

Nama : Dhimas Rizaldy

Npm : 20312030

Bagian : Backend Api

- **Penjelasan Backend Api Website E-kos**

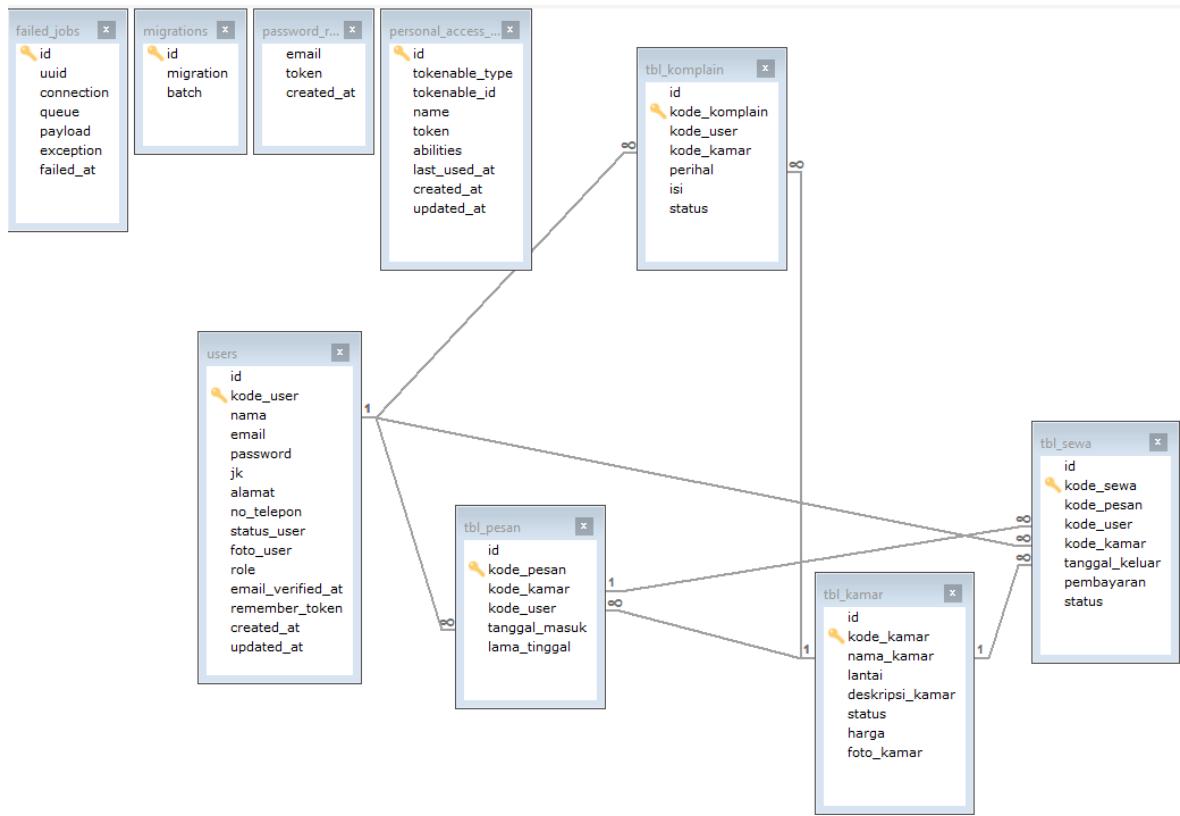
Backend API dalam website e-kost menggunakan Laravel adalah bagian dari sistem yang bertanggung jawab untuk mengelola logika dan pemrosesan data di belakang layar. Backend API ini bertindak sebagai jembatan antara database dan antarmuka pengguna (frontend) dalam website e-kost.

Dalam konteks Laravel, backend API biasanya terdiri dari serangkaian "endpoint" yang melayani permintaan dari klien (seperti aplikasi mobile atau frontend web) dan memberikan respons dengan data yang diminta. Laravel menyediakan kerangka kerja yang kuat untuk membangun dan mengelola backend API dengan mudah.

Beberapa fitur utama dalam backend API Laravel pada website E-kos mungkin meliputi:

1. Routing: Laravel menyediakan sistem routing yang fleksibel untuk menentukan URL endpoint dan menghubungkannya dengan fungsi atau metode yang sesuai.
2. Controller: Controller berperan sebagai perantara antara routing dan logika aplikasi. Mereka mengelola permintaan yang masuk, memproses data, dan memberikan respons yang tepat.
3. Model: Model dalam Laravel mewakili struktur data dalam database. Mereka digunakan untuk mengakses, memanipulasi, dan mengelola data yang diperlukan oleh aplikasi.

1. Database : db_e-kos



2. Tampilan kodingan Models :

Dalam framework Laravel, "models" adalah bagian yang penting dalam pola pengembangan aplikasi web berbasis PHP. Model bertanggung jawab untuk mengelola data dan interaksi dengan basis data.

