LAPORAN PRAKTIKUM
Pengembagan Aplikasi Berbasis Internet

Pembuatan Website Menggunakan Laravel

1. Dealova Zevanya Manurung 13323045
2. Tarorom Joseph Abel Gurning 13323014
3. Jose Rafael Lumban Raja 13323002
4. Idvend Manurung 13323011
5. Rejeki Adi putra lumban batu 13323038

DIII Teknologi Komputer

INSTITUT TEKNOLOGI DEL
FAKULTAS VOKASI
2023/2024

Bagian Codingan:

Bagian Dari env:

```
APP_NAME=Shopi
APP_ENV=local
APP_KEY=base64:lyVK9DXa+dkx6zRj6TwSmiHILSxyjf8odd5EBY8WB/k=
APP_DEBUG=true
APP_URL=http://localhost:8000

LOG_CHANNEL=stack
LOG_DEPRECATIONS_CHANNEL=null
LOG_LEVEL=debug

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=4308
DB_DATABASE=uwaz
DB_USERNAME=root
DB_PASSWORD=

BROADCAST_DRIVER=log
CACHE_DRIVER=file
FILESYSTEM_DISK=local
QUEUE_CONNECTION=sync
SESSION_DRIVER=file
SESSION_LIFETIME=120

MEMCACHED_HOST=127.0.0.1

REDIS_HOST=127.0.0.1
REDIS_PASSWORD=null
REDIS_PORT=6379

MAIL_MAILER=smtp
MAIL_HOST=smtp.googlemail.com
MAIL_PORT=465
MAIL_USERNAME=YOURKEYS@gmail.com
MAIL_PASSWORD=YOURKEYS
MAIL_ENCRYPTION=null
MAIL_FROM_ADDRESS=contact@shopi.com
MAIL_FROM_NAME="Shopi"

AWS_ACCESS_KEY_ID=
AWS_SECRET_ACCESS_KEY=
```

```
AWS_DEFAULT_REGION=us-east-1
AWS_BUCKET=
AWS_USE_PATH_STYLE_ENDPOINT=false

PUSHER_APP_ID=
PUSHER_APP_KEY=
PUSHER_APP_SECRET=
PUSHER_HOST=
PUSHER_PORT=443
PUSHER_SCHEME=https
PUSHER_APP_CLUSTER=mt1

VITE_PUSHER_APP_KEY="${PUSHER_APP_KEY}"
VITE_PUSHER_HOST="${PUSHER_HOST}"
VITE_PUSHER_PORT="${PUSHER_PORT}"
VITE_PUSHER_SCHEME="${PUSHER_SCHEME}"
VITE_PUSHER_APP_CLUSTER="${PUSHER_APP_CLUSTER}"

STRIPE_KEY="YOURKEYS"
STRIPE_SECRET="YOURKEYS"
STRIPE_WEBHOOK_SECRET=your-stripe-webhook-secret
```

Penjelasan:
File.env adalah file konfigurasi untuk aplikasi Laravel bernama "Shopi". Berikut adalah penjelasan singkat untuk setiap bagian dari file tersebut:

Informasi Aplikasi:
APP_NAME: Nama aplikasi, dalam hal ini "Shopi".
APP_ENV: Lingkungan aplikasi, di sini adalah "local" untuk pengembangan lokal.
APP_KEY: Kunci aplikasi yang digunakan untuk enkripsi.
APP_DEBUG: Menunjukkan apakah mode debug diaktifkan (true).
APP_URL: URL aplikasi, di sini adalah http://localhost:8000.

Log:
LOG_CHANNEL: Saluran logging, di sini adalah "stack".
LOG_DEPRECATIONS_CHANNEL: Saluran logging untuk deprecations, tidak ada yang ditetapkan (null).
LOG_LEVEL: Level logging, di sini adalah "debug".

Database:
DB_CONNECTION: Jenis koneksi database, di sini adalah "mysql".
DB_HOST: Host database, di sini adalah 127.0.0.1.
DB_PORT: Port database, di sini adalah 4308.
DB_DATABASE: Nama database, di sini adalah "uwaz".
DB_USERNAME: Username database, di sini adalah "root".
DB_PASSWORD: Password database, di sini kosong.

Cache dan Queue:
BROADCAST_DRIVER: Driver broadcasting, di sini adalah "log".
CACHE_DRIVER: Driver cache, di sini adalah "file".
FILESYSTEM_DISK: Disk filesystem yang digunakan, di sini adalah "local".
QUEUE_CONNECTION: Koneksi queue, di sini adalah "sync".
SESSION_DRIVER: Driver session, di sini adalah "file".
SESSION_LIFETIME: Waktu hidup session dalam menit, di sini adalah 120.
Memcached dan Redis
MEMCACHED_HOST: Host untuk Memcached, di sini adalah 127.0.0.1.
REDIS_HOST: Host untuk Redis, di sini adalah 127.0.0.1.
REDIS_PASSWORD: Password untuk Redis, di sini kosong (null).
REDIS_PORT: Port untuk Redis, di sini adalah 6379.

Mail:
MAIL_MAILER: Mailer yang digunakan, di sini adalah "smtp".
MAIL_HOST: Host SMTP, di sini adalah smtp.googlemail.com.
MAIL_PORT: Port SMTP, di sini adalah 465.
MAIL_USERNAME: Username untuk SMTP, di sini adalah YOURKEYS@gmail.com.
MAIL_PASSWORD: Password untuk SMTP, di sini adalah YOURKEYS.
MAIL_ENCRYPTION: Metode enkripsi untuk SMTP, di sini adalah null.
MAIL_FROM_ADDRESS: Alamat email pengirim default, di sini adalah contact@shopi.com.
MAIL_FROM_NAME: Nama pengirim default, di sini adalah "Shopi".

AWS:
AWS_ACCESS_KEY_ID: ID kunci akses AWS.
AWS_SECRET_ACCESS_KEY: Kunci rahasia AWS.
AWS_DEFAULT_REGION: Region default AWS, di sini adalah us-east-1.
AWS_BUCKET: Nama bucket AWS.
AWS_USE_PATH_STYLE_ENDPOINT: Penggunaan endpoint gaya path untuk AWS, di sini adalah false.

Pusher:
PUSHER_APP_ID: ID aplikasi Pusher.
PUSHER_APP_KEY: Kunci aplikasi Pusher.
PUSHER_APP_SECRET: Rahasia aplikasi Pusher.
PUSHER_HOST: Host untuk Pusher.
PUSHER_PORT: Port untuk Pusher, di sini adalah 443.
PUSHER_SCHEME: Skema untuk Pusher, di sini adalah https.
PUSHER_APP_CLUSTER: Cluster aplikasi Pusher, di sini adalah mt1.

Vite (untuk integrasi dengan Pusher):
VITE_PUSHER_APP_KEY: Mengambil nilai dari PUSHER_APP_KEY.
VITE_PUSHER_HOST: Mengambil nilai dari PUSHER_HOST.
VITE_PUSHER_PORT: Mengambil nilai dari PUSHER_PORT.
VITE_PUSHER_SCHEME: Mengambil nilai dari PUSHER_SCHEME.

VITE_PUSHER_APP_CLUSTER: Mengambil nilai dari PUSHER_APP_CLUSTER.

Stripe:
STRIPE_KEY: Kunci API Stripe.
STRIPE_SECRET: Rahasia API Stripe.
STRIPE_WEBHOOK_SECRET: Rahasia webhook Stripe.

File: routes/api.php:

```php
<?php

use Illuminate\Http\Request;
use Illuminate\Support\Facades\Route;

/*
|--------------------------------------------------------------------------
| API Routes
|--------------------------------------------------------------------------
|
| Here is where you can register API routes for your application. These
| routes are loaded by the RouteServiceProvider and all of them will
| be assigned to the "api" middleware group. Make something great!
|
*/

Route::middleware('auth:sanctum')->get('/user', function (Request $request) {
    return $request->user();
});
```

Penjelasan:

- **use Illuminate\Http\Request**: Mengimpor kelas Request dari Laravel yang digunakan untuk menangani permintaan HTTP.
- **use Illuminate\Support\Facades\Route**: Mengimpor fasad Route yang digunakan untuk mendefinisikan rute dalam aplikasi Laravel.
- Komentar ini memberikan konteks bahwa rute-rute yang didefinisikan dalam file ini akan digunakan untuk API dan akan dimuat oleh RouteServiceProvider. Semua rute ini akan menggunakan grup middleware api.
- Route::middleware('auth
- '): Menentukan bahwa rute ini menggunakan middleware auth:sanctum, yang berarti rute ini memerlukan autentikasi menggunakan Laravel Sanctum.
- ->get('/user', function (Request $request) { ... }): Mendefinisikan rute dengan metode HTTP GET ke endpoint /user. Ketika endpoint ini diakses, fungsi anonim akan dijalankan.
- function (Request $request): Fungsi anonim yang menerima objek Request.

