



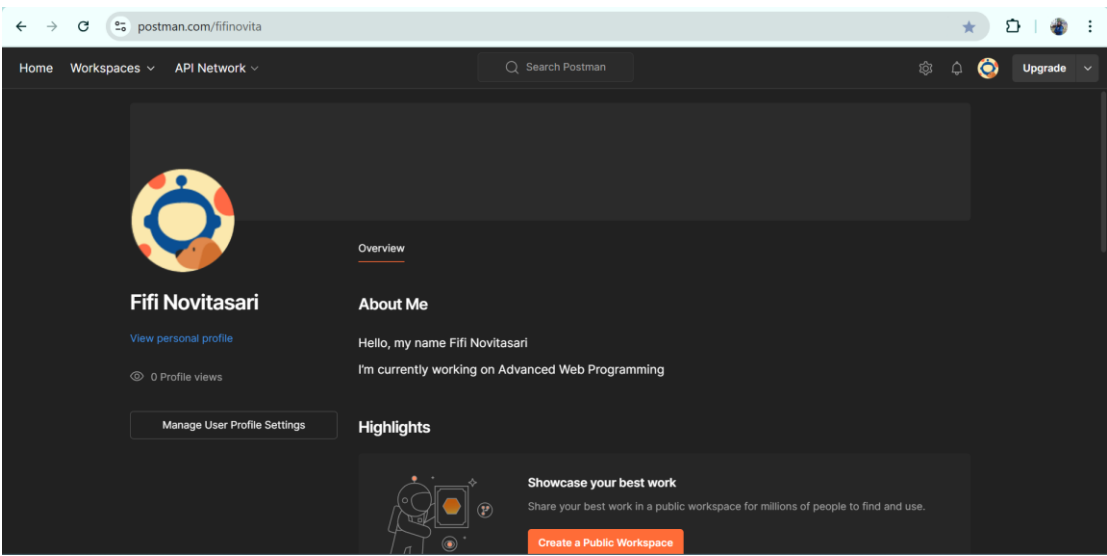
Mata Kuliah : Pemrograman Web Lanjut (PWL)
Program Studi : D4 – Sistem Informasi Bisnis
Semester : 5

Kelas : SIB
NIM : 2241760035
Nama : Fifi Novitasari
Jobsheet Ke- : 10

Laporan Jobsheet 10

RESTFUL API

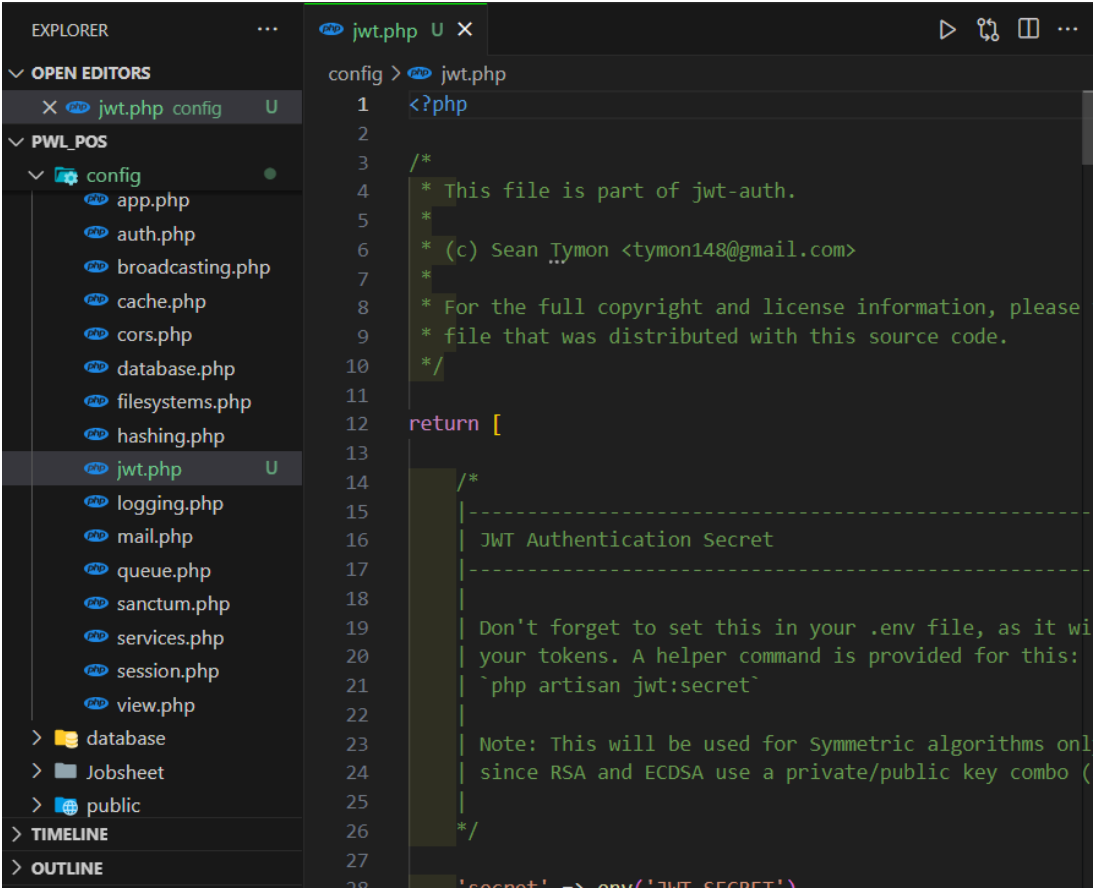
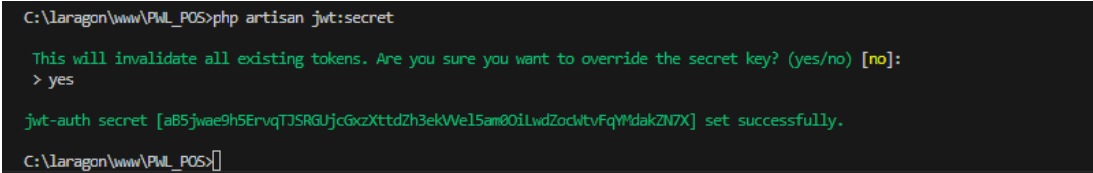
Praktikum Ke-1 Membuat RESTful API Register

Langkah	Jawaban/Deskripsi
1	<p>Download aplikasi Postman di https://www.postman.com/downloads.</p> 
2	<p>Instalasi JWT dengan mengetikkan perintah berikut:</p> <pre>composer require tymon/jwt-auth:2.1.1</pre>

