatie\Permission\Middlewares\PermissionMiddleware::class,
65         'role_or_permission' => \Spatie\Permission\Middlewares\RoleOrPermissionMiddleware::class,
66     ];
67
68 /**
69 * The priority-sorted list of middleware.
70 *
71 * This forces non-global middleware to always be in the given order.
72 *
73 * @var array
74 */
75 protected $middlewarePriority = [
76     \Illuminate\Session\Middleware\StartSession::class,
77     \Illuminate\View\Middleware\ShareErrorsFromSession::class,
78     \App\Http\Middleware\Authenticate::class,
79     \Illuminate\Session\Middleware\AuthenticateSession::class,
80     \Illuminate\Routing\Middleware\SubstituteBindings::class,
81     \Illuminate\Auth\Middleware\Authorize::class,
82 ];
83 ]

```

Terdapat dua variabel yakni \$routeMiddleware yang berfungsi untuk menugaskan penerapan global pada middleware di web dan variabel \$middlewarePriority yang berfungsi memaksa middleware non-global untuk selalu berada dalam urutan tertentu. Ini berhubungan dengan penggunaan satu file yang penggunaannya dibedakan berdasarkan role yang dipakai.

Setelah melakukan konfigurasi kernel.php maka selanjutnya kita akan mengintegrasikan login sesuai role masing-masing dengan route middleware yang ada pada file web.php

```

Route::middleware('auth')->group(function () {
    Route::get('/home', 'HomeController@index')->name('home');

    // Modul Admin
    Route::prefix('/')->middleware('role:Admin')->group(function () {})

    // Modul Karyawan
    Route::prefix('/')->middleware('role:Karyawan')->group(function () {})

    // Modul Customer
    Route::prefix('/')->middleware('role:Customer')->group(function () {})
}

```

Kode diatas adalah sepotong kode untuk menentukan role dari masing-masing user jika user masuk dari salah satu role diatas maka group function () {} akan aktif sesuai fitur yang ada dalam role tersebut.

Berikut adalah perbedaan yang ada pada halaman setelah login sesuai aktor pada web ini “Admin”

The screenshot shows the Laundry Management System dashboard for an administrator. The top navigation bar includes links for Home, Logout, and Profile (Administrator Admin). The sidebar on the left has a 'Dashboard' button highlighted in blue, along with other options: Data User, Transaksi, Data Finance, and Other. The main content area features several cards:

- Jumlah Customer:** 5
- Laundry Masuk:** 1
- Laundry Selesai:** 0
- Laundry Dambil:** 0

Pendapatan: Rp 0 (Tahun ini 2022), Rp 0 (Bulan ini June), Rp 0 (Tahun lalu 2021)

Pendapatan: Rp 0 (Hari ini Friday), Rp 0 (Kemarin Thursday)

Data Laundry Masuk Per-hari: A chart showing laundry arrivals per day from 1 to 31.

Data Laundry Masuk Per-bulan: A chart showing laundry arrivals per month from January to December.

“Karyawan”

The screenshot shows the Laundry Management System dashboard for a employee. The top navigation bar includes links for Home, Logout, and Profile (Aristo Karyawan). The sidebar on the left has a 'Dashboard' button highlighted in blue, along with other options: Data Transaksi, Laporan, and Frontend. The main content area features several cards:

Welcome: Welcome Aristo!

Statistics: 5 Customers, 1 Laundry Masuk, 0 Laundry Selesai, 0 Laundry Dambil. Updated 1 month ago.

Hari ini: 0 kg Kilogram

Kemarin: 0 kg Kilogram

Pendapatan: Bulan Ini Rp 0. Pendapatan naik 0% dari bulan kemarin.

Data Laundry Masuk Per-bulan: A chart showing laundry arrivals per month from January to December.

COPYRIGHT © 2022 Laundry Umbul Berkah, All rights Reserved Build With ❤️

“Customer”

The screenshot shows the customer dashboard for the Laundry Umbul Berkah application. At the top, there's a header with the brand name "Laundry Umbul Berkah" and a user profile for "Ayase Customer". Below the header is a sidebar with two items: "Dashboard" (selected) and "Frontend". The main content area has three main sections: a welcome message, statistics, and a transaction history table.

Welcome: Welcome Ayase! Semoga harimu menyenangkan. Friday, 24 June 2022, 14:13:18

Statistics: Updated 1 month ago

Icon	Value	Description
Checkmark	0 Total	Jumlah Laundry
Kg icon	0 Kg	Jumlah Laundry
Star icon	0 Point	

Data Transaksi Kamu:

#	No Resi	TGL Transaksi	Status	Payment	Jenis	Total
No data available in table						

Showing 0 to 0 of 0 entries

COPYRIGHT © 2022 Laundry Umbul Berkah, All rights Reserved Build With ❤️

Terlihat bahwa terdapat perbedaan tampilan dashboard dan fitur pada sidebar kiri dashboard pada masing-masing role sehingga role Admin dengan Karyawan memiliki fungsi interface yang berbeda begitu pula Admin dengan Customer maupun Karyawan dengan Customer namun tetap isi database pada web ini masih dalam satu kesatuan terintegrasi data sehingga setiap ada transaksi dari salah satu role maka role yang memiliki kepentingan untuk mengetahui transaksi tersebut dapat mengetahuinya juga secara real time

Integrasi halaman CRUD ke dashboard sesuai aktor yang login

Sesuai pada contoh sebelumnya setiap halaman aktor satu berbeda dengan aktor lainnya, terdapat tiga aktor atau role dalam website kami yakni “Admin”, “Karyawan”, dan “Customer” setiap role memiliki integrasi halaman CRUD ke dashboard masing-masing berikut kode yang kami gunakan untuk satu file yang memiliki integrasi tiga role sekaligus

```
app > Http > Controllers > HomeController.php > HomeController
14 |     * Create a new controller instance.
15 |
16 |     *
17 |     * @return void
18 |     */
19 |     public function __construct()
20 |     {
21 |         $this->middleware('auth');
22 |     }
23 |
24 |    /**
25 |     * Show the application dashboard.
26 |     *
27 |     * @return \Illuminate\Contracts\Support\Renderable
28 |     */
29 |     public function index()
30 |     {
31 |         if(Auth::check()){
32 |             if (Auth::user()->auth === "Admin") {
33 |                 ...
34 |             } elseif(Auth::user()->auth === "Karyawan") {
35 |                 ...
36 |             } elseif(Auth::user()->auth == 'Customer'){
37 |                 ...
38 |             }
39 |         }
40 |     }
41 |
42 | }
43 |
44 | }
```

Kode ini kami selipkan di setiap kode blok program agar terlihat pada laporan, disini kami menerapkan integrasi halaman dashboard dengan menggunakan if else pada program sehingga satu file controller memiliki tiga role sekaligus sehingga bisa menghemat file dalam suatu dashboard.

Setiap pengkondisian `Auth::user()->auth === “Aktor”` memiliki kodenya masing-masing sesuai kebutuhan untuk selanjutnya kami akan memperlihatkan kode program yang digunakan untuk setiap role terhadap dashboard yang kami buat

Role Admin

`Auth::user() ->auth === “Admin”`
memiliki kode pada controlernya sebagai berikut :