- return $request->user();: Mengembalikan informasi pengguna yang diautentikasi. Ini akan mengembalikan objek User yang sedang login berdasarkan token autentikasi yang dikirimkan dalam permintaan.

File: app/Console/Kernel.php:

```php
<?php

namespace App\Console;

use Illuminate\Console\Scheduling\Schedule;
use Illuminate\Foundation\Console\Kernel as ConsoleKernel;

class Kernel extends ConsoleKernel
{
    /**
     * Define the application's command schedule.
     */
    protected function schedule(Schedule $schedule): void
    {
        // $schedule->command('inspire')->hourly();
    }

    /**
     * Register the commands for the application.
     */
    protected function commands(): void
    {
        $this->load(__DIR__.'/Commands');

        require base_path('routes/console.php');
    }
}
```

Penjelasan:

- namespace App\Console: Menentukan namespace untuk kelas Kernel, yaitu App\Console.
- use Illuminate\Console\Scheduling\Schedule: Mengimpor kelas Schedule dari Laravel yang digunakan untuk menjadwalkan tugas.
- use Illuminate\Foundation\Console\Kernel as ConsoleKernel: Mengimpor kelas ConsoleKernel dari Laravel dan mengaliasinya sebagai ConsoleKernel.
- protected function schedule(Schedule $schedule): void: Metode yang digunakan untuk mendefinisikan jadwal perintah aplikasi.
- $schedule->command('inspire')->hourly();: Contoh perintah yang dijadwalkan untuk dijalankan setiap jam. Ini dikomentari (non-aktif) secara default. Anda dapat menambahkan tugas terjadwal lainnya di sini, misalnya mengirim email setiap hari, membersihkan database, dll.

- Metode commands
- protected function commands(): void: Metode yang digunakan untuk mendaftarkan perintah konsol aplikasi.
- $this->load(DIR.'/Commands');: Memuat semua perintah yang didefinisikan di direktori App\Console\Commands.
- require base_path('routes/console.php');: Mengimpor file console.php yang terletak di direktori routes, tempat Anda dapat mendefinisikan perintah konsol yang berbasis closure.

File: routes/web.php:

```
use App\Http\Controllers\InvoiceController;
use App\Http\Controllers\ProfileController;
use App\Http\Livewire\Cart;
use App\Http\Livewire\Checkout;
use App\Http\Livewire\Dashboard;
use App\Http\Livewire\Home;
use App\Http\Livewire\ProductDetails;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Route;
```

- use ...: Mengimpor berbagai controller dan Livewire components yang akan digunakan untuk mendefinisikan rute.

```
/*
|--------------------------------------------------------------------------
| Web Routes
|--------------------------------------------------------------------------
|
| Here is where you can register web routes for your application. These
| routes are loaded by the RouteServiceProvider and all of them will
| be assigned to the "web" middleware group. Make something great!
|
*/
```

- Menjelaskan bahwa file ini digunakan untuk mendefinisikan rute web yang akan dimuat oleh RouteServiceProvider dan akan menggunakan grup middleware web.

```
Route::get('/', [Home::class, 'render'])->name('home');
```

- Mendefinisikan rute GET untuk URL root (/), yang akan memanggil metode render dari komponen Livewire Home. Rute ini diberi nama home.

```

Route::get('/product/{product_id}', [ProductDetails::class, 'render'])->name('product.details');
```

- Mendefinisikan rute GET untuk URL /product/{product_id}, yang akan memanggil metode render dari komponen Livewire ProductDetails. Rute ini diberi nama product.details.

```
Route::post('/add-to-cart', [Cart::class, 'addToCart'])->name('cart.add');
Route::post('/inc-qty', [Cart::class, 'incQty'])->name('qty.up');
Route::post('/dec-qty', [Cart::class, 'decQty'])->name('qty.down');
Route::delete('/destroy-item', [Cart::class, 'destroyItem'])->name('destroy.item');
Route::delete('/destroy-cart', [Cart::class, 'destroyCart'])->name('destroy.cart');
Route::get('/cart', [Cart::class, 'render'])->name('cart');
```

- Mendefinisikan berbagai rute untuk operasi terkait keranjang belanja, seperti menambah ke keranjang, meningkatkan dan mengurangi kuantitas, menghapus item, mengosongkan keranjang, dan menampilkan keranjang. Semua rute ini menggunakan komponen Livewire Cart.

```
Route::middleware('auth')->group(function () {
    Route::get('/checkout', [Checkout::class, 'render'])->name('checkout');
    Route::post('/checkout-order', [Checkout::class, 'makeOrder'])->name('checkout.order');
    Route::get('/checkout-success', [Checkout::class, 'success'])->name('checkout.success');
    Route::get('/checkout-cancel', [Checkout::class, 'cancel'])->name('checkout.cancel');
    Route::get('/profile', [ProfileController::class, 'edit'])->name('profile.edit');
    Route::patch('/profile', [ProfileController::class, 'update'])->name('profile.update');
    Route::delete('/profile', [ProfileController::class, 'destroy'])->name('profile.destroy');

    Route::get('/dashboard/invoice/{order}', [Dashboard::class, 'invoice'])->name('invoice');
    Route::get('/dashboard/invoice/pdf/{order}', [Dashboard::class, 'invoicePdf'])->name('invoice.pdf');
    Route::get('/dashboard', [Dashboard::class, 'render'])->name('dashboard');
});
```

- Rute yang berada di dalam grup middleware auth memerlukan autentikasi.
- Rute checkout: untuk menampilkan halaman checkout, membuat pesanan, menampilkan halaman sukses, dan halaman pembatalan.
- Rute profil: untuk mengedit, memperbarui, dan menghapus profil pengguna.
- Rute dashboard: untuk menampilkan faktur (invoice), mengunduh faktur sebagai PDF, dan menampilkan dashboard.

```
require __DIR__ . '/auth.php';
```

- Mengimpor rute autentikasi dari file auth.php.

```
<?php

use Illuminate\Foundation\Inspiring;
use Illuminate\Support\Facades\Artisan;

/*
```

```
|--------------------------------------------------------------------------
| Console Routes
|--------------------------------------------------------------------------
|
| This file is where you may define all of your Closure based console
| commands. Each Closure is bound to a command instance allowing a
| simple approach to interacting with each command's IO methods.
|
*/

Artisan::command('inspire', function () {
    $this->comment(Inspiring::quote());
})->purpose('Display an inspiring quote');
```

File: routes/console.php:

- use Illuminate\Foundation\Inspiring: Mengimpor kelas Inspiring dari Laravel yang menyediakan kutipan inspirasional.
- use Illuminate\Support\Facades\Artisan: Mengimpor fasad Artisan yang digunakan untuk mendefinisikan dan menjalankan perintah Artisan.
- Menjelaskan bahwa file ini digunakan untuk mendefinisikan perintah konsol berbasis closure. Setiap closure akan terikat pada instance perintah, memungkinkan interaksi sederhana dengan metode IO (Input/Output) dari perintah tersebut.
- Artisan::command('inspire', function () { ... }): Mendefinisikan perintah konsol baru dengan nama inspire. Ketika perintah ini dijalankan, fungsi anonim (closure) akan dipanggil.
- $this->comment(Inspiring::quote());: Di dalam closure, metode comment digunakan untuk mencetak pesan ke konsol. Inspiring::quote() mengambil kutipan inspirasional secara acak dari kelas Inspiring dan menampilkannya di konsol.
- ->purpose('Display an inspiring quote'): Menentukan tujuan dari perintah ini, yaitu "Display an inspiring quote" (Menampilkan kutipan inspirasional). Ini digunakan untuk dokumentasi dan informasi ketika daftar perintah Artisan ditampilkan.

```
<?php

use Illuminate\Support\Facades\Broadcast;

/*
|--------------------------------------------------------------------------
| Broadcast Channels
|--------------------------------------------------------------------------
|
| Here you may register all of the event broadcasting channels that your
| application supports. The given channel authorization callbacks are
| used to check if an authenticated user can listen to the channel.
|
*/
```

```php
Broadcast::channel('App.Models.User.{id}', function ($user, $id) {
    return (int) $user->id === (int) $id;
});
```