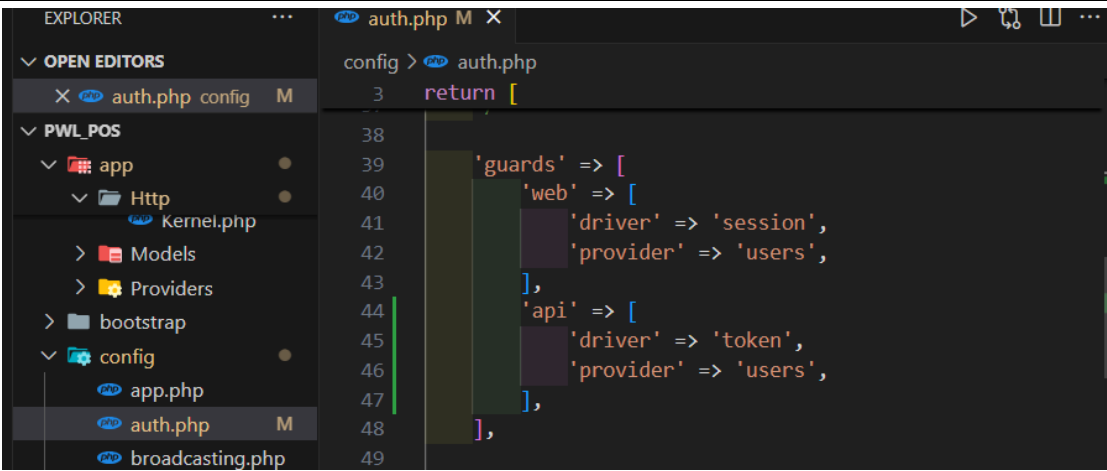
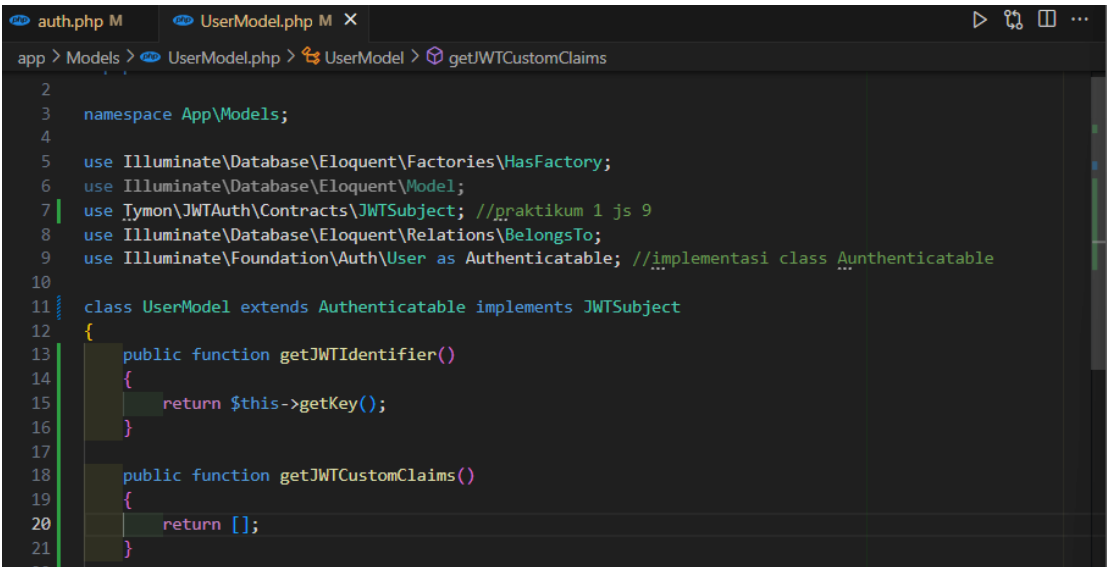
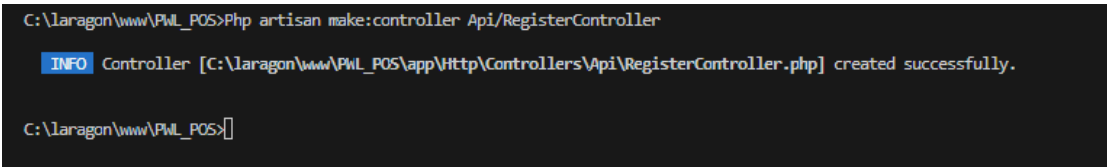


	<pre>C:\laragon\www\PNL_POS>composer require tymon/jwt-auth:2.1.1 ./composer.json has been updated Running composer update tymon/jwt-auth Loading composer repositories with package information Updating dependencies Lock file operations: 4 installs, 0 updates, 0 removals - Locking lcobucci/clock (2.3.0) - Locking lcobucci/jwt (4.0.4) - Locking stella-maris/clock (0.1.7) - Locking tymon/jwt-auth (2.1.1) Writing lock file Installing dependencies from lock file (including require-dev) Package operations: 4 installs, 0 updates, 0 removals - Downloading stella-maris/clock (0.1.7) - Downloading lcobucci/clock (2.3.0) - Downloading lcobucci/jwt (4.0.4) - Downloading tymon/jwt-auth (2.1.1) - Installing stella-maris/clock (0.1.7): Extracting archive - Installing lcobucci/clock (2.3.0): Extracting archive - Installing lcobucci/jwt (4.0.4): Extracting archive - Installing tymon/jwt-auth (2.1.1): Extracting archive Generating optimized autoload files Class App\Models\kategoriModel located in ./app/Models/KategoriModel.php does not comply with psr-4 autoloading standard (rule: App\ => ./app). Skipping. > illuminate\Foundation\ComposerScripts::postAutoloadDump > @php artisan package:discover --ansi INFO Discovering packages. barryvdh/laravel-dompdf DONE laravel/sail DONE laravel/sanctum DONE laravel/tinker DONE nesbot/carbon DONE</pre>
3	<p>Publish konfigurasi file dengan perintah berikut:</p> <pre>php artisan vendor:publish -- provider="Tymon\JWTAuth\Providers\LaravelServiceProvider"</pre> <pre>C:\laragon\www\PNL_POS>php artisan vendor:publish --provider="Tymon\JWTAuth\Providers\LaravelServiceProvider" INFO Publishing assets. Copying file [C:\laragon\www\PNL_POS\vendor\tymon\jwt-auth\config\config.php] to [C:\laragon\www\PNL_POS\config\jwt.ph p] DONE C:\laragon\www\PNL_POS></pre>