```

31     if(Auth::check()){    Undefined type 'Auth'.
32         if (Auth::user()->auth == "Admin") {    Undefined type 'Auth'.
33             $masuk = transaksi::whereIN('status_order',['Process','Done','Delivery'])->count();
34             $selesai = transaksi::where('status_order','Done')->count();
35             $diambil = transaksi::where('status_order','Delivery')->count();
36             $customer = User::where('auth','Customer')->get();
37             $sudahbayar = transaksi::where('status_payment','Success')->count();
38             $belumbayar = transaksi::where('status_payment','Pending')->count();
39             $incomeY = transaksi::where('status_payment','Success')
40             ->where('tahun',date('Y'))->sum('harga_akhir');
41
42             $incomeM = transaksi::where('status_payment','Success')
43             ->where('tahun',date('Y'))->where('bulan', ltrim(date('m'),'0'))->sum('harga_akhir');
44
45             $incomeYOld = transaksi::where('status_payment','Success')
46             ->where('tahun',date("Y",strtotime("-1 month")))->sum('harga_akhir');
47
48             $incomeD = transaksi::where('status_payment','Success')
49             ->where('tahun',date('Y'))->where('bulan', ltrim(date('m'),'0'))->where('tgl',ltrim(date('d'),'0'))->sum('harga_akhir');
50
51             $incomeDOld = transaksi::where('status_payment','Success')->where('tahun',date('Y'))
52             ->where('bulan', ltrim(date('m'),'0'))->where('tgl',ltrim(date("d",strtotime("-1 day")), '0'))->sum('harga_akhir');
53
54             $data = DB::table("transaksis")    Undefined type 'DB'.
55             ->select("id" ,DB::raw("(COUNT(*)) as customer"))    Undefined type 'DB'.
56             ->orderBy('created_at')
57             ->groupBy(DB::raw("MONTH(created_at)"))    Undefined type 'DB'.
58             ->count();
59
60             // Statistik Harian
61             $hari = DB::table('transaksis')    Undefined type 'DB'.
62             -> select('tgl', DB::raw('count(id) AS jml'))    Undefined type 'DB'.
63             -> whereYear('created_at','=',date("Y", strtotime(now())))
64             -> whereMonth('created_at','=',date("m", strtotime(now())))
65             -> groupBy('tgl')
66             -> get();
67
68             $tanggal = '';
69             $batas = 31;
70             $nilai = '';
71             for($_i=1; $_i <= $batas; $_i++){
72                 $tanggal = $tanggal . (string)$_i . ',';
73                 $check = false;
74                 foreach($hari as $data){
75                     if((int)$data->tgl === $_i){
76                         $nilai = $nilai . (string)$data->jml . ',';
77                         $check = true;
78                     }
79                 }
80                 if(!$check){
81                     $nilai = $nilai . '0,';
82                 }
83             }
84
85             // Statistik Bulanan
86             $bln = DB::table('transaksis')    Undefined type 'DB'.
87             -> select('bulan', DB::raw('count(id) AS jml'))    Undefined type 'DB'.
88             -> whereYear('created_at','=',date("Y", strtotime(now())))
89             -> whereMonth('created_at','=',date("m", strtotime(now())))
90             -> groupBy('bulan')
91             -> get();
92
93             $bulans = '';
94             $batas = 12;
95             $nilaiB = '';
96             for($_i=1; $_i <= $batas; $_i++){

```

```

97     $bulans = $bulans . (string)$_['i'] . ',';
98     $_check = false;
99     foreach($bln as $_data){
100         if((int)@$_data->bulan === $_i){
101             $nilaiB = $nilaiB . (string) $_data->jml . ',';
102             $_check = true;
103         }
104     }
105     if(!$_check){
106         $nilaiB = $nilaiB . '0,';
107     }
108 }
109
110 return view('modul_admin.index')
111     -> with('data', $data)
112     -> with('masuk', $masuk)
113     -> with('selesai', $selesai)
114     -> with('customer', $customer)
115     -> with('sudahbayar', $sudahbayar)
116     -> with('belumbayar', $belumbayar)
117     -> with('_tanggal', substr($tanggal, 0,-1))
118     -> with('_nilai', substr($nilai, 0, -1))
119     -> with('_bulan', substr($bulans, 0,-1))
120     -> with('_nilaiB', substr($nilaiB, 0, -1))
121     -> with('dambil', $dambil)
122     -> with('incomeY', $incomeY)
123     -> with('incomeM', $incomeM)
124     -> with('incomeYold', $incomeYold)
125     -> with('incomeD', $incomeD)
126     -> with('incomeDold', $incomeDold);

```

ketika kita menggunakan Auth::user() ->auth === “Admin” maka kita akan masuk ke dalam blok program yang hanya bisa diakses oleh user admin dan ketika kita mengakses controlernya melalui route yang telah dibuat maka controller ini melakukan return view(‘modul_admin.index’) yang dimana ini merupakan file blade.php untuk menampilkan interface atau tampilan program. Saat melakukan fungsi return view() fungsi ini kemudian dichaining dengan beberapa fungsi pengambilan data yakni fungsi -> with() yang digunakan sebagai method atau fungsi dalam pengambilan data dari model yang kita buat.

model yang saya buat adalah model transaksi yang terdapat dalam folder app/models. tujuan pembuatan model adalah untuk membuat aturan relasi database yang akan kita gunakan disini dalam model transaksi berisi kode program untuk membuat aturan relasi tersebut

```

app > Models >  transaksi.php >  transaksi
1  <?php
2
3  namespace App\Models;
4
5  use Illuminate\Database\Eloquent\Model;
6  use Illuminate\Notifications\Notifiable;
7
8  class transaksi extends Model
9  {
10     use Notifiable;
11     protected $fillable = [
12         'customer_id', 'user_id', 'tgl_transaksi', 'customer', 'status_order', 'status_payment', 'harga_id', 'kg', 'hari', 'harga', 'tgl', 'tgl_ambil', 'invoi
13     ];
14
15     public function price()
16     {
17         return $this->belongsTo(harga::class, 'harga_id', 'id');
18     }
19
20     public function customers()
21     {
22         return $this->belongsTo(User::class, 'customer_id', 'id')->where('auth', 'Customer');
23     }
24
25     public function user()
26     {
27         return $this->belongsTo(User::class, 'user_id', 'id');
28     }
29
30     public function detailTransaksi()
31     {
32         return $this->hasMany(transaksidetail::class, 'id_transaksi', 'id');
33     }
34 }
35

```

disini kita merancang pembuatan relasi antar tabel dalam setiap tabel yang kita buat seperti contoh dalam kode program ketika menggunakan fungsi price() maka fungsi ini akan berelasi one to one atau dalam kode program adalah belongsTo() dari model Transaksi ke model harga yang dimana harga_id melekat pada id serta beberapa fungsi fungsi untuk pengambilan data melalui database lainnya digunakan untuk membuat alur sistem backend menjadi lebih teratur

alhasil setelah kita menuju route yang telah dikontrol oleh controller maka controller akan mengembalikan view ke file blade.php disertai dengan data-data yang digunakan untuk menampilkan informasi-informasi dari database

Role Karyawan

Auth::user() ->auth === “Karyawan”

memiliki kode pada controlernya sebagai berikut :