- **UserModel.php**

```

app > Models > UserModel.php
  <?php
  namespace App\Models;
  use Illuminate\Database\Eloquent\Model;
  use Illuminate\Support\Facades\DB;
  class UserModel extends Model
  {
    // Tampil Data User
    function viewData()
    {
      $query = DB::table('users')
        ->select([
          'kode_user',
          'nama',
          'email',
          'password',
          'jk',
          'alamat',
          'no_telepon',
          'status_user',
          'role',
          'foto_user'
        ])
        ->orderBy("kode_user")
        ->get();
      return $query;
    }
    // Tampil Data User Join
    // Detail Data User
    function detailData($parameter)
    {
      $query = DB::table('users')
        ->select([
          'kode_user',
          'nama',
          'email',
          'password',
          'jk',
          'alamat',
          'no_telepon',
          'status_user',
          'role',
          'foto_user'
        ])
        ->where(DB::raw("(kode_user)"), "=", $parameter)
        ->orderBy("kode_user")
        ->get();
    }
  }

```

- **KamarModel.php**

```

app > Models > KamarModel.php
  <?php
  namespace App\Models;
  use Illuminate\Database\Eloquent\Model;
  use Illuminate\Support\Facades\DB;
  class KamarModel extends Model
  {
    // Tampil Data Kamar
    function viewData()
    {
      $query = DB::table('tbl_kamar')
        ->select([
          'kode_kamar',
          'nama_kamar',
          'lantai',
          'jumlah_lipik_kamar',
          'status',
          'harga',
          'foto_kamar'
        ])
        ->orderBy("kode_kamar")
        ->get();
      return $query;
    }
    // Detail Data Kamar
    function detailData($parameter)
    {
      $query = DB::table('tbl_kamar')
        ->select([
          'nama_kamar',
          'lantai',
          'jumlah_lipik_kamar',
          'status',
          'harga',
          'foto_kamar'
        ])
        ->where(DB::raw("(kode_kamar)"), "=", $parameter)
        ->orderBy("kode_kamar")
        ->get();
    }
    // Delete Data Kamar
    function deleteData($parameter)
    {
      DB::table('tbl_kamar')
        ->where(DB::raw("(kode_kamar)"), "=", $parameter)
        ->delete();
    }
  }

```

- **PesanModel.php**

```

app > Models > PesanModel.php
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class PesanModel extends Model
{
    // Tampil Data pesan
    function viewData()
    {
        $query = DB::table('tb_pesanan')
            ->select(
                'kode_pesan',
                'kode_kamar',
                'kode_user',
                'tanggal_masuk',
                'lama_tinggal'
            )
            ->orderBy('kode_pesan')
            ->get();
        return $query;
    }

    // Tampil Data tb_pesanan Join tb_users Join tb_kamar
    function viewDataWithJoin()
    {
        $query = DB::table('tbl_pesanan')
            ->join('users', 'users.kode_user', '=', 'tbl_pesanan.kode_user')
            ->join('tbl_kamar', 'tbl_kamar.kode_kamar', '=', 'tbl_pesanan.kode_kamar')
            ->select(
                'tbl_pesanan.kode_pesanan',
                'tbl_pesanan.tanggal_masuk',
                'tbl_kamar.nama_kamar',
                'tbl_pesanan.kode_user',
                'users.name',
                'tbl_pesanan.tanggal_masuk',
                'tbl_pesanan.lama_tinggal'
            )
            ->get();
        return $query;
    }

    // Detail Data pesan
    function detailData($parameter)
    {
        $query = DB::table('tbl_pesanan')
            ->select(
                'kode_pesan',
                'kode_kamar',
                'kode_user',
                'tanggal_masuk',
                'lama_tinggal'
            )
            ->where('kode_pesan', '=', $parameter)
            ->orderBy('kode_pesan')
            ->get();
        return $query;
    }
}

```

- **SewaModel.php**

```

app > Models > SewaModel.php
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class SewaModel extends Model
{
    // Tampil Data sewa
    function viewData()
    {
        $query = DB::table('tbl_sewa')
            ->select(
                'kode_sewa',
                'kode_pesan',
                'kode_user',
                'kode_kamar',
                'tanggal_keluar',
                'pembayaran',
                'status'
            )
            ->orderBy('kode_sewa')
            ->get();
        return $query;
    }

    // Tampil Data tol_sewa Join tb_users Join tb_kamar Join tb_pesanan
    function viewDataWithJoin()
    {
        $query = DB::table('tbl_sewa')
            ->join('users', 'users.kode_user', '=', 'tbl_sewa.kode_user')
            ->join('tbl_pesanan', 'tbl_pesanan.kode_pesanan', '=', 'tbl_sewa.kode_pesan')
            ->join('tbl_kamar', 'tbl_kamar.kode_kamar', '=', 'tbl_sewa.kode_kamar')
            ->select(
                'tbl_sewa.kode_sewa',
                'tbl_sewa.kode_kamar',
                'tbl_sewa.kode_pesanan',
                'tbl_sewa.tanggal_keluar',
                'tbl_sewa.pembayaran',
                'tbl_sewa.status',
                'users.name',
                'tbl_sewa.kode_kamar',
                'tbl_kamar.nama_kamar',
                'tbl_kamar.harga',
                'tbl_pesanan.tanggal_masuk',
                'tbl_pesanan.lama_tinggal',
                'tbl_sewa.tanggal_keluar'
            )
            ->get();
        return $query;
    }

    // Tampil View Detail Data tb_sewa Join tb_users Join tb_kamar Join tb_pesanan
    function viewDetailData($sewa)
    {
        $query = DB::table('tbl_sewa')
            ->join('users', 'users.kode_user', '=', 'tbl_sewa.kode_user')
            ->join('tbl_pesanan', 'tbl_pesanan.kode_pesanan', '=', 'tbl_sewa.kode_pesan')
            ->join('tbl_kamar', 'tbl_kamar.kode_kamar', '=', 'tbl_sewa.kode_kamar')
            ->select(
                'tbl_sewa.kode_sewa',
                'tbl_sewa.kode_user',
                'users.name',
                'users.email'
            )
            ->get();
        return $query;
    }
}

