**Ujian Akhir Semester**

**WEB PROGRAMMING**

**Name: Martinus Grady Naftali**

**NIM: 2401962453**

**Class Program: LB08**

**Date:**

**6 Februari 2023**

**Lecturer:**

**Rani Puspita, S.Kom., M.Kom**

**UNIVERSITAS BINA NUSANTARA**

**COMPUTER SCIENCE**

**2022/2023**

# **Table of Content**

[I. Project Link 4](#_Toc120295531)

[II. Create Project 4](#_Toc120295532)

[A. Composer Create Project 4](#_Toc120295533)

[B. Change the directory 4](#_Toc120295534)

[C. Run the project 4](#_Toc120295535)

[III. Configure Database and Storage 5](#_Toc120295536)

[A. Create New Database in phpMyAdmin using XAMPP 5](#_Toc120295537)

[B. Create Laravel Storage 5](#_Toc120295538)

[C. Add images to Storage 5](#_Toc120295539)

[IV. Configure ENV 7](#_Toc120295540)

[A. Change App Name 7](#_Toc120295541)

[B. Change Database 7](#_Toc120295542)

[C. Change Filesystem Driver 7](#_Toc120295543)

[V. Create Models and migration 7](#_Toc120295544)

[A. Publisher 7](#_Toc120295545)

[B. Book 10](#_Toc120295546)

[C. Category 12](#_Toc120295547)

[D. BookCategory 14](#_Toc120295548)

[E. Migrate 16](#_Toc120295549)

[VI. Create Database Seeders 16](#_Toc120295550)

[A. Publisher 16](#_Toc120295551)

[B. Book 17](#_Toc120295552)

[C. Category 19](#_Toc120295553)

[D. BookCategory 20](#_Toc120295554)

[E. Create Call to All Seeders 23](#_Toc120295555)

[VII. Create Routing 24](#_Toc120295556)

[A. Initialization 24](#_Toc120295557)

[B. Home 24](#_Toc120295558)

[C. Book Detail 24](#_Toc120295559)

[D. Category 24](#_Toc120295560)

[E. Publisher 24](#_Toc120295561)

[F. Contact 25](#_Toc120295562)

[G. Result 25](#_Toc120295563)

[VIII. Create Controllers 26](#_Toc120295564)

[A. Initialization 26](#_Toc120295565)

[B. Home 26](#_Toc120295566)

[1) Index 26](#_Toc120295567)

[2) Category 26](#_Toc120295568)

[3) View Book 27](#_Toc120295569)

[4) Contact 27](#_Toc120295570)

[5) Result 28](#_Toc120295571)

[C. Publisher 29](#_Toc120295572)

[1) Index 29](#_Toc120295573)

[2) Publisher Detail 29](#_Toc120295574)

[3) Result 30](#_Toc120295575)

[IX. Create Views 30](#_Toc120295576)

[A. Initialization 30](#_Toc120295577)

[B. Import bootstrap 31](#_Toc120295578)

[C. Layout 32](#_Toc120295579)

[D. Components 34](#_Toc120295580)

[E. Page Initialization 36](#_Toc120295581)

[F. Home & Category 36](#_Toc120295582)

[G. Book Detail 37](#_Toc120295583)

[H. Contact 38](#_Toc120295584)

[I. Publisher List 39](#_Toc120295585)

[J. Publisher Detail 40](#_Toc120295586)

[X. Screenshot 41](#_Toc120295587)

[A. Home 41](#_Toc120295588)

[B. Category 42](#_Toc120295589)

[C. Book Detail 44](#_Toc120295590)

[D. Publisher List 45](#_Toc120295591)

[E. Publisher Detail 45](#_Toc120295592)

[F. Contact 47](#_Toc120295593)

[XI. Reference 48](#_Toc120295594)

[A. Picture and Text 48](#_Toc120295595)

[B. Documentation 48](#_Toc120295596)

# Project Link

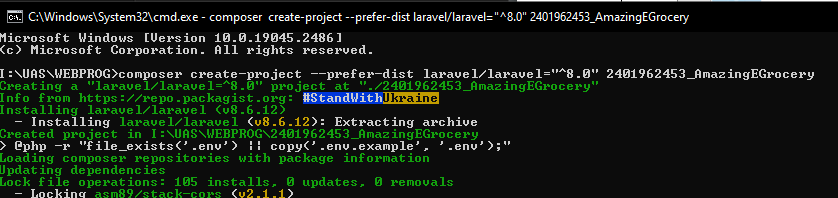
llll

# Create Project

## Composer Create Project

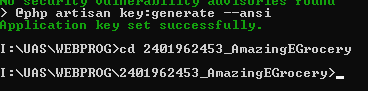
Create a new laravel project by typing “composer create-project -–prefer-dist laravel/laravel=”^8.0" 2401962453\_AmazingEGrocery” in the terminal and press enter.

I use -–prefer-dist laravel/laravel=”^8.0" to install a specific laravel version, which is version 8. “2401962453\_AmazingEGrocery” is the name of the project.



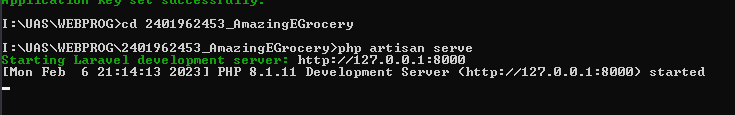
## Change the directory

Change directory to project folder using cd command.



## Run the project

Run the project typing “php artisan serve” and then press enter.

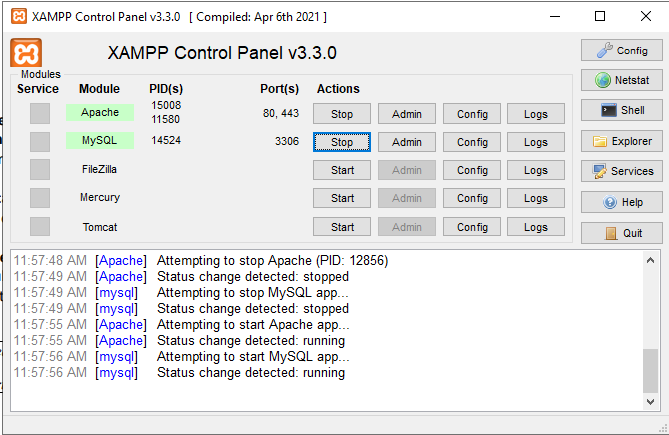


Then go to <http://127.0.0.1:8000/> in your browser.

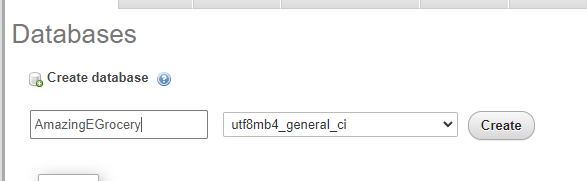
# Configure Database and Storage

## Create New Database in phpMyAdmin using XAMPP

Open XAMPP and start Apache and MySQL.

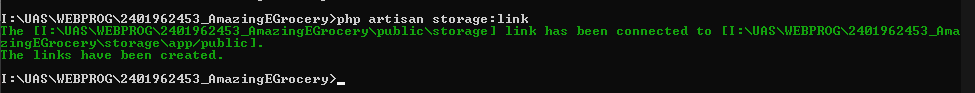


Go to MySQL admin and create new database named “giantbooksupplier”.



## Create Laravel Storage

Go to terminal and type “php artisan storage:link” to create new storage to store files.



# Configure ENV

## Change App Name

Inside the .env file in the root directory of the project, change the “APP\_NAME” into “Amazing E-Grocery”.

APP\_NAME=Amazing E-Grocery

APP\_ENV=local

APP\_KEY=base64:DAh+XUvsovqGP1U4hBd7Pe7El4YjSDfA2RVofXwtyN8=

APP\_DEBUG=true

APP\_URL=http://localhost

## Change Database

Once again, inside the .env file in the root directory of the project, change the “DB\_DATABASE” into “amazingegrocery”.

DB\_CONNECTION=mysql

DB\_HOST=127.0.0.1

DB\_PORT=3306

DB\_DATABASE=amazingegrocery

DB\_USERNAME=root

DB\_PASSWORD=

## Change Filesystem Driver

Finally, inside the .env file in the root directory of the project, change the “FILESYSTEM\_DRIVER” into “public”.

BROADCAST\_DRIVER=log

CACHE\_DRIVER=file

FILESYSTEM\_DRIVER=public

QUEUE\_CONNECTION=sync

SESSION\_DRIVER=file

SESSION\_LIFETIME=120

# Create Models and migration

## Role

Go to the terminal and type “php artisan make:model Role –migration” to create Role model and the migration.



Go to database/migrations folder and go to create\_roles\_table file.

In the public function up(), implement Schema∷create to create table fields which later be migrated to the database. Add increments(‘role\_id’) (which automatically assigned by laravel as PK and as an int that is auto incremented), and role\_name (varchar(10)). I also set all the attributes to nullable(false) so that they are set as “not null”.

class CreateRolesTable extends Migration

{

*/\*\**

*\* Run the migrations.*

*\**

*\* @return void*

*\*/*

    public function up()

    {

        Schema::create('roles', function (Blueprint $table) {

            $table->increments('role\_id')->nullable(false);

            $table->string('role\_name', 10)->nullable(false);

        });

    }

*/\*\**

*\* Reverse the migrations.*

*\**

*\* @return void*

*\*/*

    public function down()

    {

        Schema::dropIfExists('roles');

    }

}

Go to Role model in app/Models folder and fill the Role class with table, primary key, fillable and function for table relationship.

class Role extends Model

{

    use HasFactory;

public $timestamps = false;

    protected $table = 'roles';

protected $primaryKey = 'role\_id';

    protected $fillable = [

        'role\_name'

    ];

    public function account()

    {

        return $this->hasOne(Account::class,

        'role\_id', 'role\_id');

    }

}

Set table to ‘roles’ and timestamps to false

protected $table = 'roles';

Add primary key

protected $primaryKey = 'role\_id';

Add fillable table fields to fillable.

    protected $fillable = [

        'role\_name'

    ];

Add account function to add table relationship to acount table (one to one) using hasOne method by referencing acount ‘role\_id’ to role ‘role\_id’.

    public function account()

    {

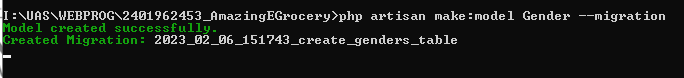
        return $this->hasOne(Account::class,

        'role\_id', 'role\_id');

    }

## Gender

Go to the terminal and type “php artisan make:model Gender –migration” to create Gender model and the migration.



Go to database/migrations folder and go to create\_genders\_table file.

class CreateGendersTable extends Migration

{

*/\*\**

*\* Run the migrations.*

*\**

*\* @return void*

*\*/*

    public function up()

    {

        Schema::create('genders', function (Blueprint $table) {

            $table->increments('gender\_id')->nullable(false);

            $table->string('gender\_desc', 10)->nullable(false);

        });

    }

*/\*\**

*\* Reverse the migrations.*

*\**

*\* @return void*

*\*/*

    public function down()

    {

        Schema::dropIfExists('genders');

    }

}

In the public function up(), implement Schema∷create to create table fields which later be migrated to the database. Add increments(‘gender\_id’) (which automatically assigned by laravel as PK and as an int that is auto incremented), and gender\_desc (varchar(10)). I also set all the attributes to nullable(false) so that they are set as “not null”.

Go to Gender model in app/Models folder and fill the Gender class with table, fillable and function for table relationship.

class Gender extends Model

{

    use HasFactory;

public $timestamps = false;

    protected $table = 'genders';

protected $primaryKey = 'gender\_id';

    protected $fillable = [

        'gender\_desc'

    ];

    public function account()

    {

        return $this->hasOne(Account::class,

        'gender\_id', 'gender\_id');

    }

}

Set table to ‘genders’ ande timestamps to false

 protected $table = 'genders';

Add primary key

protected $primaryKey = 'role\_id';

Add fillable table fields to fillable.

protected $fillable = [

        'gender\_desc'

    ];

Add account function to add table relationship to acount table (one to one) using hasOne method by referencing acount ‘gender\_id’ to role ‘gender\_id’.

    public function account()

    {

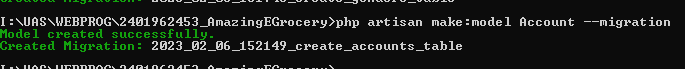
        return $this->hasOne(Account::class,

        'gender\_id', 'gender\_id');

    }

## Account

Go to the terminal and type “php artisan make:model Account –migration” to create Account model and the migration.



Go to database/migrations folder and go to create\_accounts\_table file.

class CreateAccountsTable extends Migration

{

*/\*\**

*\* Run the migrations.*

*\**

*\* @return void*

*\*/*

    public function up()

    {

        Schema::create('accounts', function (Blueprint $table) {

            $table->increments('account\_id')->nullable(false);

            $table->foreignId('role\_id')->reference('role\_id')->on('roles')->onUpdate('cascade')->onDelete('cascade');

            $table->foreignId('gender\_id')->reference('gender\_id')->on('genders')->onUpdate('cascade')->onDelete('cascade');

            $table->string('first\_name', 25)->nullable(false);

            $table->string('last\_name', 25)->nullable(false);

            $table->string('email', 100)->nullable(false);

            $table->string('display\_picture\_link', 100)->nullable(false);

            $table->string('password');

            $table->rememberToken();

            $table->timestamps();

        });

    }

*/\*\**

*\* Reverse the migrations.*

*\**

*\* @return void*

*\*/*

    public function down()

    {

        Schema::dropIfExists('accounts');

    }

}

In the public function up(), implement Schema∷create to create table fields which later be migrated to the database. Add increments(‘acount\_id’) (which automatically assigned by laravel as PK and as an int that is auto incremented), role\_id, gender\_id (foreignId which reference the id on role and gender table, and add cascade so when parent record is updated or deleted, the child record will also be updated or deleted), first\_name (varchar(25)), last\_name (varchar(25)), email (varchar(100)), display\_picture\_link (varchar(100)), password, rememberToken, and timestamp. I also set all the attributes to nullable(false) so that they are set as “not null”.

Go to Account model in app/Models folder and fill the Account class with table, fillable and function for table relationship. Change extends model to autheticatable and use HasApiTokens and Notifiatble.

class Account extends Authenticatable

{

    use HasApiTokens, HasFactory, Notifiable;

    protected $table = 'accounts';

protected $primaryKey = 'account\_id';

    protected $fillable = [

        'role\_id',

        'gender\_id',

        'first\_name',

        'last\_name',

        'email',

        'display\_picture\_link',

        'password'

    ];

    public function role()

    {

        return $this->belongsTo(Role::class, 'role\_id', 'role\_id');

    }

    public function gender()

    {

        return $this->belongsTo(Gender::class, 'gender\_id', 'gender\_id');

    }

}

Set table to ‘account’.

protected $table = 'accounts';

Add primary key

protected $primaryKey = 'account\_id';

Add fillable table fields to fillable.

    protected $fillable = [

        'role\_id',

        'gender\_id',

        'first\_name',

        'last\_name',

        'email',

        'display\_picture\_link',

        'password'

    ];

Add role function to add table relationship to roles table (acount is own by one role) using belongsTo method by referencing role ‘role\_id’ to account ‘role\_id’.

    public function role()

    {

        return $this->belongsTo(Role::class, 'role\_id', 'role\_id');

    }

Add gender function to add table relationship to roles table (acount is own by one gender) using belongsTo method by referencing gender ‘gender\_id’ to account ‘gender\_id’.

    public function gender()

    {

        return $this->belongsTo(Gender::class, 'gender\_id', 'gender\_id');

    }

Add orders function to add table relationship to roles table (acount has many orders) using hasMany method by referencing order ‘acount\_id’ to account ‘account\_id’.

    public function orders()

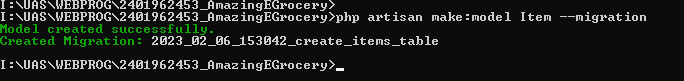
    {

        return $this->hasMany(Order::class, 'account\_id', 'account\_id');

    }

## Item

Go to the terminal and type “php artisan make:model Item –migration” to create Item model and the migration.

  
Go to database/migrations folder and go to create\_items\_table file.

class CreateItemsTable extends Migration

{

*/\*\**

*\* Run the migrations.*

*\**

*\* @return void*

*\*/*

    public function up()

    {

        Schema::create('items', function (Blueprint $table) {

            $table->increments('item\_id')->nullable(false);

            $table->string('item\_name', 50)->nullable(false);

            $table->text('item\_desc', 500)->nullable(false);

            $table->integer('price')->nullable(false);

        });

    }

*/\*\**

*\* Reverse the migrations.*

*\**

*\* @return void*

*\*/*

    public function down()

    {

        Schema::dropIfExists('items');

    }

}

In the public function up(), implement Schema∷create to create table fields which later be migrated to the database. Add increments(‘item\_id’) (which automatically assigned by laravel as PK and as an int that is auto incremented), item\_name (varchar(50)), item\_desc (varchar only support up to 255 characters, so I use text which supports more than 255 characters (500), and price(int). I also set all the attributes to nullable(false) so that they are set as “not null”.