```

128 } elseif(Auth::user()->auth === "Karyawan") {      Undefined type 'Auth'.
129     $masuk = transaksi::whereIN('status_order',['Process','Done','Delivery'])->where('user_id',auth::user()->id)->count();    Undefined
130     $selesai = transaksi::where('status_order','Done')->where('user_id',auth::user()->id)->count();      Undefined type 'Auth'.
131     $diambil = transaksi::where('status_order','Delivery')->where('user_id',auth::user()->id)->count();      Undefined type 'Auth'.
132     $customer = User::where('karyawan_id',auth::user()->id)->get();      Undefined type 'Auth'.
133
134     $kgToday = transaksi::where('user_id',Auth::id())->where('tahun',date('Y'))      Undefined type 'Auth'.
135     ->where('bulan', ltrim(date('m'),'0'))->where('tgl',ltrim(date('d'),'0'))->sum('kg');
136
137     $kgTodayOld = transaksi::where('user_id',Auth::id())->where('tahun',date('Y'))      Undefined type 'Auth'.
138     ->where('bulan', ltrim(date('m'),'0'))->where('tgl',ltrim(date("d",strtotime("-1 day")),'0'))->sum('kg');
139
140     $incomeM = transaksi::where('user_id',Auth::id())->where('status_payment','Success')      Undefined type 'Auth'.
141     ->where('tahun',date('Y'))->where('bulan', ltrim(date('m'),'0'))->sum('harga_akhir');
142
143     $incomeMOld = transaksi::where('user_id',Auth::id())->where('status_payment','Success')      Undefined type 'Auth'.
144     ->where('tahun',date('Y'))->where('bulan', ltrim(date('m',strtotime("-1 month")),'0'))->sum('harga_akhir');
145
146     $persen = 0;
147     if ($incomeMOld != null || $incomeM != null) {
148         $persen = ($incomeM - $incomeMOld) / $incomeM * 100;
149     }
150
151     // Statistik Bulanan
152     $bln = DB::table('transaksis')      Undefined type 'DB'.
153     -> select('bulan', DB::raw('count(id) AS jml'))      Undefined type 'DB'.
154     -> whereYear('created_at','=',date("Y", strtotime(now())))
155     -> whereMonth('created_at','=',date("m", strtotime(now())))
156     -> groupBy('bulan')
157     -> get();
158
159     $bulans = '';
160     $batas = 12;
161     $nilaiB = '';
162     for($_i=1; $_i <= $batas; $_i++){
163         $bulans = $bulans . (string)$_i . ',';
164         $_check = false;
165         foreach($bln as $_data){
166             if((int)@$_data->bulan === $_i){
167                 $nilaiB = $nilaiB . (string)$_['data]->jml . ',';
168                 $_check = true;
169             }
170         }
171         if(!$check){
172             $nilaiB = $nilaiB . '0,';
173         }
174     }
175
176     return view('karyawan.index')
177     -> with('diambil', $diambil)
178     -> with('masuk',$masuk)
179     -> with('selesai',$selesai)
180     -> with('customer', $customer)
181     -> with('kgToday', $kgToday)
182     -> with('kgTodayOld', $kgTodayOld)
183     -> with('incomeM',$incomeM)
184     -> with('incomeMOld',$incomeMOld)
185     -> with('persen',$persen)
186     -> with('_bulan', substr($bulans, 0,-1))
187     -> with('_nilaiB', substr($nilaiB, 0, -1));

```

Sama seperti pada role admin analogi penggunaannya juga sama yakni pertama-tama masuk ke dalam blok program karyawan yang hanya bisa diakses oleh user karyawan dan controller melakukan return view ke file blade.php dengan dichaining oleh beberapa pengambilan data yang diperlukan

Role Customer

Auth::user() ->auth === "Customer"

memiliki kode pada controllernya sebagai berikut :

```
189 }elseif(Auth::user()->auth == 'Customer'){
190     $totalLaundry = transaksi::where('customer_id',Auth::id())->count();      Undefined type 'Auth'.
191     $totalLaundryKg = transaksi::where('customer_id',Auth::id())->sum('kg');      Undefined type 'Auth'.
192
193     $transaksi = transaksi::where('customer_id',Auth::id())->get();      Undefined type 'Auth'.
194
195     return view('customer.index',\compact('totalLaundry','totalLaundryKg','transaksi'));
196 }
197 }
198 }
199 }
200 }
```

Sama seperti pada role admin dan karyawan analogi penggunaannya juga sama yakni pertama-tama masuk ke dalam blok program customer yang hanya bisa diakses oleh user customer dan controller melakukan return view ke file blade.php dengan dichaining oleh beberapa pengambilan data yang diperlukan namun chaining method yang digunakan adalah method compact(). menggunakan method ini karena variabel data yang diambil tidak sebanyak sebelumnya sehingga bisa cukup untuk satu baris.

CRUD (Master Entitas dan Transaksi)

Insert Post

Laravel membuat koneksi dengan database dan menjalankan Query menjadi sangat sederhana. Agar postingan dapat muncul dan dilihat dalam website, maka diperlukan INSERT terlebih dahulu. Dalam kasus laundry ini, postingan dapat berupa data user (karyawan / customer) atau data transaksi.

Disini dicontohkan ketika karyawan akan menambahkan data customer baru. Akan dibuat method / function baru di dalam controller karyawan, method ini akan kita gunakan untuk proses melakukan insert data ke dalam database. Dilakukan pada file app\Http\Controllers\Karyawan\CustomerController.php dan kemudian silahkan tambahkan kode berikut ini di bawah function index dan detail.

```
// Create
public function create()
{
    return view('karyawan.customer.create');
}

// Store
public function store(AddCustomerRequest $request)
{
    try {
        // Your logic here
    } catch (\Exception $e) {
        // Handle exception
    }
}
```

```

        DB::beginTransaction();
        $cekNumber = substr($request->no_telp,0,1); // ambil
angka pertama dari string
        $cekNumber1 = substr($request->no_telp,0,2); // ambil
angka pertama & kedua dari string

        if ($cekNumber == 0) { // cek jika angka pertama sama
dengan 0, jalankan perintah ini
            $removeNol = '62'. ltrim($request->no_telp, 0); //
Hapus angka kosong
        } elseif($cekNumber1 == 62) { // cek jika angka pertama &
kedua sama dengan 62, jalankan perintah ini
            $removeNol = $request->no_telp; // Balikan jika format
sudah benar
        }

$password = $this->acakpass(8);

$addCustomer = User::create([
    'karyawan_id' => Auth::id(),
    'name'         => $request->name,
    'email'        => $request->email,
    'auth'          => 'Customer',
    'status'        => 'Active',
    'no_telp'       => $removeNol,
    'alamat'        => $request->alamat,
    'password'      => Hash::make($password)
]) ;

