

Code Lab 2: Framework Laravel

Daftar Isi

A.	Skema Code Lab	2
1.	<i>ERD (Entity Relational Diagram).....</i>	<i>2</i>
2.	<i>Mock Up Website</i>	<i>3</i>
B.	Installing Fresh Laravel 11	6
C.	Login dan Register (dengan Layout)	7
D.	Database Model, Migration, Seeder, dan Faker	12
1.	<i>Membuat Jabatan</i>	<i>12</i>
2.	<i>Membuat Barang</i>	<i>14</i>
3.	<i>Membuat User.....</i>	<i>16</i>
4.	<i>Membuat Inventaris</i>	<i>18</i>
5.	<i>Memanggil Seeder.....</i>	<i>21</i>
E.	Authentication + Validation.....	23
F.	Registration + Validation	25

A. Skema Code Lab

1. ERD (Entity Relational Diagram)

kita akan membuat suatu aplikasi manajemen inventaris dengan skema table seperti berikut



Dengan menggunakan skema ERD diatas

2. Mock Up Website

a. Login

A Web Page

https://localhost/auth/login.php

Form Login

Email

Password

Login

Belum punya akun ? [daftar](#)

This mockup shows a web browser window titled 'A Web Page' with the address bar displaying 'https://localhost/auth/login.php'. The main content area contains a 'Form Login' box. Inside the box, there are three input fields: 'Email' (text), 'Password' (password), and a 'Login' button. Below the inputs, there is a link that says 'Belum punya akun ? daftar'.

b. Register

A Web Page

https://localhost/auth/register.php

Form Register

Email

Password

Password Confirmation

Nama Lengkap

Jenis Kelamin

Submit

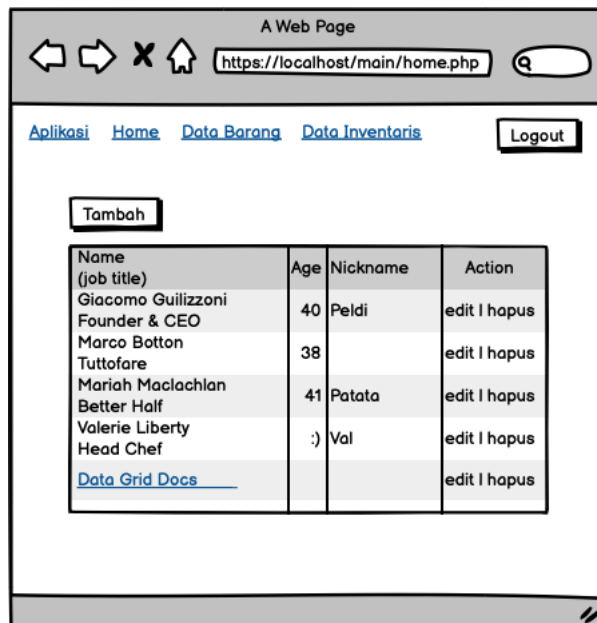
Sudah punya akun ? [Masuk](#)

This mockup shows a web browser window titled 'A Web Page' with the address bar displaying 'https://localhost/auth/register.php'. The main content area contains a 'Form Register' box. Inside the box, there are five input fields: 'Email' (text), 'Password' (password), 'Password Confirmation' (password), 'Nama Lengkap' (text), and 'Jenis Kelamin' (text). Below the inputs, there is a 'Submit' button. At the bottom of the form, there is a link that says 'Sudah punya akun ? Masuk'.