```

- **KomplainModel.php**

The screenshot shows a code editor interface with the following details:

- File Path:** app > Models > KomplainModel.php
- Code Content (KomplainModel.php):**

```
1 <?php
2 namespace App\Models;
3 use Illuminate\Database\Eloquent\Model;
4 use Illuminate\Support\Facades\DB;
5
6 class KomplainModel extends Model
7 {
8     // Tampil Data komplain
9     function viewData()
10    {
11        $query = DB::table("tbl_komplain")
12            ->select(
13                "tbl_komplain",
14                "kode_user",
15                "kode_kamar",
16                "perihal",
17                "isi",
18                "status"
19            )
20            ->orderBy("kode_komplain")
21            ->get();
22
23        return $query;
24    }
25
26    // Tampil Data tbl_komplain Join tb_users Join tb_kamar
27    function viewDataKomplainJoin()
28    {
29        $query = DB::table("tbl_komplain")
30            ->join("tb_users", "tbl_komplain.kode_user", '=', "tb_users.kode_user")
31            ->join("tbl_kamar", "tbl_komplain.kode_kamar", '=', "tbl_kamar.kode_kamar")
32            ->select("tbl_komplain.kode_komplain", "tbl_komplain.kode_user", "users.name",
33                    "tbl_komplain.kode_kamar", "tbl_kamar.nama_kamar", "tbl_komplain.perihal",
34                    "tbl_komplain.isi", "tbl_komplain.status")
35            ->get();
36
37        return $query;
38    }
39
40    // Detail Data komplain
41    function detailData($parameter)
42    {
43        $query = DB::table("tbl_komplain")
44            ->select(
45                "tbl_komplain",
46                "kode_komplain",
47                "kode_user",
48                "kode_kamar",
49                "perihal",
50                "isi",
51                "status"
52            )
53            ->where('tbl_komplain.kode_komplain', $parameter);
54
55        return $query;
56    }
57}
```

- Explorer View:** Shows the project structure with files like KamarModel.php, PesanModel.php, and User.php.
- Bottom Status Bar:** Shows file details (Ln 1, Col 1), workspace status (01 11, 0 0), and system status (Partly cloudy, 9:23 PM, 5/31/2023).

3. Tampilan Kodingan Controller :

Dalam framework Laravel, "controller" adalah komponen yang bertanggung jawab untuk menangani permintaan dan mengendalikan alur aplikasi web. Controller berperan sebagai perantara antara rute dan model, serta mengatur logika bisnis dan respons yang akan dikirimkan kembali ke pengguna.

- **UserController.php**

The screenshot shows a code editor interface with the file `UserController.php` open in the center. The file is located in the `app > Http > Controllers` directory. The code implements a `UserController` class that extends `Controller`. It contains several methods: `viewUser()`, `detailUser($parameter)`, and `deleteUser($parameter)`. The code uses the `Illuminate\Http\Request` and `App\Models\UserModel` classes. The right side of the screen displays a vertical code preview pane. The bottom status bar shows the file path as `master_dimas 5/31/2023` and the current time as `8:50 PM`.

```
app > Http > Controllers > UserController.php
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6 use App\Models\UserModel;
7
8 class UserController extends Controller
9 {
10     // Function Construct()
11     function __construct()
12     {
13         $this->model = new UserModel();
14     }
15
16     // Function Untuk View Data User
17     function viewUser()
18     {
19         // Ambil Function viewData dari UserModel
20         $data = $this->model->viewData();
21
22         // Tampilkan hasil dari "tbl_user"
23         return response([
24             "data" => $data
25         ], http_response_code());
26     }
27
28     // Function Untuk Detail Data User
29     function detailUser($parameter)
30     {
31         // Ambil fungsi detailData dari UserModel
32         $data = $this->model->detailData($parameter);
33
34         // Tampilkan Hasil dari "tbl_user"
35         return response([
36             "detailUser" => $data
37         ], http_response_code());
38     }
39
40     // Function Untuk Delete Data User
41     function deleteUser($parameter)
42     {
43         // cek data dari tbl_user (berdasarkan Kode_User)
44         $data = $this->model->deleteData($parameter);
45
46         // Jika data ditemukan
47         if (count($data) != 0) {
48             // Lakukan penghapusan data
49             $data = $this->model->deleteData($parameter);
50             // Buat pesan dan status hasil Penghapusan Data
51             $status = 1;
52             $pesan = "Data Berhasil diHapus";
53         }
54     }
55 }
```