Penjelasan:

- Definisi Saluran Siaran: Mendaftarkan saluran siaran App.Models.User.{id} yang menggunakan callback otorisasi untuk memeriksa apakah pengguna yang diautentikasi dapat mendengarkan saluran tersebut.
- Callback Otorisasi: Memeriksa apakah ID pengguna yang diautentikasi sama dengan ID yang diminta untuk saluran tersebut. Jika cocok, pengguna diizinkan untuk mendengarkan saluran ini

File: routes/auth.php:

```php
<?php

use App\Http\Controllers\Auth\AuthenticatedSessionController;
use App\Http\Controllers\Auth\ConfirmablePasswordController;
use App\Http\Controllers\Auth\EmailVerificationNotificationController;
use App\Http\Controllers\Auth\EmailVerificationPromptController;
use App\Http\Controllers\Auth\NewPasswordController;
use App\Http\Controllers\Auth\PasswordController;
use App\Http\Controllers\Auth\PasswordResetLinkController;
use App\Http\Controllers\Auth\RegisteredUserController;
use App\Http\Controllers\Auth\VerifyEmailController;
use Illuminate\Support\Facades\Route;

Route::middleware('guest')->group(function () {
    Route::get('register', [RegisteredUserController::class, 'create'])
                ->name('register');

    Route::post('register', [RegisteredUserController::class, 'store']);

    Route::get('login', [AuthenticatedSessionController::class, 'create'])
                ->name('login');

    Route::post('login', [AuthenticatedSessionController::class, 'store']);

    Route::get('forgot-password', [PasswordResetLinkController::class, 'create'])
                ->name('password.request');

    Route::post('forgot-password', [PasswordResetLinkController::class, 'store'])
                ->name('password.email');

    Route::get('reset-password/{token}', [NewPasswordController::class, 'create'])
                ->name('password.reset');
```

```
    Route::post('reset-password', [NewPasswordController::class, 'store'])
                ->name('password.store');
});

Route::middleware('auth')->group(function () {
    Route::get('verify-email', EmailVerificationPromptController::class)
                ->name('verification.notice');

    Route::get('verify-email/{id}/{hash}', VerifyEmailController::class)
                ->middleware(['signed', 'throttle:6,1'])
                ->name('verification.verify');

    Route::post('email/verification-notification', [EmailVerificationNotificationController::class, 'store'])
                ->middleware('throttle:6,1')
                ->name('verification.send');

    Route::get('confirm-password', [ConfirmablePasswordController::class, 'show'])
                ->name('password.confirm');

    Route::post('confirm-password', [ConfirmablePasswordController::class, 'store']);

    Route::put('password', [PasswordController::class, 'update'])->name('password.update');

    Route::post('logout', [AuthenticatedSessionController::class, 'destroy'])
                ->name('logout');
});
```

Penjelasan:

 Rute untuk Tamu: Mengatur proses registrasi, login, dan pengaturan ulang kata sandi.
 Rute untuk Pengguna yang Diautentikasi: Mengatur verifikasi email, konfirmasi kata sandi, pembaruan kata sandi, dan logout.

File: app/Console/Kernel.php:

```
<?php

namespace App\Console;

use Illuminate\Console\Scheduling\Schedule;
use Illuminate\Foundation\Console\Kernel as ConsoleKernel;

class Kernel extends ConsoleKernel
{
    /**
     * Define the application's command schedule.
     */
    protected function schedule(Schedule $schedule): void
    {
```

```
        // $schedule->command('inspire')->hourly();
    }

    /**
     * Register the commands for the application.
     */
    protected function commands(): void
    {
        $this->load(__DIR__.'/Commands');

        require base_path('routes/console.php');
    }
}
```

Penjelasan:

- Penjadwalan Tugas: Metode schedule mendefinisikan penjadwalan tugas yang akan dijalankan secara otomatis oleh Laravel.
- Pendaftaran Perintah Konsol: Metode commands digunakan untuk memuat perintah konsol dari direktori Commands dan file routes/console.php.

File: app/Events/OrderStatusChanged.php:

```
<?php

namespace App\Events;

use App\Models\Order;
use Illuminate\Broadcasting\Channel;
use Illuminate\Broadcasting\InteractsWithSockets;
use Illuminate\Broadcasting\PresenceChannel;
use Illuminate\Broadcasting\PrivateChannel;
use Illuminate\Contracts\Broadcasting\ShouldBroadcast;
use Illuminate\Foundation\Events\Dispatchable;
use Illuminate\Queue\SerializesModels;

class OrderStatusChanged
{
    use Dispatchable, InteractsWithSockets, SerializesModels;

    public Order $order;
    /**
     * Create a new event instance.
     */
    public function __construct($order)
    {
        $this->order = $order;
    }
```

```
    /**
     * Get the channels the event should broadcast on.
     *
     * @return array<int, \Illuminate\Broadcasting\Channel>
     */
    public function broadcastOn(): array
    {
        return [
            new PrivateChannel('channel-name'),
        ];
    }
}
```

Penjelasan:

- Event: OrderStatusChanged digunakan untuk menginformasikan bahwa status pesanan telah berubah.
- Broadcasting: Event ini mendukung broadcasting dengan menggunakan saluran privat.
- Trait: Menggunakan trait Dispatchable, InteractsWithSockets, dan SerializesModels untuk memberikan fitur tambahan pada event ini.

File: routes/api.php

```
<?php

use Illuminate\Http\Request;
use Illuminate\Support\Facades\Route;

/*
|--------------------------------------------------------------------------
| API Routes
|--------------------------------------------------------------------------
|
| Here is where you can register API routes for your application. These
| routes are loaded by the RouteServiceProvider and all of them will
| be assigned to the "api" middleware group. Make something great!
|
*/

Route::middleware('auth:sanctum')->get('/user', function (Request $request) {
    return $request->user();
});
```

- File api.php adalah tempat untuk mendefinisikan rute API dalam aplikasi Laravel.
- Rute-rute ini dapat disesuaikan sesuai kebutuhan aplikasi dan middleware dapat ditambahkan untuk mengontrol akses ke rute tersebut.

- Dalam contoh tersebut, terdapat satu rute yang memberikan informasi tentang pengguna yang sedang diautentikasi.

Definisi Rute untuk Pengguna yang Belum Diautentikasi (Guest):

```php
<?php

use App\Http\Controllers\Auth\AuthenticatedSessionController;
use App\Http\Controllers\Auth\ConfirmablePasswordController;
use App\Http\Controllers\Auth\EmailVerificationNotificationController;
use App\Http\Controllers\Auth\EmailVerificationPromptController;
use App\Http\Controllers\Auth\NewPasswordController;
use App\Http\Controllers\Auth\PasswordController;
use App\Http\Controllers\Auth\PasswordResetLinkController;
use App\Http\Controllers\Auth\RegisteredUserController;
use App\Http\Controllers\Auth\VerifyEmailController;
use Illuminate\Support\Facades\Route;

Route::middleware('guest')->group(function () {
    Route::get('register', [RegisteredUserController::class, 'create'])
                ->name('register');

    Route::post('register', [RegisteredUserController::class, 'store']);

    Route::get('login', [AuthenticatedSessionController::class, 'create'])
                ->name('login');

    Route::post('login', [AuthenticatedSessionController::class, 'store']);

    Route::get('forgot-password', [PasswordResetLinkController::class, 'create'])
                ->name('password.request');

    Route::post('forgot-password', [PasswordResetLinkController::class, 'store'])
                ->name('password.email');

    Route::get('reset-password/{token}', [NewPasswordController::class, 'create'])
                ->name('password.reset');

    Route::post('reset-password', [NewPasswordController::class, 'store'])
                ->name('password.store');
});

Route::middleware('auth')->group(function () {
    Route::get('verify-email', EmailVerificationPromptController::class)
                ->name('verification.notice');

    Route::get('verify-email/{id}/{hash}', VerifyEmailController::class)
                ->middleware(['signed', 'throttle:6,1'])
```

```
            ->name('verification.verify');

    Route::post('email/verification-notification', [EmailVerificationNotificationController::class, 'store'])
            ->middleware('throttle:6,1')
            ->name('verification.send');

    Route::get('confirm-password', [ConfirmablePasswordController::class, 'show'])
            ->name('password.confirm');

    Route::post('confirm-password', [ConfirmablePasswordController::class, 'store']);

    Route::put('password', [PasswordController::class, 'update'])->name('password.update');

    Route::post('logout', [AuthenticatedSessionController::class, 'destroy'])
            ->name('logout');
});
```