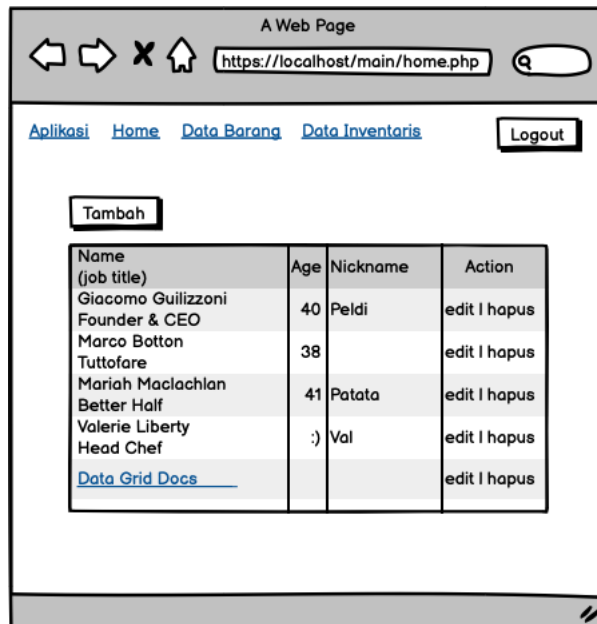
c. Home



d. Data Barang



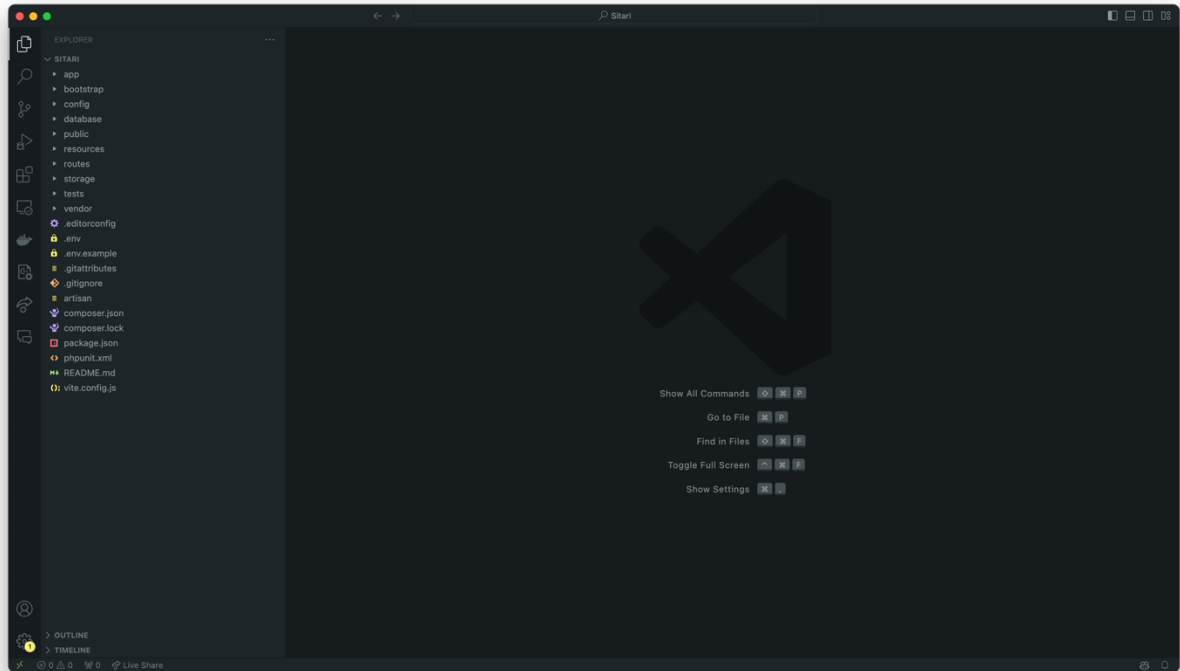
e. Data Inventaris



B. Installing Fresh Laravel 11

Install project Laravel bernama Sitari (Sistem Informasi Inventaris) menggunakan composer:

```
composer create-project laravel/laravel:^11 Sitari
```