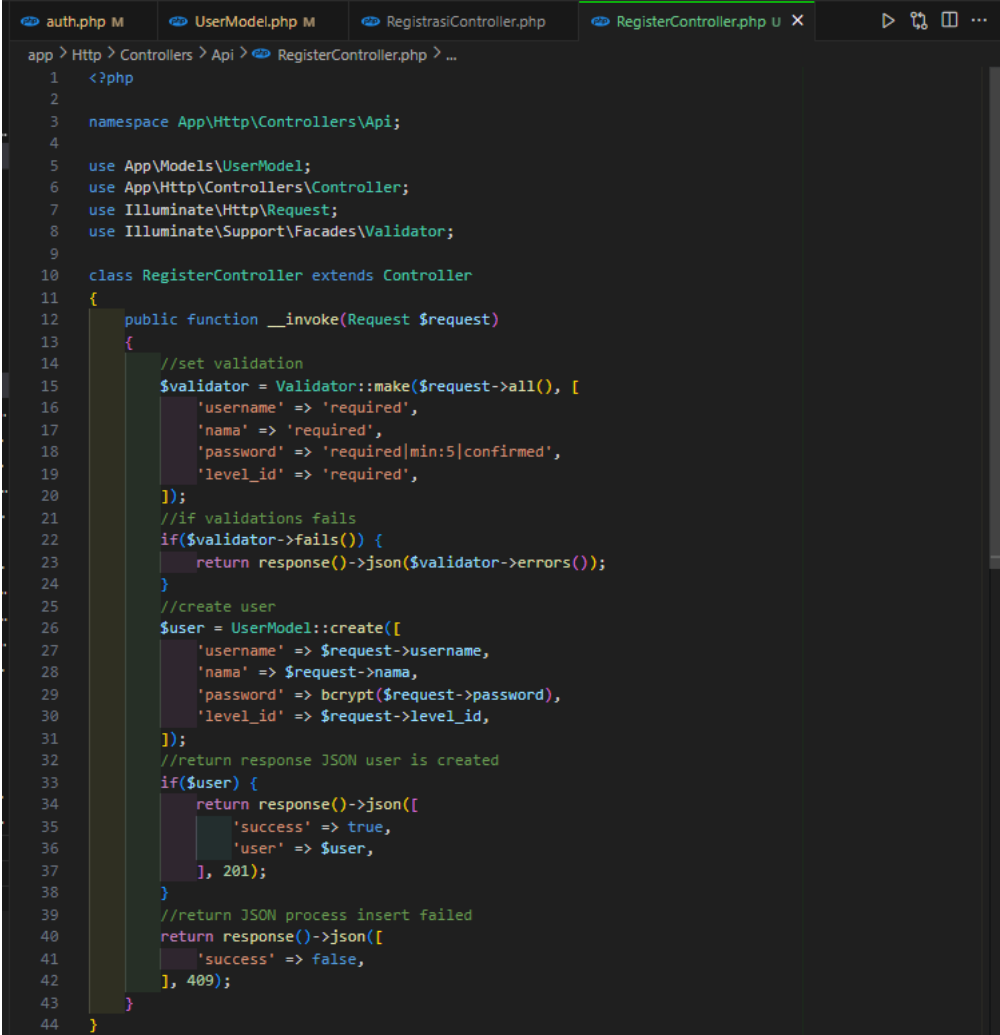


	<p>Jika perintah di atas berhasil, maka kita akan mendapatkan 1 file baru yaitu config/jwt.php. Pada file ini dapat dilakukan konfigurasi jika memang diperlukan</p> 
4	<p>Membuat secret key JWT. php artisan jwt:secret</p> <p>Jika berhasil, maka pada file .env akan ditambahkan sebuah baris berisi nilai key JWT_SECRET.</p> 
5	<p>Selanjutnya konfigurasi guard API. Buka config/auth.php.</p>

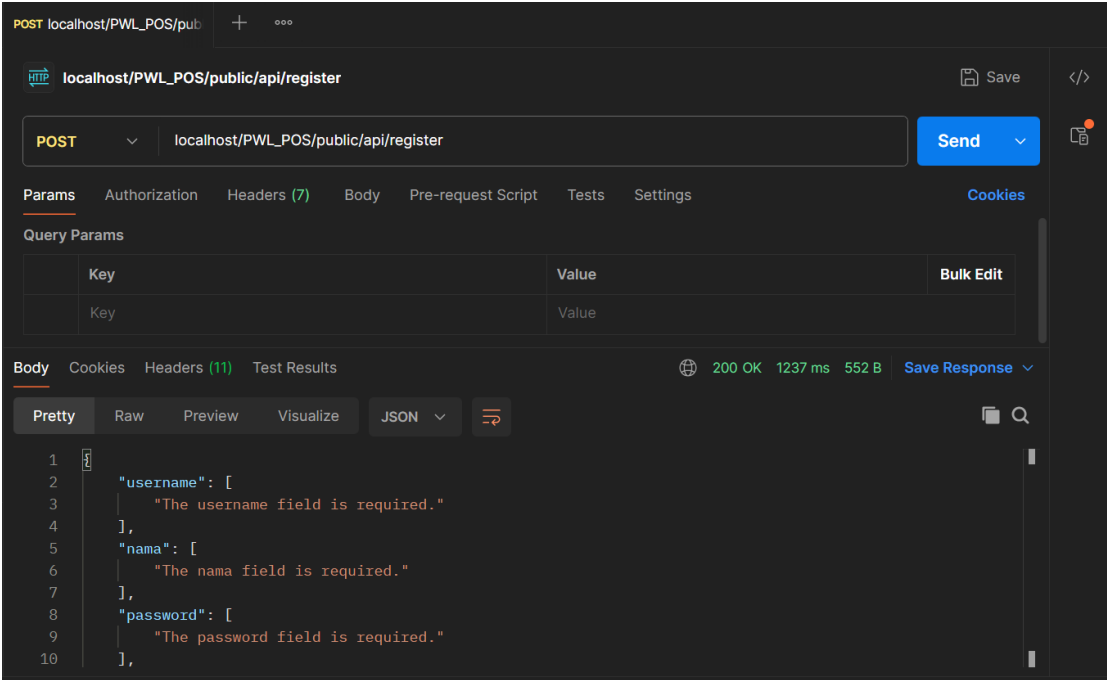


	
6	<p>Menambahkan kode di model UserModel</p> 
7	<p>Membuat controller untuk register dengan menjalankan perintah berikut.</p> <p>Php artisan make:controller Api/RegisterController</p> <p>Jika berhasil maka akan ada tambahan controller pada folder Api dengan nama RegisterController.</p> 



	 <pre>1 <?php 2 3 namespace App\Http\Controllers\Api; 4 5 use App\Models\UserModel; 6 use App\Http\Controllers\Controller; 7 use Illuminate\Http\Request; 8 use Illuminate\Support\Facades\Validator; 9 10 class RegisterController extends Controller 11 { 12 public function __invoke(Request \$request) 13 { 14 //set validation 15 \$validator = Validator::make(\$request->all(), [16 'username' => 'required', 17 'nama' => 'required', 18 'password' => 'required min:5 confirmed', 19 'level_id' => 'required', 20]); 21 //if validations fails 22 if(\$validator->fails()) { 23 return response()->json(\$validator->errors()); 24 } 25 //create user 26 \$user = UserModel::create([27 'username' => \$request->username, 28 'nama' => \$request->nama, 29 'password' => bcrypt(\$request->password), 30 'level_id' => \$request->level_id, 31]); 32 //return response JSON user is created 33 if(\$user) { 34 return response()->json([35 'success' => true, 36 'user' => \$user, 37], 201); 38 } 39 //return JSON process insert failed 40 return response()->json([41 'success' => false, 42], 409); 43 } 44 }</pre>
8	Selanjutnya buka routes/api.php.



	<pre>routes > api.php > ... 1 <?php 2 3 use App\Http\Controllers\Api\RegisterController; 4 use Illuminate\Http\Request; 5 use Illuminate\Support\Facades\Route; 6 7 /* 8 ----- 9 API Routes 10 ----- 11 12 Here is where you can register API routes for your application. These 13 routes are loaded by the RouteServiceProvider and all of them will 14 be assigned to the "api" middleware group. Make something great! 15 16 */ 17 18 // praktikum 1 js 9 19 Route::post('/register', App\Http\Controllers\Api\RegisterController::class)->name('register'); 20 21 Route::middleware('auth:sanctum')->get('/user', function (Request \$request) { 22 return \$request->user(); 23 });</pre>
9	<p>Melakukan uji coba REST API melalui aplikasi Postman. Buka aplikasi Postman, isi URL localhost/PWL_POS/public/api/register serta method POST. Klik Send.</p>  <p>Jika berhasil akan muncul error validasi seperti gambar di atas.</p>
10	<p>Sekarang memasukkan data. Klik tab Body dan pilih form-data. Isikan key sesuai dengan kolom data, serta isikan data registrasi menggunakan nilai yang diinginkan.</p>

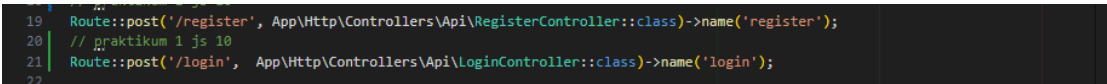


Setelah klik tombol Send, jika berhasil maka akan keluar pesan sukses seperti gambar di atas.

