

فروشگاه خانه سفارشات ورود ثبت نام سیدخريد (0) جستجو جستجو

نام و نام خانوادگی

fatemeh

آدرس ایمیل

khodadadi9124@gmail.com

رمز عبور

.....

تکرار رمز عبور

.....

ثبت نام

فروشگاه خانه سفارشات ورود ثبت نام سیدخريد (0) جستجو جستجو

ورود


ایمیل

khodadadi9124@gmail.com

رمز عبور

.....

ورود





محصولات پرتعداد



لپ تاب لنوو

لپ تاب 15.6 اینچی لنوو مدل IdeaPad L340-R16

3490000 تومان



لپ تاب ایسر

لپ تاب 15.6 اینچی ایسر مدل Nitro 5 AN515-45-R6WC

76422000 تومان



لپ تاب اچ پی

لپ تاب 15.6 اینچی اچ پی مدل Pavilion Gaming 15 DK2044-A

39570000 تومان





برگشت

لپ تاپ اچ پی

قیمت: 39570000 تومان

شاخه: لپ تاپ

توضیحات: اچ پی در پیکره بندی این لپ تاپ از جدیدترین نسل پردازنده مرکزی قدرتمند شرکت اینتل، یعنی مدل Core i7 11370H با حافظه کش 12 مگابایت، پردازشگر گرافیکی GeForce RTX 3050 انویدیا با 4 گیگابایت حافظه ی GDDR6 اختصاصی و 16 گیگابایت رم از نوع DDR4 استفاده کرده است

افزودن به سبد خرید

fatemeh

عالی

fatemeh

عالی





ش پسندیدن پاسخ به اشتراک گذاشتن

خوب

ارسال

Panel content

[ادامه خرید ←](#)

شما 2 کالا در سبد خرید خود دارید		
	39570000 تومان	لپ تاپ اچ پی Pavilion Gaming 15 DK2044-A 
	76422000 تومان	لپ تاپ ایسر Nitro 5 AN515-45-R6WC 
		سفارش

Panel content
Footer

پرداخت

خلاصه سفارش	
تومان 115992000	قیمت
تومان 100	هزینه حمل و نقل
مجموع تومان 115992100	
<div>آدرس خود را وارد کنید</div>	
روش پرداخت	
<input checked="" type="radio"/> پرداخت آنلاین	
<input type="radio"/> پرداخت در محل	
<input type="checkbox"/>	
<input type="checkbox"/>	
شرایط و ضوابط را قبول دارم.	
در باره به روز رسانی ها و رویدادهای محصول ایمیل دریافت کنید. اگر نظر خود را تغییر دادید، می توانید در هر زمانی اشتراک خود را لغو کنید.	
ثبت سفارش	

Panel content

فروشگاه

خانه سفارشات

لیست سفارشات

لپ تاپ اچ پی


وضعیت حمل و نقل:
Pending

وضعیت پرداخت: در انتظار پرداخت

روش پرداخت: Online Payment

آدرس تحویل: کرمان-رفسنجان...


قیمت: 39570000 تومان



لپ تاپ ایسر

وضعیت حمل و نقل:
Pending

وضعیت پرداخت: در انتظار



فروشگاه

خانه سفارشات

لیست سفارشات


Panel content

Start

فروشگاه


خانه سفارشات

نتیجه جستجو




گوشی موبایل شیائومی

گوشی موبایل شیائومی مدل Mi 11
Ultra M2102K1C دو سیم کارت
ظرفیت 512 گیگابایت و رم 12 گیگابایت
34800000 تومان



گوشی موبایل اپل

گوشی موبایل اپل مدل iPhone 13 Pro
Max A2644 دو سیم کارت ظرفیت 256
گیگابایت و رم 6 گیگابایت
55752000 تومان



گوشی موبایل سامسونگ

گوشی موبایل سامسونگ مدل Galaxy
Note 20 Ultra 5G SM-N986 دو سیم
کارت ظرفیت 256 گیگابایت
32700000 تومان