$addCustomer->assignRole($addCustomer->auth);

```

Function Create

Function create akan kita gunakan untuk menampilkan form, dimana form ini akan digunakan oleh karyawan untuk melakukan input data, seperti memasukkan data nama customer, content dan gambar.

Function Store

Function store akan digunakan untuk melakukan proses insert data ke dalam database, di dalam function store kita membuat sebuah validasi, dimana data akan dimasukkan ke dalam database jika validasi tersebut sudah terpenuhi.

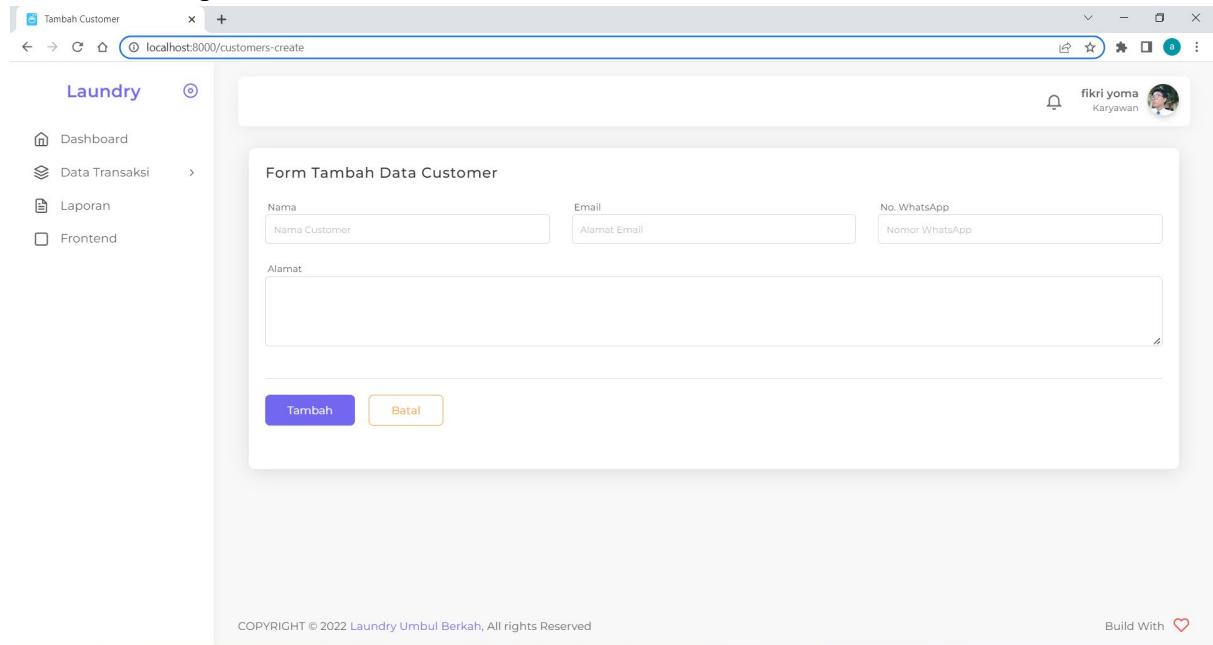
Membuat View Create

Setelah berhasil menambahkan 2 function baru untuk proses insert data, sekarang dilanjutkan untuk membuat halaman view atau templatanya. Buat file baru dengan nama create.blade.php di dalam folder resources\views\karyawan\customer\create.blade.php dan masukkan kode berikut ini

```
resources > views > karyawan > customer > create.blade.php > ...  
1 @extends('layouts.backend')  
2 @section('title', 'Tambah Customer')  
3 @section('header', 'Tambah Data Customer')  
4 @section('content')  
5 <div class="col-lg-12">  
6   <div class="card card-outline-info">  
7     <div class="card-header">  
8       <h4 class="card-title">Form Tambah Data Customer</h4>  
9     </div>  
10    <div class="card-body">  
11      <form action="{{url('customers-store')}}" method="POST">  
12        @csrf  
13        <div class="form-body">  
14          <div class="row p-t-20">  
15            <div class="col-md-4">  
16              <div class="form-group has-success">  
17                <label class="control-label">Nama</label>  
18                <input type="text" class="form-control form-control-danger @error('name') is-invalid @enderror" name="name" value="{{old('name')}}" placeholder="Masukkan Nama" required="">  
19                @error('name')  
20                  <span class="invalid-feedback text-danger" role="alert">  
21                    <strong>{{ $message }}</strong>  
22                  </span>  
23                @enderror  
24              </div>  
25            <!--/span-->  
26            <div class="col-md-4">  
27              <div class="form-group has-success">  
28                <label class="control-label">Email</label>  
29                <input type="email" class="form-control form-control-danger @error('email') is-invalid @enderror" name="email" value="{{old('email')}}" placeholder="Masukkan Email" required="">  
30                @error('email')  
31                  <span class="invalid-feedback text-danger" role="alert">  
32                    <strong>{{ $message }}</strong>  
33                  </span>  
34                @enderror  
35              </div>  
36            </div>  
37          </div>
```

```
resources > views > karyawan > customer > create.blade.php > div.col-lg-12 > div.card.card-outline-info > div.card-body > form > div.form-body > div.row.p-t-20 > div.col-md-4  
1 <div class="col-md-4">  
2   <div class="form-group has-success">  
3     <label class="control-label">WhatsApp</label>  
4     <input type="number" class="form-control form-control-danger @error('no_telp') is-invalid @enderror" name="no_telp" value="{{old('no_telp')}}" placeholder="Masukkan WhatsApp" required="">  
5     @error('no_telp')  
6       <span class="invalid-feedback text-danger" role="alert">  
7         <strong>{{ $message }}</strong>  
8       </span>  
9     @enderror  
10   </div>  
11   <!--/span-->  
12   <div class="col-md-12">  
13     <div class="form-group has-success">  
14       <label class="control-label">Alamat</label>  
15       <textarea name="alamat" class="form-control @error('alamat') is-invalid @enderror" rows="3" placeholder="Masukkan Alamat" value="{{old('alamat')}}" required="">  
16       @error('alamat')  
17         <span class="invalid-feedback text-danger" role="alert">  
18           <strong>{{ $message }}</strong>  
19         </span>  
20       @enderror  
21     </div>  
22   </div>  
23   <!--/row-->  
24 </div>  
25 <input type="hidden" name="auth" value="Admin">  
26 <div class="form-actions">  
27   <button type="submit" class="btn btn-primary mr-1 mb-1">Tambah</button>  
28   <a href="{{url('customers')}}" class="btn btn-outline-warning mr-1 mb-1">Batal</a>  
29 </div>  
30 </form>  
31 </div>  
32 </div>  
33 </div>  
34 <!--/section-->
```

screenshot tampilan



Update and Delete Post

• Update

untuk membuat fitur update hal pertama yang perlu dilakukan adalah membuat route yang akan berfungsi menghubungkan view dan controller

```
Route::get('customers-
edit/{id}', 'Karyawan\CustomerController@editcust');
```

{id} sebagai parameter untuk menangkap 1 id customer yang ingin di edit, selanjutnya di dalam file Karyawan\CustomerController kita buat function editcust yang didalam function ini kita isi code berikut

```
//Edit pelanggan
public function editcust($id)
{
    $pelanggan = DB::table('users')->where('id', $id)->get();
    return view('karyawan.customer.edit', ['pelanggan' =>
$pelanggan]);
}
```

function editcust(\$id) memiliki parameter \$id yang berfungsi menangkap id yang ingin di edit dengan menggunakan syntax PHP Query Builder, variable \$pelanggan berisi syntax untuk melakukan get() kolom id yang ada dalam tabel users dan selanjutnya kita kembalikan nilai atau lakukan return ke dalam view karyawan.customer.edit yang membawa associative array ['pelanggan' => \$pelanggan]

berikut isi code dari view karyawan.customer.edit