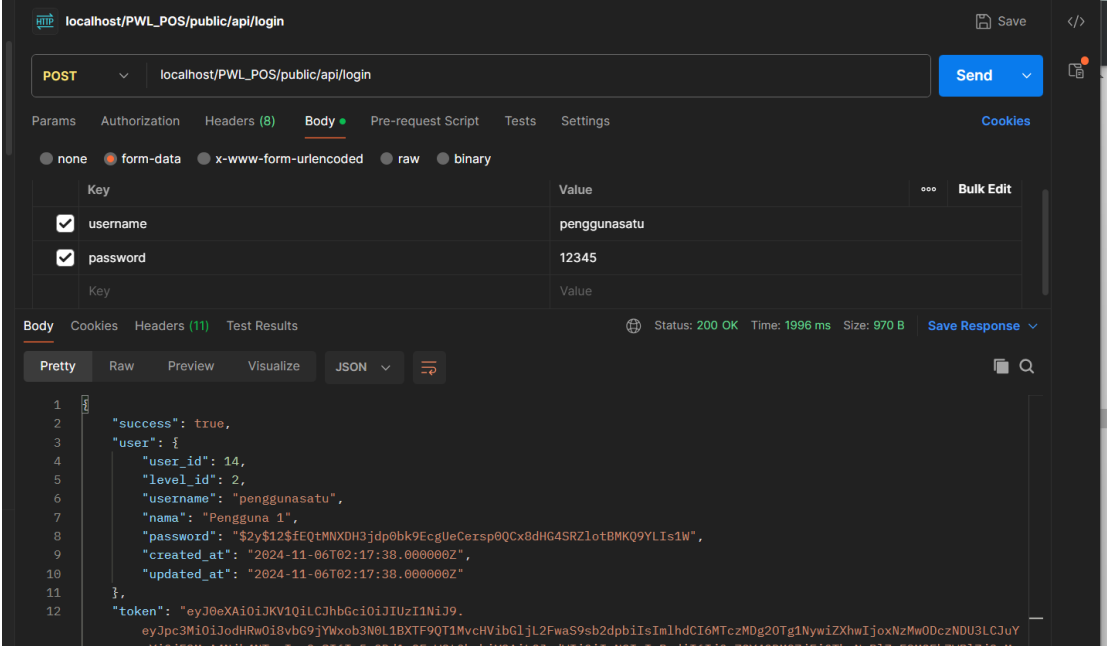
Praktikum Ke-2 Membuat RESTful API Login

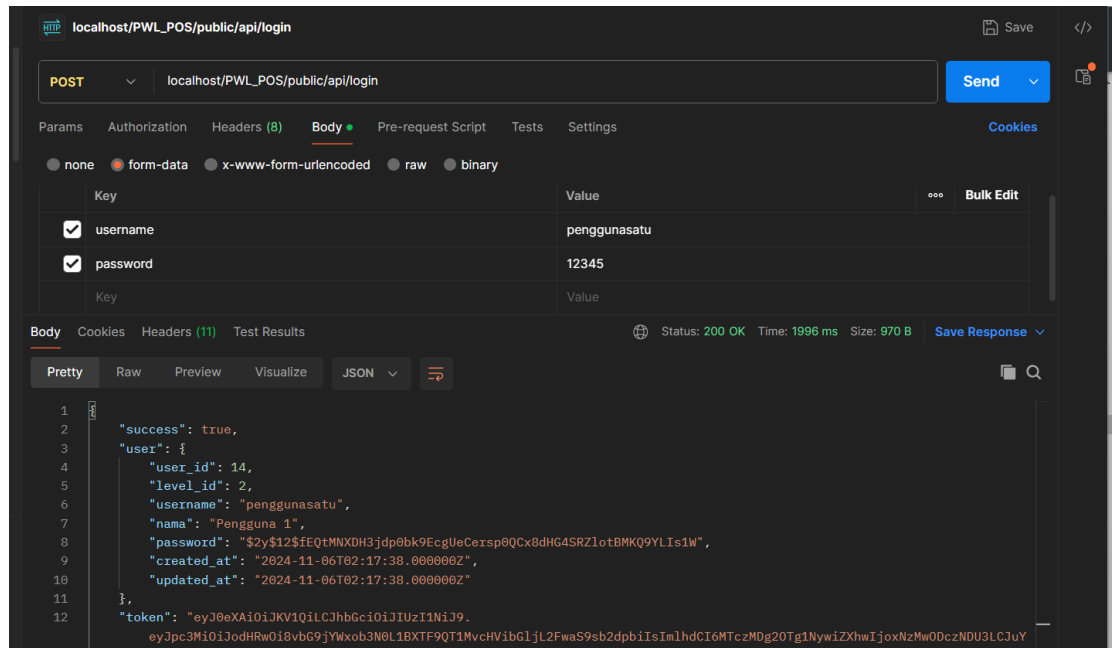
Langkah	Jawaban/Deskripsi
1	<p>Buat file controller dengan nama LoginController.</p> <p>php artisan make:controller Api/LoginController</p> <p>Jika berhasil maka akan ada tambahan controller pada folder Api dengan nama LoginController</p> <pre>C:\laragon\www\PWL_POS>Php artisan make:controller Api/RegistrasiController INFO Controller [C:\laragon\www\PWL_POS\app\Http\Controllers\Api\RegistrasiController.php] created successfully. C:\laragon\www\PWL_POS>php artisan make:controller Api/LoginController</pre>



2	<p>Buka file tersebut, dan ubah kode</p> 
3	<p>Berikutnya menambahkan route baru pada file api.php yaitu /login dan /user</p> 
4	<p>Jika sudah, maka melakukan uji coba REST API melalui aplikasi Postman. Buka aplikasi Postman, isi URL localhost/PWL_POS/public/api/login serta method POST. Klik Send.</p>



	 <p>Jika berhasil akan muncul error validasi seperti gambar di atas.</p>
5	Selanjutnya, isikan username dan password sesuai dengan data user yang ada pada database. Klik tab Body dan pilih form-data. Isikan key sesuai dengan kolom data, serta isikan data user. Klik tombol Send, jika berhasil maka akan keluar tampilan seperti berikut. Copy nilai token yang diperoleh pada saat login karena akan diperlukan pada saat logout.
6	Lakukan percobaan yang untuk data yang salah dan berikan screenshot hasil percobaan.