- **KamarController.php**

The screenshot shows a code editor interface with the file `KamarController.php` open in the center. The file is located in the `app > Http > Controllers` directory. The code implements a `KamarController` class that extends `Controller`. It contains several methods: `viewKamar()`, `detailKamar($parameter)`, and `deleteKamar($parameter)`. The code uses the `Illuminate\Http\Request` and `App\Models\KamarModel` classes. The right side of the screen displays a vertical code preview pane. The bottom status bar shows the file path as `master_dimas 5/31/2023` and the current time as `8:51 PM`.

```
app > Http > Controllers > KamarController.php
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6 use App\Models\KamarModel;
7
8 class KamarController extends Controller
9 {
10     //Function Construct()
11     function __construct()
12     {
13         $this->model = new KamarModel();
14     }
15
16     // Function Untuk View Data
17     function viewKamar()
18     {
19         // Ambil Function viewData dari UserModel
20         $data = $this->model->viewData();
21
22         // Tampilkan hasil dari "tbl_user"
23         return response([
24             "dataKamar" => $data
25         ], http_response_code());
26     }
27
28     // Function Untuk Detail Data User
29     function detailKamar($parameter)
30     {
31         // Ambil fungsi detailData dari UserModel
32         $data = $this->model->detailData($parameter);
33
34         // Tampilkan Hasil dari "tbl_user"
35         return response([
36             "detailKamar" => $data
37         ], http_response_code());
38     }
39
40     // Function Untuk Delete Data User
41     function deleteKamar($parameter)
42     {
43         // cek data dari tol_user (berdasarkan kode_kamar)
44         $data = $this->model->deleteData($parameter);
45
46         // Jika data ditemukan
47         if (count($data) != 0) {
48             // Lakukan penghapusan data
49             $data = $this->model->deleteData($parameter);
50             // Buat pesan dan status hasil Penghapusan Data
51             $status = 1;
52             $pesan = "Data Berhasil diHapus";
53         }
54     }
55 }
```

- **PesanController.php**

```

File Edit Selection View Go Run Terminal Help < > Backend
EXPLORER BACKEND app Controllers Http Controllers
    <?php
    namespace App\Http\Controllers;
    use Illuminate\Http\Request;
    use App\Models\PesanModel;
    class PesanController extends Controller
    {
        //Function Construct()
        function __construct()
        {
            $this->model = new PesanModel();
        }
        // Function Untuk View Data
        function viewPesan()
        {
            // Ambil Function ViewData dari PesanModel
            $data = $this->model->viewData();
            // Tampilkan hasil dari "tbl_pesan"
            return response([
                "pesan" => $data
            ], http_response_code());
        }
        // Function untuk view data tbl_pesan join tbl_users join tbl_kamar
        function viewPesanJoin()
        {
            // ambil Function ViewDataPesanJoin dari PesanModel
            $data = $this->model->viewDataPesanJoin();
            // Tampilkan hasil dari "tbl_pesan join "tbl_users" Join "tbl_kamar"
            return response([
                "Data Pesan Join" => $data
            ], http_response_code());
        }
        // Function Untuk Detail Data pesan
        function detailPesan($parameter)
        {
            // Ambil Fungsi detailData dari PesanModel
            $data = $this->model->detailData($parameter);
            // Tampilkan hasil dari "tbl_pesan"
            return response([
                "Detail Pesan" => $data
            ], http_response_code());
        }
        // Function Untuk Delete Data pesan
        function deletePesan($parameter)
        {
        }
    }