Go to Item model in app/Models folder and fill the Item class with table, fillable and function for table relationship.

class Item extends Model

{

    use HasFactory;

public $timestamps = false;

    protected $table = 'items';

protected $primaryKey = 'item\_id';

    protected $fillable = [

        'item\_name',

        'item\_desc',

        'price'

    ];

    public function order()

    {

        return $this->hasOne(Role::class, 'item\_id', 'item\_id');

    }

}

Add primary key

protected $primaryKey = 'item\_id';

Set table to ‘items’, timestamps to false and add fillable table fields to fillable

    protected $fillable = [

        'item\_name',

        'item\_desc',

        'price'

    ];

Add order function to add table relationship to order table (one to one) using hasOne method by referencing order ‘item\_id’ to item ‘item\_id’.

    public function order()

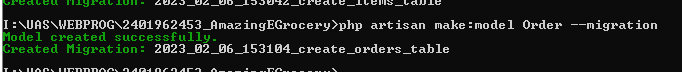
    {

        return $this->hasOne(Role::class, 'item\_id', 'item\_id');

    }

## Order

Go to the terminal and type “php artisan make:model Order –migration” to create Item model and the migration.

  
Go to database/migrations folder and go to create\_orders\_table file.

class CreateOrdersTable extends Migration

{

*/\*\**

*\* Run the migrations.*

*\**

*\* @return void*

*\*/*

    public function up()

    {

        Schema::create('orders', function (Blueprint $table) {

            $table->increments('order\_id')->nullable(false);

            $table->foreignId('account\_id')->reference('account\_id')->on('accounts')->onUpdate('cascade')->onDelete('cascade');

            $table->foreignId('item\_id')->reference('item\_id')->on('items')->onUpdate('cascade')->onDelete('cascade');

            $table->integer('price')->nullable(false);

        });

    }

*/\*\**

*\* Reverse the migrations.*

*\**

*\* @return void*

*\*/*

    public function down()

    {

        Schema::dropIfExists('orders');

    }

}

In the public function up(), implement Schema∷create to create table fields which later be migrated to the database. Add increments(‘order\_id’) (which automatically assigned by laravel as PK and as an int that is auto incremented), account\_id, item\_id (foreignId which reference the id on account and item table, and add cascade so when parent record is updated or deleted, the child record will also be updated or deleted), and price (int). I also set all the attributes to nullable(false) so that they are set as “not null”.

Go to Order model in app/Models folder and fill the Order class with table, fillable and function for table relationship.

class Order extends Model

{

    use HasFactory;

public $timestamps = false;

    protected $table = 'orders';

protected $primaryKey = 'order\_id';

    protected $fillable = [

        'account\_id',

        'item\_id',

        'price'

    ];

    public function account()

    {

        return $this->belongsTo(Account::class, 'account\_id', 'account\_id');

    }

    public function items()

    {

        return $this->belongsTo(Item::class, 'item\_id', 'item\_id');

    }

}

Set table to ‘orders’, timestamps to false, and add fillable table fields to fillable

protected $table = 'orders';

Add primary key

protected $primaryKey = 'order\_id';

Add fillable

    protected $fillable = [

        'account\_id',

        'item\_id',

        'price'

    ];

Add account function to add table relationship to accounts table (order is own by one account) using belongsTo method by referencing order ‘account\_id’ to account ‘account\_id’.

    public function account()

    {

        return $this->belongsTo(Account::class, 'account\_id', 'account\_id');

    }

Add items function to add table relationship to items table (order is own by many item) using belongsTo method by referencing order ‘item\_id’ to item ‘item\_id’.

    public function items()

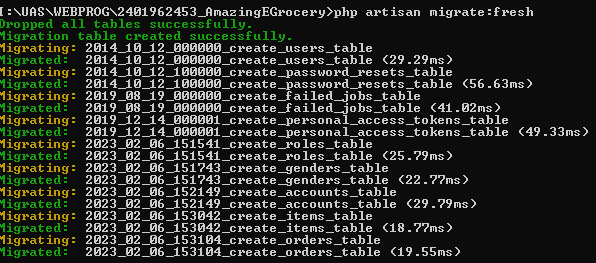
    {

        return $this->belongsTo(Item::class, 'item\_id', 'item\_id');

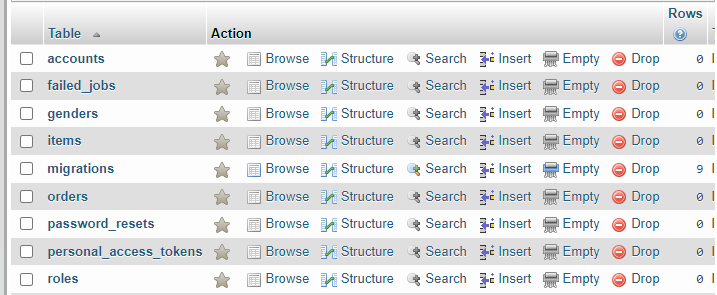
    }

## Migrate

Type ‘php artisan migrate:fresh’ in the terminal and enter to migrate to the database



Tables are successfully migrated to the database



# Create Database Seeders

## Item

Go to terminal and type ‘php artisan make:seeder ItemSeeder’ to make item seeder.



Go to ItemSeeder in database/seeders folder and add use App\Models\Item after use Illuminate\Database\Seeder.

use App\Models\Item;

In the public function run() by referencing the Item model, call firstOrCreate method to create a new record if the record is not yet created. Pass fillable fields value such as item\_name, item\_desc, and price from data you have collected in an array as an argument in firtsOrCreate method.

class ItemSeeder extends Seeder

{

*/\*\**

*\* Run the database seeds.*

*\**

*\* @return void*

*\*/*

    public function run()

    {

        Item::firstOrCreate([

            'item\_name' => "Kailan Baby Conventional",

            'item\_desc' => "Baby Kailan adalah sayuran berdaun tebal, datar, mengkilap dan berwarna hijau dengan batang tebal. Tahukah kamu bahwa sayur kailan memiliki kandungan nutrisi yang cukup lengkap bagi tubuh.",

            'price'=> 15800

        ]);

        Item::firstOrCreate([

            'item\_name' => "Pakchoy Pakcoy",

            'item\_desc' => "Buah yang kami kirim setiap harinya, sehingga, sayur yang baru dipanen akan dikirim setiap harinya waktu dini hari. Setelah datang, kami bersihkan dan siapkan untuk dikirimkan ke pelanggan.",

            'price'=> 2400

        ]);

        Item::firstOrCreate([

            'item\_name' => "Selada Keriting",

            'item\_desc' => "Daun selada adalah sumber vitamin A dan vitamin K yang sangat tinggi. Selain itu, daun selada juga mengandung berbagai nutrisi penting seperti zat besi, kalium, kalsium, folat, dan serat.",

            'price'=> 8100

        ]);

    }

}

## Gender

Go to terminal and type ‘php artisan make:seeder GenderSeeder’ to make gender seeder.



Go to GenderSeeder in database/seeders folder and add use App\Models\Gender after use Illuminate\Database\Seeder.

use App\Models\Gender;

In the public function run() by referencing the Item model, call firstOrCreate method to create a new record if the record is not yet created. Pass fillable fields value such as gender\_desc from data you have collected in an array as an argument in firtsOrCreate method.

class GenderSeeder extends Seeder

{

*/\*\**

*\* Run the database seeds.*

*\**

*\* @return void*

*\*/*

    public function run()

    {

        Gender::firstOrCreate([

            'gender\_desc' => "Male",

        ]);

        Gender::firstOrCreate([

            'gender\_desc' => "Female",

        ]);

    }

}

## Role

Go to terminal and type ‘php artisan make:seeder RoleSeeder’ to make role seeder.



Go to RoleSeeder in database/seeders folder and add use App\Models\Role after use Illuminate\Database\Seeder.

use App\Models\Role;

In the public function run() by referencing the Role model, call firstOrCreate method to create a new record if the record is not yet created. Pass fillable fields value such as role\_name from data you have collected in an array as an argument in firtsOrCreate method.

class RoleSeeder extends Seeder

{

*/\*\**

*\* Run the database seeds.*

*\**

*\* @return void*

*\*/*

    public function run()

    {

        Role::firstOrCreate([

            'role\_name' => "User",

        ]);

        Role::firstOrCreate([

            'role\_name' => "Admin",

        ]);

    }

}

## Create Call to All Seeders

To call all the seeders we have created, go to database/seeders folder and go to DatabaseSeeder.php. Inside that file, in the public function run() use method call and reference it to all seeders we have created.

class DatabaseSeeder extends Seeder

{

*/\*\**

*\* Seed the application's database.*

*\**

*\* @return void*

*\*/*

    public function run()

    {

        $this->call(ItemSeeder::class);

        $this->call(GenderSeeder::class);

        $this->call(RoleSeeder::class);

        $this->call(AccountSeeder::class);

    }

}

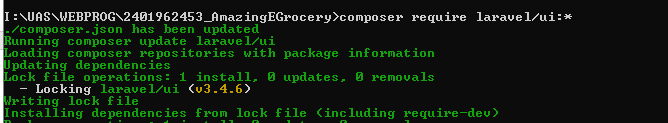
Go to terminal and type “php artisan db:seed” to seed all of our data to the database



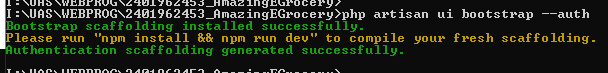
# Create Index

## Install Laravel UI

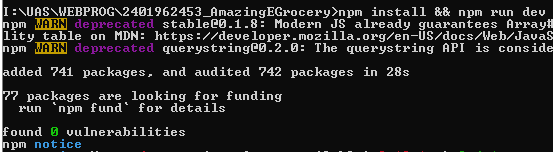
Type “composer require laravel/ui:\*” in terminal



Type “php artisan ui bootstrap –auth” in terminal



Type “npm install && npm run dev” in terminal

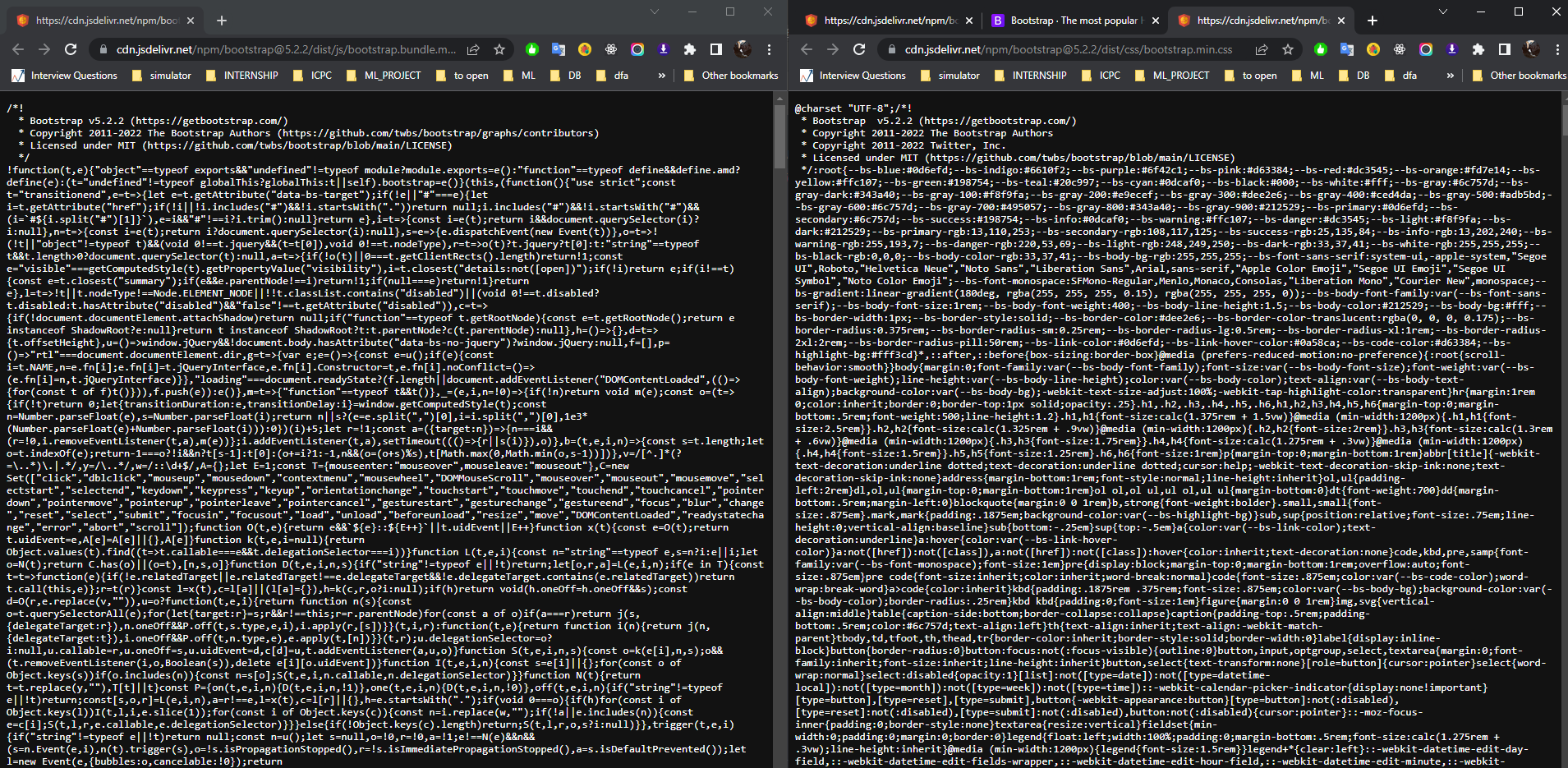


## Import bootstrap

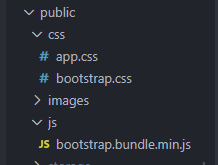
Go to bootstrap website, <https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js/>

<https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css/>

Then copy all the css and js code.



Put the css code to public/css/bootstrap.css and js code to public/js/bootstrap.bundle.min.js



Import css.bootstrap.css in head and js/bootstrap.bundle.min.js script in body at views/layouts/app.blade.php

<head>

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1">

*<!-- CSRF Token -->*

    <meta name="csrf-token" content="{{ csrf\_token() }}">

    <title>{{ config('app.name', 'Laravel') }}</title>

*<!-- Scripts -->*

    <script src="{{ asset('js/app.js') }}" defer></script>

*<!-- Fonts -->*

    <link rel="dns-prefetch" href="//fonts.gstatic.com">

    <link href="https://fonts.googleapis.com/css?family=Nunito" rel="stylesheet">

*<!-- Styles -->*

    <link href="{{ asset('css/app.css') }}" rel="stylesheet">

    <link href="{{ asset('css/bootstrap.css') }}" rel="stylesheet">

</head>

*<!-- Scripts -->*

    <script src="{{ asset('js/bootstrap.bundle.min.js') }}"></script>

</body>

</html>

## Customize app.blade.php

Add footer below main.

        <main class="py-4">

            @yield('content')

        </main>

        <footer class="footer" style="position: absolute; bottom: 0; width: 100%; background-color: #adead2">

            <div class="text-center p-4">

                © Amazing E-Grocery 2023

            </div>

        </footer>

## Customize index.blade.php

Change welcome.blade.php into index.blade.php



Follow the code below to add bootstrap component and other element in index

@section('content')

<div class="container">

    <div class="row justify-content-center">

        <div class="d-flex col-md-8 justify-content-center align-items-center">

            <div class="d-flex justify-content-center align-items-center rounded-circle bg-white" style="width: 500px; height: 500px; border: 12px solid #f5da55;">

                <h1 class="text-center">{{\_\_('index.title')}}</h5>

            </div>

        </div>

    </div>

</div>

@endsection

# Create Register

## Change Register View

Open resources/views/auth/register.blade.php. Then follow the following code with bootstrap component and other elements.