فروشگاه

خانه سفارشات

نتیجه جستجو

گوشی موبایل شیائومی

گوشی موبایل اپل

گوشی موبایل سامسونگ

```

Route::get('/logout', function () {
    Session::forget('user');
    return redirect('login');
});

Auth::routes();

Route::get('/', [ProductController::class, 'index']);

Route::get('/product/{id}', [ProductController::class, 'product']->name('product'));

Route::get('/search', [ProductController::class, 'search']);

Route::post('/add_to_cart', [ProductController::class, 'addToCart']);

Route::get('/cart_list', [ProductController::class, 'cartList']);

Route::get('/remove_cart/{id}', [ProductController::class, 'removeCart']);

Route::get('/order_now', [ProductController::class, 'orderNow']);

Route::post('/order_place', [ProductController::class, 'orderPlace']);

Route::get('/my_order', [ProductController::class, 'myOrder']);

Route::post('/comment', [CommentController::class, 'store']->name('comment'));

```

ابتدا برای صفحه ای که می‌خواهیم بسازیم
route مورد نظر را درست می‌کنیم که
در اینجا تمام Route ها تعریف شده

```

public function __construct()
{
    $this->middleware('auth')->except('index', 'product', 'search', 'removeCart');
}

```

برای اینکه کاربر رو مجبور کنیم که قبل از دسترسی به متد ها حتما لاگین کنه باید از middleware auth استفاده کنیم. در اینجا ما از middleware auth برای تمام متد ها به جز index, product, search و removeCart استفاده کردیم.

```

function index(){
    $data = Product::all();
    return view('product',['products'=>$data]);
}

```

این متد که مربوط به روت / است تمام محصول ها
را فرستادیم به خروجی که در فایل
product.blade.php است.

```
function product($id){
    $product = Product::find($id);
    $comments = Comment::getByProductId($id)->with('user')->get();
    return view('detail', compact('product', 'comments'));
}
```

این متد برای صفحه محصول میباشد، محصول مربوطه با Id خاص خودش رو از جدول products توسط مدل Product دریافت میکنه و داخل متغیر product قرار میدهد. برای اینکه تمام نظرهای این محصول رو دریافت کنیم باید Id این محصول رو به اسکوپ getByProductId پاس بدیم و با relationship مدل user دریافت کنیم و در متغیر comments ذخیره میکنیم و در نهایت به ویو detail میفرستیم به عنوان خروجی.

```
function search(Request $req){
    $data = Product::where('name','like', '%'.$req->input('query').'%')->get();
    return view('search', ['products'=>$data]);
}
```

این متد کاری که انجام میدهد این است که به دنبال محصول هایی میگردد که اسم آنها تشکیل شده از کلمه ای که

داخل فیلد search در header سایت وارد کردیم و نتیجه رو داخل صفحه جداگانه نمایش میده.

```
function addToCart(Request $req){
    $cart = new Cart;
    $cart->user_id = auth()->id();
    $cart->product_id = $req->product_id;
    $cart->save();
    return redirect('/');
}
```

این متد مربوط به زمانی میشود که وقتی داخل صفحه محصول درخواست افزودن به سبد خرید میدهیم که بعد از اضافه کردن رکورد جدید به جدول redirect میشود به صفحه اول

```
function cartList(){
    $user_id = auth()->id();
    $result = DB::table('carts')
    ->join('products', 'carts.product_id', 'products.id')
    ->select('products.*', 'carts.id as cart_id')
    ->where('carts.user_id', $user_id)
    ->get();
    return view('cartlist', ['products'=>$result]);
}
```

متد cartList تمام محصول هایی که کاربر به سبدخرید خود اضافه کرده را براساس ID کاربر از جدول استخراج کرده و داخل صفحه لیست سبدخرید نمایش میدهد

```
function removeCart($id){
    Cart::destroy($id);
    return redirect('/cart_list');
}
```