```

Ln 1, Col 1 Spaces:4 UTF-8 LF PHP Go Live Prettier 8:52 PM 5/31/2023

- **SewaController.php**

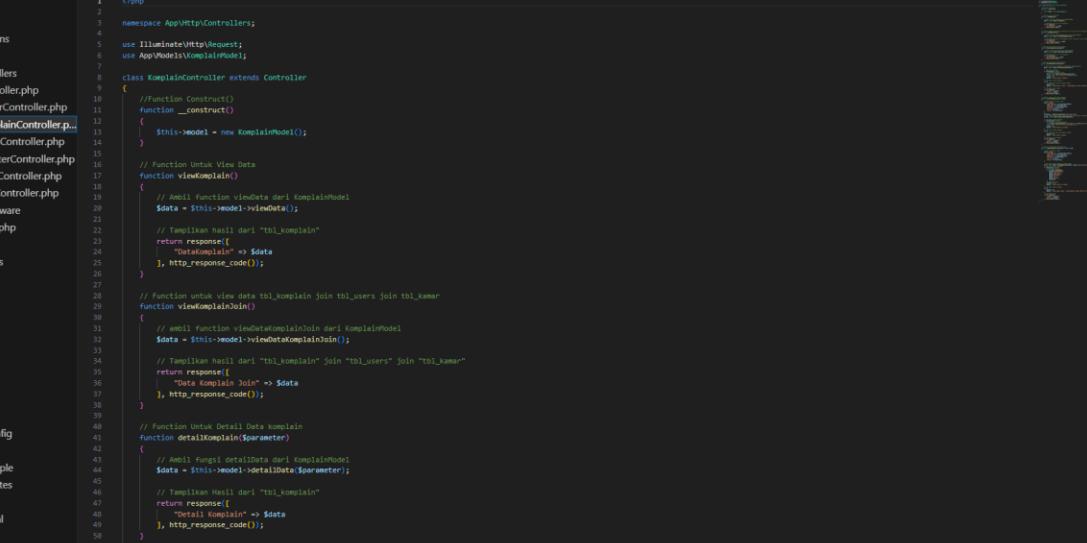
```

File Edit Selection View Go Run Terminal Help < > Backend
EXPLORER BACKEND app Controllers Http Controllers
    <?php
    namespace App\Http\Controllers;
    use Illuminate\Http\Request;
    use App\Models\SewaModel;
    class SewaController extends Controller
    {
        //Function Construct()
        function __construct()
        {
            $this->model = new SewaModel();
        }
        // Function Untuk View Data
        function viewSewa()
        {
            // Ambil Function ViewData dari SewaModel
            $data = $this->model->viewData();
            // Tampilkan hasil dari "tbl_sewa"
            return response([
                "DataSewa" => $data
            ], http_response_code());
        }
        // Function untuk view data tbl_pesan join tbl_users join tbl_kamar join tbl_sewa
        function viewSewaJoin()
        {
            // ambil Function ViewDataSewaJoin dari SewaModel
            $data = $this->model->viewDataSewaJoin();
            // Tampilkan hasil dari "tbl_sewa join "tbl_users" join "tbl_kamar" join "tbl_pesan"
            return response([
                "Data Sewa Join" => $data
            ], http_response_code());
        }
        // Function untuk view detail data tbl_pesan join tbl_users join tbl_kamar join tbl_sewa
        function viewDetailSewaJoin()
        {
            // ambil Function viewDataDetailSewaJoin dari SewaModel
            $data = $this->model->viewDataDetailSewaJoin();
            // Tampilkan hasil dari "tbl_sewa" join "tbl_users" join "tbl_kamar" join "tbl_pesan"
            return response([
                "Detail Tampil Data Sewa Join" => $data
            ], http_response_code());
        }
        // Function Untuk Detail Data sewa
        function detailSewa($parameter)
        {
        }
    }

```

Ln 21, Col 1 Spaces:4 UTF-8 LF PHP Go Live Prettier 8:54 PM 5/31/2023

- **KomplainController.php**



The screenshot shows a PHP development environment with the following details:

- File Explorer:** Shows the project structure under "BACKEND". Key files include `KamarController.php`, `PesanController.php`, `SewaController.php`, and `KomplainController.php`.
- Code Editor:** The main editor window displays the `KomplainController.php` file. The code is a Controller class named `KomplainController` extending `Controller`. It contains several methods:
 - `viewKomplain()`: Returns a response with the template `"DataKomplain"` and its data.
 - `viewKomplainJoin()`: Returns a response with the template `"DataKomplainJoin"` and its data.
 - `detailKomplain($parameter)`: Returns a response with the template `"DetailKomplain"` and its data.
 - `deleteKomplain($parameter)`: Deletes a record from the database and returns a response.
- Terminal:** Shows the command `master_dimas`.
- Environment:** Shows `.env`, `.env.example`, `.gitattributes`, `.gitignore`, `stylecyml`, and `artisan`.
- Timeline:** Shows the current date and time as `5/31/2023 9:54 PM`.
- Bottom Bar:** Includes icons for file operations, search, and various system status indicators.

4. Tampilan Codingan Route API

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface with the following details:

- File Explorer:** Shows the project structure under "PWBS-TA-DEV".
- Editor:** The main window displays the content of the file "api.php".
- Bottom Status Bar:** Provides information such as file paths, line numbers, and other development details.

```
// Here is where you can register API routes for your application. These
// routes are loaded by the RouteServiceProvider within a group which
// is assigned the "api" middleware group. Enjoy building your API!
|
| API Routes
|
|
| Here is where you can register API routes for your application. These
| routes are loaded by the RouteServiceProvider within a group which
| is assigned the "api" middleware group. Enjoy building your API!
|
|
Route::middleware('auth:sanctum')->get('/user', function (Request $request) {
    return $request->user();
});

// =====ROUTE REGISTER=====
// =====ROUTE USER=====

// Api Route Register User
Route::post('/register', [RegisterController::class, 'register']);

// Api Route Login User
Route::get('/login', [RegisterController::class, 'login']);

// Api Route Logout User
Route::post('/logout', [RegisterController::class, 'logout']);

// =====ROUTE DETAIL=====
// =====ROUTE DETAIL USER=====

// Route untuk tampil data user
Route::get('/viewUser', [UserController::class, 'viewUser']);
// route untuk detail data user
Route::get('/detailUser/{parameter}', [UserController::class, 'detailUser']);
// route untuk update data user
Route::put('/updateUser/{parameter}', [UserController::class, 'updateUser']);
// route untuk hapus data user
Route::delete('/deleteUser/{parameter}', [UserController::class, 'deleteUser']);
// Route untuk insert data user
Route::post('/insertUser', [UserController::class, 'insertUser']);
```