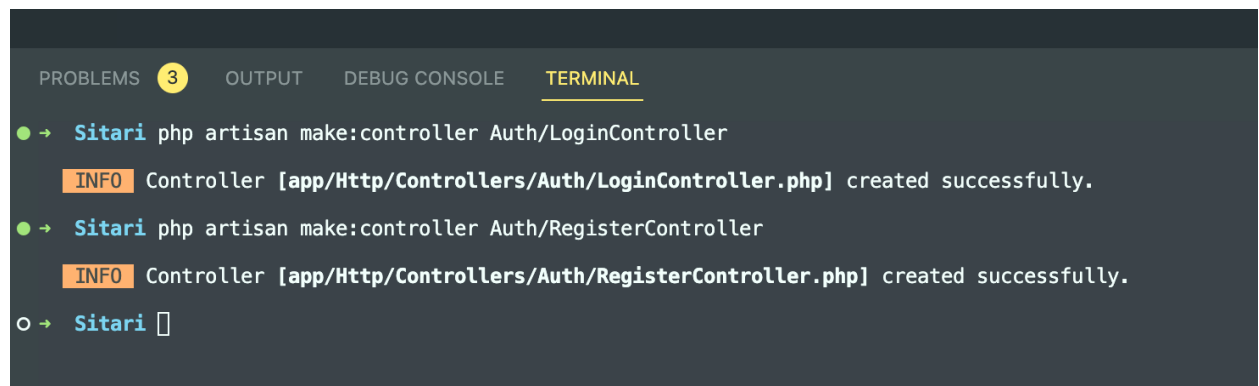


C. Login dan Register (dengan Layout)

1. Buat Controller untuk Login dan Auth

```
php artisan make:controller Auth/LoginController
```

```
php artisan make:controller Auth/RegisterController
```



The screenshot shows a terminal window with a dark background. At the top, there are tabs for 'PROBLEMS' (with a yellow circle containing the number 3), 'OUTPUT', 'DEBUG CONSOLE', and 'TERMINAL' (which is selected). The terminal shows two commands being executed by 'Sitari'. The first command is 'php artisan make:controller Auth/LoginController', followed by an 'INFO' message: 'Controller [app/Http/Controllers/Auth/LoginController.php] created successfully.' The second command is 'php artisan make:controller Auth/RegisterController', followed by another 'INFO' message: 'Controller [app/Http/Controllers/Auth/RegisterController.php] created successfully.' The prompt 'Sitari' is visible at the bottom.

2. Buat Layout untuk halaman Auth: buat folder bernama layouts pada folder resources/views/**layouts**
3. Buat file bernama **layout_auth.blade.php** dan tuliskan script seperti berikut

```
<!doctype html>
<html lang="en">

<head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
fit=no">

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet">

    <title>@yield('title')</title>

    @stack("header")
</head>

<body>
    @yield("content")

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js"></script>
```

```

    @stack('footer')
</body>

</html>

```

4. Buat folder bernama auth untuk mengelompokkan view bagian autentikasi resources/views/**auth**
5. Buat file bernama login.blade.php dan isikan script berikut

```

@extends('layouts.layout_auth')

@section('title', "Login")

@section("content")

<div class="container" style="padding-top:50px">
    <div class="row">
        <div class="col-md-4"></div>
        <div class="col-md-4">
            <h2 style="text-align: center;">Login</h2>
            <form method="post" action="{{ url('auth/login') }}">
                @csrf
                <div class="mb-3">
                    <label for="exampleInputEmail1" class="form-label">Email
address</label>
                    <input type="email" class="form-control" id="exampleInputEmail1"
aria-describedby="emailHelp" name="email">
                    <div id="emailHelp" class="form-text">
                        We'll never share your email with anyone else.
                    </div>
                </div>
                <div class="mb-3">
                    <label for="exampleInputPassword1" class="form-
label">Password</label>
                    <input type="password" class="form-control"
id="exampleInputPassword1" name="password">
                </div>
                <button type="submit" class="btn btn-primary">Submit</button>
            </form>

            <br>
            belum punya akun ? <a href="{{ route('register') }}">Register</a>
        </div>
    </div>
</div>

@endsection

```


6. Buat file bernama register.blade.php dan isikan script berikut

```
@extends('layouts.layout_auth')

@section('title', "Register")

@section("content")
<div class="container">
    <div class="row" style="padding-top: 50px">
        <div class="col-md-4"></div>
        <div class="col-md-4">
            <h2 class="text-center">Register</h2>

            @if (session("success"))
                <div class="alert alert-primary">{{ session('success') }}</div>
            @endif
            <form method="post" action="#">
                <div class="mb-3">
                    <label class="form-label">Email address</label>
                    <input type="email" name="email" class="form-control"
placeholder="Enter email">
                </div>
                <div class="mb-3">
                    <label class="form-label">Password</label>
                    <input type="password" name="password" class="form-control"
placeholder="Password">
                </div>

                <div class="mb-3">
                    <label class="form-label">Password Confirmation</label>
                    <input type="password" name="password_confirmation" class="form-
control" placeholder="Retype Password">
                </div>

                <div class="mb-3">
                    <label class="form-label">Gender</label>
                    <select name="gender" id="gender" class="form-control">
                        <option disabled selected>Jenis Kelamin</option>
                        <option value="laki">Laki-Laki</option>
                        <option value="perempuan">Perempuan</option>
                    </select>
                </div>

                <button type="submit" class="btn btn-primary">Register</button>
            </form>

            <br>
        </div>
    </div>
</div>
```

```

        sudah punya akun ? <a href="{{ url('/auth/login') }}">Login</a>
    </div>
</div>
</div>
@endsection