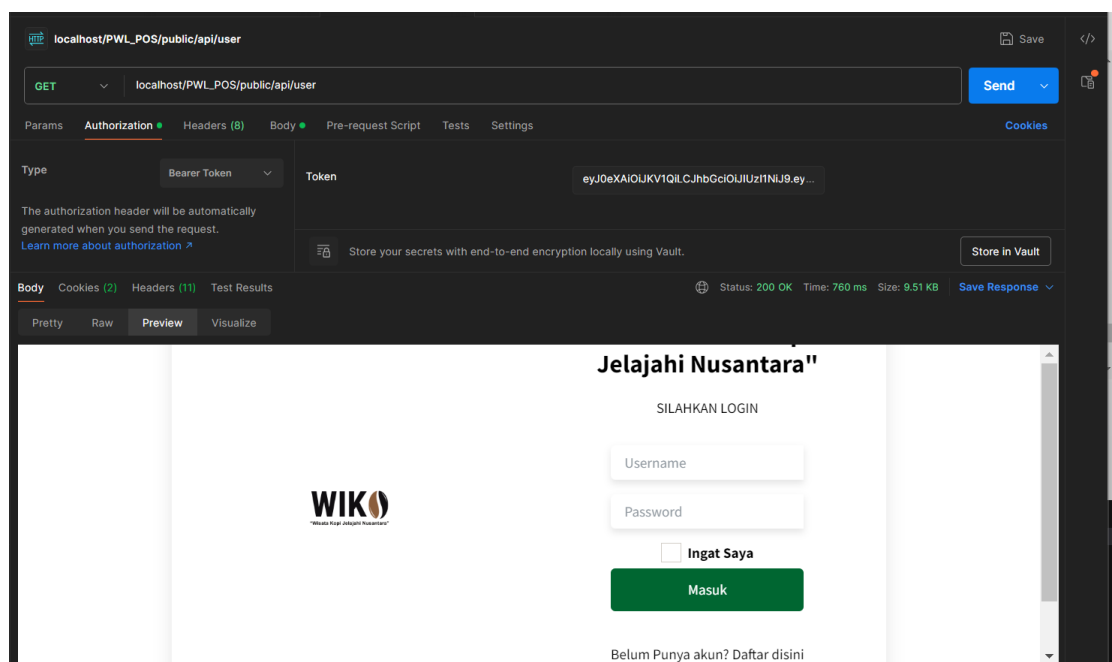
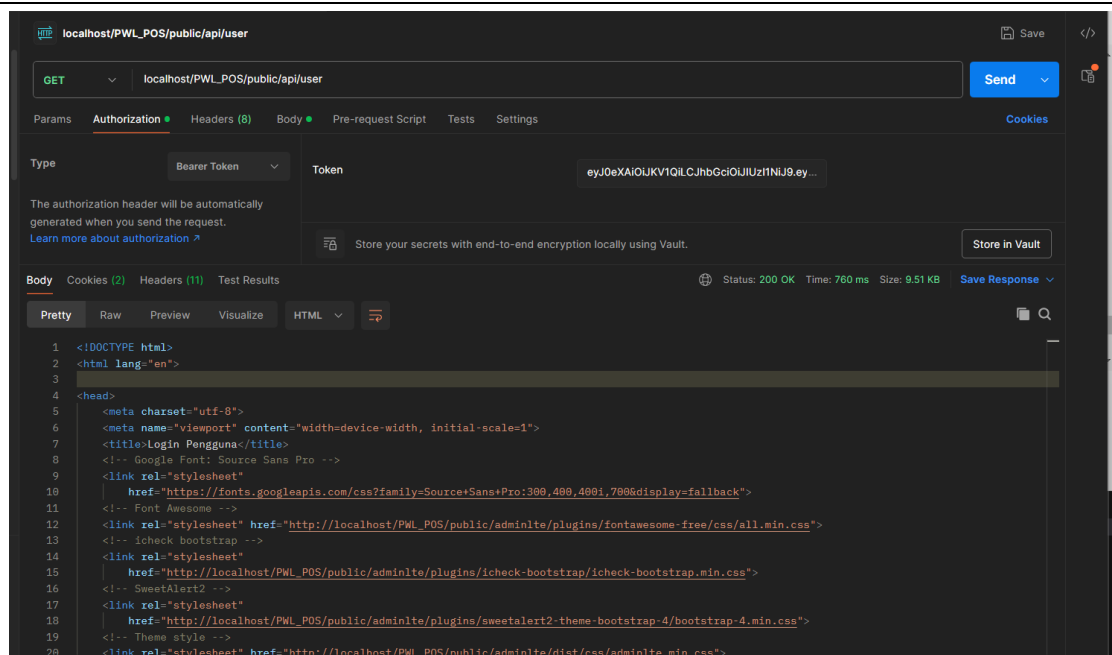


Berikut adalah token yang sudah didapat

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwOi8vbG9jYXRob3N0L1BXTF9QT1MvcHVibGljL2FwaS9sb2dpbiIsImhhdCI6MTczMDg2OTg1NywiaXhwIjozNzYwODczNDU3LCJuYmYiOiJlZjE3MzYwODczNDU3LCJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwOi8vbG9jYXRob3N0L1BXTF9QT1MvcHVibGljL2FwaS9sb2dpbiIsImhhdCI6MTczMDg2OTg1NywiaXhwIjozNzYwODczNDU3LCJuYmYiOiJlZjE3MzYwODczNDU3LCJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9

7

Coba kembali melakukan login dengan data yang benar. Sekarang mari kita coba menampilkan data user yang sedang login menggunakan URL localhost/PWL_POS/public/api/user dan method GET