5. Tampilan Seetting .env

```
APP_NAME=StimE-KOS
APP_ENV=local
APP_KEY=bases64:tbFSp4Ucv1uCBjPn5y3e1195v10m+eNV1X82GRY-
APP_DEBUG=true
APP_URL=http://127.0.0.1:8000/
LOG_CHANNEL=stack
LOG_DEPRECATIONS_CHANNEL=null
LOG_LEVEL=debug

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=web-e-kos
DB_USERNAME=root
DB_PASSWORD=

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

MEMCACHED_HOST=127.0.0.1

REDIS_HOST=127.0.0.1
REDIS_PASSWORD=null
REDIS_PORT=3790

MAIL_MAILER=smtp
MAIL_HOST=mailhog
MAIL_PORT=1025
MAIL_USERNAME=null
MAIL_PASSWORD=null
MAIL_ENCRYPTION=null
MAIL_FROM_ADDRESS=null
MAIL_FROM_NAME="${APP_NAME}"

AWS_ACCESS_KEY_ID=
AWS_SECRET_ACCESS_KEY=
AWS_DEFAULT_REGION=nous-east-1
AWS_BUCKET=
AWS_USE_PATH_STYLE_ENDPOINT=false

PUSHER_APP_ID=
PUSHER_APP_KEY=
PUSHER_APP_SECRET=
PUSHER_APP_CLUSTER=at1

MIX_PUSHER_APP_KEY="${PUSHER_APP_KEY}"
MIX_PUSHER_APP_CLUSTER="${PUSHER_APP_CLUSTER}"
```

6. Tampilan Testing Backend Api via postman

Test backend API pada website e-kos adalah proses pengujian yang dilakukan untuk memastikan bahwa backend API berfungsi dengan baik dan memberikan respons yang diharapkan sesuai dengan spesifikasi yang telah ditentukan. Test ini melibatkan pengujian fungsi dan logika bisnis yang ada di dalam backend API.

Berikut adalah beberapa jenis pengujian yang umum dilakukan pada backend API website e-kos:

1. Pengujian Unit: Pengujian unit dilakukan untuk menguji setiap unit atau komponen individual dalam backend API secara terisolasi. Ini memastikan bahwa setiap bagian dari kode berfungsi dengan baik dan menghasilkan respons yang diharapkan.
 2. Pengujian Integrasi: Pengujian integrasi melibatkan pengujian interaksi antara berbagai komponen backend API dengan sistem lain, seperti basis data, layanan pihak ketiga, atau sistem eksternal lainnya. Tujuannya adalah untuk memastikan bahwa integrasi tersebut berjalan lancar dan tidak ada masalah dalam komunikasi antara komponen-komponen tersebut.
 3. Pengujian Fungsionalitas: Pengujian fungsionalitas bertujuan untuk memastikan bahwa fungsi-fungsi yang ada dalam backend API berjalan sesuai dengan spesifikasi yang telah ditetapkan. Pengujian ini mencakup pengujian berbagai endpoint API, seperti permintaan GET, POST, PUT, DELETE, dan lain-lain, serta memastikan bahwa respons yang diberikan sesuai dengan harapan.

4. Pengujian Performa: Pengujian performa dilakukan untuk menguji kinerja backend API dalam menghadapi beban kerja yang tinggi atau skenario yang kompleks. Pengujian ini melibatkan pengujian kecepatan respons, waktu respons, dan skalabilitas backend API.

a. **Test Api Tbl_user**

- **Test API tambah user :**

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'My Workspace' containing various API requests like 'GET View_AllUser', 'POST Add_User', etc. The main area shows a 'POST Add_User' request. The 'Body' tab is selected, showing a JSON payload:

```
1 {  
2   "jk": "pria",  
3   "alamat": "Jakarta",  
4   "no_telp": "089277726728",  
5   "status_user": "Pegawai",  
6   "foto_user": "toni.png",  
7   "role": "customer"  
}
```

The status bar at the bottom indicates: Status: 200 OK Time: 223 ms Size: 353 B. The system tray shows the date and time as 5/31/2023 11:14 PM.