Penjelasan:

- Kode tersebut mendefinisikan berbagai rute yang diperlukan untuk proses otentikasi pengguna dalam aplikasi Laravel, seperti registrasi, login, reset kata sandi, verifikasi email, dan lainnya.
- Rute-rute tersebut dikelompokkan berdasarkan status otentikasi pengguna, yaitu pengguna yang belum diautentikasi (guest) dan pengguna yang sudah diautentikasi.
- Setiap rute memiliki URL dan pengontrol yang sesuai untuk menangani permintaan yang diberikan.

Definisi Saluran Broadcasting:

```
<?php

use Illuminate\Support\Facades\Broadcast;

/*
|--------------------------------------------------------------------------
| Broadcast Channels
|--------------------------------------------------------------------------
|
| Here you may register all of the event broadcasting channels that your
| application supports. The given channel authorization callbacks are
| used to check if an authenticated user can listen to the channel.
|
*/

Broadcast::channel('App.Models.User.{id}', function ($user, $id) {
    return (int) $user->id === (int) $id;
});
```

Penjelasan:

- Konfigurasi saluran broadcasting ini memungkinkan Anda untuk menentukan izin pengguna yang diautentikasi untuk mendengarkan siaran di saluran tertentu dalam aplikasi Laravel Anda. Dengan cara ini, Anda dapat mengendalikan siapa yang memiliki akses ke data yang disiarkan dalam waktu nyata di aplikasi Anda.

File: routes/console.php:

```php
<?php

use Illuminate\Foundation\Inspiring;
use Illuminate\Support\Facades\Artisan;

/*
|--------------------------------------------------------------------------
| Console Routes
|--------------------------------------------------------------------------
|
| This file is where you may define all of your Closure based console
| commands. Each Closure is bound to a command instance allowing a
| simple approach to interacting with each command's IO methods.
|
*/

Artisan::command('inspire', function () {
    $this->comment(Inspiring::quote());
})->purpose('Display an inspiring quote');
```

Penjelasan:

- File console.php dalam aplikasi Laravel Anda digunakan untuk mendefinisikan perintah konsol berbasis closure. Ini memberikan cara cepat dan mudah untuk menambahkan fungsionalitas khusus ke perintah konsol Anda. Dalam contoh di atas, perintah inspire menampilkan kutipan inspiratif, tetapi Anda dapat menyesuaikan dengan perintah konsol sesuai kebutuhan aplikasi Anda.

File: routes/web.php:

```php
<?php

use App\Http\Controllers\InvoiceController;
use App\Http\Controllers\ProfileController;
use App\Http\Livewire\Cart;
```

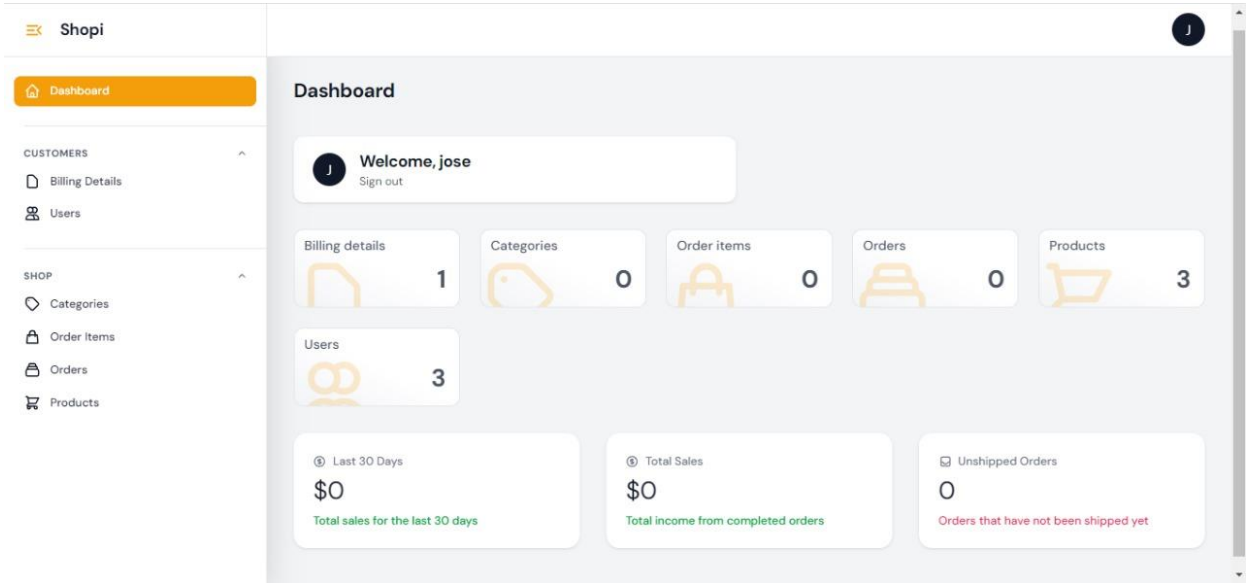
```php
use App\Http\Livewire\Checkout;
use App\Http\Livewire\Dashboard;
use App\Http\Livewire\Home;
use App\Http\Livewire\ProductDetails;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Route;

/*
|--------------------------------------------------------------------------
| Web Routes
|--------------------------------------------------------------------------
|
| Here is where you can register web routes for your application. These
| routes are loaded by the RouteServiceProvider and all of them will
| be assigned to the "web" middleware group. Make something great!
|
*/

Route::get('/', [Home::class, 'render'])->name('home');

Route::get('/product/{product_id}', [ProductDetails::class, 'render'])->name('product.details');

Route::post('/add-to-cart', [Cart::class, 'addToCart'])->name('cart.add');
Route::post('/inc-qty', [Cart::class, 'incQty'])->name('qty.up');
Route::post('/dec-qty', [Cart::class, 'decQty'])->name('qty.down');
Route::delete('/destroy-item', [Cart::class, 'destroyItem'])->name('destroy.item');
Route::delete('/destroy-cart', [Cart::class, 'destroyCart'])->name('destroy.cart');
Route::get('/cart', [Cart::class, 'render'])->name('cart');

Route::middleware('auth')->group(function () {
    Route::get('/checkout', [Checkout::class, 'render'])->name('checkout');
    Route::post('/checkout-order', [Checkout::class, 'makeOrder'])->name('checkout.order');
    Route::get('/checkout-success', [Checkout::class, 'success'])->name('checkout.success');
    Route::get('/checkout-cancel', [Checkout::class, 'cancel'])->name('checkout.cancel');
    Route::get('/profile', [ProfileController::class, 'edit'])->name('profile.edit');
    Route::patch('/profile', [ProfileController::class, 'update'])->name('profile.update');
    Route::delete('/profile', [ProfileController::class, 'destroy'])->name('profile.destroy');

    Route::get('/dashboard/invoice/{order}', [Dashboard::class, 'invoice'])->name('invoice');
    Route::get('/dashboard/invoice/pdf/{order}', [Dashboard::class, 'invoicePdf'])->name('invoice.pdf');
    Route::get('/dashboard', [Dashboard::class, 'render'])->name('dashboard');
});


require __DIR__ . '/auth.php';
```

Penjelasan:

- ile web.php dalam aplikasi Laravel Anda digunakan untuk mendefinisikan rute-rute web yang akan digunakan oleh pengguna.
- Rute-rute ini memetakan URL ke pengontrol dan metode yang sesuai untuk menangani permintaan yang diberikan.
- Penggunaan Route::middleware('auth') memastikan bahwa rute-rute tertentu hanya dapat diakses oleh pengguna yang sudah diautentikasi.