```

7. Sambungkan Controller Login dengan view, buka LoginController dan tambahkan script seperti berikut

```

public function index()
{
    return view("auth.login");
}

```

8. Sambungkan Controller Register dengan view, buka RegisterController dan tambahkan script seperti berikut

```

public function index()
{
    return view("auth.register");
}

```

9. Buat Route dan hubungkan dengan Controller dengan mengubah script **web.php** menjadi seperti berikut

```

<?php

use App\Http\Controllers\Auth\LoginController;
use App\Http\Controllers\Auth\RegisterController;
use Illuminate\Support\Facades\Route;

Route::redirect('/', '/auth/login');
Route::prefix('auth')->group(function () {
    Route::get('login', [LoginController::class, 'index'])->name('login');
    Route::get('register', [RegisterController::class, 'index'])->name('register');
});

```

Login

Email address

We'll never share your email with anyone else.

Password

Submit

belum punya akun ? [Register](#)

Register

Email address

Password

Password Confirmation

Gender

Register

sudah punya akun ? [Login](#)

D. Database Model, Migration, Seeder, dan Faker

Buat database bernama **sitari**, dan sesuaikan konfigurasi database pada **.env**
Kita akan membuat 4 tabel seperti skema diatas yaitu table

1. Membuat Jabatan

- a. `php artisan make:model Jabatan -f -s -m`

```
● → Sitari php artisan make:model Jabatan -f -s -m
INFO Model [app/Models/Jabatan.php] created successfully.
INFO Factory [database/factories/JabatanFactory.php] created successfully.
INFO Migration [database/migrations/2024_06_27_001839_create_jabatans_table.php] created successfully.
INFO Seeder [database/seeder/JabatanSeeder.php] created successfully.
```

- b. Setting Migration

Ubah fungsi **up()** dan **down()** menjadi seperti berikut

```
/**
 * Run the migrations.
 */
public function up(): void
{
    Schema::create('jabatan', function (Blueprint $table) {
        $table->id('jab_id');
        $table->string('jab_nama', 50);
        $table->timestamps();
    });
}

/**
 * Reverse the migrations.
 */
public function down(): void
{
    Schema::dropIfExists('jabatan');
}
```

- c. Setting model Jabatan seperti berikut, tambahkan **\$primaryKey** dan **\$table**

```
<?php

namespace App\Models;

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

class Jabatan extends Model
{
    use HasFactory;

    protected $table = 'jabatan';
    protected $primaryKey = 'jab_id';
}
```

- d. Setting seeder jabatan

```
<?php

namespace Database\Seeders;

use App\Models\Jabatan;
use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;

class JabatanSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        Jabatan::create(['jab_nama' => 'admin']);
        Jabatan::create(['jab_nama' => 'staff']);
    }
}
```

- e. Setting factory jabatan
Tidak menggunakan faker, karena jabatannya Cuma 2

2. Membuat Barang

a. `php artisan make:model Barang -f -s -m`

```
● → Sitari php artisan make:model Barang -f -s -m
INFO Model [app/Models/Barang.php] created successfully.
INFO Factory [database/factories/BarangFactory.php] created successfully.
INFO Migration [database/migrations/2024_06_27_002355_create_barangs_table.php] created successfully.
INFO Seeder [database/seeder/BarangSeeder.php] created successfully.
```

b. Setting migration

```
/**
 * Run the migrations.
 */
public function up(): void
{
    Schema::create('barang', function (Blueprint $table) {
        $table->id('barang_id');
        $table->string("barang_nama");
        $table->string("barang_vendor");
        $table->enum("barang_jenis", ['smartphone', 'laptop', 'tv']);
        $table->text("barang_spesifikasi");
        $table->timestamps();

    });
}

/**
 * Reverse the migrations.
 */
public function down(): void
{
    Schema::dropIfExists('barang');
}
```

c. Setting model

```
<?php

namespace App\Models;

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

class Barang extends Model
```

```
{
    use HasFactory;

    protected $table = 'barang';
    protected $primaryKey = 'barang_id';
    protected $fillable = ['barang_nama', 'barang_harga', 'barang_stok'];
}
```

d. Setting seeder

```
<?php

namespace Database\Seeders;

use App\Models\Barang;
use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;

class BarangSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        Barang::factory(10)->create();
    }
}
```

e. Setting factory

```
public function definition(): array
{
    return [
        'barang_nama' => $this->faker->name,
        'barang_vendor' => $this->faker->company,
        'barang_jenis' => $this->faker->randomElement(['smartphone', 'laptop',
'tv']),
        'barang_spesifikasi' => $this->faker->text,
    ];
}
```