- **Test API Edit data user :**

Hasil update data user :

- Test API Delete data user :

My Workspace

DELETE http://127.0.0.1:8000/api/deleteUser/1012991

Key	Value	Description

Status: 200 OK Time: 414 ms Size: 352 B

```

1
2
3
4
      "status": 1,
      "pesan": "Data Berhasil diHapus"
    
```

b. Test Api Tbl_kamar

- Test API tambah kamar :

POST http://127.0.0.1:8000/api/insertKamar

Key	Value	Description
kode_kamar	km06	
nama_kamar	Kamar C02	
lantai	Lantai 2	
deskripsi_kamar	Spesifikasi : -	
status	Kosong	
harga	750000	
foto_kamar	KamarC2.png	

Status: 200 OK Time: 599 ms Size: 353 B

```

1
2
3
4
      "status": 1,
      "pesan": "Data Berhasil Disimpan"
    
```

- Test API Edit data kamar :

The screenshot shows the Postman interface with a collection named "Apl_E-KOS". A specific request titled "Update_Kamar" is selected. The method is set to PUT, and the URL is `http://127.0.0.1:8000/api/updateKamar/km06`. The request body contains the following JSON:

```

{
    "kode_kamar": "km06",
    "nama_kamar": "Kamar C06",
    "lantai": "lantai 3",
    "deskripsi_kamar": "Kamar Lengkap : sudah tersedia Kasur,Meja Belajar Kursi dan Lemari",
    "status": "Penuh",
    "harga": "750000",
    "foto_kamar": "Kamar C3.png"
}

```

The response status is 200 OK, and the response body is:

```

{
    "status": "1",
    "pesan": "Data Berhasil di Ubah"
}

```

Hasil update data kamar :

The screenshot shows the Postman interface with a collection named "Apl_E-KOS". A specific request titled "Show_Kamar" is selected. The method is set to GET, and the URL is `http://127.0.0.1:8000/api/detailKamar/km06`. The response status is 200 OK, and the response body is:

```

{
    "Detail_Kamar": [
        {
            "nm_kamar": "Kamar C06",
            "lantai": "lantai 3",
            "deskripsi_kamar": "Kamar Lengkap : sudah tersedia Kasur,Meja Belajar Kursi dan Lemari",
            "status": "Penuh",
            "harga": "750000",
            "foto_kamar": "Kamar C3.png"
        }
    ]
}

```

- Test API Delete data kamar :

DELETE <http://127.0.0.1:8000/api/deleteKamar/kmr06>

Key	Value	Description
Key		Description

```

1
2
3
4
{
  "status": 1,
  "pesan": "Data Berhasil diHapus"
}

```

c. Test Api Tbl_pesan

- Test API tambah pesan :

POST <http://127.0.0.1:8000/api/insertPesan>

Key	Value	Description
kode_pesan	KPN03	
kode_kamar	kmr03	
kode_user	23432424	
tanggal_awal	2023-06-01	
lama_stingal	2 Bulan	

```

1
2
3
4
{
  "status": 1,
  "pesan": "Data Berhasil Disimpan"
}

```

- Test API Edit data pesan :

The screenshot shows the Postman interface with a collection named 'Aplication-KOS'. A specific request titled 'Update_Pesan' is selected. The method is set to 'PUT' with the URL `http://127.0.0.1:8000/api/updatePesan/KPSN03?kode_pesan=KPSN03&kode_kamar=kmr04&kode_user=23432424&tanggal_masuk=2023-06-01&lama_tinggal=4 Bulan`. The 'Params' tab shows five parameters: 'kode_pesan' (KPSN03), 'kode_kamar' (kmr04), 'kode_user' (23432424), 'tanggal_masuk' (2023-06-01), and 'lama_tinggal' (4 Bulan). The 'Body' tab is selected, showing a JSON response with status: "1", pesan: "Data Berhasil di Ubah". The status bar at the bottom indicates a 200 OK response.

Hasil update data pesan :

The screenshot shows the Postman interface with the same collection 'Aplication-KOS'. A request titled 'Show_Pesan' is selected, using the 'GET' method with the URL `http://127.0.0.1:8000/api/detailPesan/KPSN03`. The 'Params' tab shows three parameters: 'name', 'email', and 'password'. The 'Body' tab is selected, displaying a JSON response with a key 'Detail Pesan' containing a single object with fields: 'kode_pesan' (KPSN03), 'kode_kamar' (kmr04), 'kode_user' (23432424), 'tanggal_masuk' (2023-06-01), and 'lama_tinggal' (4 Bulan). The status bar at the bottom indicates a 200 OK response.

- Test API Delete data pesan :

DELETE <http://127.0.0.1:8000/api/deletePesan/KPSN03>

Key	Value	Description
Key		Description

```

1
2
3
4
{
  "status": 1,
  "pesan": "Data Berhasil diHapus"
}
    
```

d. Test Api Tbl_sewa

- Test API tambah sewa :

POST <http://127.0.0.1:8000/api/insertSewa>

Key	Value	Description
kode_sewa	KDSW03	
kode_pesan	KPSN03	
kode_user	29897728	
kode_kamar	kmr02	
tanggal_keluar	2023-8-21	
pembayaran	750000	
status	Dicicil	

```

1
2
3
4
{
  "status": 1,
  "pesan": "Data Berhasil Disimpan"
}
    
```