```
resources > views > karyawan > customer > edit.blade.php > div.col-lg-12 > div.card.card-outline-info > div.card-header
1  @extends('layouts.backend')
2  @section('title','Edit Customer')
3  @section('header','Edit Data Customer')
4  @section('content')
5  <div class="col-lg-12">
6      <div class="card card-outline-info">
7          <div class="card-header">
8              <h4 class="card-title">Form Edit Data Customer</h4>
9          </div>
10         <div class="card-body">
11             @foreach($pelanggan as $p)
12                 <form action="{{url('customers-saveeditcust')}}" method="POST">
13                     {{ csrf_field() }}
14                     <input type="hidden" name="id" value="{{$p->id}}> <br/>
15                     <div class="form-body">
16                         <div class="row p-t-20">
17                             <div class="col-md-4">
18                                 <div class="form-group has-success">
19                                     <label class="control-label">Nama</label>
20                                     <input type="text" class="form-control form-control-danger @error('name') is-invalid @enderror" value="{{$p->name}}>
21                                         @error('name')
22                                             <span class="invalid-feedback text-danger" role="alert">
23                                                 <strong>{{ $message }}</strong>
24                                             </span>
25                                         @enderror
26                                     </div>
27                                 </div>
28                                 <!--/span-->
29                                 <div class="col-md-4">
30                                     <div class="form-group has-success">
31                                         <label class="control-label">Email</label>
32                                         <input type="email" class="form-control form-control-danger @error('email') is-invalid @enderror" value="{{$p->email}}>
33                                         @error('email')
34                                             <span class="invalid-feedback text-danger" role="alert">
35                                                 <strong>{{ $message }}</strong>
36                                             </span>
37                                         @enderror

```

associative array ['pelanggan' => \$pelanggan] dilakukan perulangan @foreach(\$pelanggan as \$p) yang dimana perulangan ini melakukan print_r ke seluruh data dari variabel \$pelanggan untuk ditampilkan ke dalam tag html sehingga saat kita masuk ke dalam form edit nilai atribut dalam form edit yang dulu masih ada.

#	Nama	Email	Alamat	No Telpon	Action
1	Yogi	yogi@gmail.com	Surabaya	6282011633004	<button>Detail</button> <button>Edit</button> <button>Hapus</button>

Form Edit Data Customer

Nama: Yogi

Email: yogi@gmail.com

No. WhatsApp: 6282011633004

Alamat: Surabaya

Simpan Batal

terlihat data dengan nama Yogi masih sama setelah mengklik tombol edit karena ditambah atribut value = “{{\$p->id}}”

didalam form kita menggunakan attribute action=... dan method="POST" yang dimana setelah form kita input dan kirim akan dikirim dalam url('customers-saveeditcust') dan url ini akan kita buat routenya untuk diproses dengan controller

```
<form action="{!!url('customers-saveeditcust')}> method="POST">
```

```
Route::post('customers-
saveeditcust', 'Karyawan\CustomerController@saveeditcust');
```

Route yang digunakan adalah Route::post yang berfungsi melakukan post atau kirim data dengan dilakukan kontrol oleh CustomerController menggunakan function saveeditcust.

selanjutnya isi dari function saveeditcust berisi code sebagai berikut

```
//save edit pelanggan
public function saveeditcust(Request $request) {
    try {
        DB::beginTransaction();
        // update data pelanggan
        DB::table('users')->where('id', $request->id)->update([
            'name' => $request->name,
            'email' => $request->email,
            'no_telp' => $request->no_telp,
            'alamat' => $request->alamat
        ]);
        DB::commit();
        Session::flash('success', 'Perubahan data pelanggan sukses');
        return redirect('customers');
    } catch (ErrorException $e) {
        DB::rollback();
        throw new ErrorException($e->getMessage());
    }
}
```

dimana variabel \$request sebagai parameter dichaining dengan name dari inputan dalam form.

```
DB::table('users')->where('id', $request->id)->update([
    'name' => $request->name,
    'email' => $request->email,
    'no_telp' => $request->no_telp,
    'alamat' => $request->alamat
```

Berikut adalah hasil dari form yang telah diedit

Form Edit Data Customer

Nama Yogi	Email yogi@gmail.com	No. WhatsApp 6282011633004
Alamat Surabaya		
<button>Simpan</button> <button>Batal</button>		

Form Edit Data Customer

Nama Yondaime	Email yondaime@gmail.com	No. WhatsApp 6282011633005
Alamat Jepang		
<button>Simpan</button> <button>Batal</button>		

Perubahan data pelanggan sukses

Tambah Customer					
#	Nama	Email	Alamat	No Telpon	Action
1	Yondaime	yondaime@gmail.com	Jepang	6282011633005	<button>Detail</button> <button>Edit</button> <button>Hapus</button>
2	Ayase	ayase@gmail.com	Jepang	6282011633002	<button>Detail</button> <button>Edit</button> <button>Hapus</button>
3	Ikura	ikura@gmail.com	Jepang	6282011633001	<button>Detail</button> <button>Edit</button> <button>Hapus</button>
4	Alex	Alex@gmail.com	Surabaya	6282112345678	<button>Detail</button> <button>Edit</button> <button>Hapus</button>

- Delete

Delete merupakan salah satu fungsi yang tercakup dalam CRUD. Melalui fungsi ini data yang sudah tidak dibutuhkan dapat dihapus dari database. Contoh pemanfaatan fitur ini misalnya adalah untuk menghapus blog post yang tidak perlu, menghapus transaksi yang salah, dan berbagai fungsi lainnya.

Untuk membuat fitur delete hal pertama yang perlu dilakukan adalah membuat route yang akan berfungsi menghubungkan view dan controller

```
Route::get('customers-
delete/{id}', 'Karyawan\CustomerController@delete');
```

{id} sebagai parameter untuk menangkap 1 id customer yang ingin dihapus, selanjutnya kita buat function delete di dalam file Karyawan\CustomerController

```
public function delete($id)
{
    $customer = User::where('karyawan_id', Auth::user()->id)
        ->where('auth', 'Customer')
        ->where('id', $id)->delete();
    $customer = User::where('karyawan_id', Auth::user()->id)
        ->where('auth', 'Customer')
        ->orderBy('id', 'DESC')->get();
    return view('karyawan.customer.index', compact('customer'));
}
```

Menggunakan parameter \$id untuk menangkap id customer yang ingin dihapus. Setelah melalui tahap validasi, digunakan function dari PHP yaitu delete() pada variabel \$customer. Setelah itu, dilakukan -> OrderBy -> get() untuk memanggil kembali list database terbaru setelah salah satu data customer dihapus.

untuk dapat menjalankan program delete dari tampilan website, maka dibuat button delete pada view. Untuk melakukannya kita akan menambahkan syntax berikut ke dalam tabel Action pada tampilan

```
<a href="#" {{url('customers-delete', $item->id)}} " class="btn btn-sm
btn-danger" style="color:white">Delete</a>
```

Ditambahkan juga href untuk menghubungkan ke route customers-delete yang telah dibuat tadi.

Screenshot Tampilan

#	Nama	Email	Alamat	No Telpon	Action
1	alex	jonhalex@gmail.com	jl.pahlawan	628251736220	Detail Delete Edit

Upload and Update Image

Dikarenakan website kami yang bertemakan Laundry, maka tidak begitu dibutuhkan banyak gambar. Tetapi di sini akan dicontohkan proses mengupload gambar profil karyawan pada menu profile.