Bagian Website:

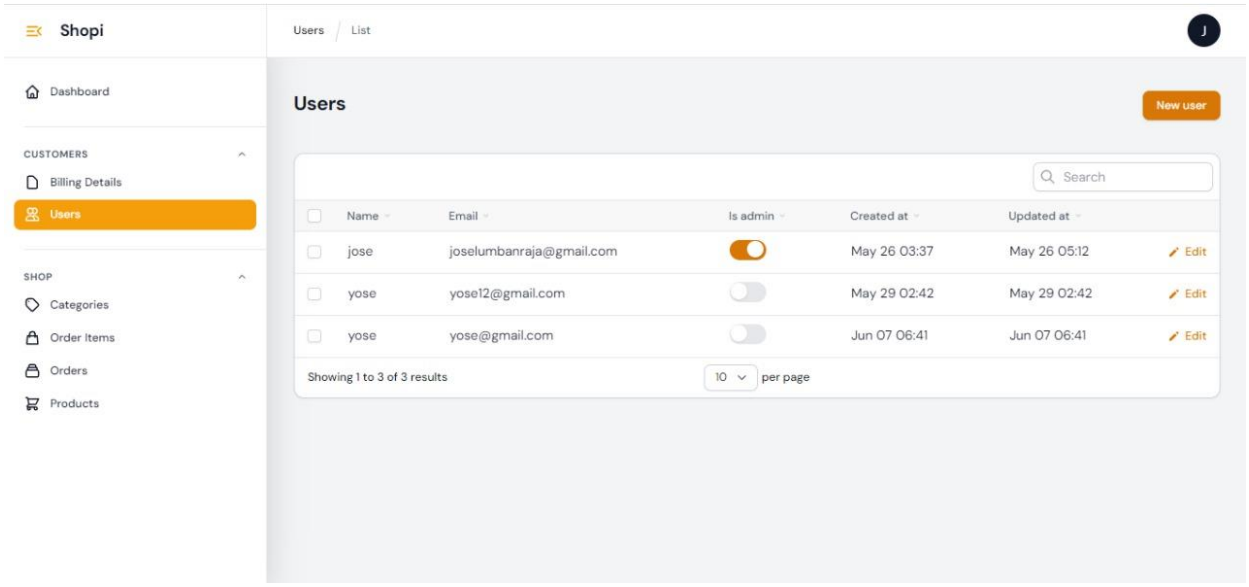
Dashboard:



Penjelasan:

Dashboard adalah halaman yang menyajikan informasi penting atau fungsi administratif, seperti statistik, laporan, atau kontrol. Dalam aplikasi Laravel, Dashboard mungkin merupakan halaman admin yang menampilkan data penting atau menyediakan akses ke fungsi administratif tertentu, seperti manajemen pengguna atau laporan.

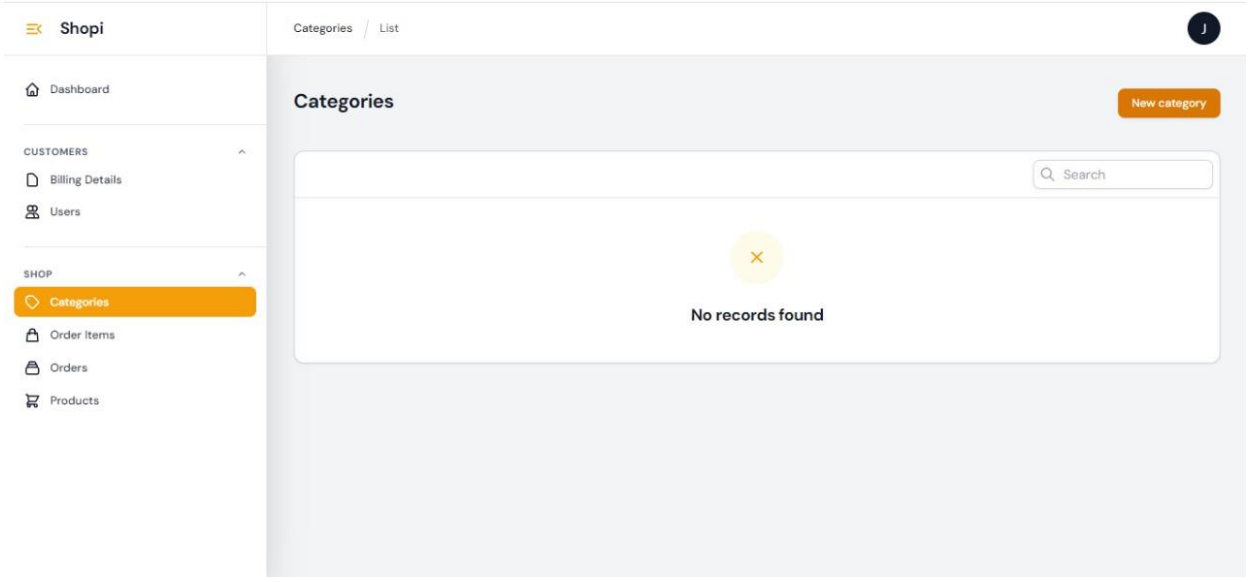
Users



Penjelasan:

users adalah istilah yang digunakan untuk merujuk pada individu atau pengguna yang menggunakan aplikasi. Dalam Laravel, "users" biasanya merujuk pada model Eloquent yang mewakili pengguna aplikasi, serta tabel dalam basis data yang menyimpan informasi tentang pengguna tersebut.

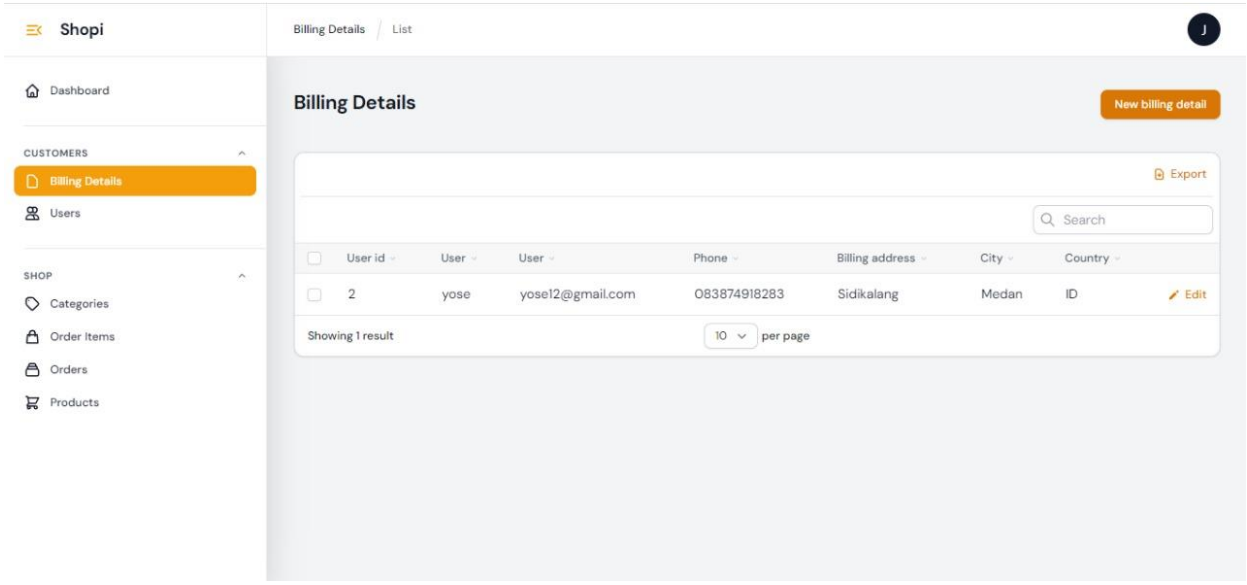
Categories



Penjelasan :

Categoris mungkin merupakan istilah yang kurang umum atau salah eja dari "categories". Dalam konteks umum, "categories" biasanya merujuk pada kumpulan atau klasifikasi berbagai objek atau entitas berdasarkan karakteristik atau atribut tertentu.