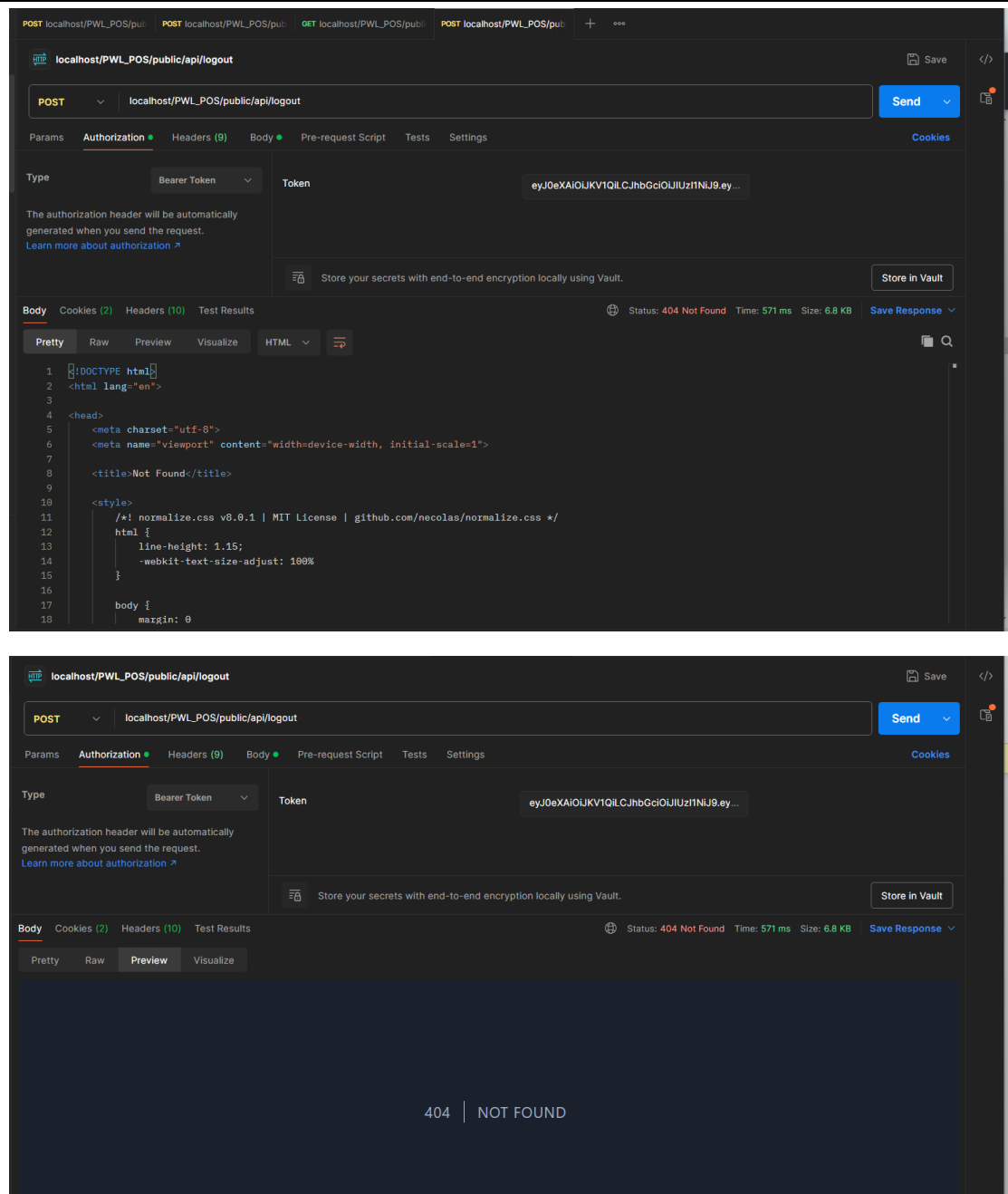


Praktikum Ke-3 Membuat RESTful API Logout

Langkah	Jawaban/Deskripsi
1	Tambahkan kode berikut pada file .env <code>JWT_SHOW_BLACKLIST_EXCEPTION=true</code>



	<pre>60 61 JWT_SECRET=ab5jwae9h5ErvqTJSRGUjcGxzXtttdZh3ekWe15am00iLwdZocWtvFqYMdakZN7X 62 JWT_SHOW_BLACKLIST_EXCEPTION=true 63</pre>
2	<p>Buat Controller baru dengan nama LogoutController.</p> <p><code>php artisan make:controller Api/LogoutController</code></p> <pre>C:\laragon\www\PWL_POS>php artisan make:controller Api/LogoutController INFO Controller [C:\laragon\www\PWL_POS\app\Http\Controllers\Api\LogoutController.php] created successfully.</pre>
3	<p>Buka file tersebut dan ubah kode</p> <pre>app > Http > Controllers > Api > LogoutController.php > LogoutController > __invoke 1 <?php 2 3 namespace App\Http\Controllers\Api; 4 5 use App\Http\Controllers\Controller; 6 use Illuminate\Http\Request; 7 use Tymon\JWTAuth\Facades\JWTAuth; 8 use Tymon\JWTAuth\Exceptions\JWTException; 9 use Tymon\JWTAuth\Exceptions\TokenExpiredException; 10 use Tymon\JWTAuth\Exceptions\TokenInvalidException; 11 12 class LogoutController extends Controller 13 { 14 public function __invoke(Request \$request) 15 { 16 //remove token 17 \$removeToken = JWTAuth::invalidate(JWTAuth::getToken()); 18 19 if(\$removeToken) { 20 //return response JSON 21 return response()->json([22 'success' => true, 23 'message' => 'Logout berhasil!' 24]); 25 } 26 } 27 }</pre>
4	<p>Tambahkan routes pada api.php</p> <pre>23 //praktikum 3 js 10 24 Route::post('/logout', App\Http\Controllers\Api\LogoutController::class)->name('logout'); 25</pre>
5	<p>Melakukan uji coba REST API melalui aplikasi Postman. Buka aplikasi Postman, isi URL localhost/PWL_POS/public/api/logout serta method POST.</p>



- 6 Isi token pada tab Authorization, pilih Type yaitu Bearer Token. Isikan token yang didapat saat login. Jika sudah klik Send.

Praktikum Ke-4 Implementasi CRUD dalam RESTful API

Langkah	Jawaban/Deskripsi
1	Pertama, buat controller untuk mengolah API pada data level.

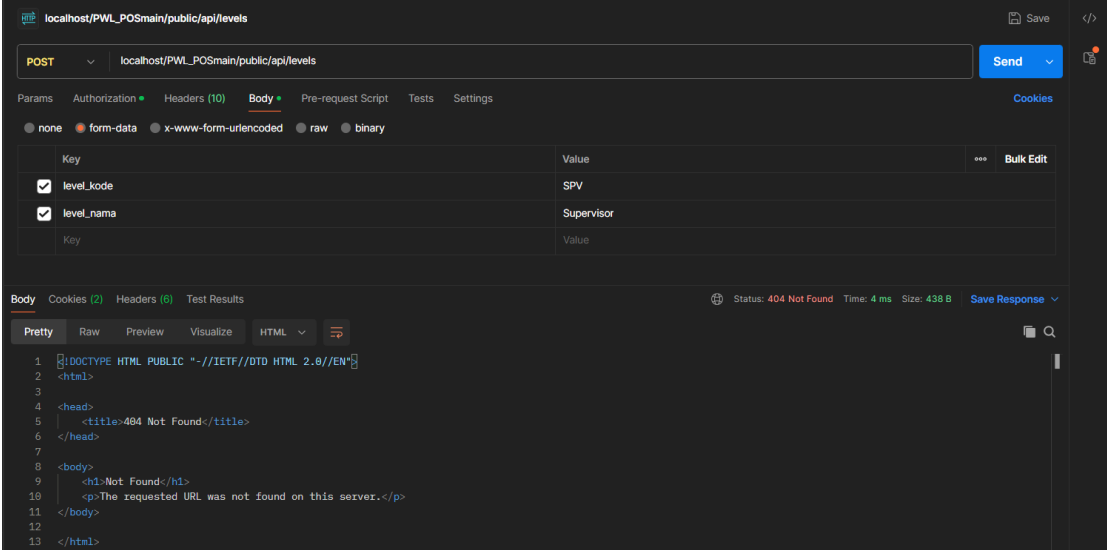
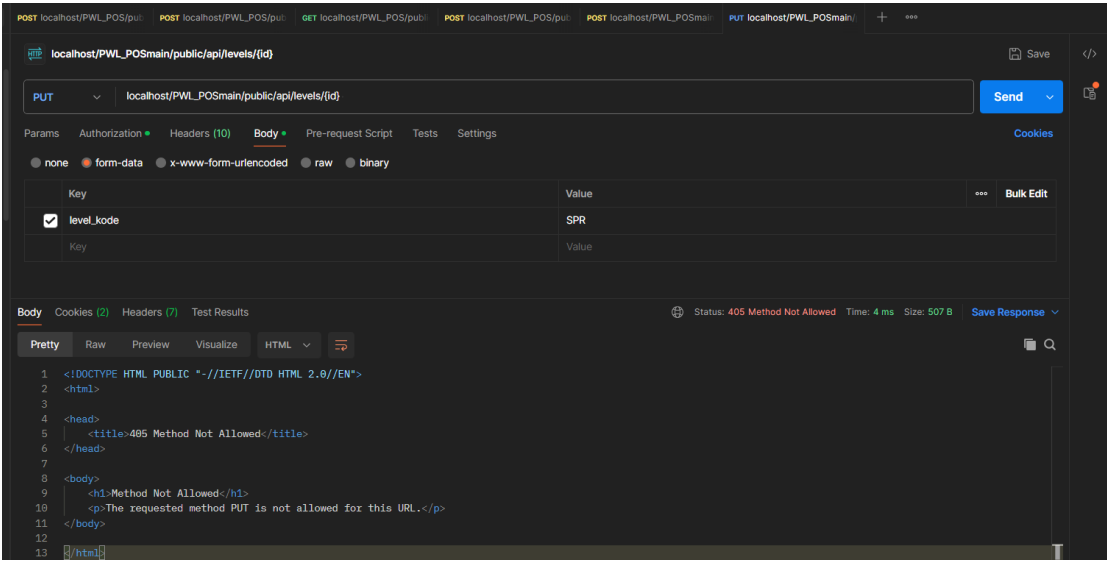