3. Membuat User

Karena secara default User model, UserFactory, dan User Migration sudah tersedia, maka kita hanya perlu memodifikasi dan membuat User Seeder

- a) `php artisan make:seeder UserSeeder`
- b) Setting migration
pada `/database/migrations/0001_01_01_000000_create_users_table.php` update Schema user menjadi berikut:

Function up()

```
Schema::create('user', function (Blueprint $table) {  
    $table->id('user_id');  
    $table->foreignId('user_jab_id')->nullable()->constraint('jabatan',  
'jab_id');  
    $table->string('fullname')->nullable();  
    $table->string('email')->unique();  
    $table->timestamp('email_verified_at')->nullable();  
    $table->string('password');  
    $table->enum('gender', ['laki', 'perempuan'])->nullable();  
    $table->rememberToken();  
    $table->timestamps();  
});
```

Function down()

```
Schema::dropIfExists('user');
```

- c) Setting model


```

class User extends Authenticatable
{
    use HasFactory, Notifiable;

    protected $table = 'user';
    protected $primaryKey = 'user_id';

    /**
     * The attributes that are mass assignable.
     *
     * @var array<int, string>
     */
    protected $fillable = [
        // 'name', // hilangkan name
        'email',
        'password',
        'fullname',
        'gender',
        'user_jab_id'
    ];
}

```

d) Setting seeder

```

<?php

namespace Database\Seeders;

use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;
use App\Models\User;

class UserSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        User::factory(10)->create();
    }
}

```

Hapus data default User::create pada database/seeder/DatabaseSeeder.php

e) Setting factory

```
public function definition(): array
{
    $listJabatan = Jabatan::all()->pluck('jab_id')->toArray();
    return [
        'fullname' => fake()->name(),
        'email' => fake()->unique()->safeEmail(),
        'email_verified_at' => now(),
        'user_jab_id' => fake()->randomElement($listJabatan),
        'gender' => fake()->randomElement(['laki', 'perempuan']),
        'password' => static::$password ??= Hash::make('password'),
        'remember_token' => Str::random(10),
    ];
}
```

4. Membuat Inventaris

a) `php artisan make:model Inventaris -f -s -m`

```
● → Sitari php artisan make:model Inventaris -f -s -m
INFO Model [app/Models/Inventaris.php] created successfully.
INFO Factory [database/factories/InventarisFactory.php] created successfully.
INFO Migration [database/migrations/2024_06_27_005238_create_inventaris_table.php] created successfully.
INFO Seeder [database/seeds/InventarisSeeder.php] created successfully.
```

b) Setting migration

```
/**
 * Run the migrations.
 */
public function up(): void
{
    Schema::create('inventaris', function (Blueprint $table) {
        $table->id('inv_id');
        $table->unsignedBigInteger('inv_user_id');
        $table->unsignedBigInteger('inv_barang_id');
        $table->string('lokasi');
        $table->enum('inv_status', ['aktif', 'rusak', 'hilang', 'dilelang']);
        $table->date('inv_tgl_beli');
        $table->timestamps();

        $table->foreign('inv_barang_id')->references('barang_id')->on('barang')->onDelete('cascade');
        $table->foreign('inv_user_id')->references('user_id')->on('user')->onDelete('cascade');

    });
}

/**
 * Reverse the migrations.
 */
public function down(): void
{
    Schema::dropIfExists('inventaris');
}
```

c) Setting model

```
<?php

namespace App\Models;

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

class Inventaris extends Model
{
    use HasFactory;

    protected $table = 'inventaris';
    protected $primaryKey = 'inv_id';
}
```

d) Setting seeder

```
<?php

namespace Database\Seeders;

use App\Models\Inventaris;
use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;

class InventarisSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        Inventaris::factory(40)->create();
    }
}
```

e) Setting factory

```
public function definition(): array
{
    return [
        "inv_user_id" => $this->faker->randomElement(range(1, 10)),
        "inv_barang_id" => $this->faker->randomElement(range(1, 10)),
        "lokasi" => $this->faker->address,
        "inv_status" => $this->faker->randomElement(['aktif', 'rusak', 'hilang',
        'dilelang']),
        "inv_tgl_beli" => $this->faker->date,
    ];
}
```