برای حذف محصول از سبدخرید به متد removeCart نیاز داریم که بعد از درخواستی که از سمت کاربر در صفحه لیست سبدخرید فرستاده میشود با ID مربوطه، آن محصول از سبد حذف میشود.

```
function orderNow(){
    $user_id = auth()->id();
    $result = DB::table('carts')
    ->join('products', 'carts.product_id', 'products.id')
    ->where('carts.user_id', $user_id)
    ->sum('products.price');
    return view('ordernow', ['total'=>$result]);
    // return $result;
}
```

متد orderNow مجموع ارزش محصولات سبدخرید را به عنوان مرحله پیش پرداخت به مشتری یا کاربر نمایش میدهد


```
function orderPlace(Request $req){
    $user_id = auth()->id();
    $allCart = Cart::where('user_id', $user_id)->get();
    foreach($allCart as $cart){
        $order = new Order;
        $order->product_id = $cart['product_id'];
        $order->user_id = $cart['user_id'];
        $order->address = $req->address;
        $order->status = "Pending";
        $order->payment_method = $req->payment;
        $order->payment_status = "در انتظار پرداخت";
        $order->save();
    }
    Cart::where('user_id', $user_id)->delete();
    return redirect('/');
}
```

برای ثبت سفارش کاربر در جدول orders با استفاده از این متد و مدل Order اقدام می کنیم که بعد از ثبت سفارش، سفارش مشتری به وضعیت در انتظار پرداخت تغییر میکند

```
function myOrder(){
    $user_id = auth()->id();
    $result = DB::table('orders')
    ->join('products', 'orders.product_id', 'products.id')
    ->where('orders.user_id', $user_id)
    ->get();
    return view('myorder', ['orders'=>$result]);
    // return $result;
}
```

متد myOrder تمام محصولات که مشتری سفارش داده را از جدول orders براساس ID مشتری یا کاربر استخراج کرده و به مشتری در صفحه لیست سفارشات نمایش میدهد.

```
<?php
```

```
namespace App\Models;
```

```
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
```

```
class Comment extends Model
```

```
{
    use HasFactory;

    protected $fillable = ['user_id', 'product_id', 'content'];

    public function scopeGetByProductId($query, $productId)
    {
        $query->where('product_id', $productId);
    }

    public function user()
    {
        return $this->belongsTo(User::class);
    }
}
```

مدل Comment: property دارد به اسم fillable که به دلایل امنیتی باید مشخص کنیم که مقدار چه فیلدهایی باید در جدول ذخیره بشوند..

این مدل یک متد scope دارد به اسم getByProductId که در کل متدهای scope باعث میشوند که کمتر کد query بزنیم در application و یک متد رابطه داریم به اسم user که با این متد میتوانیم کاربری که کامنتی را نوشته به دست بیاوریم.

برای ساخت جدول users از این مایگریشن استفاده میکنیم
در تمام migration ها از متدهای از پیش تعریف شده استفاده میکنیم.

متد string به فیلد متنی و متد timestamp به فیلد مربوط timestamp درست میکنه

```
public function up()
{
    Schema::create('users', function (Blueprint $table) {
        $table->id();
        $table->string('name');
        $table->string('email')->unique();
        $table->timestamp('email_verified_at')->nullable();
        $table->string('password');
        $table->rememberToken();
        $table->timestamps();
    });
}
```

ساختار جدول products

```
public function up()
{
    Schema::create('products', function (Blueprint $table) {
        $table->id();
        $table->string('name');
        $table->string('price');
        $table->string('category');
        $table->string('gallery');
        $table->string('description');
        $table->string('short_description');
        $table->timestamps();
    });
}
```

با اجرا شدن این migration جدول carts ایجاد میشود.
متدهای integer ی که اینجا استفاده شده به فیلد عددی در جدول درست میکند

```
public function up()
{
    Schema::create('carts', function (Blueprint $table) {
        $table->id();
        $table->integer('product_id');
        $table->integer('user_id');
        $table->timestamps();
    });
}
```