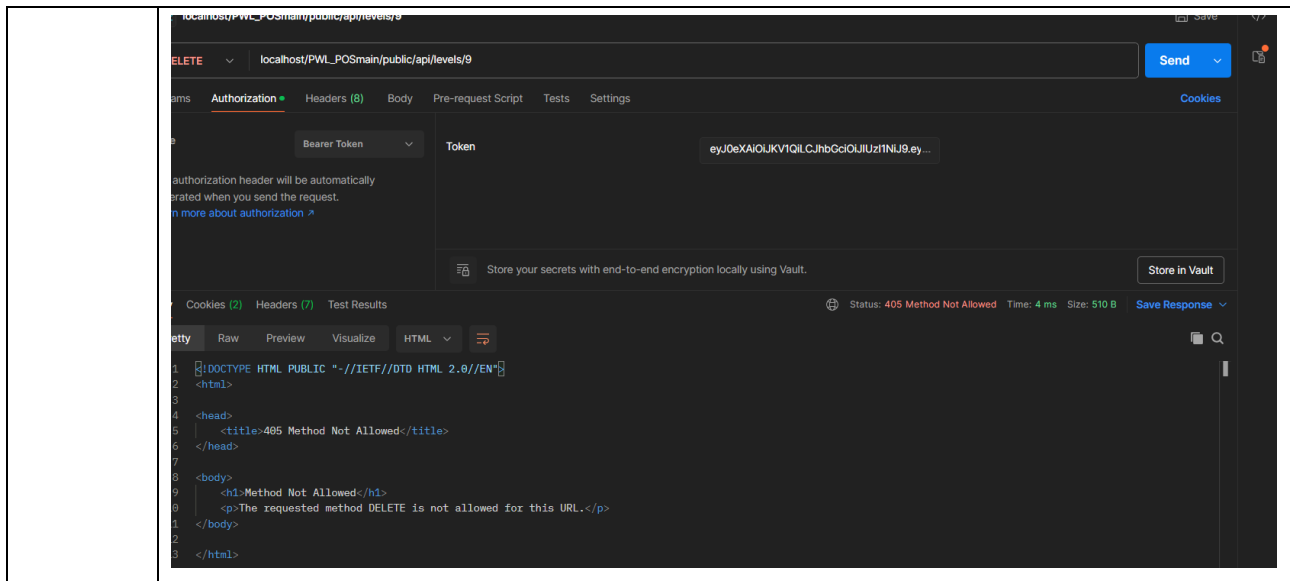
	<p>php artisan make:controller Api/LevelController</p> <pre>C:\laragon\www\PNL_POS>php artisan make:controller Api/LevelController INFO Controller [C:\laragon\www\PNL_POS\app\Http\Controllers\Api\LevelController.php] created successfully. C:\laragon\www\PNL_POS></pre>
2	<p>Setelah berhasil, buka file tersebut dan tuliskan kode seperti berikut yang berisi fungsi CRUDnya.</p> <pre>LoginController.php U LogoutController.php U api.php M LevelController.php U X app > Http > Controllers > Api > LevelController.php > LevelController 1 <?php 2 3 namespace App\Http\Controllers\Api; 4 5 use App\Http\Controllers\Controller; 6 use Illuminate\Http\Request; 7 use App\Models\LevelModel; 8 9 class LevelController extends Controller 10 { 11 public function index() 12 { 13 return LevelModel::all(); 14 } 15 16 public function store(Request \$request) 17 { 18 \$level = LevelModel::create(\$request->all()); 19 return response()->json(\$level, 201); 20 } 21 22 public function show(LevelModel \$level) 23 { 24 return LevelModel::find(\$level); 25 } 26 27 public function update(Request \$request, LevelModel \$level) 28 { 29 \$level->update(\$request->all()); 30 return LevelModel::find(\$level); 31 } 32 33 public function destroy(LevelModel \$user) 34 { 35 \$user->delete(); 36 return response()->json([37 'success' => true, 38 'message' => "Data terhapus", 39]); 40 } 41 }</pre>
3	<p>Lengkapi routes pada api.php</p>



4	<p>Melakukan uji coba API mulai dari fungsi untuk menampilkan data. Gunakan URL: localhost/PWL_POS-main/public/api/levels dan method GET</p>
5	<p>Kemudian, melakukan percobaan penambahan data dengan URL : localhost/PWL_POSmain/public/api/levels dan method POST</p>



	
6	Melakukan percobaan menampilkan detail data.
7	Melakukan edit data menggunakan localhost/PWL_POSmain/public/api/levels/{id} dan method PUT. Isikan data yang ingin diubah pada tab Param.
	
8	Terakhir lakukan percobaan hapus data.



Tugas Praktikum

Langkah	Jawaban/Deskripsi
1	Implementasikan CRUD API pada tabel lainnya
2	<div>Tabel m_user</div> <div><pre>C:\laragon\www\PWL_POS>php artisan make:controller Api\UserController</pre><pre>INFO Controller [C:\laragon\www\PWL_POS\app\Http\Controllers\Api\UserController.php] created successfully.</pre><pre>C:\laragon\www\PWL_POS></pre></div> <div>UserController.php</div>



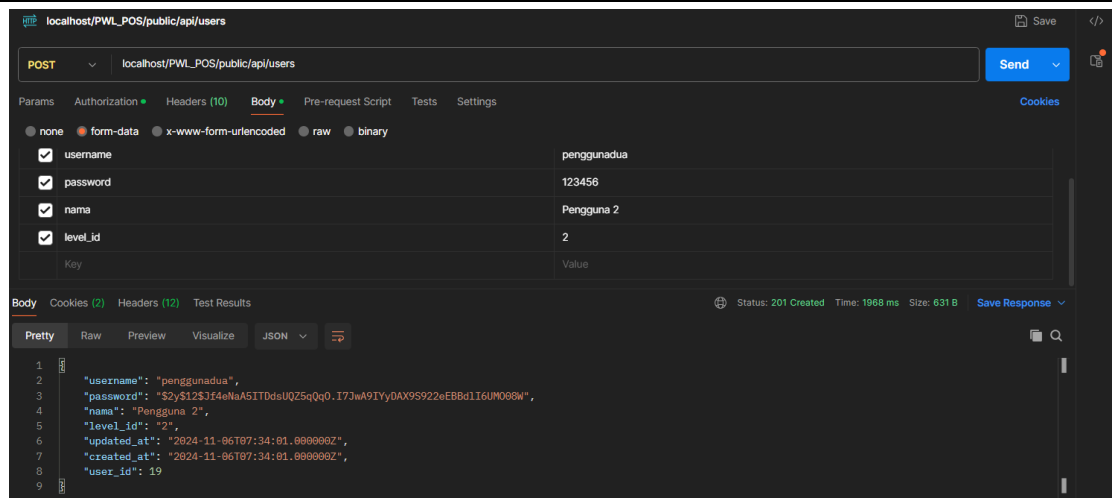
```
api.php M  UserController.php X  LevelController.php
app > Http > Controllers > Api > UserController.php > UserController
1  <?php
2
3  namespace App\Http\Controllers\Api;
4
5  use App\Http\Controllers\Controller;
6  use Illuminate\Http\Request;
7  use App\Models\UserModel;
8
9  class UserController extends Controller
10 {
11     public function index()
12     {
13         return UserModel::all();
14     }
15
16     public function store(Request $request)
17     {
18         $id = UserModel::create($request->all());
19         return response()->json($id, 201);
20     }
21
22     public function show(UserModel $id)
23     {
24         return UserModel::find($id);
25     }
26
27     public function update(Request $request, UserModel $id)
28     {
29         $id->update($request->all());
30         return UserModel::find($id);
31     }
32
33     public function destroy(UserModel $id)
34     {
35         $id->delete();
36         return response()->json([
37             'success' => true,
38             'message' => "Data terhapus",
39         ]);
40     }
41 }
42 }
```