    @extends('layouts.app')

    @section('content')

        <div class="container">

            <div class="row justify-content-center">

                <div class="col-md-12">

                    <form method="POST" action="{{ route('register') }}" id="registration\_form"">

                        @csrf

                        <div class="form-group mb-4">

                            <h3 class="text-left"><u>{{ \_\_('form.titleRegister') }}</u></h3>

                        </div>

                        <div class="form-group d-flex justify-content-between">

                            <div class="col-6 pe-2">

                                <div class="form-group row mb-2">

                                    <label for="firstName" class="col-sm-3 col-form-label">{{ \_\_('form.first\_name') }} :</label>

                                    <div class="col-sm-9">

                                        <input type="text" class="form-control" @error('first\_name') is-invalid @enderror"

                                            id="firstName" name="firstName">

                                    </div>

                                    <span class="invalid-feedback" role="alert">

                                        @error('first\_name')

                                            {{ $message }}

                                        @enderror

                                    </span>

                                </div>

                                <div class="form-group row mb-2">

                                    <label for="email" class="col-sm-3 col-form-label">{{ \_\_('form.email') }} :</label>

                                    <div class="col-sm-9">

                                        <input type="email" class="form-control" @error('email') is-invalid @enderror"

                                            id="email" name="email">

                                    </div>

                                    <span class="invalid-feedback" role="alert">

                                        @error('email')

                                            {{ $message }}

                                        @enderror

                                    </span>

                                </div>

                                <div class="form-group row mb-2">

                                    <label for="gender" class="col-sm-3 col-form-label">{{ \_\_('form.gender') }} :</label>

                                    <div class="col-sm-9 d-flex align-items-center">

                                        <div class="input-group-prepend @error('gender') is-invalid @enderror">

                                            <div class="form-check form-check-inline">

                                                <input class="form-check-input" type="radio" name="gender" id="male"

                                                    value="male">

                                                <label class="form-check-label" for="male">@lang('form.male')</label>

                                            </div>

                                            <div class="form-check form-check-inline">

                                                <input class="form-check-input" type="radio" name="gender" id="female"

                                                    value="female">

                                                <label class="form-check-label" for="female">@lang('form.female')</label>

                                            </div>

                                        </div>

                                    </div>

                                    <span class="invalid-feedback" role="alert">

                                        @error('gender')

                                            {{ $message }}

                                        @enderror

                                    </span>

                                </div>

                                <div class="form-group row mb-2">

                                    <label for="password" class="col-sm-3 col-form-label">{{ \_\_('form.password') }} :</label>

                                    <div class="col-sm-9">

                                        <input type="password" class="form-control" @error('password') is-invalid @enderror"

                                            id="password" name="password">

                                    </div>

                                    <span class="invalid-feedback" role="alert">

                                        @error('password')

                                            {{ $message }}

                                        @enderror

                                    </span>

                                </div>

                            </div>

                            <div class="col-6 ps-2">

                                <div class="form-group row mb-2">

                                    <label for="lastName" class="col-sm-3 col-form-label">{{ \_\_('form.last\_name') }} :</label>

                                    <div class="col-sm-9">

                                        <input type="text" class="form-control" @error('lastName') is-invalid @enderror"

                                            id="lastName" name="lastName">

                                    </div>

                                    <span class="invalid-feedback" role="alert">

                                        @error('lastName')

                                            {{ $message }}

                                        @enderror

                                    </span>

                                </div>

                                <div class="form-group row mb-2">

                                    <label for="role" class="col-sm-3 col-form-label">{{ \_\_('form.role') }} :</label>

                                    <div class="col-sm-9">

                                        <div class="input-group @error('role') is-invalid @enderror">

                                            <select class="form-control" id="role" form="registration\_form" name="role">

                                                <option value="admin">@lang('form.admin')</option>

                                                <option value="user">@lang('form.user')</option>

                                            </select>

                                        </div>

                                    </div>

                                    <span class="invalid-feedback" role="alert">

                                        @error('role')

                                            {{ $message }}

                                        @enderror

                                    </span>

                                </div>

                                <div class="form-group row mb-2">

                                    <label for="displayPicture"

                                        class="col-sm-3 col-form-label">{{ \_\_('form.display\_picture') }} :</label>

                                    <div class="col-sm-9 d-flex align-items-center">

                                        <input type="file" class="form-control-file" id="displayPicture"

                                            name="displayPicture">

                                        <span class="text-danger">

                                    </div>

                                    <span class="invalid-feedback" role="alert">

                                        @error('displayPicture')

                                            {{ $message }}

                                        @enderror

                                    </span>

                                </div>

                                <div class="form-group row mb-2">

                                    <label for="confirmPassword" class="col-sm-3 col-form-label">{{ \_\_('form.confirm\_password') }} :</label>

                                    <div class="col-sm-9">

                                        <input type="password" class="form-control" @error('confirmPassword') is-invalid @enderror"

                                            id="confirmPassword" name="confirmPassword">

                                    </div>

                                    <span class="invalid-feedback" role="alert">

                                        @error('confirmPassword')

                                            {{ $message }}

                                        @enderror

                                    </span>

                                </div>

                            </div>

                        </div>

                    </form>

                </div>

            </div>

        </div>

    @endsection

## Change Auth

Change user model in config/auth.php into Account∷class

    'providers' => [

        'users' => [

            'driver' => 'eloquent',

            'model' => App\Models\Account::class,

        ],

## Edit Register Controller

Go to registerController at app/Http/Auth/RegisterController.php

Edit the validator function according to question’s requirements :

Alpha\_dash is used so that no symbol is allowed is the registration name

    protected function validator(array $data)

    {

        return Validator::make($data, [

            'first\_name' => ['required', 'string', 'alpha\_dash', 'max:25'],

            'last\_name' => ['required', 'string', 'alpha\_dash', 'max:25'],

            'email' => ['required', 'string', 'email', 'max:100', 'unique:users'],

            'role' => ['required', 'string'],

            'gender' => ['required', 'string'],

            'display\_picture' => ['required', 'image', 'mimes:jpg,jpeg,png'],

            'password' => ['required', 'string', Password::min(8), Password::min(1)->numbers(), 'confirmed'],

        ]);

    }

Edit the create function, find id for gender and role using where, and store display picture using storeAs.

    protected function create(array $data)

    {

        $request = app('request');

        $gender\_id = Gender::where('gender\_desc',$data['gender']) -> first()->gender\_id;

        $role\_id = Role::where('role\_name',$data['role']) -> first()->role\_id;

        $file = $request->file('display\_picture');

        $ext = $file->getClientOriginalExtension();

        $filename = time().'.'.$ext;

        $file->storeAs('account/display\_picture', $filename);

        return Account::create([

            'role\_id' => $role\_id,

            'gender\_id' => $gender\_id,

            'first\_name' => $data['first\_name'],

            'last\_name' => $data['last\_name'],

            'email' => $data['email'],

            'display\_picture\_link' => $filename,

            'password' => Hash::make($data['password']),

        ]);

    }

# Create Login

## Change Login view

Open resources/views/auth/login.blade.php. Then follow the following code with bootstrap component and other elements

@extends('layouts.app')

@section('content')

    <div class="container">

        <div class="row justify-content-center">

            <div class="col-md-6">

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

                    @csrf

                    <div class="form-group mb-4">

                        <h3 class="text-left"><u>{{ \_\_('form.titleLogin') }}</u></h3>

                    </div>

                    <div class="form-group row mb-3">

                        <label for="email" class="col-sm-3 col-form-label">{{ \_\_('form.email') }} :</label>

                        <div class="col-sm-9">

                            <input type="email" class="form-control @error('email') is-invalid @enderror"

                                value="{{ old('email') }}" required autocomplete="email" autofocus id="email"

                                name="email">

                        </div>

                        <span class="text-danger" role="alert">

                            @error('email')

                                {{ $message }}

                            @enderror

                        </span>

                    </div>

                    <div class="form-group row mb-3">

                        <label for="password" class="col-sm-3 col-form-label">{{ \_\_('form.password') }}

                            :</label>

                        <div class="col-sm-9">

                            <input type="password" class="form-control @error('password') is-invalid @enderror"

                                id="password" name="password">

                        </div>

                        <span class="text-danger" role="alert">

                            @error('password')

                                {{ $message }}

                            @enderror

                        </span>

                    </div>

                    <div class="form-group row mb-3">

                        <div class="form-check ms-2">

                            <input class="form-check-input" type="checkbox" name="remember" id="remember"

                                {{ old('remember') ? 'checked' : '' }}>

                            <label class="form-check-label" for="remember">

                                {{ \_\_('Remember Me') }}

                            </label>

                        </div>

                    </div>

                    <div class="d-flex justify-content-center row">

                        <div class="w-auto mb-2 mt-4">

                            <div class="col text-center" style="background-color: #f5da55 !important">

                                <button type="submit" class="btn">

                                    {{ \_\_('form.submit') }}

                                </button>

                            </div>

                        </div>

                        <div class="row  justify-content-center text-center">

                            <a href="{{ route('register') }}">{{ \_\_('form.no\_account') }}</a>

                        </div>

                    </div>

                </form>

            </div>

        </div>

    </div>

@endsection

# Create Middleware

## Redirect the unregistered user to “index page”

Open app/Http/Middleware/Autheticate.php.

Change route inside redirectTo into ‘index’.

class Authenticate extends Middleware

{

*/\*\**

*\* Get the path the user should be redirected to when they are not authenticated.*

*\**

*\* @param  \Illuminate\Http\Request  $request*

*\* @return string|null*

*\*/*

    protected function redirectTo($request)

    {

        if (! $request->expectsJson()) {

            return route('index');

        }

    }

}

Open routes/web.php and change Route for ‘/’ into:

Route::get('/', function () {

    return view('index');

})->name('index');

And surround route that needs auth with Route∷middleware([‘auth’])

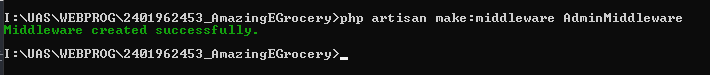
Route::middleware(['auth'])->group(function (){

    Route::get('/home', [App\Http\Controllers\HomeController::class, 'index'])->name('home');

});

## Redirect non Admin user to Home page

Go to terminal and type “php artisan make:middleware AdminMiddleware” then enter to create admin middleware.



Go to app/Http/Kernel.php and edit the $routeMiddleware by adding

'isAdmin' => \App\Http\Middleware\AdminMiddleware::class

    protected $routeMiddleware = [

        'auth' => \App\Http\Middleware\Authenticate::class,

        'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,

        'cache.headers' => \Illuminate\Http\Middleware\SetCacheHeaders::class,

        'can' => \Illuminate\Auth\Middleware\Authorize::class,

        'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,

        'password.confirm' => \Illuminate\Auth\Middleware\RequirePassword::class,

        'signed' => \Illuminate\Routing\Middleware\ValidateSignature::class,

        'throttle' => \Illuminate\Routing\Middleware\ThrottleRequests::class,

        'verified' => \Illuminate\Auth\Middleware\EnsureEmailIsVerified::class,

        'isAdmin' => \App\Http\Middleware\AdminMiddleware::class

    ];

Go to app/Http/Middleware/AdminMiddleware.php and edit handle function which checks the auth first and then check role of user. If user is not admin, then redirect to home.

    public function handle(Request $request, Closure $next)

    {

        if(Auth::check())

        {

            if(Auth::user()->role()->first()->role\_name == "Admin")

            {

                return $next($request);

            }

            else

            {

                return redirect(route('home'))->with('status', 'Access Denied! You are not the admin');

            }

        }

        else

        {

            return redirect('/')->with('status', 'You need to login first');

        }

    }

Go to app/Providers/AppServiceProvider.php and edit the boot function, add check at boot at blade for admin and add paginator for later in paging.

    public function boot()

    {

        Schema::defaultStringLength(191);

        Blade::if('isAdmin', function () {

            return Auth::check() && Auth::user()->role()->first()->role\_name == "Admin";

        });

        Paginator::useBootstrap();

    }

# Create Navbar for Logged User

## Navbar for All User and Admin

Add code below at navbar after <!-- Left Side Of Navbar -->, wrap it in @auth, so that it only appears when user has logged in. For account maintennance, wrap it inside @isAdmin, so it only appears when admin has logged in.

<ul class="navbar-nav me-auto">

    @auth

        <li class="nav-item" style="background-color: #f5da55 !important">

            <a class="nav-link" href="{{ route('home') }}">{{ \_\_('page.home') }}</a>

        </li>

        <li class="nav-item" style="background-color: #f5da55 !important">

            <a class="nav-link" href="{{ route('cart') }}">{{ \_\_('page.cart') }}</a>

        </li>

        <li class="nav-item" style="background-color: #f5da55 !important">

            <a class="nav-link" href="{{ route('profile') }}">{{ \_\_('page.profile') }}</a>

        </li>

        @isAdmin

            <li class="nav-item" style="background-color: #f5da55 !important">

                <a class="nav-link" href="{{ route('account-maintenance') }}">{{ \_\_('page.account\_maintenance') }}</a>

            </li>

        @endisAdmin

    @endauth

</ul>

# Create Profile

## Make Profile View

Inside views, add profile.blade.php.



Use bootstrap component, html element, and blade directive to construct view below:

@extends('layouts.app')

@section('content')

    <div class="container">

        <div class="row justify-content-center">

            <div class="col-md-12">

                <div class="row mt-4">

                    <div class="col-md-2 p-1 d-flex justify-content-center">

                        <img src="{{ asset('storage/account/display\_picture/' . $account->display\_picture\_link) }}"

                            class="img-fluid rounded border" alt="display picture">

                    </div>

                    <div class="col-md-10">

                        <form method="POST" action="{{ route('update-profile', ['id' => $account->account\_id]) }}"

                            id="registration\_form" enctype="multipart/form-data">

                            @method('PATCH')

                            @csrf

                            <div class="form-group d-flex justify-content-between">

                                <div class="col-6 pe-2">

                                    <div class="form-group row mb-2">

                                        <label for="first\_name" class="col-sm-3 col-form-label">{{ \_\_('form.first\_name') }}

                                            :</label>

                                        <div class="col-sm-9">

                                            <input type="text"

                                                class="form-control

                                        @error('first\_name') is-invalid @enderror"

                                                id="first\_name" name="first\_name" value="{{ $account->first\_name }}">

                                        </div>

                                        @error('first\_name')

                                            <span class="text-danger" role="alert">

                                                {{ $message }}

                                            </span>

                                        @enderror

                                    </div>

                                    <div class="form-group row mb-2">

                                        <label for="email" class="col-sm-3 col-form-label">{{ \_\_('form.email') }}

                                            :</label>

                                        <div class="col-sm-9">

                                            <input type="email" class="form-control @error('email') is-invalid @enderror"

                                                id="email" name="email" value="{{ $account->email }}"">

                                        </div>

                                        <span class="text-danger" role="alert">

                                            @error('email')

                                                {{ $message }}

                                            @enderror

                                        </span>

                                    </div>

                                    <div class="form-group row mb-2">

                                        <label for="gender" class="col-sm-3 col-form-label">{{ \_\_('form.gender') }}

                                            :</label>

                                        <div class="col-sm-9 d-flex align-items-center">

                                            <div class="input-group-prepend @error('gender') is-invalid @enderror">

                                                <div class="form-check form-check-inline">

                                                    @if ($account->gender == 'Male')

                                                        <input class="form-check-input" type="radio" name="gender"

                                                            id="male" value="Male" checked="checked">

                                                    @else

                                                        <input class="form-check-input" type="radio" name="gender"

                                                            id="male" value="Male">

                                                    @endif

                                                    <label class="form-check-label"

                                                        for="male">{{ \_\_('form.male') }}</label>

                                                </div>

                                                <div class="form-check form-check-inline">

                                                    @if ($account->gender == 'Female')

                                                        <input class="form-check-input" type="radio" name="gender"

                                                            id="female" value="Female" checked="checked">

                                                    @else

                                                        <input class="form-check-input" type="radio" name="gender"

                                                            id="female" value="Female">

                                                    @endif

                                                    <label class="form-check-label"

                                                        for="female">{{ \_\_('form.female') }}</label>

                                                </div>

                                            </div>

                                        </div>

                                        <span class="text-danger" role="alert">

                                            @error('gender')

                                                {{ $message }}

                                            @enderror

                                        </span>

                                    </div>

                                    <div class="form-group row mb-2">

                                        <label for="password" class="col-sm-3 col-form-label">{{ \_\_('form.password') }}

                                            :</label>

                                        <div class="col-sm-9">

                                            <input type="password"

                                                class="form-control @error('password') is-invalid @enderror" id="password"

                                                name="password">

                                        </div>

                                        <span class="text-danger" role="alert">

                                            @error('password')

                                                {{ $message }}

                                            @enderror

                                        </span>

                                    </div>

                                </div>

                                <div class="col-6 ps-2">

                                    <div class="form-group row mb-2">

                                        <label for="last\_name" class="col-sm-3 col-form-label">{{ \_\_('form.last\_name') }}

                                            :</label>

                                        <div class="col-sm-9">

                                            <input type="text"

                                                class="form-control @error('last\_name') is-invalid @enderror" id="last\_name"

                                                name="last\_name" value="{{ $account->last\_name }}">

                                        </div>

                                        <span class="text-danger" role="alert">

                                            @error('last\_name')

                                                {{ $message }}

                                            @enderror

                                        </span>

                                    </div>

                                    <div class="form-group row mb-2">

                                        <label for="role" class="col-sm-3 col-form-label">{{ \_\_('form.role') }}

                                            :</label>

                                        <div class="col-sm-9">

                                            <div class="input-group @error('role') is-invalid @enderror">

                                                <select class="form-control" id="role" form="registration\_form"

                                                    name="role">

                                                    @if ($account->role == 'Admin')

                                                        <option value="Admin" selected="selected">

                                                            {{ \_\_('form.admin') }}</option>

                                                        <option value="User">

                                                            {{ \_\_('form.user') }}</option>

                                                    @else

                                                        <option value="Admin">

                                                            {{ \_\_('form.admin') }}</option>

                                                        <option value="User" selected="selected">

                                                            {{ \_\_('form.user') }}</option>

                                                    @endif

                                                </select>

                                            </div>

                                        </div>

                                        <span class="text-danger" role="alert">

                                            @error('role')

                                                {{ $message }}

                                            @enderror

                                        </span>

                                    </div>

                                    <div class="form-group row mb-2">

                                        <label for="display\_picture"

                                            class="col-sm-3 col-form-label">{{ \_\_('form.display\_picture') }}

                                            :</label>

                                        <div class="col-sm-9 d-flex align-items-center">

                                            <input type="file"

                                                class="form-control @error('display\_picture') is-invalid @enderror"

                                                id="display\_picture" placeholder="No file chosen" name="display\_picture">

                                        </div>

                                        <span class="text-danger" role="alert">

                                            @error('display\_picture')

                                                {{ $message }}

                                            @enderror

                                        </span>

                                    </div>

                                    <div class="form-group row mb-2">

                                        <label for="password-confirm"

                                            class="col-sm-3 col-form-label">{{ \_\_('form.confirm\_password') }}

                                            :</label>

                                        <div class="col-sm-9">

                                            <input type="password"

                                                class="form-control

                                        @error('password') is-invalid @enderror"

                                                id="password-confirm" name="password\_confirmation">

                                        </div>

                                    </div>

                                </div>

                            </div>

                            <div class="d-flex justify-content-center row">

                                <div class="w-auto mb-2 mt-4">

                                    <div class="col text-center" style="background-color: #f5da55 !important">

                                        <button type="submit" class="btn">

                                            {{ \_\_('form.save') }}

                                        </button>

                                    </div>

                                </div>

                            </div>

                    </div>

                    </form>

                </div>

            </div>

        </div>

    </div>

    </div>

@endsection

View above uses blade directive such as @error to show error and @if to assist some condition.