5. Memanggil Seeder

- a. Memanggil fungsi seeder pada database/seeder/**DatabaseSeeder.php**, kemudian ubah menjadi seperti berikut

```
public function run(): void
{
    $this->call([
        JabatanSeeder::class,
        BarangSeeder::class,
        UserSeeder::class,
        InventarisSeeder::class,
    ]);
}
```

- b. Jalankan `php artisan migrate:refresh --seed` akan generate table beserta isinya

```

● → Sitari php artisan migrate:refresh --seed

INFO Rolling back migrations.

2024_06_27_005238_create_inventaris_table ..... 19.79ms DONE
2024_06_27_002355_create_barangs_table ..... 8.43ms DONE
2024_06_27_001839_create_jabatans_table ..... 7.31ms DONE
0001_01_01_000002_create_jobs_table ..... 21.10ms DONE
0001_01_01_000001_create_cache_table ..... 12.52ms DONE
0001_01_01_000000_create_users_table ..... 21.15ms DONE

INFO Running migrations.

0001_01_01_000000_create_users_table ..... 60.46ms DONE
0001_01_01_000001_create_cache_table ..... 21.58ms DONE
0001_01_01_000002_create_jobs_table ..... 57.65ms DONE
2024_06_27_001839_create_jabatans_table ..... 11.03ms DONE
2024_06_27_002355_create_barangs_table ..... 15.63ms DONE
2024_06_27_005238_create_inventaris_table ..... 78.37ms DONE

INFO Seeding database.

Database\Seeders\JabatanSeeder ..... RUNNING
Database\Seeders\JabatanSeeder ..... 9 ms DONE

Database\Seeders\BarangSeeder ..... RUNNING
Database\Seeders\BarangSeeder ..... 42 ms DONE

Database\Seeders\UserSeeder ..... RUNNING
Database\Seeders\UserSeeder ..... 291 ms DONE

Database\Seeders\InventarisSeeder ..... RUNNING
Database\Seeders\InventarisSeeder ..... 110 ms DONE
○ → Sitari █

```

E. Authentication + Validation

Laravel telah menyediakan fitur Auth untuk autentikasi secara mudah, berikut langkahnya

1. Pastikan views/auth/**login.blade.php** sudah memiliki **action** dan **@csrf**

```
resources > views > auth > login.blade.php > ...
1  @extends('layouts.layout_auth')
2
3  @section('title', "Login")
4
5  @section("content")
6
7  <div class="container" style="padding-top:50px">
8      <div class="row">
9          <div class="col-md-4"></div>
10         <div class="col-md-4">
11             <h2 class="text-center">Login</h2>
12             <form method="post" action="{{ url('auth/login') }}">
13                 @csrf
14                 <div class="mb-3">
15                     <label for="exampleInputEmail1" class="form-label">Email address</label>
16                     <input type="email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp" name="email">
17                     <div id="emailHelp" class="form-text">
18                         We'll never share your email with anyone else.
19                     </div>
20                 </div>
21                 <div class="mb-3">
22                     <label for="exampleInputPassword1" class="form-label">Password</label>
23                     <input type="password" class="form-control" id="exampleInputPassword1" name="password">
24                 </div>
25                 <button type="submit" class="btn btn-primary">Submit</button>
26             </form>
27
28             <br>
29             belum punya akun ? <a href="{{ route('register') }}">Register</a>
30         </div>
31     </div>
32 </div>
33
34 @endsection
```

2. Buat fungsi bernama **processLogin** pada auth/**LoginController**, tuliskan script berikut

```
use Illuminate\Support\Facades\Auth;
```

```
public function processLogin(Request $request)
{
    $request->validate([
        'email' => 'required|email',
        'password' => 'required|min:4'
    ]);

    $credential = ['email' => $request->email, 'password' => $request->password];
    $login = Auth::attempt($credential);
    if ($login) {
```

```

        dd(Auth::check());
    } else {
        return redirect()->back()->withInput()->withErrors("Invalid Credential");
    }
}