- **Test API Edit data sewa :**

The screenshot shows the Postman interface with a collection named "Api_E-KOS". A specific request titled "Update_Sewa" is selected, which is a PUT method to the URL `http://127.0.0.1:8000/api/updateSewa/KDSW03?kode_sewa=KDSW03&kode_pesan=KPSN03&kode_user=29897728&kode_kamar=kmr02&tanggal_keluar=2023-08-21&pembayaran=750000&status=Sudah Bayar`. The "Params" tab shows the following key-value pairs:

Key	Value
kode_sewa	KDSW03
kode_pesan	KPSN03
kode_user	29897728
kode_kamar	kmr02
tanggal_keluar	2023-08-21
pembayaran	750000
status	Sudah Bayar

The "Body" tab displays the JSON response:

```

1  {
2   "status": "1",
3   "pesan": "Data Berhasil di Ubah"
4 }

```

Hasil update data sewa :

The screenshot shows the Postman interface with a collection named "Api_E-KOS". A specific request titled "Show_Sewa" is selected, which is a GET method to the URL `http://127.0.0.1:8000/api/detailSewa/KDSW03`. The "Params" tab shows the following key-value pairs:

Key	Value
nama	
email	
password	

The "Body" tab displays the JSON response:

```

1  {
2   "Detail Sewa": [
3     {
4       "kode_sewa": "KDSW03",
5       "kode_pesan": "KPSN03",
6       "kode_user": "29897728",
7       "kode_kamar": "kmr02",
8       "tanggal_keluar": "2023-08-21",
9       "pembayaran": "750000",
10      "status": "Sudah Bayar"
11    }
12  ]

```

- **Test API Delete data sewa :**

My Workspace

DELETE http://127.0.0.1:8000/api/deleteSewa/KDSW03

Key	Value	Description
Key	Value	Description

```

1
2   "status": 1,
3   "pesan": "Data berhasil dihapus"
4

```

e. Test Api Tbl_komplain

- Test API tambah komplain :**

POST http://127.0.0.1:8000/api/insertKomplain

Key	Value	Description
kode_komplain	KDKMP02	
kode_user	10102939	
kode_kamar	kmr01	
perihal	keran air rusak	
isi	pak tolong keran air saya rusak, tolong diperbaiki pak	
status	Belum	

```

1
2   "status": 1,
3   "pesan": "Data berhasil disimpan"
4

```

- Test API Edit data komplain :

The screenshot shows the Postman interface with a collection named "Api_E-KOS". A specific request titled "Update_Komplain" is selected, which performs a PUT operation on the URL `http://127.0.0.1:8000/api/updateKomplain/KDKMP02?kode_komplain=KDKMP02&kode_user=10102939&kode_kamar=kmr01&perihal=Genteng Bocor, Air Mati, Token Listrik Habis&isi=Pak ini kamar saya genteng nya bocor, mohon diperbaiki ya pak&status=Belum`. The response status is 200 OK, time 397 ms, and size 354 B. The response body is:

```

1
2   "status": "1",
3   "pesan": "Data Berhasil di Ubah"
4

```

Hasil update data komplain :

The screenshot shows the Postman interface with a collection named "Api_E-KOS". A specific request titled "Show_Komplain" is selected, which performs a GET operation on the URL `http://127.0.0.1:8000/api/detailKomplain/KDKMP02`. The response status is 200 OK, time 432 ms, and size 543 B. The response body is:

```

1
2   "Detail Komplain": [
3     {
4       "kode_komplain": "KDKMP02",
5       "kode_user": "10102939",
6       "kode_kamar": "kmr01",
7       "perihal": "Genteng Bocor, Air Mati, Token Listrik Habis",
8       "isi": "Pak ini kamar saya genteng nya bocor, mohon diperbaiki ya pak",
9       "status": "Belum"
10    }
11 ]

```

- Test API Delete data komplain :

Home Workspaces API Network Explore

My Workspace

Collections Environments History

GET View_All_Pesan JOIN users J...
GET View_All_Sewa JOIN users J...
GET View_All_Detail Sewa JOIN u...
GET View_All_Sewa
GET View_All_Komplain
GET View_All_Kamar
GET View_All_Pesan
PUT Update_User
PUT Update_Sewa
PUT Update_Pesan
PUT Update_Kamar
DEL Delete_User
DEL Delete_Kamar
DEL Delete_Pesan
DEL Delete_Sewa
DEL Delete_Komplain
GET Show_User
GET Show_Pesan
GET Show_Sewa
GET Show_SewaJoin
GET Show_Komplain
GET Show_Kamar
New Collection
RESTful API basics: CRUD, test & va...

DELETE http://127.0.0.1:8000/api/deleteKomplain/KOKMP02

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description
Key	Value	Description

Body Cookies Headers (10) Test Results

Pretty Raw Preview Visualize JSON

```
1
2   "status": 1,
3   "pesan": "Data Berhasil diHapus"
4
```

Status: 200 OK Time: 375 ms Size: 352 B Save as Example

Runner Capture requests Cookies Trash

83°F Haze

10:06 AM 6/2/2023