## Make Profile Controller

Type “php artisan make:controller ProfileController” in terminal and press enter.



Make profile function and pass data from account database to view.

    public function profile()

    {

        $account = Auth::user();

        $account['gender'] = Auth::user()->gender()->first()->gender\_desc;

        $account['role'] = Auth::user()->role()->first()->role\_name;

        return view('profile', ['account' => $account]);

    }

Make update function, this function will update profile data of account according to id of account\_id. It is assumed that the previous password is not shown to the profile because it has been hashed.

First it validate the data excluding the photo, then if photo is changed, then validate it. The previous photo is then deleted from storage and the new one is added to the storage. After that, the process is simillar with register, but this one doesn’t use create function, it uses update function instead.

public function update(Request $request, $id)

    {

        $this->validate($request, [

            'first\_name' => ['required', 'string', 'alpha\_dash', 'max:25'],

            'last\_name' => ['required', 'string', 'alpha\_dash', 'max:25'],

            'email' => ['required', 'string', 'email', 'max:100'],

            'role' => ['required', 'string'],

            'gender' => ['required', 'string'],

            'password' => ['required', 'string', Password::min(8), Password::min(1)->numbers(), 'confirmed'],

        ]);

        $account = Account::find($id);

        if(!$account)

        {

            return redirect(route('home'))->with('status-error', "Found no account that match id");

        }

        if($request->hasFile('display\_picture'))

        {

            $this->validate($request, [

                'display\_picture' => ['required', 'image', 'mimes:jpg,jpeg,png'],

            ]);

            if (Storage::disk('public')->exists('account/display\_picture/' . $account->display\_picture\_link)) {

                Storage::delete('account/display\_picture/' . $account->display\_picture\_link);

            }

            $file = $request->file('display\_picture');

            $ext = $file->getClientOriginalExtension();

            $filename = time().'.'.$ext;

            $file->storeAs('account/display\_picture', $filename);

            $account->display\_picture\_link = $filename;

        }

        $gender\_id = Gender::where('gender\_desc',$request->input('gender')) -> first()->gender\_id;

        $role\_id = Role::where('role\_name', $request->input('role')) -> first()->role\_id;

        $account->role\_id = $role\_id;

        $account->gender\_id = $gender\_id;

        $account->first\_name = $request->input('first\_name');

        $account->last\_name = $request->input('last\_name');

        $account->email = $request->input('email');

        $account->password = Hash::make($request->input('password'));

        $account->update();

        return redirect(route('home'))->with('status-success', "Saved");

    }

## Add to Web.php

Add new route for showing profile and updating profile inside auth middleware

Route::middleware(['auth'])->group(function (){

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

    Route::get('/profile', [ProfileController::class, 'profile'])->name('profile');

    Route::patch('/update-profile/{id}', [ProfileController::class, 'update'])->name('update-profile');

});

# Create Home Page

## Make Card Component

Make new component called card by typing “php artisan make:component card” in terminal and enter.

Go to app/View/Components/card.php and add code below:

class card extends Component

{

    public $item;

*/\*\**

*\* Create a new component instance.*

*\**

*\* @return void*

*\*/*

    public function \_\_construct($item)

    {

        $this->item = $item;

    }

This is to construct data for the card

Then go to resources/views/components/card.blade.php and add element below

<div class="card m-2" style="width: 18rem;">

    <img src="{{ asset('item\_image/item.jpg') }}" class="card-img-top p-2" alt="item image">

    <div class="card-body">

        <h5 class="card-title">{{ $item->item\_name }}</h5>

        <a href="{{ route('detail-item', ['id' => $item->item\_id]) }}" class="btn btn-primary">Detail</a>

    </div>

</div>

## Edit Home View

Go to resources/views/home.blade.php and edit inside section content:

@extends('layouts.app')

@section('content')

<div class="container">

    <div class="row justify-content-center">

        <div class="col-md">

            <div class="row p-4 mb-5 justify-content-center">

                @foreach ($items as $item)

                    <x-card :item="$item" />

                @endforeach

            </div>

            <div class="d-flex justify-content-end">

                {{ $items->links() }}

            </div>

        </div>

    </div>

</div>

@endsection

I am using @foreach directive to go through items and show it into the card component. →links() is for the pagination

## Edit Home Controller

    public function index()

    {

        $items = Item::paginate(10);;

        return view('home', ['items' => $items]);

    }

First it gets items from item model and paginate it to 10 item per page. Then pass it to view.

# Create Detail Item

## Create Detail Item View

Add detail\_item.blade.php to view.



Put elements and bootstrap component in detail. Detail will get item from controller and it will show details such as name, description, and price. When buy button is pressed, it will add item to cart.

@section('content')

    <div class="container">

        <div class="row justify-content-center">

            <div class="col-md-10 ">

                <div class="card mb-3">

                    <div class="row g-0">

                        <div class="col-md-4 p-4 d-flex justify-content-center">

                            <img src="{{ asset('item\_image/item.jpg') }}" class="img-fluid rounded border" alt="item photo">

                        </div>

                        <div class="col-md-8">

                            <div class="card-body">

                                <h3 class="card-title">{{ $item->item\_name }}</h3>

                                <div class="row mt-4">

                                    <div class="col-md-3" scope="row">{{ \_\_('Detail') }}</div>

                                    <div class="col">{{ $item->item\_desc }}</div>

                                </div>

                                <div class="row mt-3">

                                    <div class="col-md-3" scope="row">{{ \_\_('Price') }}</div>

                                    <div class="col">{{ 'Rp. ' . ' ' . $item->price }}</div>

                                </div>

                                <div class="row">

                                    <div class="col-10"></div>

                                    <div class="col-2">

                                        <form method="POST" action="{{ route('add-to-cart', ['item\_id' => $item->item\_id]) }}"

                                        id="add\_cart\_form" enctype="multipart/form-data">

                                        @csrf

                                        <button type="submit" class="btn mt-5"

                                            style="background-color: #f5da55 !important">

                                            {{ \_\_('transaction.buy') }}

                                        </button>

                                        </form>

                                    </div>

                                </div>

                            </div>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    </div>

@endsection

## Edit Controller

Add view function to HomeController. This will find id from item that is selected and will be passed to detail page. If item does not exist, it will redirect to home.

    public function view($item\_id)

    {

        $item = Item::find($item\_id);

        if ($item) {

            return view('home.detail\_item', ['item' => $item]);

        }

        return redirect(route('home'));

    }

Add new controller called CartController



Add addToCart function. This function will add item to cart (Order model) using firstOrCreate function.

class CartController extends Controller

{

    public function \_\_construct()

    {

        $this->middleware('auth');

    }

    public function addToCart($item\_id)

    {

        $item = Item::find($item\_id);

        $cart = Order::firstOrCrate([

            'account\_id' => Auth::user()->id,

            'item\_id' => $item\_id,

            'price' => $item->price

        ]);

    }

## Add to Web.php

Add this new route inside auth middleware

Route::get('item/{item\_id}', [HomeController::class, 'view'])->name('detail-item');

Route::post('add-to-cart/{item\_id}', [CartController::class, 'addToCart'])->name('add-to-cart');

# Create Cart Menu

## Create Cart Component

Create a new component called cartProduct using terminal 

Go to CartProduct in app/View/Components/CartProduct.php and add $item to construct

class cartProduct extends Component

{

    public $item;

*/\*\**

*\* Create a new component instance.*

*\**

*\* @return void*

*\*/*

    public function \_\_construct($item)

    {

        $this->item = $item;

    }

Go to resources/views/components/cart-product.blade.php and add title, price, and delete button on each item in cart layout using bootstrap and other elements. It will use item object that is passed from parent.

<div class="card mb-3">

    <div class="row g-0">

        <div class="col-md-3 d-flex justify-content-center">

            <img src="{{ asset('item\_image/item.jpg') }}" class="p-3 img-fluid rounded-start" alt="product photo"

                style="min-width: 160px; min-height: 160px;">

        </div>

        <div class="col-md-4">

            <div class="card-body h-100 d-flex align-items-center">

                <h3 class="card-title">{{ $item->item()->first()->item\_name }}</h3>

            </div>

        </div>

        <div class="col-md-3">

            <div class="card-body h-100 d-flex align-items-center">

                <div class="row">

                    <div class="col">

                        <h3 class="card-title">{{ 'Rp. ' . ' ' . $item->item()->first()->price }}</h3>

                    </div>

                </div>

            </div>

        </div>

        <div class="col-md-2">

            <div class="card-body d-flex justify-content-end h-100 align-items-center">

                <form action="{{ route('delete-from-cart', ['order\_id' => $item->order\_id]) }}" method="POST">

                    @csrf

                    @method('DELETE')

                    <button type="submit" class="btn btn-outline-danger">

                        {{ \_\_('transaction.delete') }}

                    </button>

                </form>

            </div>

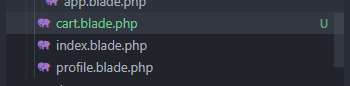
        </div>

    </div>

</div>

## Create Cart View

Add Cart blade view to view folder.



Add elements and bootstrap components.

@extends('layouts.app')

@section('content')

    <div class="container mt-5 mb-5 d-flex" style="min-height: 100%">

        <div class="d-flex justify-content-center row" style="flex:1 !important">

          <div>

            <h3 class="text-left"><u>{{ \_\_('page.cart') }}</u></h3>

          </div>

            <div class="col" style="flex:1 !important">

                <div class="row-md-10" style="min-height: 80%">

                    @foreach ($items as $item)

                        <x-cart-product :item="$item" />

                    @endforeach

                </div>

                <div class="row-md-2">

                    <div class="col-md-6">

                    </div>

                    <div class="col-md-6 ms-auto">

                        <div class="row justify-content-md-end">

                                <h5 class="text-end">{{ \_\_('transaction.total') . ('  :  Rp. ' . $totalPrice) }}</h5>

                        </div>

                        <div class="row-md-auto p-2 d-flex align-items-center justify-content-end">

                            <form method="POST" action="{{ route('check-out') }}" id="check-out"

                                enctype="multipart/form-data">

                                @csrf

                                @method('DELETE')

                                <button type="submit" class="btn mt-5" style="background-color: #f5da55 !important">

                                    {{ \_\_('transaction.check\_out') }}

                                </button>

                            </form>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    </div>

@endsection

It will loop though order, and display it on the componenets I created earlier. On the bottom page, show total price of cart. I assume when check out, the items in order for the user will be deleted. Then the order for the user will be resetted.

## Edit CartController

Go to app/Http/Controllers/CartController.php and add cart(), destroyItem(), and checkOut() functions.

public function cart()

{

    $items = Order::where('account\_id', Auth::user()->account\_id)->get();

    $totalPrice = 0;

    foreach($items as $item)

    {

        $totalPrice = $totalPrice + $item->item()->first()->price;

    }

    return view('cart', ['items' => $items, 'totalPrice' => $totalPrice]);

}

Cart function will display all items in cart. It will get all order from certain user using where. Then the function will iterate through item and add the price of the item to total price, so then the cart page can show it.

public function destroyItem($order\_id)

{

    $order = Order::find($order\_id);

    if($order){

        $order->delete();

    }

    return redirect(route('cart'));

}

This function is for deleting item in the cart, it will find the order\_id, and delete it using delete(). Then redirect to cart page.

public function checkOut()

{

    $items = Order::where('account\_id', Auth::user()->account\_id)->get();

    foreach($items as $item)

    {

        $item->delete();

    }

    return redirect(route('home'));

}

I assume that checkout will empty the order owned by a person. It will iterate each item and delete it using delete function. Finally it will redirect home.

## Add Routes to Web.php

Go to web.php and add delete-from-cart, cart, and check-out.

    Route::delete('delete-from-cart/{order\_id}', [CartController::class, 'destroyItem'])->name('delete-from-cart');

    Route::get('cart', [CartController::class, 'cart'])->name('cart');

    Route::delete('check-out', [CartController::class, 'checkOut'])->name('check-out');

# Create Account Maintenance

## Make Controller

Make controller by typing “php artisan make:controller AccountMaintenanceController” in terminal and enter



Add construct function with middleware that only allows admin to access.

    public function \_\_construct()

    {

        $this->middleware(['auth','isAdmin']);

    }

Add maintenance function. This function will get all accounts in the database and pass it to the view into a list. I have added a pagination in the list.

    public function maintenance()

    {

        $accounts = Account::paginate(10);

        return view('admin.account-maintenance', ['accounts' => $accounts]);

    }

Add updateRole function. This function will be triggered when admin click on update role on one of the account in the maintenance list. It will navigate the admin to update role page.

    public function updateRole($id)

    {

        $account = Account::find($id);

        $account['role'] = $account->role()->first()->role\_name;

        return view('admin.update-role', ['account' => $account]);

    }

Add deleteAccount function which will find the account with certain id and delete it using delete() function.

    public function deleteAccount($id)

    {

        $account = Account::find($id);

        if($account){

            $account->delete();

        }

        return redirect(route('account-maintenance'))->with('status-success', "Deleted");

    }

## Make Componenet

Make a new component called accountCard, by typing “php artisan make:component accountCard” in terminal and enter.



Go to app/View/Componenets/accountCard.php and add account to construct.

class accountCard extends Component

{

    public $account;

*/\*\**

*\* Create a new component instance.*

*\**

*\* @return void*

*\*/*

    public function \_\_construct($account)

    {

        $this->account = $account;

    }

Add elements to the account card in the component view.

<div class="card mb-3">

    <div class="row g-0">

        <div class="col-md-3 p-2 d-flex justify-content-center">

            <img src="{{ asset('storage/account/display\_picture/' . $account->display\_picture\_link) }}"

                class="img-fluid rounded border" alt="product photo" style="min-height: 180px; min-width: 180px;">

        </div>

        <div class="col-md-6">

            <div class="card-body">

                <h5 class="card-title">

                    {{ $account->first\_name . ' ' . $account->last\_name . ' - ' . $account->role()->first()->role\_name }}

                </h5>

            </div>

        </div>

        <div class="col-md-3">

            <div class="card-body d-flex justify-content-end">

                <a href="{{ route('update-role', ['account\_id' => $account->account\_id]) }}"

                    class="btn btn-outline-warning me-1">{{ \_\_('transaction.update\_role') }}</a>

                <form action="{{ route('delete-account', ['account\_id' => $account->account\_id]) }}" method="POST">

                    @csrf

                    @method('DELETE')

                    <button type="submit" class="btn btn-outline-danger">

                        {{ \_\_('transaction.delete') }}

                    </button>

                </form>

            </div>

        </div>

    </div>

</div>

## Make View

Add HTML, CSS, and bootstrap elements to the view. Foreach directive will function as iterator of the accounts object. Links() function as the paginator.

@extends('layouts.app')

@section('content')

    <div class="container mt-5 mb-5">

        <div class="d-flex justify-content-center row">

            <div class="col-md-10">

                @foreach ($accounts as $account)

                    <x-account-card :account="$account" />

                @endforeach

                <div class="d-flex justify-content-start">

                    {{ $accounts->links() }}

                </div>

            </div>

        </div>

    </div>

@endsection

## Add to Routes Web.php

All of pages maintained by admin will be wrapped inside admin middleware. The link prefix will also starts with admin.

Route::prefix('admin/')->middleware(['auth', 'isAdmin'])->group(function(){

    Route::get('/account-maintenance', [AccountMaintenanceController::class, 'maintenance'])->name('account-maintenance');

    Route::get('/update-role/{account\_id}', [AccountMaintenanceController::class, 'updateRole'])->name('update-role');

    Route::delete('/delete-account/{account\_id}', [AccountMaintenanceController::class, 'deleteAccount'])->name('delete-account');

});

# Create Change Role

## Add View

First we construct the view using HTML, CSS elements and bootstrap component. Inside this view there are select input for selecting role and form with patch method.

@extends('layouts.app')

@section('content')

    <div class="container">

        <div class="row justify-content-center">

            <div class="col-md-12">

                <div>

                    <h3 class="text-left"><u>{{ $account->first\_name . ' ' . $account->last\_name }}</u></h3>

                </div>

                <div class="row mt-4">

                    <div class="col-md-4 p-1 d-flex justify-content-center">

                        <img src="{{ asset('storage/account/display\_picture/' . $account->display\_picture\_link) }}"

                            class="img-fluid rounded border" alt="display picture">

                    </div>

                    <div class="col-md-8">

                        <form method="POST" action="{{ route('change-role', ['account\_id' => $account->account\_id]) }}"

                            id="registration\_form" enctype="multipart/form-data">

                            @method('PATCH')

                            @csrf

                            <div class="form-group d-flex justify-content-between">

                                <div class="col-6 pe-2">

                                    <div class="form-group row mb-2">

                                        <label for="role" class="col-sm-3 col-form-label">{{ \_\_('form.role') }}

                                            :</label>

                                        <div class="col-sm-9">

                                            <div class="input-group @error('role') is-invalid @enderror">

                                                <select class="form-control" id="role" form="registration\_form"

                                                    name="role">

                                                    @if ($account->role == 'Admin')

                                                        <option value="Admin" selected="selected">

                                                            {{ \_\_('form.admin') }}</option>

                                                        <option value="User">

                                                            {{ \_\_('form.user') }}</option>

                                                    @else

                                                        <option value="Admin">

                                                            {{ \_\_('form.admin') }}</option>

                                                        <option value="User" selected="selected">

                                                            {{ \_\_('form.user') }}</option>

                                                    @endif

                                                </select>

                                            </div>

                                        </div>

                                        <span class="text-danger" role="alert">

                                            @error('role')

                                                {{ $message }}

                                            @enderror

                                        </span>

                                    </div>

                                    <div class="d-flex justify-content-end row">

                                        <div class="w-auto mb-2 mt-4">

                                            <div class="col text-center" style="background-color: #f5da55 !important">

                                                <button type="submit" class="btn">

                                                    {{ \_\_('form.save') }}

                                                </button>

                                            </div>

                                        </div>

                                    </div>

                                </div>

                            </div>

                    </div>

                    </form>

                </div>

            </div>

        </div>

    </div>

    </div>

@endsection

## Add Controller

ChangeRole function will get the select input value selected by admin and update it to the account associated.

    public function changeRole(Request $request, $id)

    {

        $account = Account::find($id);

        $role\_id = Role::where('role\_name', $request->input('role')) -> first()->role\_id;

        $account->role\_id = $role\_id;

        $account->update();

        return redirect(route('account-maintenance'))->with('status-success', "Saved");

    }

## Add to Routes Web.php

Wrap the route inside admin middleware like earlier step.