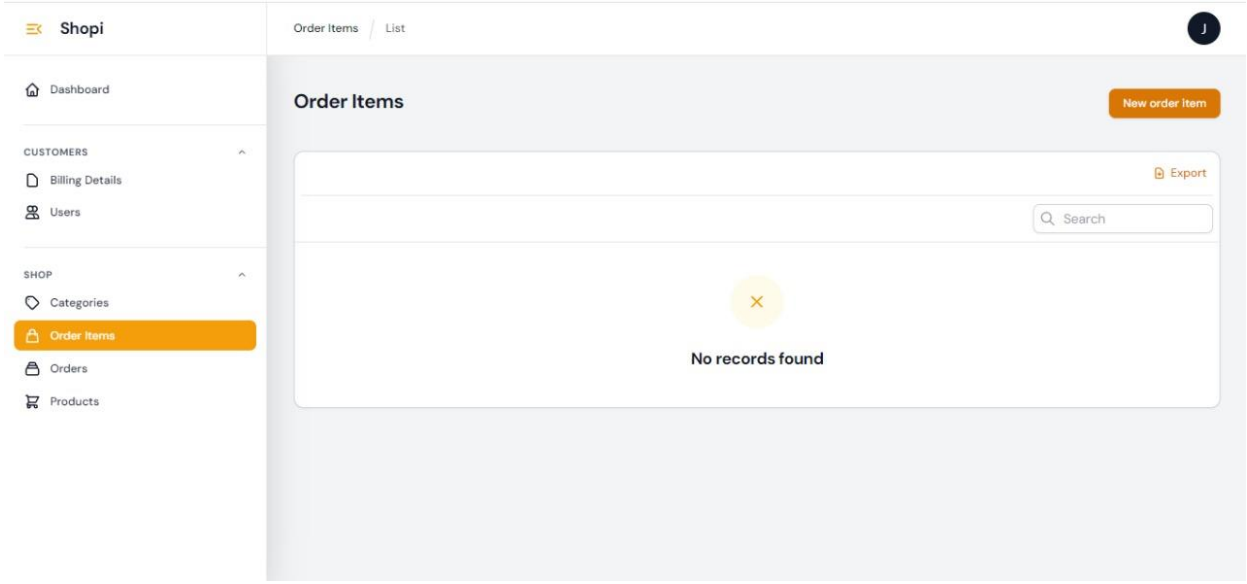
Billing Details



Penjelasan:

Billing Details adalah informasi yang dibutuhkan untuk melakukan pembayaran, seperti nama, alamat, nomor kartu kredit, dan informasi pembayaran lainnya. Informasi ini digunakan untuk memproses transaksi pembayaran atau penagihan dalam layanan atau pembelian.

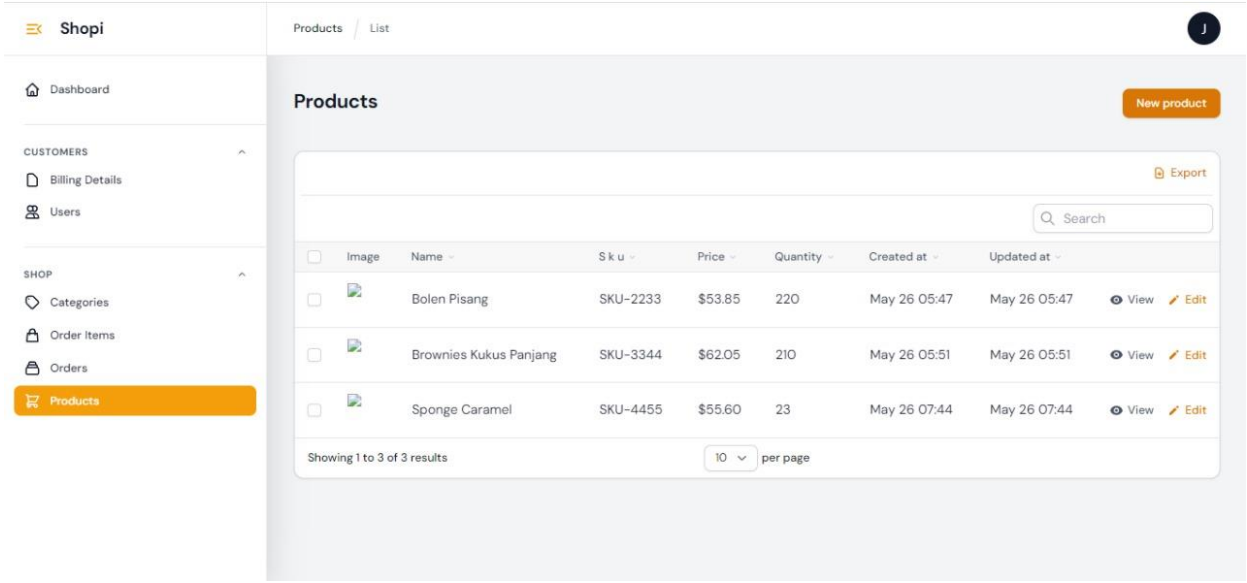
Order Items



Penjelasan:

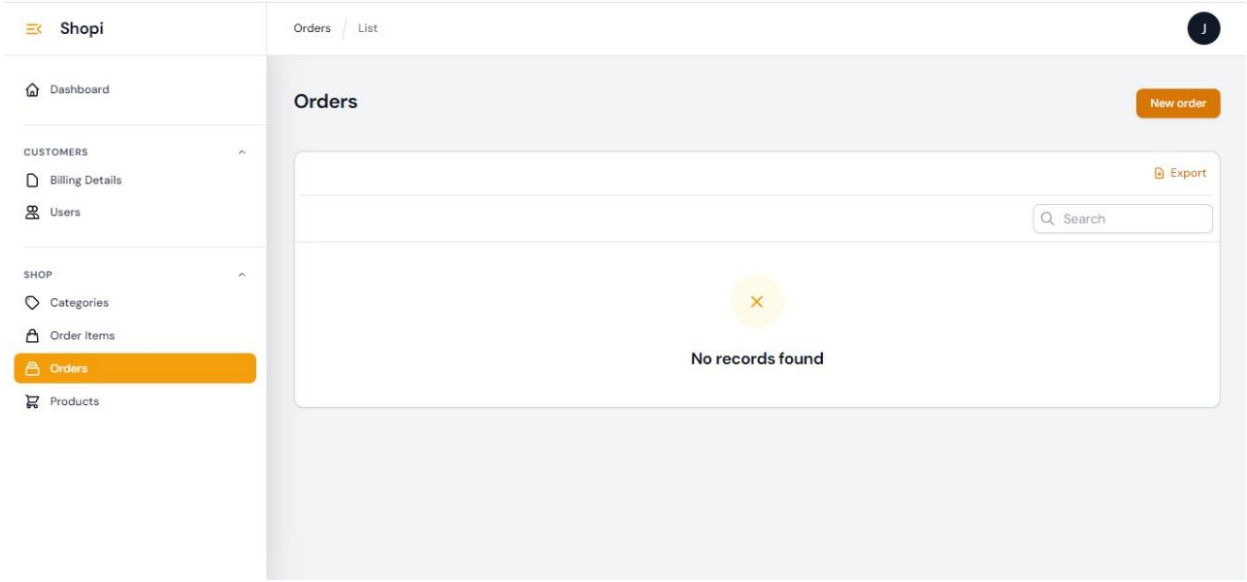
order items adalah barang atau produk yang dipesan oleh pelanggan dalam suatu transaksi pembelian. Ini mencakup detail seperti nama produk, jumlah yang dipesan, dan harga.