```

3. Tambahkan route untuk method post

```
Route::post('login', [LoginController::class, 'processLogin']);
```

Menjadi seperti berikut:

```

Route::prefix('auth')->group(function () {
    Route::get('login', [LoginController::class, 'index'])->name('login');
    Route::post('login', [LoginController::class, 'processLogin']);
    Route::get('register', [RegisterController::class, 'index'])->name('register');
});

```

4. Untuk menampilkan error dan parameter email yang lama maka ubah login.blade.php seperti berikut

Tambahkan script ini diatas tag form

```

@if ($errors->any())
    <div class="alert alert-danger" role="alert">
        {!! implode(' ', $errors->all('<li>:message</li>')) !!}
    </div>
@endif

```

Ubah input email menjadi seperti berikut

```

<input type="email" class="form-control" id="exampleInputEmail1" aria-
describedby="emailHelp" name="email" value="{{ old('email') }}">

```

Login

- The password field is required.

Email address

donal@google.com

We'll never share your email with anyone else.

Password

Submit

belum punya akun ? [Register](#)

F. Registration + Validation

1. Tambahkan URL untuk processRegister

```
Route::post('register', [RegisterController::class, 'processRegister']);
```

maka tampilannya akan seperti berikut

```
routes > web.php > ...
1  <?php
2
3  use App\Http\Controllers\Auth\LoginController;
4  use App\Http\Controllers\Auth\RegisterController;
5  use Illuminate\Support\Facades\Route;
6
7  Route::redirect('/', '/auth/login');
8  Route::prefix('auth')->group(function () {
9      Route::get('login', [LoginController::class, 'index'])->name('login');
10     Route::post('login', [LoginController::class, 'processLogin']);
11     Route::get('register', [RegisterController::class, 'index'])->name('register');
12     Route::post('register', [RegisterController::class, 'processRegister']);
13 });
14
```

2. Buat fungsi **processRegister** pada **RegisterControlller**

```
use App\Models\User;
```

```
public function processRegister(Request $request)
{
    $request->validate([
        'email' => 'required|email',
        'password' => 'required|min:4|confirmed',
        'gender' => "required"
    ]);

    $user = new User();
    $user->email = $request->email;
    $user->user_jab_id = 2; // staff
    $user->password = bcrypt($request->password);
    $user->gender = $request->gender;

    if ($user->save()) {
        return redirect()->route('login')->with("success", "Register Success");
    } else {
        return redirect()->back()->withInput()->withErrors("Something Error !");
    }
}
```

```
}
}
```

3. Ubah section('content') pada register.blade.php menjadi seperti berikut

```
@section("content")
<div class="container">
  <div class="row" style="padding-top: 50px">
    <div class="col-md-4"></div>
    <div class="col-md-4">
      <h2 class="text-center">Register</h2>

      @if (session("success"))
      <div class="alert alert-primary">{{ session('success') }}</div>
      @endif
      @if ($errors->any())
      <div class="alert alert-danger" role="alert">
        {!! implode(' ', $errors->all('<li>:message</li>')) !!}
      </div>
      @endif

      <form method="post" action="{{ route('register') }}">
        @csrf
        <div class="mb-3">
          <label class="form-label">Email address</label>
          <input type="email" name="email" class="form-control"
placeholder="Enter email" value="{{ old('email') }}">
        </div>
        <div class="mb-3">
          <label class="form-label">Password</label>
          <input type="password" name="password" class="form-control"
placeholder="Password">
        </div>
        <div class="mb-3">
          <label class="form-label">Password Confirmation</label>
          <input type="password" name="password_confirmation" class="form-
control" placeholder="Retype Password">
        </div>
        <div class="mb-3">
          <label class="form-label">Gender</label>
          <select name="gender" id="gender" class="form-control">
            <option disabled selected>Jenis Kelamin</option>
            <option value="laki">Laki-Laki</option>
            <option value="perempuan">Perempuan</option>
          </select>
        </div>
      </form>
    </div>
  </div>
</div>
```

```
        </div>

        <button type="submit" class="btn btn-primary">Register</button>
    </form>

    <br>
    sudah punya akun ? <a href="{{ url('/auth/login') }}">Login</a>
</div>
</div>
</div>
@endsection
```