Route::prefix('admin/')->middleware(['auth', 'isAdmin'])->group(function(){

    Route::get('/account-maintenance', [AccountMaintenanceController::class, 'maintenance'])->name('account-maintenance');

    Route::get('/update-role/{account\_id}', [AccountMaintenanceController::class, 'updateRole'])->name('update-role');

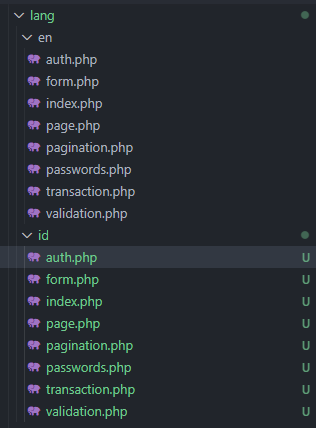
    Route::patch('/change-role/{account\_id}', [AccountMaintenanceController::class, 'changeRole'])->name('change-role');

    Route::delete('/delete-account/{account\_id}', [AccountMaintenanceController::class, 'deleteAccount'])->name('delete-account');

});

# Create Localization

## Add lang file in lang folder for en and id.



For example auth.php in lang\en will look like this

<?php

return [

    'failed' => 'These credentials do not match our records.',

    'password' => 'The provided password is incorrect.',

    'throttle' => 'Too many login attempts. Please try again in :seconds seconds.',

];

While the auth.php in lang\id will look like this

<?php

return [

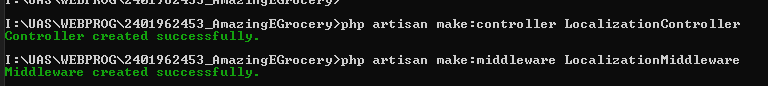
    'failed' => 'Identitas ini tidak sesuai dengan catatan kami.',

    'password' => 'Kata sandi yang diberikan salah.',

    'throttle' => 'Terlalu banyak upaya masuk. Silakan coba lagi dalam :detik detik.',

];

## Add Middleware and Controller for Localization



Go to the localization controller and add setLang function to set app and session.

    public function setLang($locale)

    {

        App::setLocale($locale);

        Session::put("locale", $locale);

        return redirect()->back();

    }

Go to the middleware for localization and add this to handle function:

    public function handle(Request $request, Closure $next)

    {

        if(Session::get("locale") != null)

        {

            App::setlocale(Session::get("locale"));

        }

        else

        {

            Session::put("locale", "en");

            App::setLocale(Session::get("locale"));

        }

        return $next($request);

    }

Then go to kernel and add

 \App\Http\Middleware\LocalizationMiddleware::class,

In middlewareGroups

Finally, add localization route to web.php

Route::get("/locale/{lange}", [LocalizationController::class, 'setLang'])->name('change-locale');

## Add toggle switch to navbar

Put these buttons on right side of navbar in app.blade.php.

*<!-- Right Side Of Navbar -->*

<ul class="navbar-nav ms-auto">

    <li class="nav-item">

<div class="btn-group btn-group-toggle me-4" data-toggle="buttons">

    <a href="{{ route('change-locale', ['lange' => 'en']) }}"

        class="btn btn btn-light language-switch" data-language="en">EN</a>

    <a href="{{ route('change-locale', ['lange' => 'id']) }}"

        class="btn btn btn-light language-switch" data-language="id">ID</a>

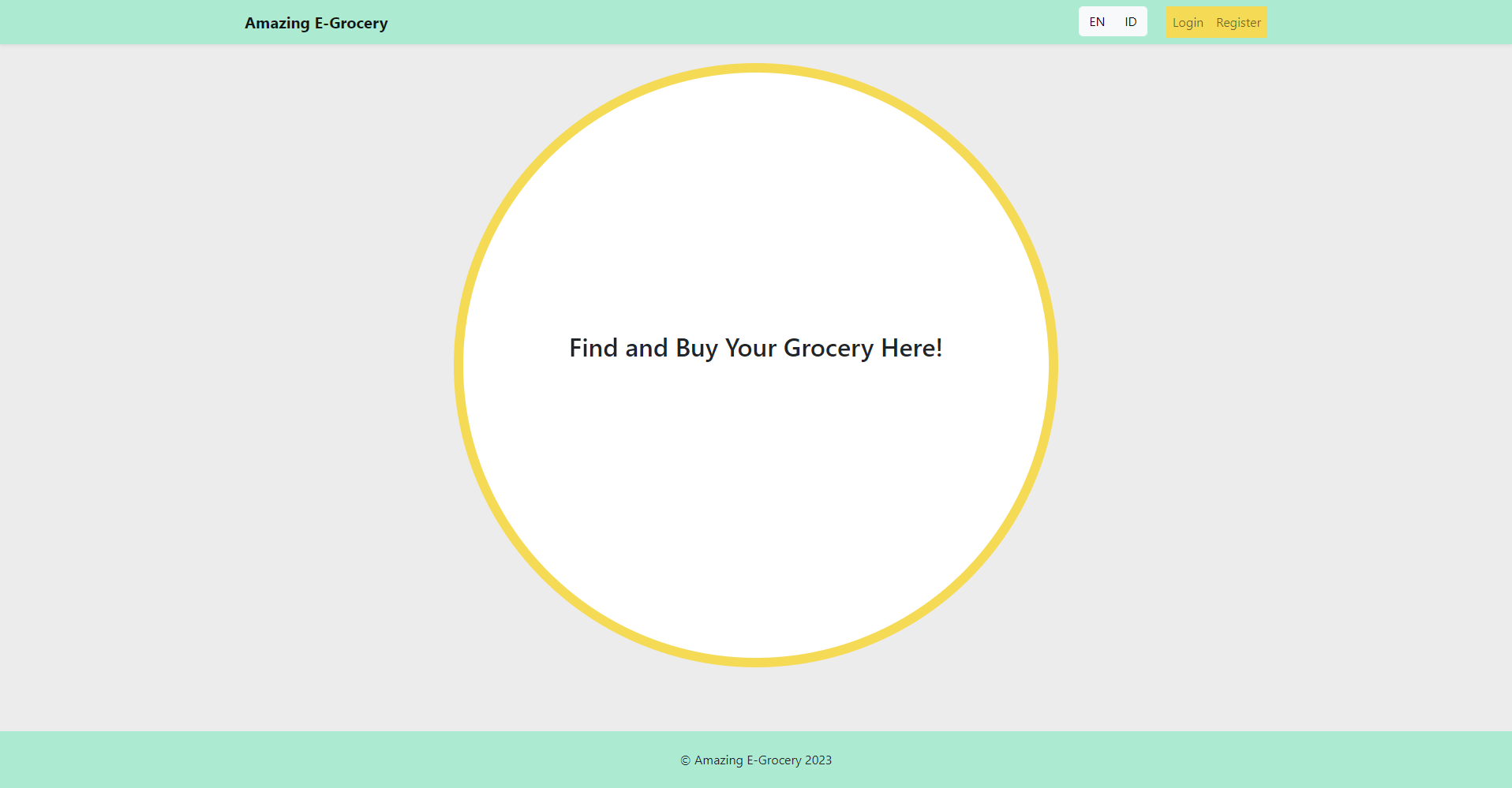
</div>

## Implement in View and Blade

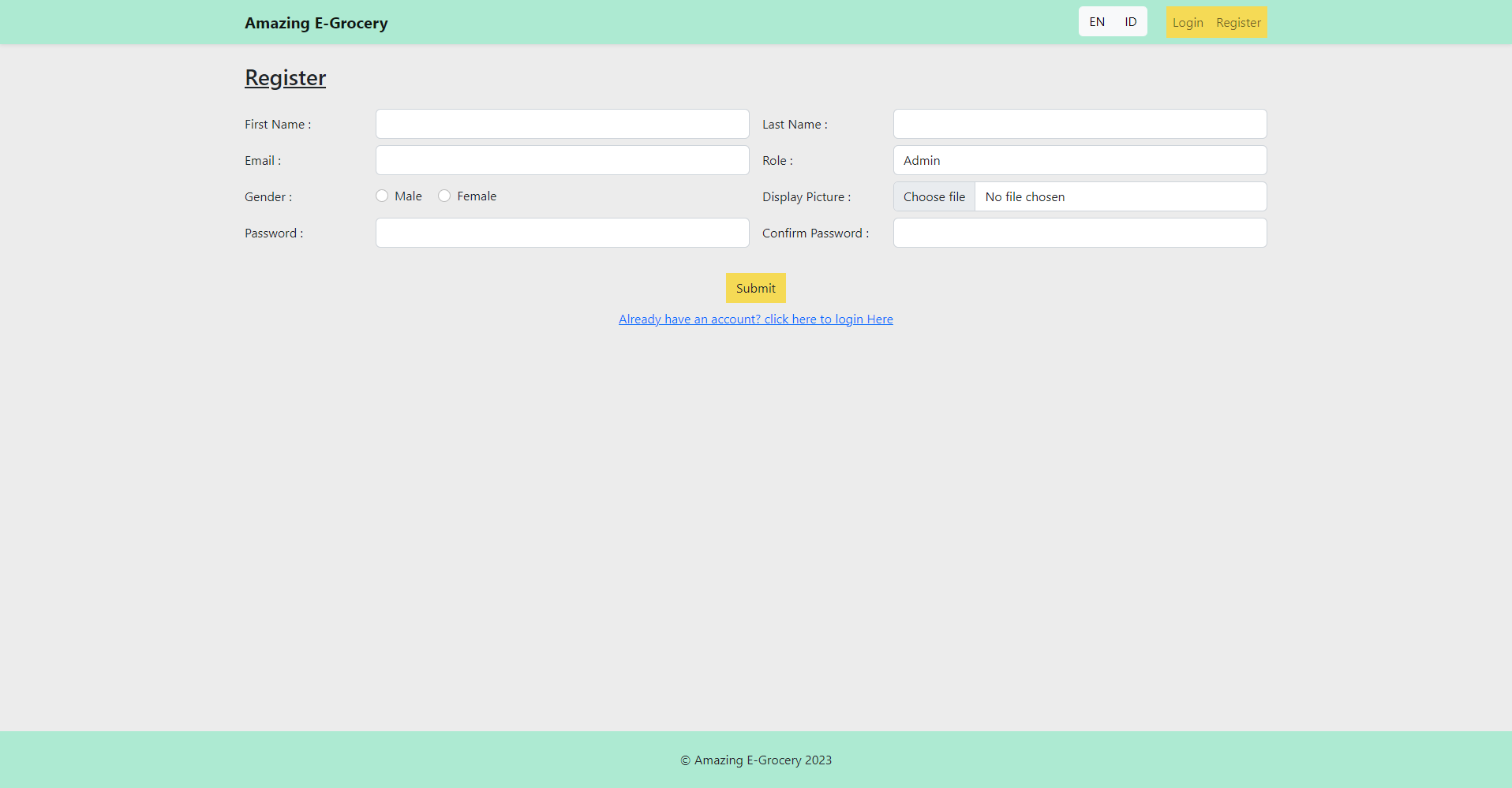
In all previous step before localization, I have also added localization support in blade and view by adding {{ \_\_('file.text') }}.