Products



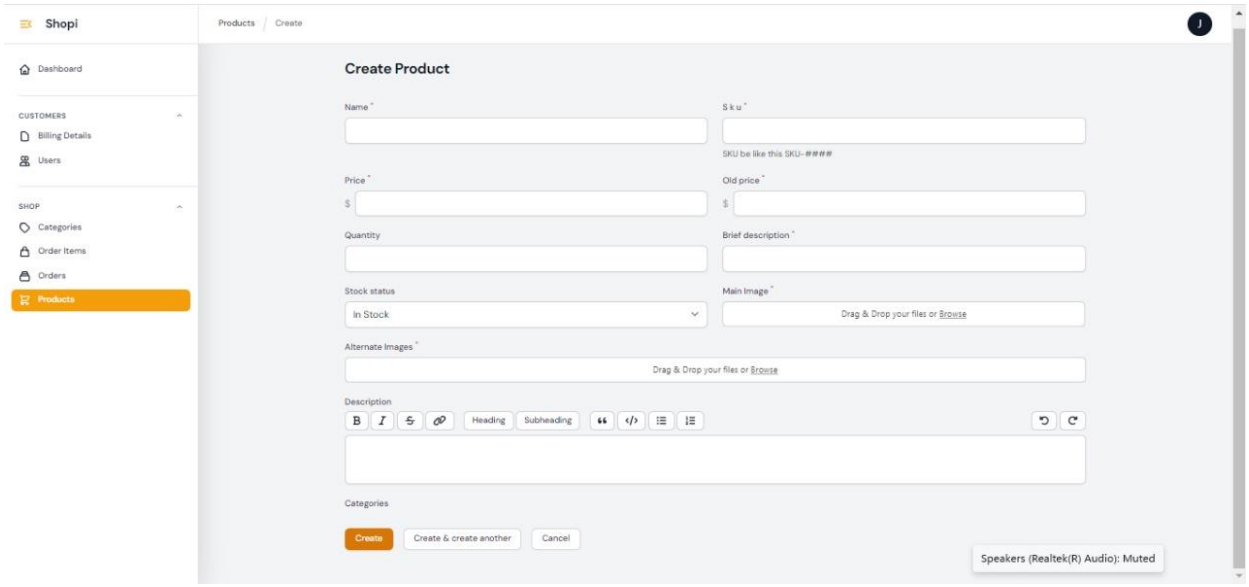
Penjelasan:
Products adalah barang atau produk yang ditawarkan untuk dijual dalam suatu toko atau platform. Ini mencakup detail seperti nama, deskripsi, dan harga.

Orders



Penjelasan:
orders adalah pesanan atau transaksi pembelian yang dilakukan oleh pelanggan dalam suatu toko atau platform. Ini mencakup detail seperti barang yang dibeli, jumlah, dan informasi pembayaran.

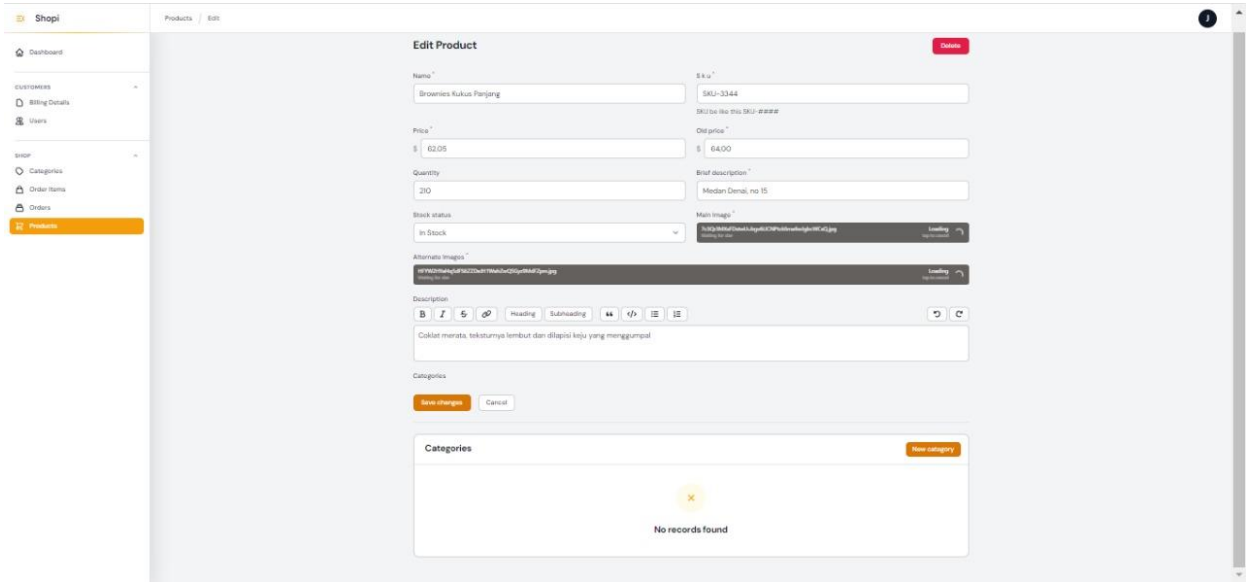
Create Product:



Penjelasan:

Create product adalah istilah yang digunakan untuk menyatakan tindakan membuat atau menambahkan produk baru ke dalam sistem atau platform, seperti toko online atau aplikasi manajemen inventaris. Ini adalah langkah yang dilakukan oleh administrator atau pengguna yang memiliki akses untuk menambahkan entri baru ke dalam katalog produk.

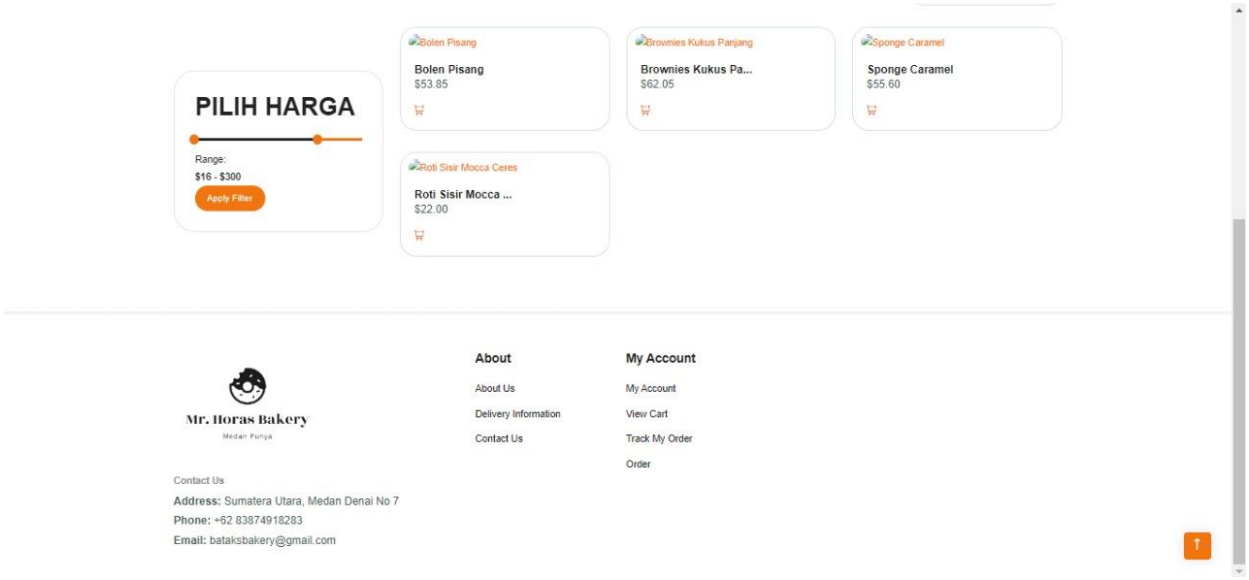
Edit Product



Penjelasan:

Edit product adalah istilah yang digunakan untuk menyatakan tindakan memodifikasi atau mengubah informasi yang terkait dengan suatu produk yang sudah ada dalam sistem atau platform, seperti toko online atau aplikasi manajemen inventaris. Ini dilakukan oleh administrator atau pengguna yang memiliki akses untuk mengubah detail produk yang sudah ada.