```
@elseif(auth::user()->auth == 'Karyawan')
    <a class="dropdown-item"
    href="{!!url('profile-karyawan', auth::user()->id ) !!}"><i class="feather
icon-user"></i>Profile
    </a>
```

Dalam membuat menu upload image kita membuat menu dropdown Profile yang melakukan href="{!!url('profile-karyawan', auth::user()->id) !!}"

selanjutnya kita buat route untuk bisa dikoneksikan dengan controller dan database

```
Route::get('profile-
karyawan/{id}', 'Karyawan\ProfileController@karyawanProfile');
```

setelah kita buat route, kita buat Controller yang memiliki function karyawanProfile disertai parameter \$id yang dimana kodennya berisi

```
// Profile Karyawan Cabang
public function karyawanProfile($id)
{
    $user = User::find($id);
    return view('karyawan.profile.index', compact('user'));
}
```

function ini melakukan return view karyawan.profile.index dengan menggaet variabel \$user yang variabel ini berisi User::find(\$id)

Selanjutnya isi dari file index.blade.php adalah sebagai form isian upload image dan beberapa update lainnya

```

resources > views > karyawan > profile > index.blade.php > ...
1  @extends('layouts.backend')
2  @section('title','Profile')
3  @section('content')
4  @if ($message = Session::get('success'))
5      <div class="alert alert-success alert-block">
6          <button type="button" class="close" data-dismiss="alert">Ã—</button>
7          <strong>{{ $message }}</strong>
8      </div>
9  @elseif($message = Session::get('error'))
10     <div class="alert alert-danger alert-block">
11         <button type="button" class="close" data-dismiss="alert">Ã—</button>
12         <strong>{{ $message }}</strong>
13     </div>
14  @endif
15  <div class="row">
16      <div class="col-lg-4 col-xlg-3 col-md-5">
17          <div class="card">
18              <div class="card-body">
19                  <div class="col text-center">
20                      <div class="m-t-30">
21                          
22                          <h4 class="card-title mt-1">{{Auth::user()->name}}</h4>
23                          <h6 class="small">customer</h6>
24                      </div>
25                  </div>
26              </div>
27              <div>
28                  <hr>
29              <div class="card-body"> <small class="text-muted">Email address </small>
30                  <h6>{{Auth::user()->email}}</h6> <small class="text-muted p-t-30 db">Phone</small>
31                  <h6>{{Auth::user()->no_telp}}</h6> <small class="text-muted p-t-30 db">Address</small>
32                  <h6>{{Auth::user()->alamat}}</h6>
33                  <small class="text-muted p-t-30 db">Social Profile</small>
34                  <br/>
35                  <button class="btn btn-circle btn-secondary"><i class="fa fa-facebook"></i></button>
36                  <button class="btn btn-circle btn-secondary"><i class="fa fa-twitter"></i></button>
37                  <button class="btn btn-circle btn-secondary"><i class="fa fa-youtube"></i></button>

```

letak form upload image berupa input type="file" adalah di kode berikut ini

```

102 <div class="col-md-6">
103     <div class="form-group has-success">
104         <label class="control-label">Foto</label>
105         <input type="file" name="foto" class="form-control @error('foto') is-invalid @enderror">
106         <span class="small text-warning">Biarkan kosong jika tidak ingin di update.</span>
107         @error('foto')
108             <span class="invalid-feedback text-danger" role="alert">
109                 <strong>{{ $message }}</strong>
110             </span>
111         @enderror
112     </div>
113 </div>

```

yang hasil inputannya akan dikirim melalui action form

```

<form action=" {{url('profile-karyawan/update', Auth::id())}} " method="POST" enctype="multipart/form-data">

```

Berikut adalah hasil tampilan form

Foto

Biarkan kosong jika tidak ingin di update.

Langkah selanjutnya kita membuat route profile-karyawan/update', Auth::id() dengan menggandeng Controller sebagai koneksi database

```

Route::put('profile-
karyawan/update/{id}', 'Karyawan\ProfileController@karyawanProfileSave')
;

```

kita menggunakan Route::put sebagai penempatan saja

selanjutnya kita membuat function karyawanProfileSave di dalam ProfileController yang memiliki parameter {\$id} sebagai penangkap id user dan {\$request} sebagai validasi file upload image

```
// Profile Karyawan Cabang - Save
public function karyawanProfileSave(Request $request, $id)
{
    $foto = $request->file('foto');
    if ($foto) {
        $nama_foto = time()."_" . $foto->getClientOriginalName();
        // isi dengan nama folder tempat kemana file diupload
        $tujuan_upload = 'public/images/foto_profile';
        $foto->storeAs($tujuan_upload, $nama_foto);
    }

    if ($request->password) {
        $password = Hash::make($request->password);
    }

    $profile = User::findOrFail($id);
    $profile->name = $request->name;
    $profile->email = $request->email;
    $profile->alamat = $request->alamat;
    $profile->nama_cabang = $request->nama_cabang;
    $profile->alamat_cabang = $request->alamat_cabang;
    $profile->foto = $nama_foto ?? Auth::user()->foto;
    $profile->password = $password ?? Auth::user()->password;
    $profile->save();

    Session::flash('success', 'Data profile berhasil diupdate !');
    return back();
}
```

Foto atau Image dikirim ke query dalam variabel menggunakan chaining method \$profile->foto sebelumnya query tersebut diambil dari nilai \$nama_foto ?? Auth::user()->foto; Kita dapat membuat pengkondisian file('foto') untuk mengubah nama file dan menempatkannya ke local storage yang ada dalam komputer

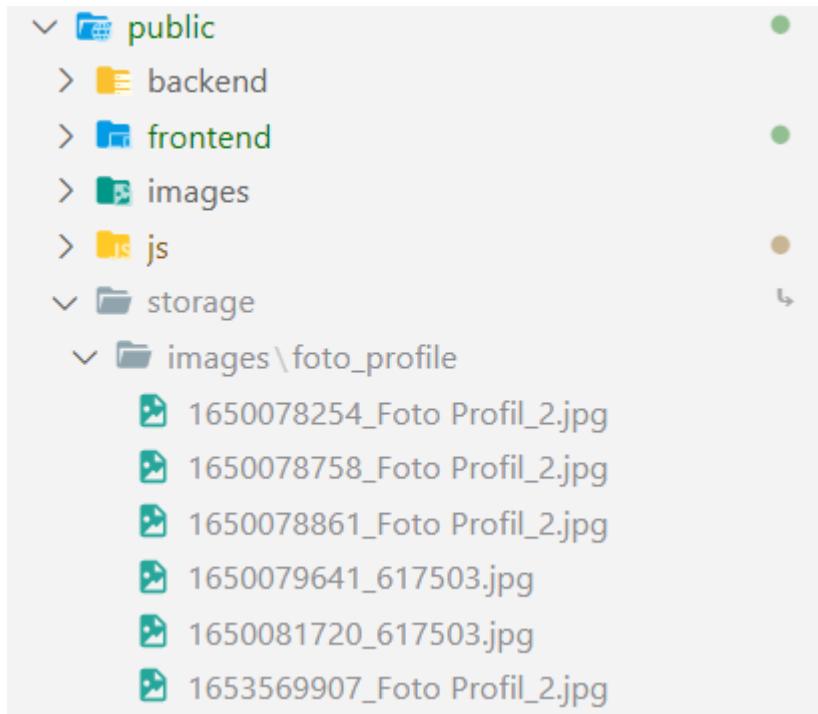
```
$foto = $request->file('foto');
if ($foto) {
    $nama_foto = time()."_" . $foto->getClientOriginalName();
    // isi dengan nama folder tempat kemana file diupload
    $tujuan_upload = 'public/images/foto_profile';
```