# Screenshot

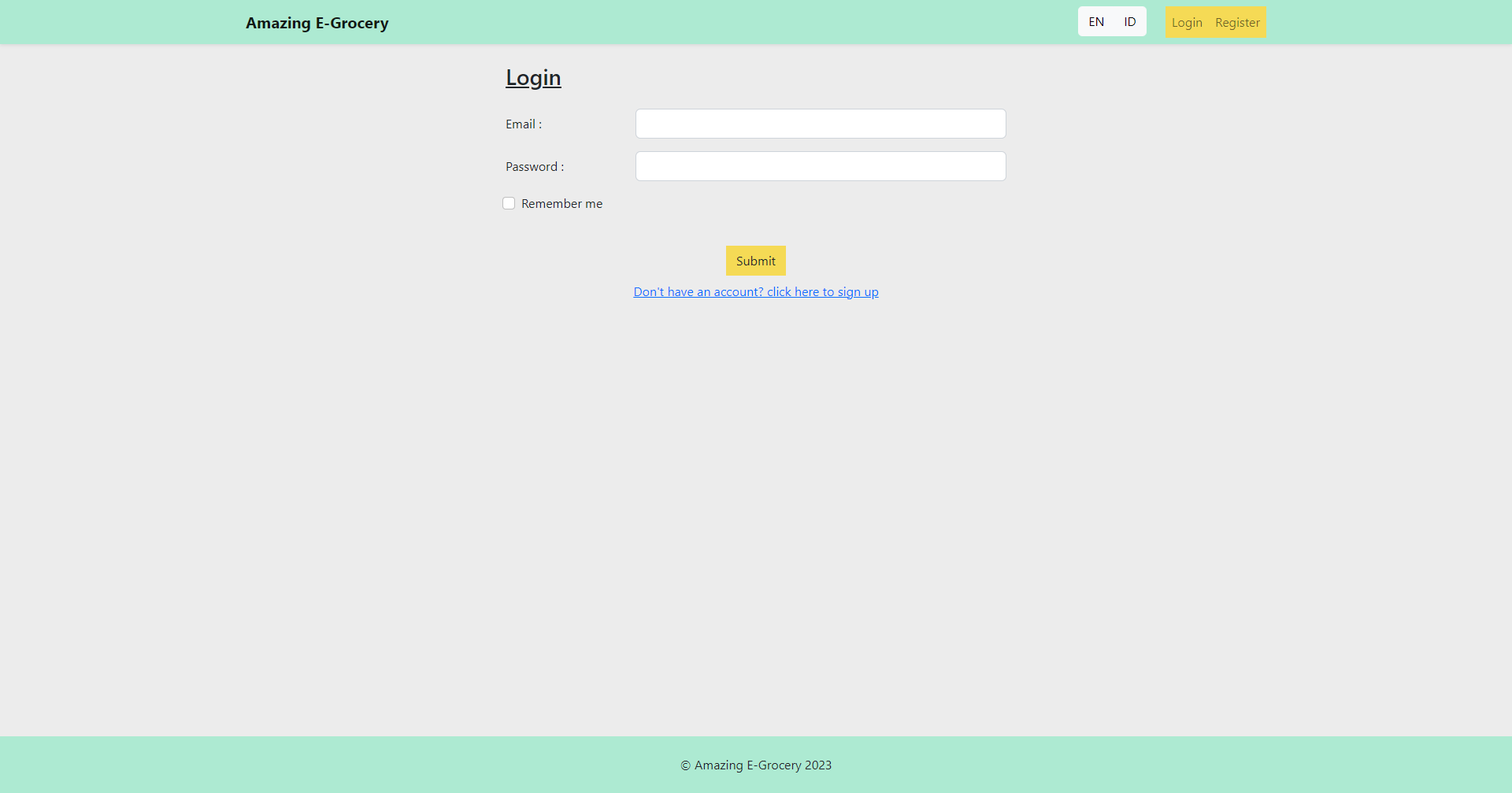
## Index



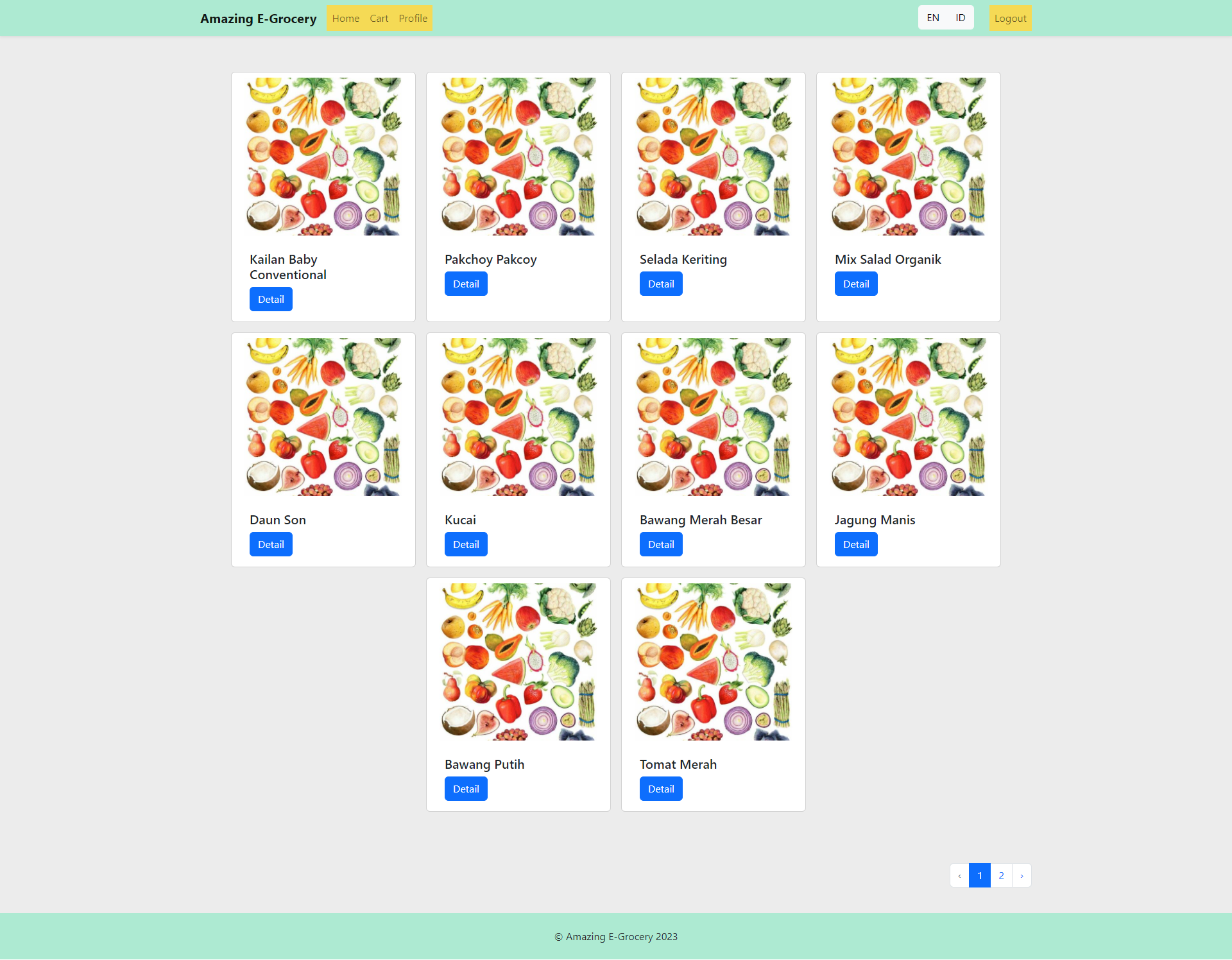
## Register



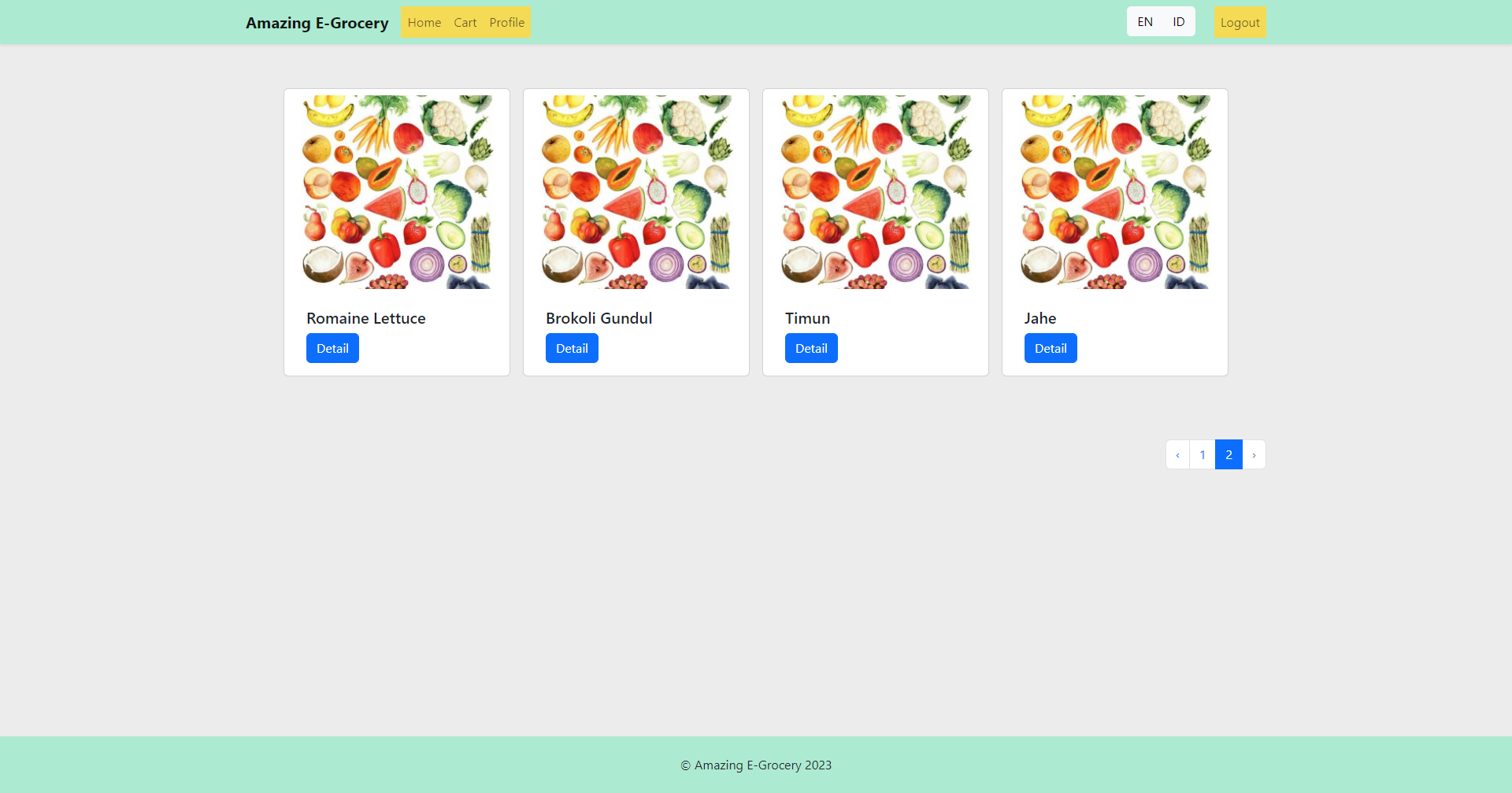
## Log In



## Home



## Home Page 2



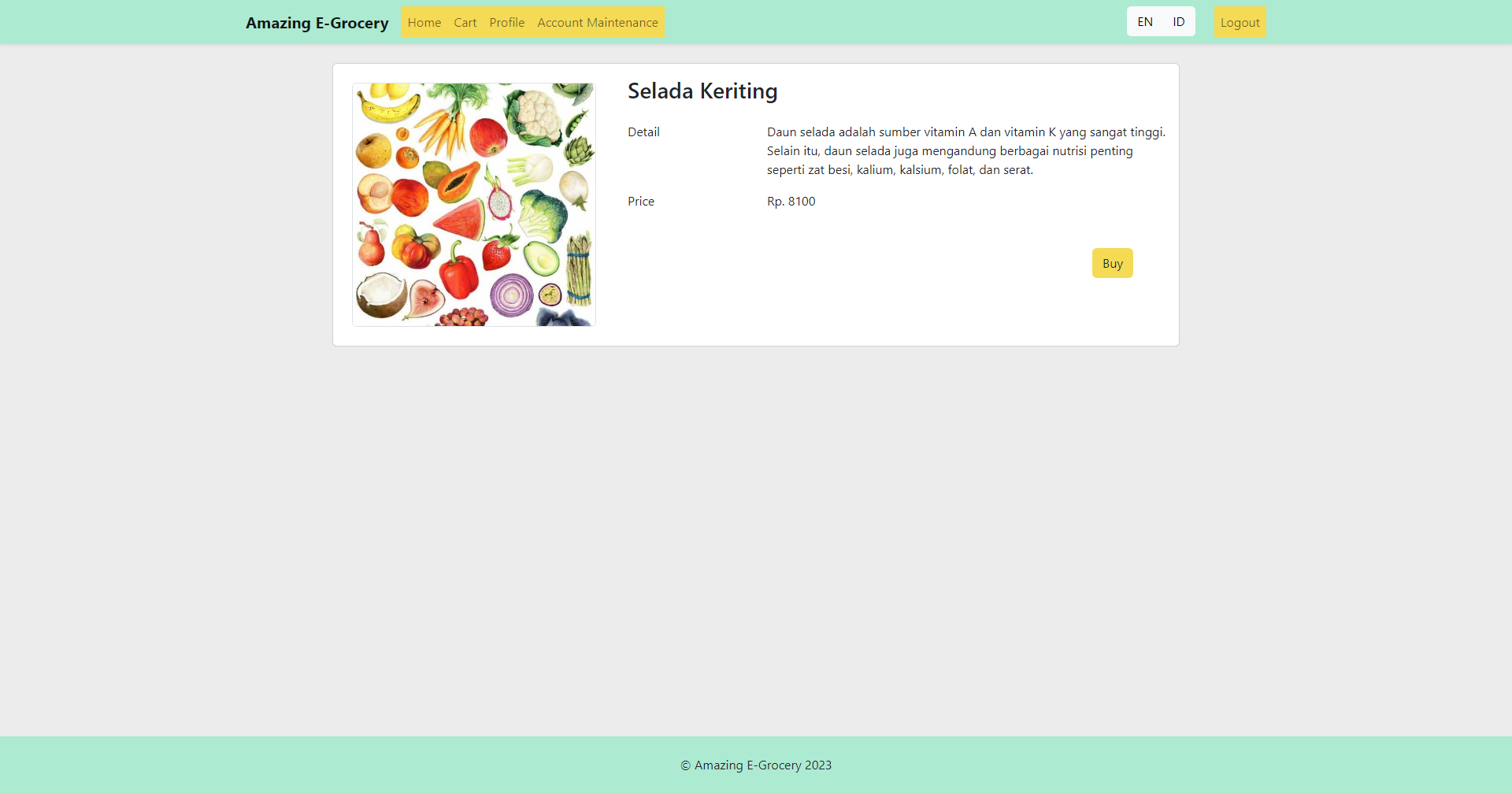
## User Navigation



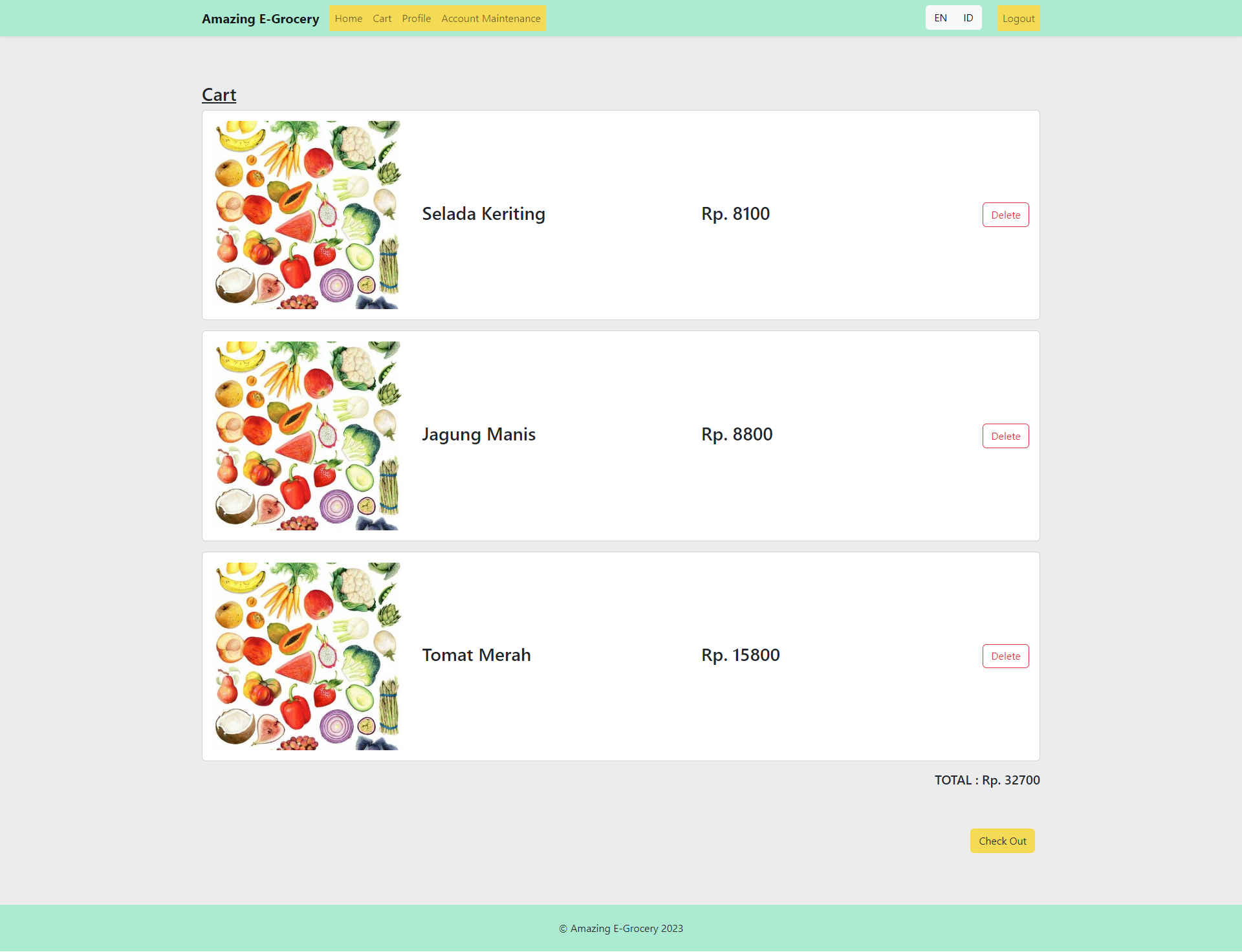
## Admin Navigation



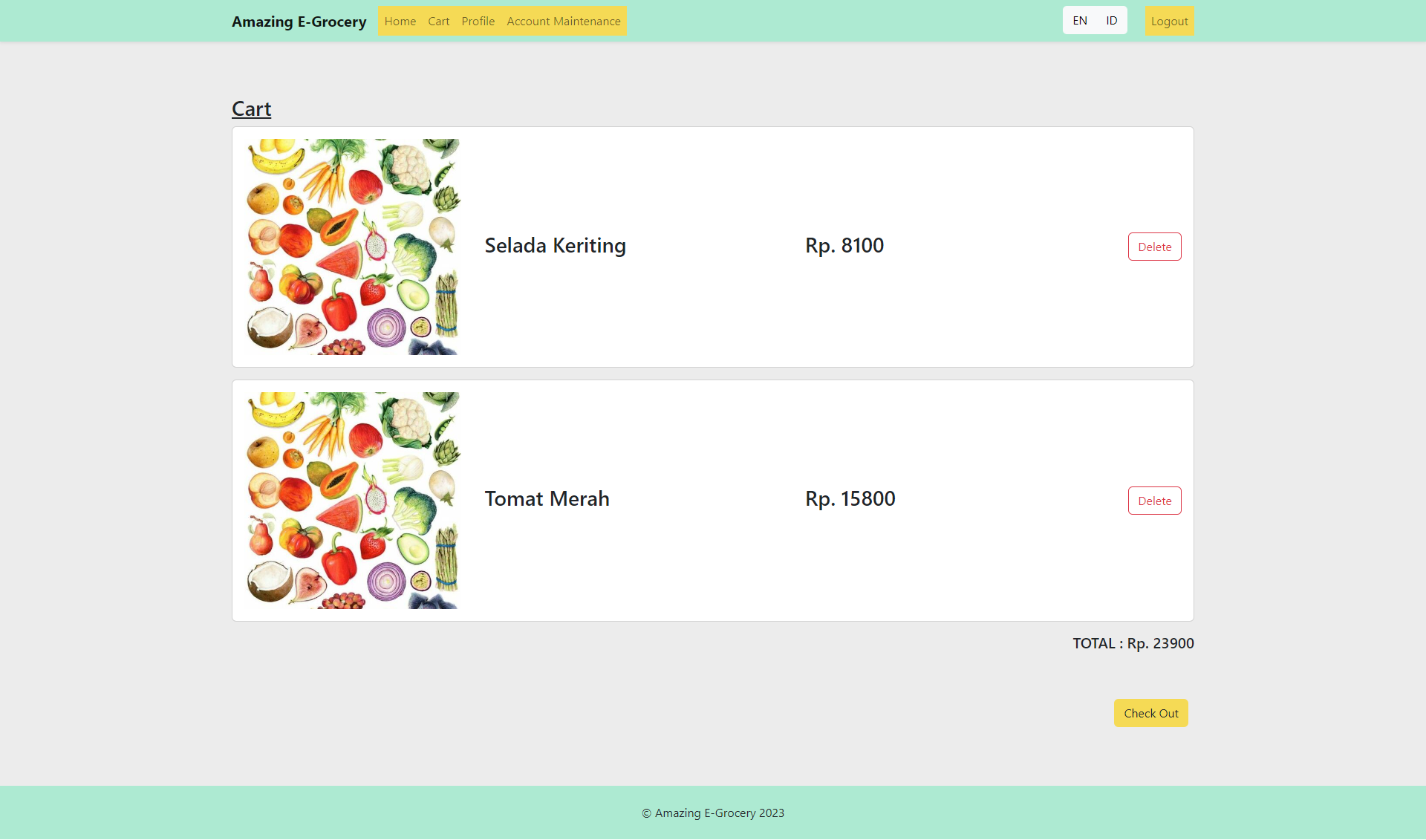
## Item Detail



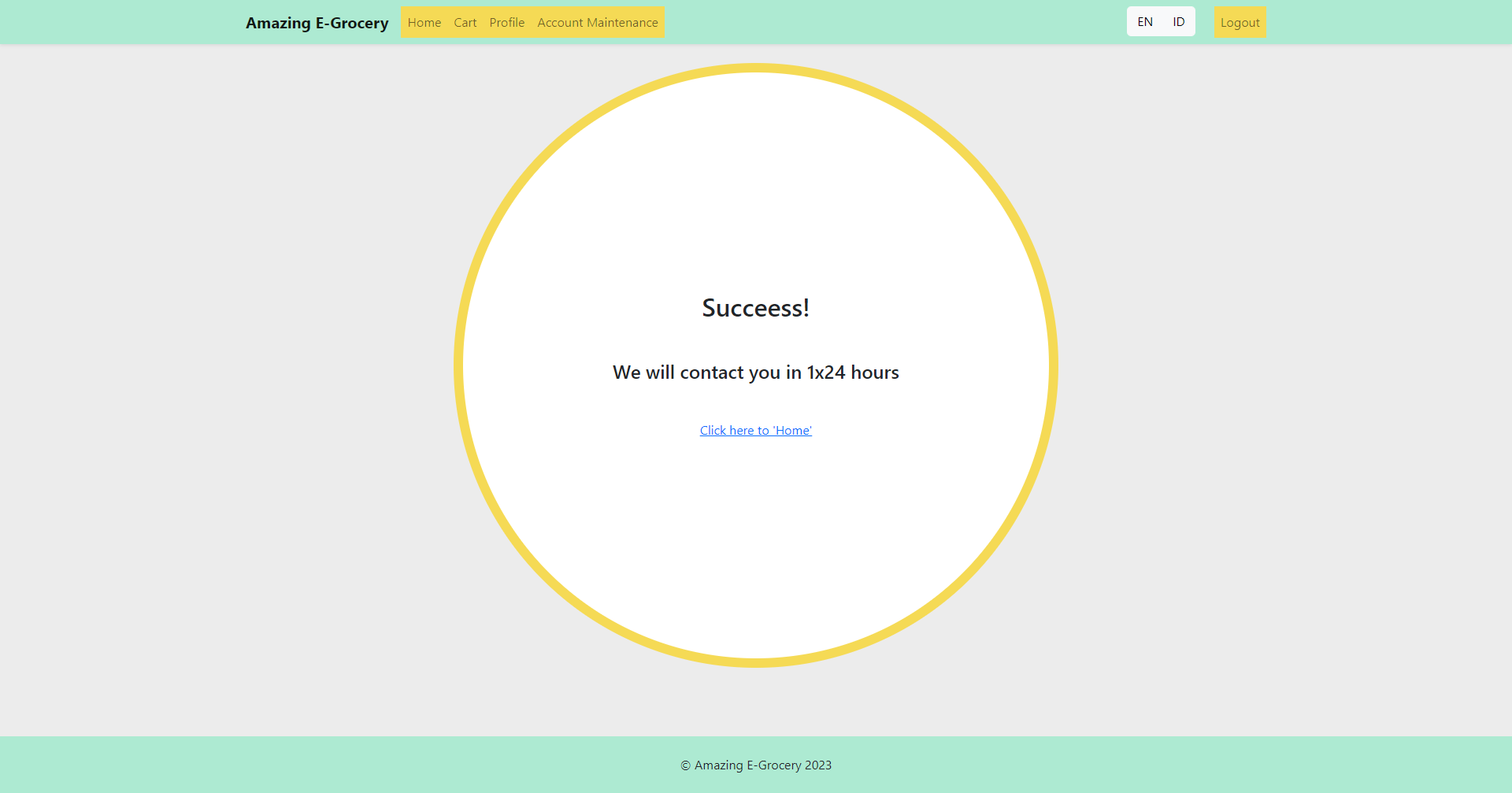
## Cart



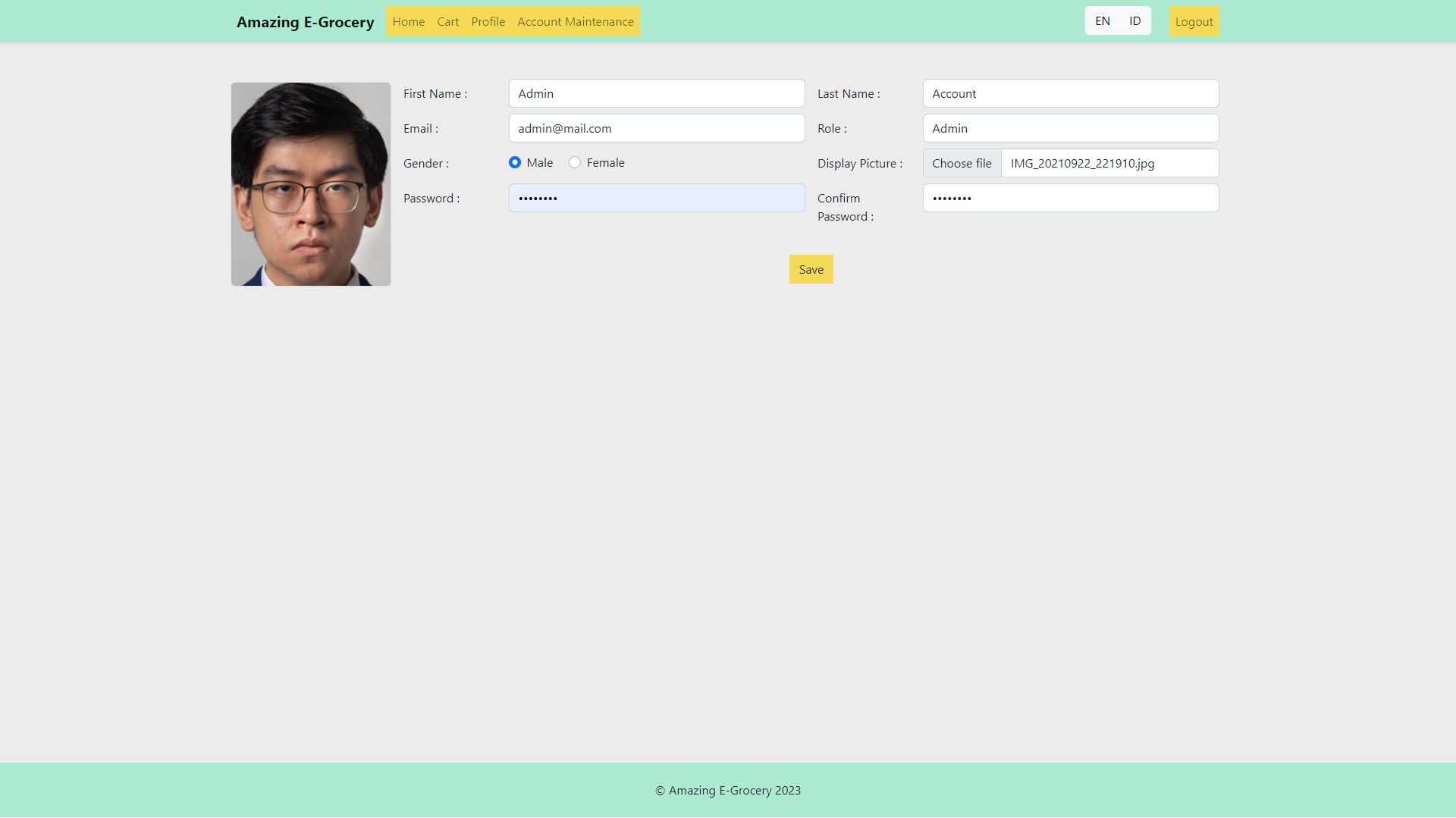