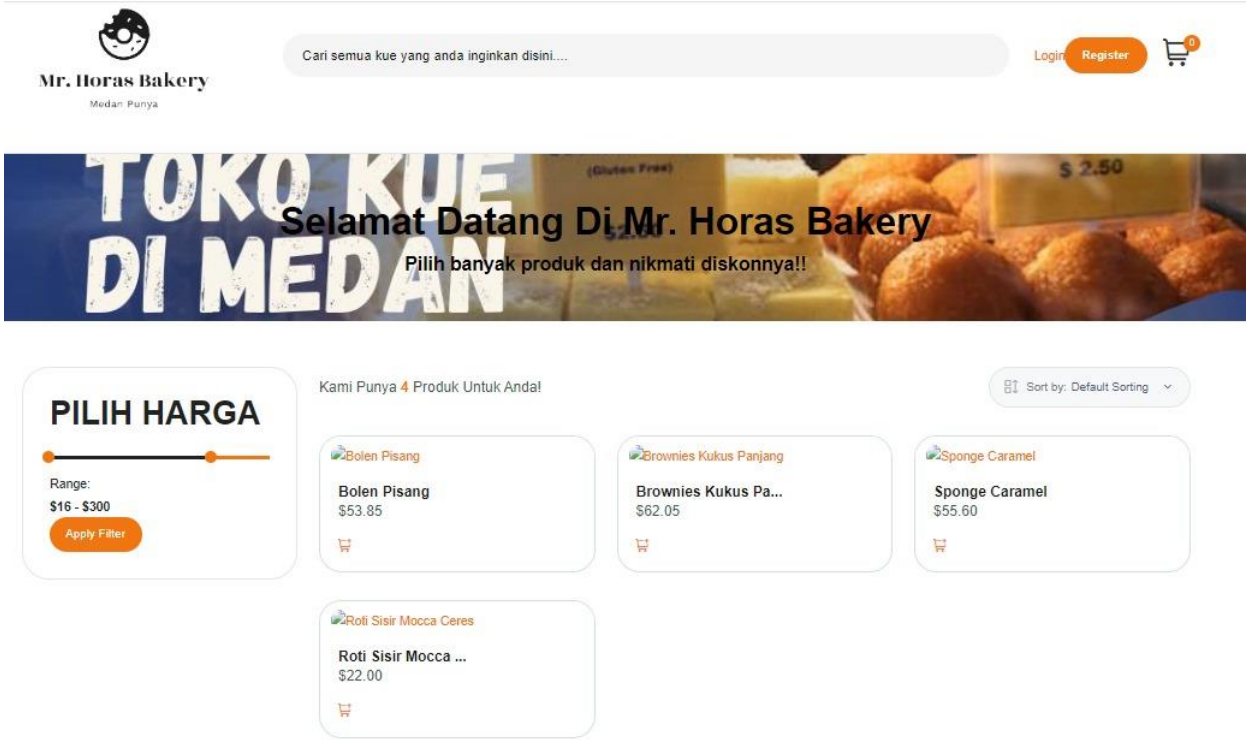
Pilih Harga:



Penjelasan:

Pilih harga adalah istilah yang mungkin digunakan untuk merujuk pada tindakan memilih atau menentukan harga untuk suatu produk atau layanan. Ini terjadi saat seseorang harus membuat keputusan tentang harga yang akan ditetapkan untuk produk yang mereka jual.

Home

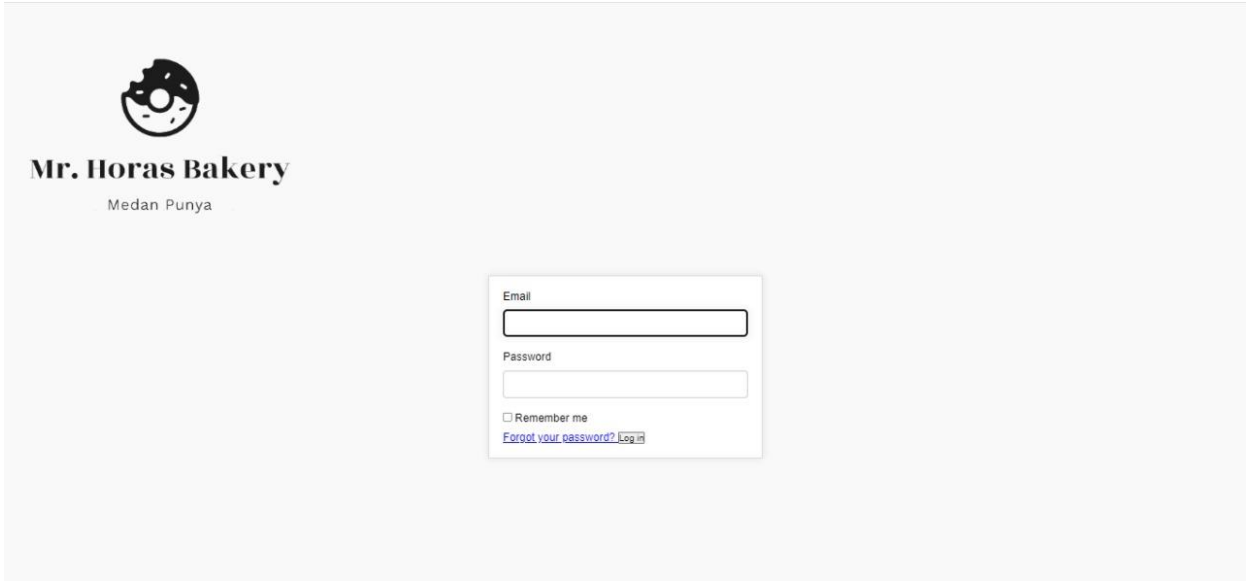


Penjelasan:

Home adalah istilah yang umumnya digunakan dalam konteks aplikasi atau situs web untuk merujuk pada halaman awal atau halaman utama yang pertama kali

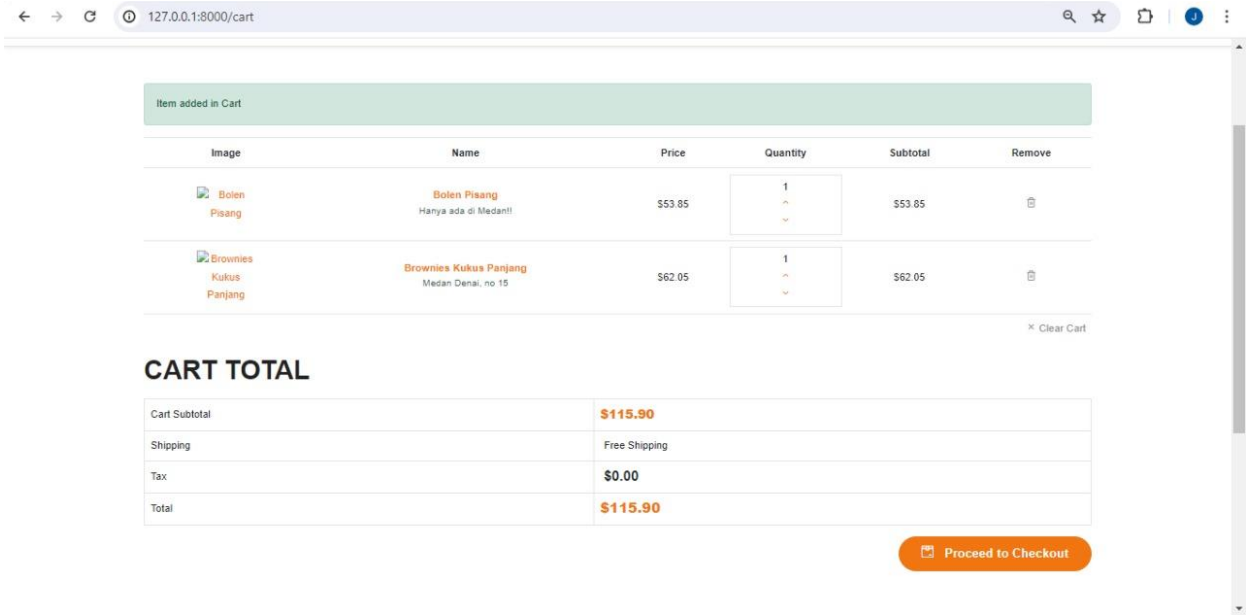
dilihat oleh pengguna ketika mereka mengunjungi aplikasi atau situs tersebut. Halaman "Home" sering kali berfungsi sebagai titik awal bagi pengguna untuk menjelajahi konten atau fitur yang tersedia dalam aplikasi atau situs tersebut.

Login



Penjelasan:
Login adalah istilah yang digunakan dalam konteks aplikasi atau situs web untuk merujuk pada proses autentikasi pengguna. Ini adalah langkah di mana pengguna harus memasukkan kredensial atau informasi identifikasi, seperti nama pengguna dan kata sandi, untuk mengakses akun mereka.

Cart Total



Penjelasan:

Cart total adalah jumlah total biaya dari semua barang atau produk yang telah ditambahkan ke dalam keranjang belanja oleh pengguna dalam suatu aplikasi e-commerce atau situs web toko online.