```

    $foto->storeAs($tujuan_upload,$nama_foto);
}

```

\$tujuan_upload sebagai direktori tempat kita hendak menyimpan file dan \$nama_foto sebagai nama file foto yang kita simpan serta \$foto->storeAs() function sebagai penempatannya



\$tujuan_upload = 'public/images/foto_profile'; menempatkan file kedalam folder public/storage/images/foto_profile

File Explorer Path: This PC > Acer (C:) > xampp > htdocs > laundry > public > storage > images > foto_profile



Selanjutnya apabila file foto berhasil lolos validasi ,query melakukan save \$profile->save(); dan kami menggunakan keterangan Session::flash('success','Data profile berhasil diupdate !'); kami melakukan return back(); untuk kembali ke halaman semula

berikut ilustrasi melakukan upload sekaligus update image profile

The screenshot illustrates the process of updating a user's profile picture and other information in a Laundry management system.

Top Left: Laundry management system dashboard with navigation links: Dashboard, Data Transaksi, Laporan, Frontend.

Top Right: User profile header showing "Aristo Karyawan" with a small profile picture.

Left Panel: Profile details for "Aristo" (Customer). It shows a profile picture, name, email, phone number, address, and social media links (Facebook, Twitter, LinkedIn).

Form Fields (Top Right):

- Informasi:**
 - Nama:** Aristo
 - Email:** aristoriza@gmail.com
 - Alamat:** Tegal, Jawa Tengah
 - No WhatsApp:** 081212345678
 - Foto:** [Browse...]
- Data Laundry:**
 - Nama Laundry (cabang):** Umbul Berkah Laundry
 - Alamat Laundry (cabang):** [Empty input field]

File Upload Dialog (Bottom Left):

- Shows a file selection dialog titled "File Upload".
- File path: "Downloads > Foto Profil > Foto-Foto".
- File list:
 - Foto 3.jpg
 - Foto ACT.jpg
 - Foto profil.png
 - Foto Profil_2.jpg
- File name: "Foto profil.png"
- Buttons: "Open" and "Cancel".

Bottom Center:

- Browse...** button followed by the selected file name: "Foto profil.png".
- Biarkan kosong jika tidak ingin di update.** (Leave empty if you don't want to update.)
- Update** and **Batal** buttons.

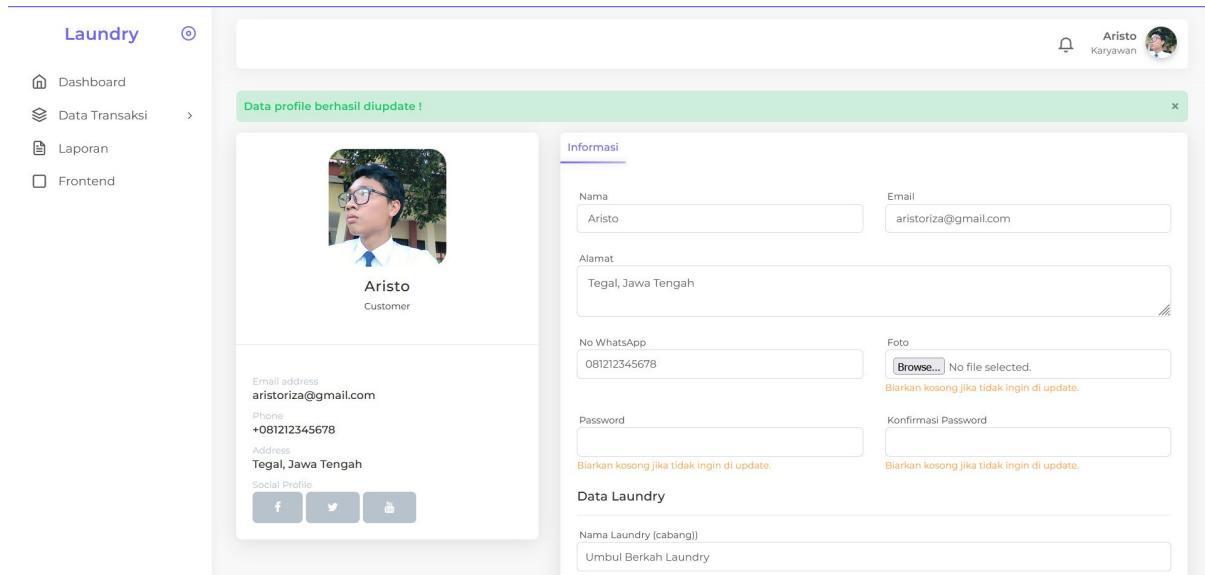


foto tersebut ditampilkan dalam file blade.php melalui code berikut ini