## Cart After Delete



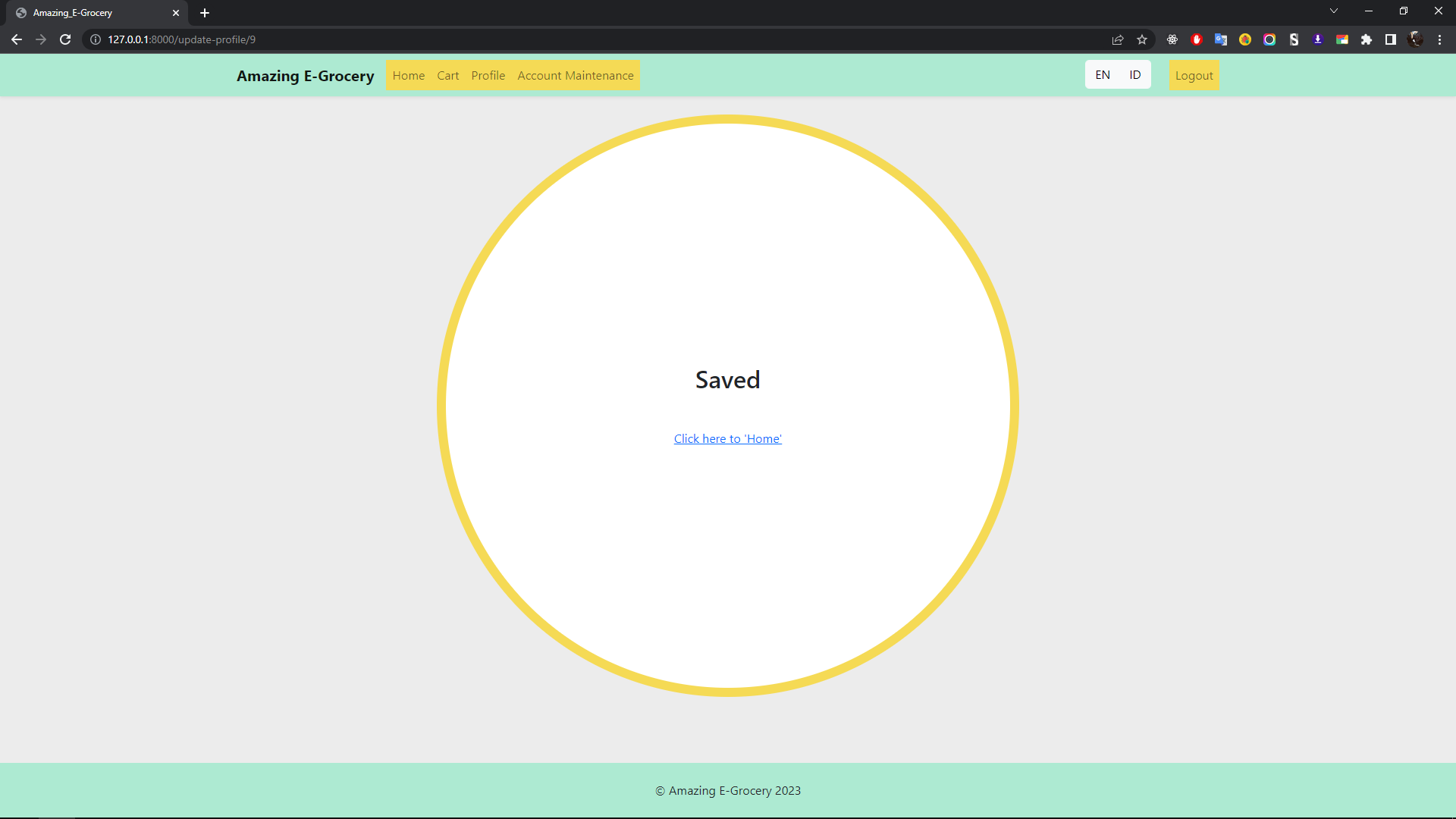
## Success Page



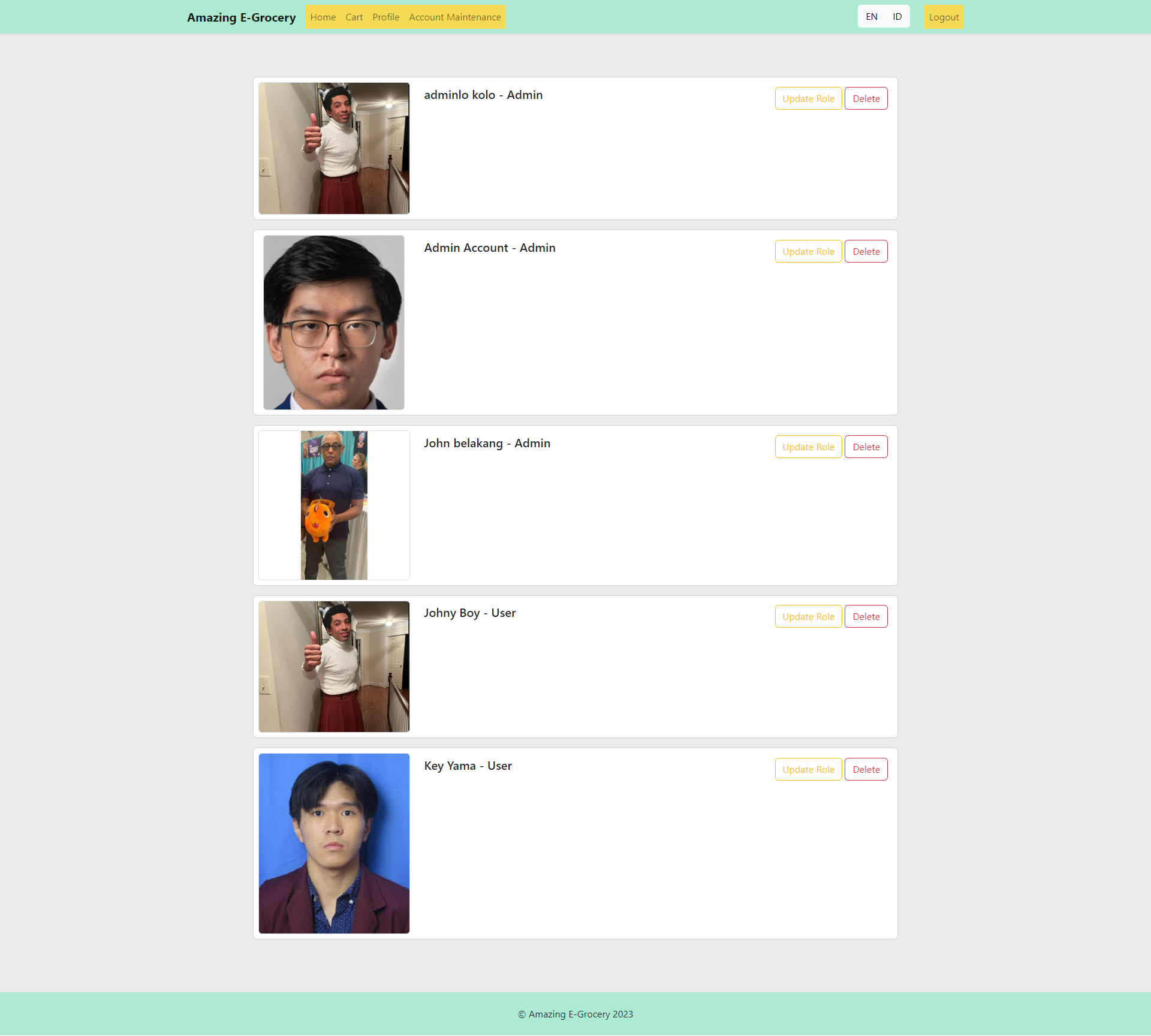
## Profile



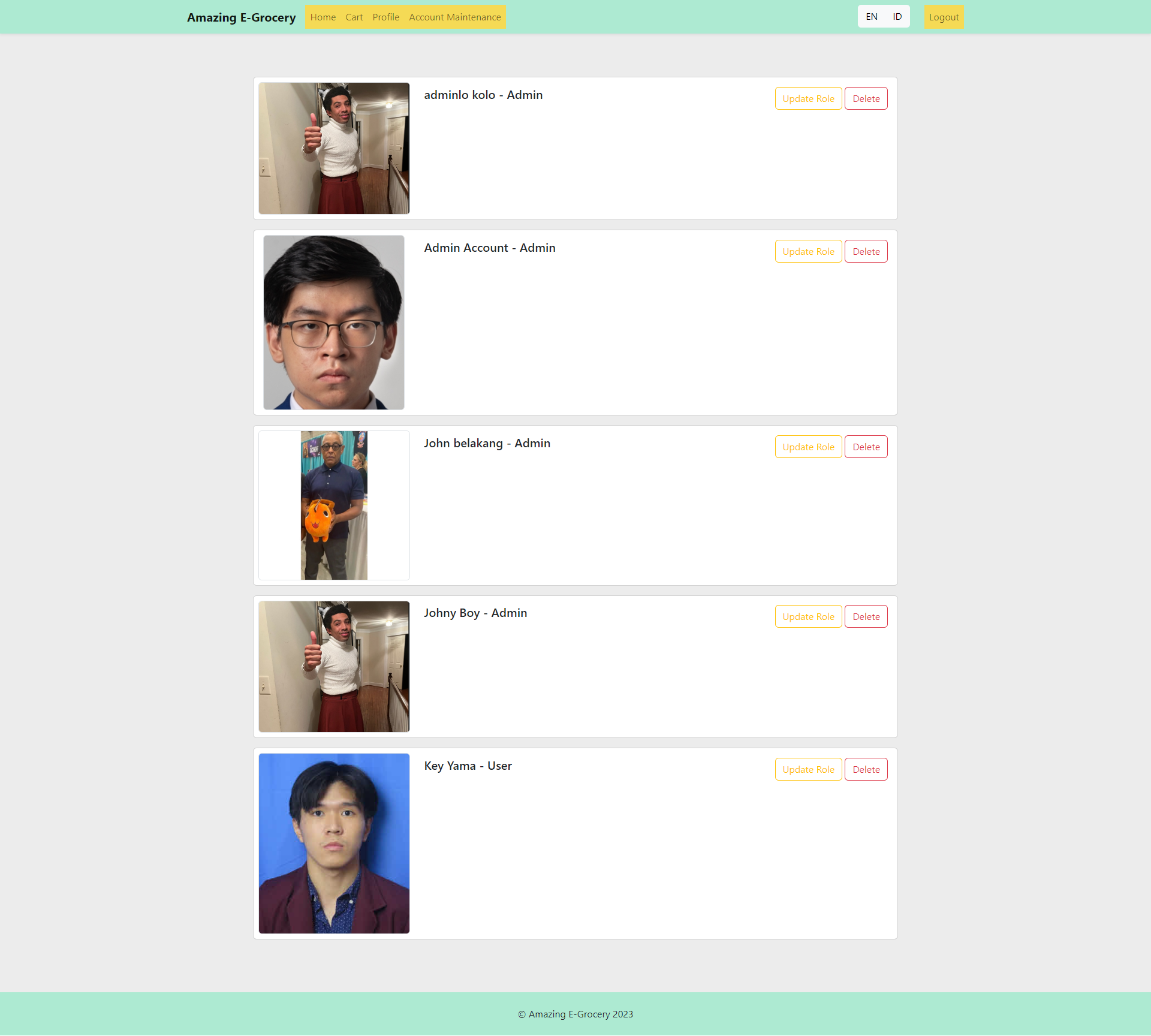
## Saved Page



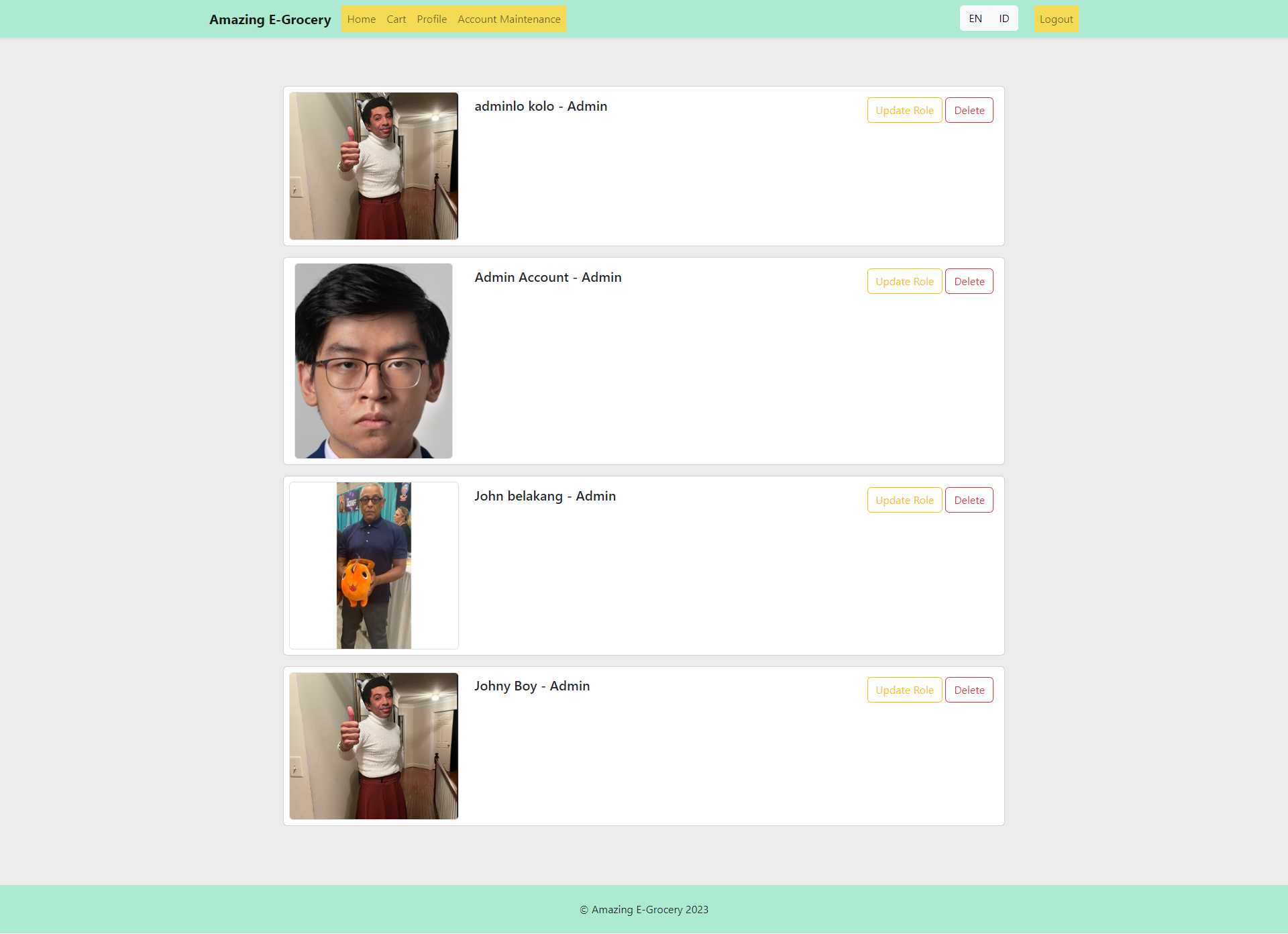
## Account Maintenance



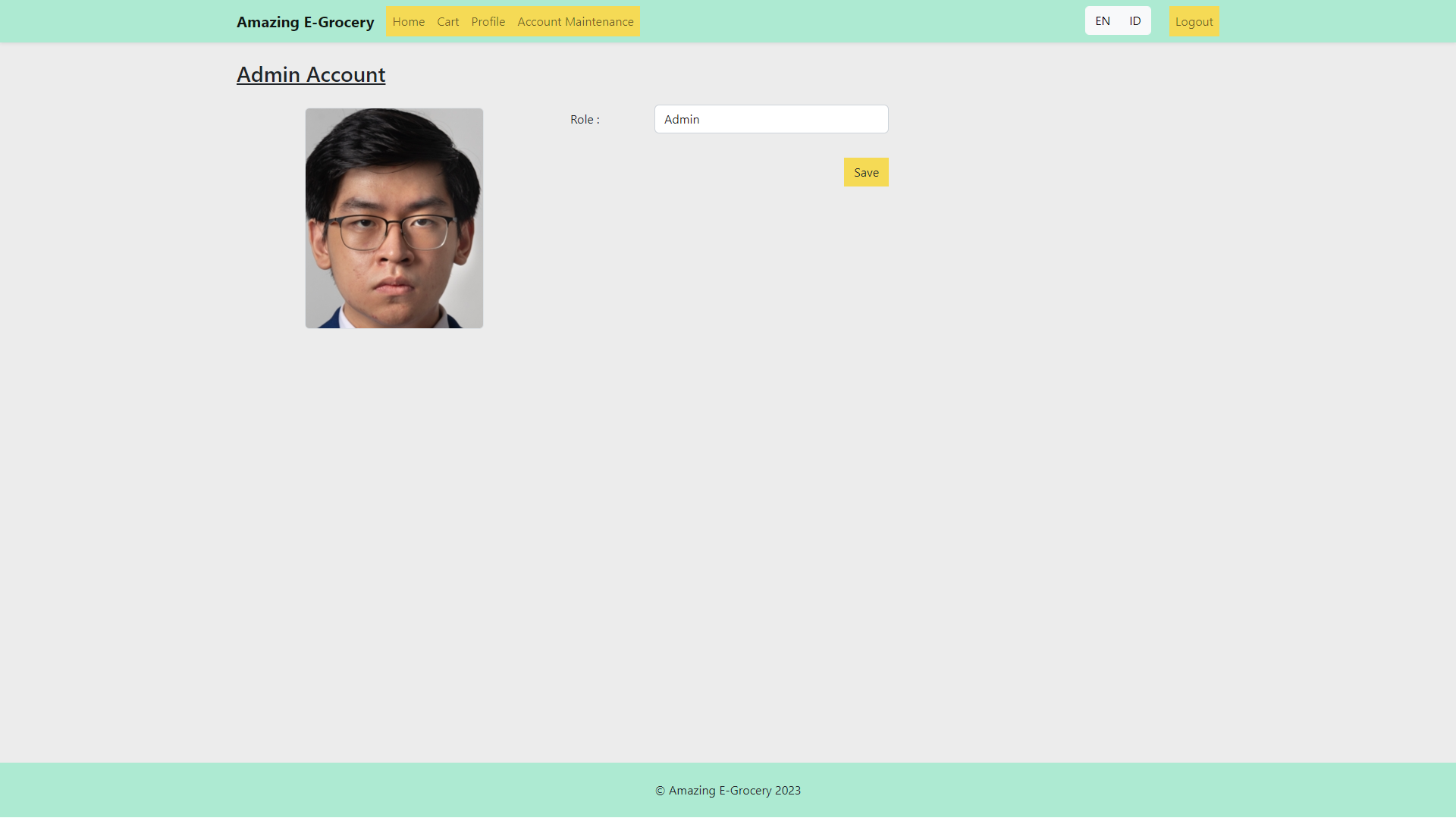
## Account Maintenane Johny Boy Changed



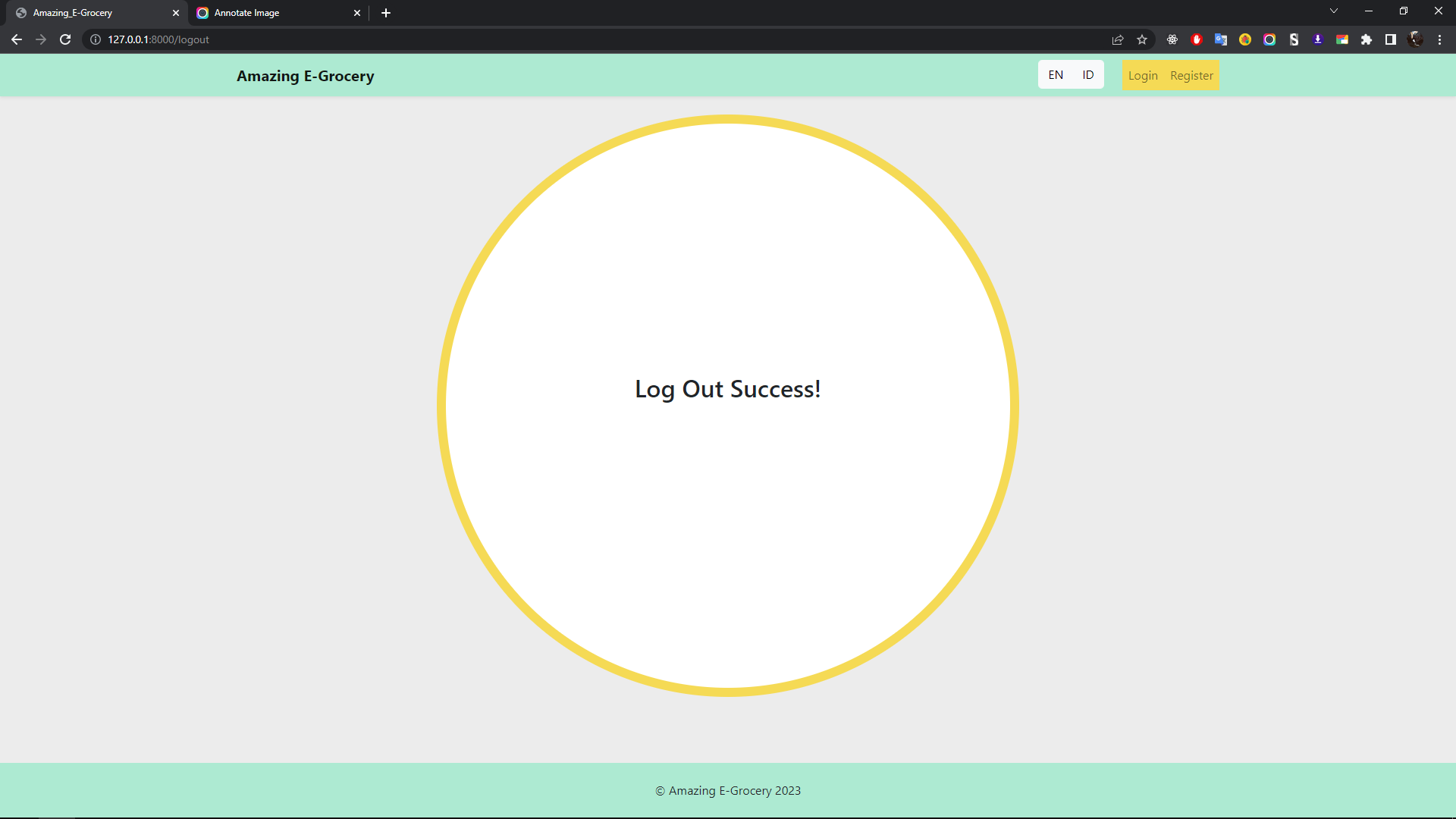
## Account Maintenance after Key Yama is deleted



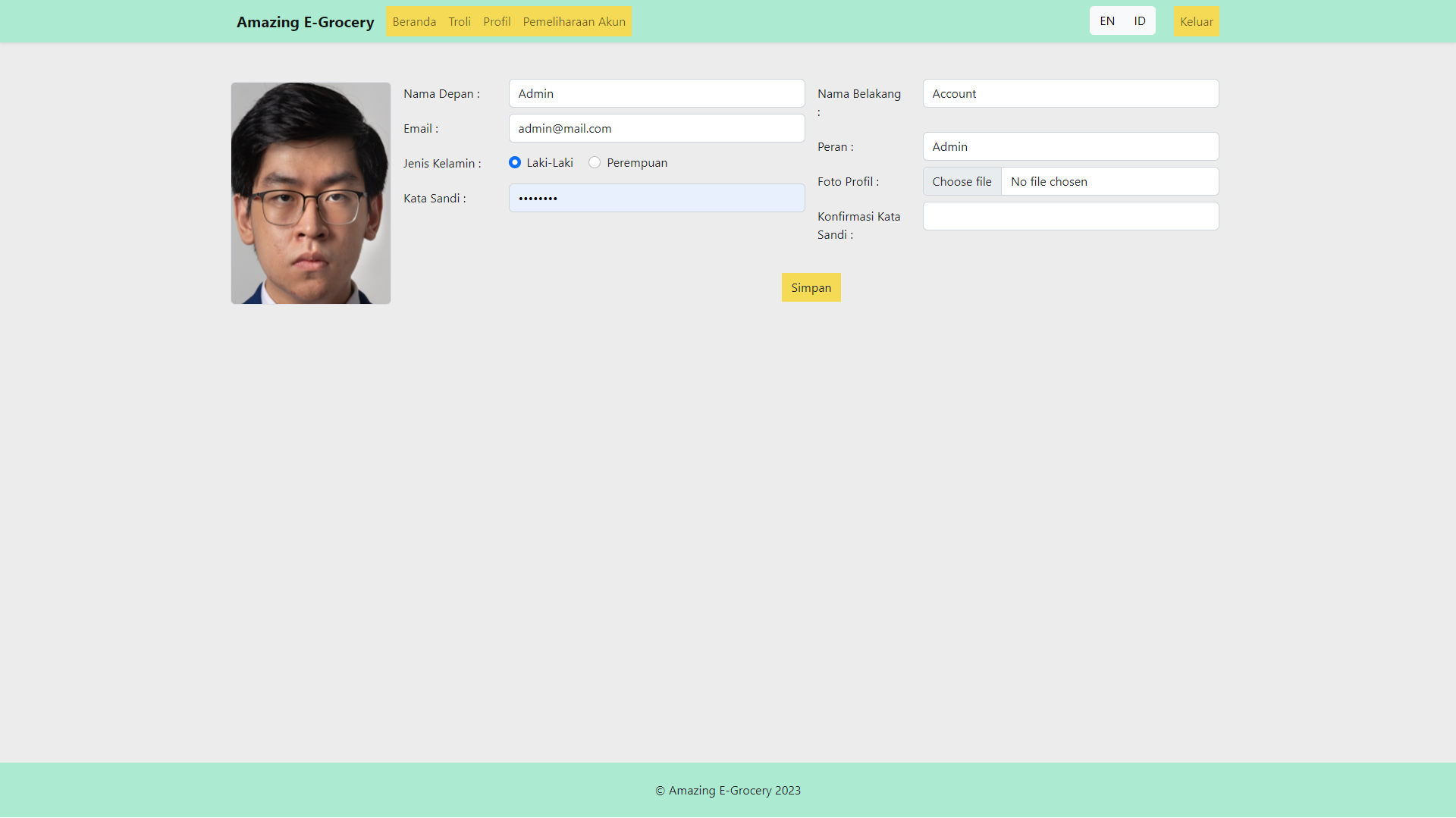
## Update Role



## Log Out Successs



## One Example Page with Indonesia Language



# Reference

## Picture and Text

1. <https://id.wikipedia.org/wiki/Buah>
2. <https://cookpad.com/id/cari/sayur>

## Documentation

1. https://laravel.com/docs/8.x/
2. https://getbootstrap.com/
3. https://getbootstrap.com/docs/5.2/components/navbar/
4. https://getbootstrap.com/docs/5.0/components/card/
5. https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js/
6. https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css/