این مایگریشن هم برای ساخت
جدول orders هست

```
public function up()
{
    Schema::create('orders', function (Blueprint $table) {
        $table->id();
        $table->integer('product_id');
        $table->integer('user_id');
        $table->string('address');
        $table->string('status');
        $table->string('payment_method');
        $table->string('payment_status');
        $table->timestamps();
    });
}
```

```

public function up()
{
    Schema::create('comments', function (Blueprint $table) {
        $table->id();
        $table->foreignId('user_id')->constrained();
        $table->string('product_id');
        $table->string('content');
        $table->timestamps();
    });
}

```

ساخت جدول comments با این مایگریشن انجام میشود. با
 متد foreignId یک فیلد index بدون علامت عددی درست
 میکند که بعد با استفاده از متد constrained و براساس
 استانداردی که در لاراول تعریف شده متصل میکنیم به جدول
 users که بعدا با استفاده از متد user در مدل comment
 که قبلا هم ذکر شد یک ارتباط یک به چند یا یک به یک
 ایجاد کنیم.

```

public function run()
{
    Comment::truncate();
    $userId = User::first()->id;
    $products = Product::all()->toArray();

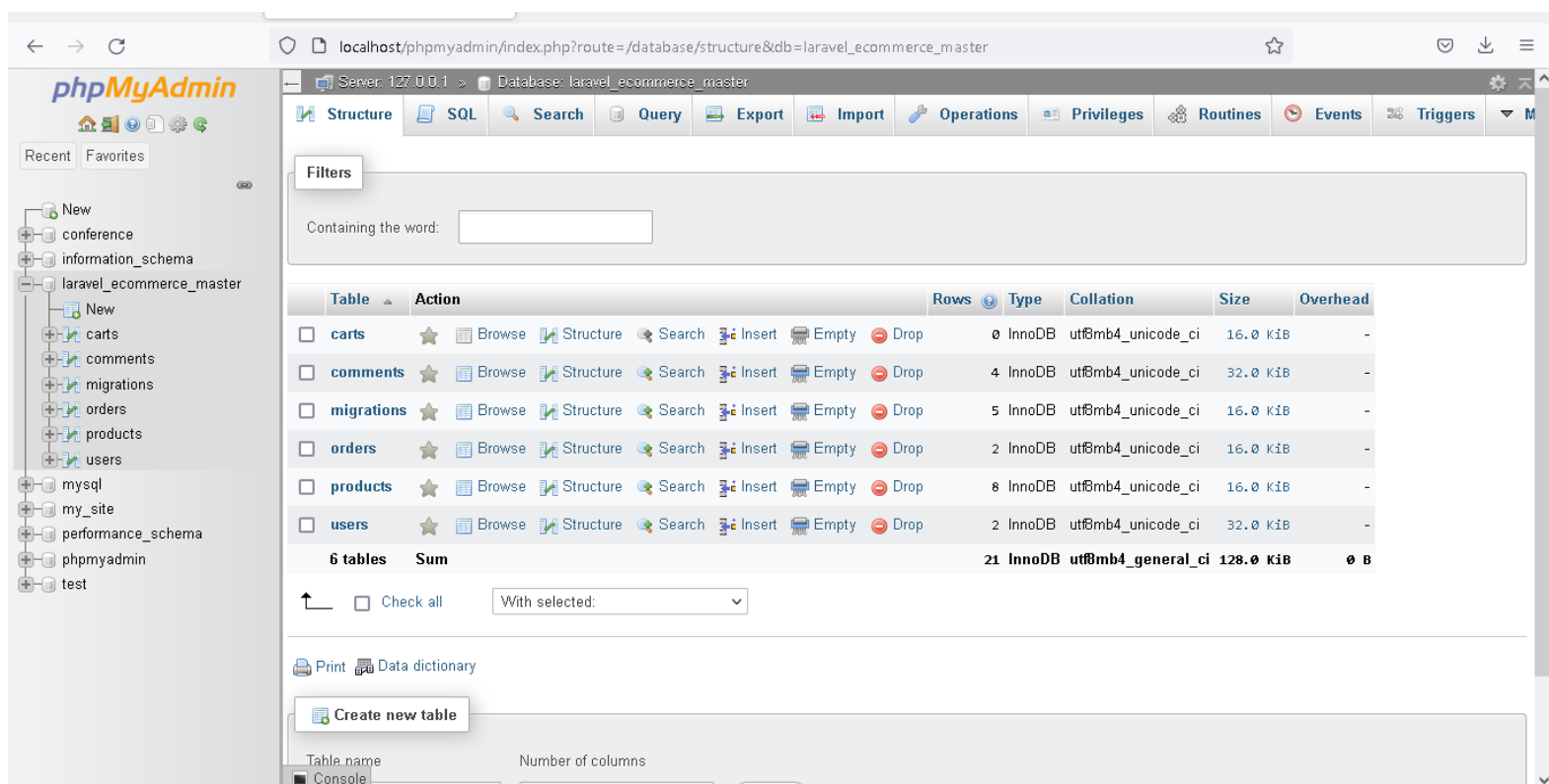
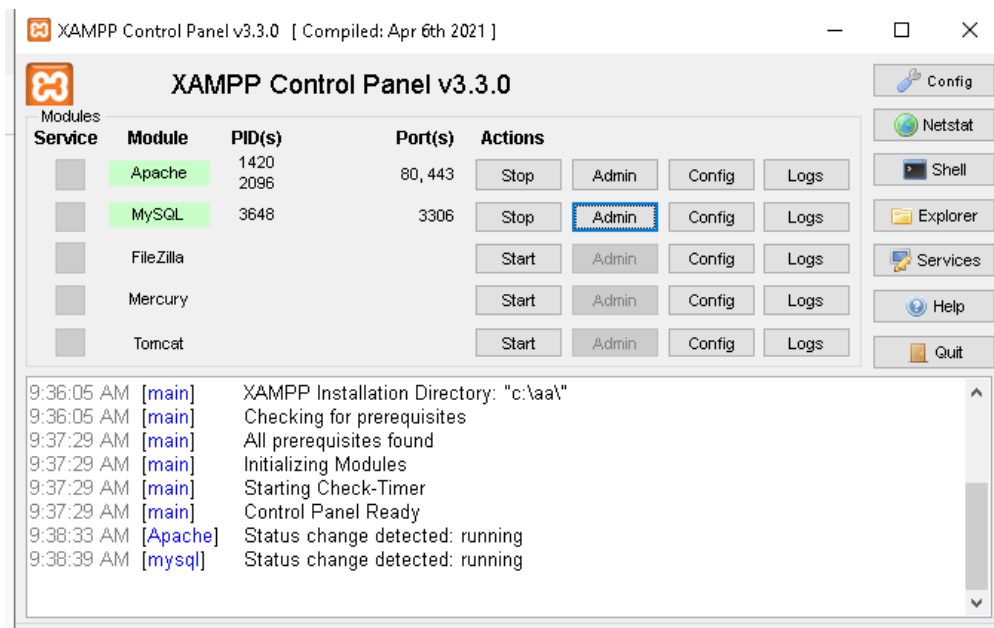
    $comments = [
        [
            'user_id' => $userId,
            'content' => 'متن آزمایشی'
        ],
        [
            'user_id' => $userId,
            'content' => 'متن آزمایشی'
        ],
        [
            'user_id' => $userId,
            'content' => 'متن آزمایشی'
        ],
        [
            'user_id' => $userId,
            'content' => 'متن آزمایشی'
        ]
    ];

    foreach ($comments as $comment) {
        $productKey = array_rand($products, 1);
    }
}

```

با کمک seeder ها میتوانیم داده های پیش فرضی برای جداولمان در نظر
 بگیریم.

در همه seeder ها از متدی استفاده شده به اسم truncate که مربوط به
 مدل همان seeder میشود برای ریست کردن جدول به کار می آید.



97143013-خدادادی