```


</span>

```

Kita tidak menggunakan fitur delete image karena karyawan diharuskan memiliki gambar profil agar mudah dikenali. Karyawan cukup update gambar apabila ingin mengganti dengan gambar yang baru.

Transaksi

Data Customer yang melakukan transaksi di input oleh karyawan. Berikut tampilan transaksi atau tambah order yang dilakukan oleh karyawan

The screenshot shows the 'Laundry' application interface. On the left, there's a sidebar with navigation options: Dashboard, Data Transaksi, Order Masuk, Tambah Order (which is highlighted in blue), Data Customer, Laporan, and Frontend. The main area is titled 'Form Tambah Data Order' and contains fields for 'Nama Customer' (dropdown with placeholder '-- Pilih Customer --'), 'No Transaksi' (input field with value '600922022'), 'Jenis Pembayaran' (dropdown with placeholder '-- Pilih Jenis Pembayaran --'), 'Status Pembayaran' (dropdown with placeholder '-- Pilih Status Payment --'), 'Berat Pakaian' (input field), 'Pilih Pakaian' (dropdown with placeholder '-- Jenis Pakaian --'), 'Pilih Hari' (input field), 'Harga' (input field), 'Disc' (input field with value '0'), 'Total Berat Pakaian' (input field with value '0'), 'Lama Hari' (input field with value '0'), 'Total Harga' (input field with value '0'), 'Total Disc' (input field with value '0'), and 'Total' (input field with value '0'). At the bottom are 'Tambah' and 'Reset' buttons. The footer includes copyright information 'COPYRIGHT © 2022 Laundry Umbul Berkah, All rights Reserved' and a 'Build With ❤️' link.

Pertama, karyawan memilih nama customer. Apabila customer belum terdaftar, maka karyawan dapat memilih menu +Customer Baru yang akan mengarahkan ke halaman tambah data customer. Setelah dipilih nama dari customer yang memesan, karyawan menentukan jenis pembayaran apakah tunai atau transfer dan status pembayaran apakah “belum dibayar” atau “sudah dibayar”. Setelahnya, karyawan menentukan berat pakaian dan jenis pakaian, pilih hari dan harga akan otomatis ter-generate karena sudah ditentukan oleh admin. lalu karyawan menginputkan jumlah diskon yang didapat oleh customer. Nomor transaksi akan otomatis ditambahkan oleh sistem, sehingga karyawan tinggal pilih Tambah dan transaksi akan otomatis ditambahkan.

This part of the image shows expanded dropdown menus from the previous screenshot:

- Jenis Pembayaran**: Shows two options: "Tunai" and "Transfer".
- Status Pembayaran**: Shows two options: "Belum Dibayar" and "Sudah Dibayar".
- Pilih Pakaian**: Shows three service types: "Cuci Setrika", "Cuci Kering", and "Setrika".

Berikut tampilan ketika karyawan telah mengisi data-data yang diperlukan

Form Tambah Data Order [+ Customer Baru](#)

Nama Customer Alex	No Transaksi 600922022	Jenis Pembayaran Tunai	Status Pembayaran Belum Dibayar
Berat Pakaian 3	Pilih Pakaian Cuci Setrika	Pilih Hari 3	Harga Rp. 7.000
Total Berat Pakaian 3	Lama Hari 3	Total Harga 21000	Disc 0
		Total Disc 0	Total 21000

Tambah **Reset**

terdapat fitur (+) yang digunakan untuk menambah data order yang total harganya akan diakumulasikan seluruhnya menjadi satu transaksi.

Form Tambah Data Order [+ Customer Baru](#)

Nama Customer Alex	No Transaksi 600922022	Jenis Pembayaran Tunai	Status Pembayaran Belum Dibayar
Berat Pakaian 3	Pilih Pakaian Cuci Setrika	Pilih Hari 3	Harga Rp. 7.000
Berat Pakaian 4	Pilih Pakaian Setrika	Pilih Hari 1	Harga Rp. 3.000
Total Berat Pakaian 7	Lama Hari 3	Total Harga 33000	Disc 10
		Total Disc 1200	Total 31800

Tambah **Reset**

setelah semua data lengkap maka tekan button Tambah untuk mendaftarkan order yang dibuat.

Berikut adalah tampilan website setelah order berhasil dibuat

The screenshot shows a web-based laundry management system. On the left, there's a sidebar with navigation links: Dashboard, Data Transaksi, Order Masuk (which is highlighted in purple), Tambah Order, Data Customer, Laporan, and Frontend. At the top right, there's a user profile for 'Aristo Karyawan' with a notification icon. The main content area has a green header bar that says 'Order Berhasil Ditambah !'. Below it, there's a table titled 'Tambah' with columns: #, No Resi, TGL Transaksi, Customer, Status Laundry, Payment, Jenis, Total, and Action. Two rows of data are shown:

#	No Resi	TGL Transaksi	Customer	Status Laundry	Payment	Jenis	Total	Action
1	600922022	03-07-22	Alex	Diproses	Pending	Cuci Setrika	Rp 31.800	<button>Bayar</button> <button>Invoice</button>
2	203522022	27-05-22	Taufik	Diproses	Pending	Cuci Setrika	Rp 17.000	<button>Bayar</button> <button>Invoice</button>

At the bottom of the table, it says 'Showing 1 to 2 of 2 entries'. There are also 'Previous' and 'Next' buttons. The footer of the page includes 'COPYRIGHT © 2022 Laundry Umbul Berkah, All rights Reserved' and 'Build With ❤️'.

terdapat action button “Bayar” ditekan ketika customer telah melakukan pembayaran.

#	No Resi	TGL Transaksi	Customer	Status Laundry	Payment	Jenis	Total	Action
1	600922022	03-07-22	Alex	Diproses	Pending	Cuci Setrika	Rp 31.800	<button>Bayar</button> <button>Invoice</button>

terdapat action button “Selesai” ditekan ketika laundry telah selesai.

#	No Resi	TGL Transaksi	Customer	Status Laundry	Payment	Jenis	Total	Action
1	600922022	03-07-22	Alex	Diproses	Lunas	Cuci Setrika	Rp 31.800	<button>Selesai</button> <button>Invoice</button>

terdapat action button “Diambil” ditekan ketika laundry telah diambil oleh customer.

#	No Resi	TGL Transaksi	Customer	Status Laundry	Payment	Jenis	Total	Action
1	600922022	03-07-22	Alex	Selesai	Lunas	Cuci Setrika	Rp 31.800	<button>Diambil</button> <button>Invoice</button>

Ketika semua action telah semua dilakukan maka transaksi yang dilakukan oleh customer Alex direcord oleh sistem database.

#	No Resi	TGL Transaksi	Customer	Status Laundry	Payment	Jenis	Total	Action
1	600922022	03-07-22	Alex	Diambil	Lunas	Cuci Setrika	Rp 31.800	<button>Invoice</button>

Berikut adalah tampilan dashboard yang telah ditambah data order atau data transaksi.

The screenshot shows a laundry management system dashboard. On the left, a sidebar menu includes 'Dashboard' (selected), 'Data Transaksi', 'Laporan', and 'Frontend'. The main area has a header 'Laundry' and a profile for 'Aristo Karyawan'. A 'Welcome' message says 'Semoga harimu menyenangkan.' Below it is the date 'Sunday, 03 July 2022, 14:52:50'. A 'Statistics' section shows 5 Customers, 2 Laundry Masuk, 0 Laundry Selesai, and 1 Laundry Dambil, all updated '1 month ago'. Below this are two cards: 'Hari ini' (7 kg Kilogram) and 'Kemarin' (0 kg Kilogram). A 'Pendapatan' card shows 'Bulan Ini Rp 31.800' with a note: 'Pendapatan naik 100% dari bulan kemarin.' To the right is a chart titled 'Data Laundry Masuk Per-bulan' showing a single peak of 1 unit in July. At the bottom, a copyright notice reads 'COPYRIGHT © 2022 Laundry Umbul Berkah, All rights Reserved' and a 'Build With ❤️' link.

Dan ini adalah tampilan dashboard yang data transaksinya telah direcord oleh database yang terintegrasi antara karyawan dengan admin.

The screenshot shows a laundry management system dashboard for an administrator. The sidebar menu includes 'Dashboard' (selected), 'Data User', 'Transaksi', 'Data Finance', and 'Other'. The main area has a header 'Laundry' and a profile for 'Administrator Admin'. A 'Statistics' section shows 5 Jumlah Customer, 2 Laundry Masuk, 0 Laundry Selesai, and 1 Laundry Dambil. Below this are two 'Pendapatan' cards: 'Tahun ini Rp 31.800', 'Bulan ini Rp 31.800', and 'Tahun lalu Rp 31.800'. To the right are two more 'Pendapatan' cards: 'Hari ini Rp 31.800' and 'Kemarin Rp 0'. Below these are two charts: 'Data Laundry Masuk Per-hari' showing a single peak of 1 unit on July 3rd, and 'Data Laundry Masuk Per-bulan' showing a single peak of 1 unit in July. The x-axis for the daily chart ranges from 1 to 31, and for the monthly chart from January to December.