Menambahkan di api.php

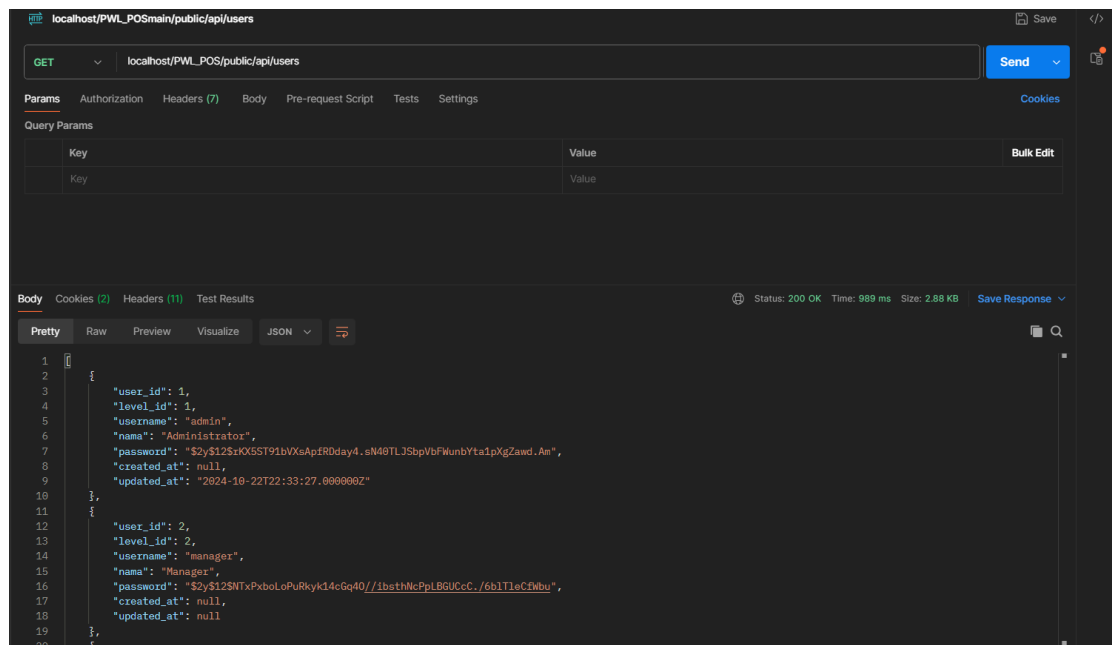
```
use App\Http\Controllers\Api\UserController;
// User
Route::get('users', [UserController::class, 'index']);
Route::post('users', [UserController::class, 'store']);
Route::get('users/{id}', [UserController::class, 'show']);
Route::put('users/{id}', [UserController::class, 'update']);
Route::delete('users/{id}', [UserController::class, 'destroy']);
```

Postman:

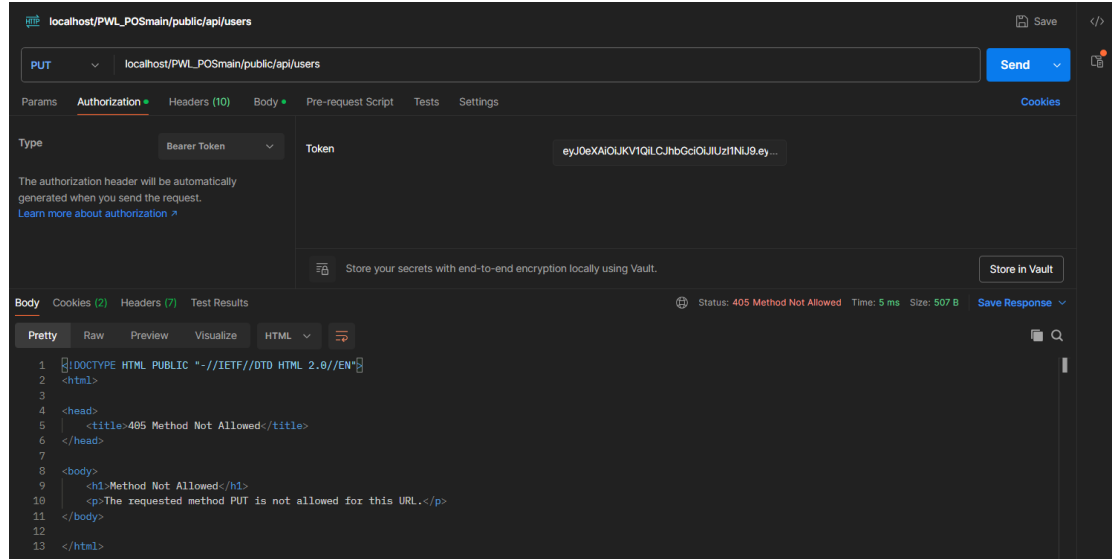
POST



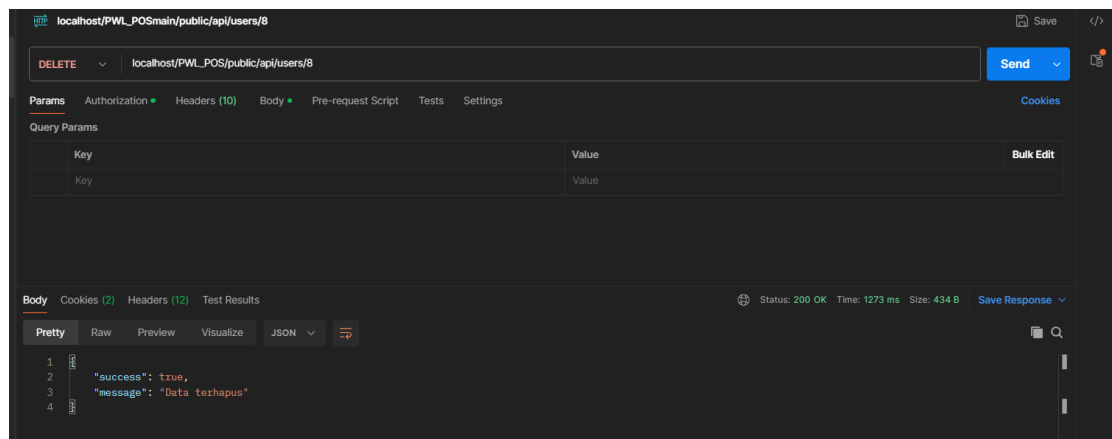
GET



PUT

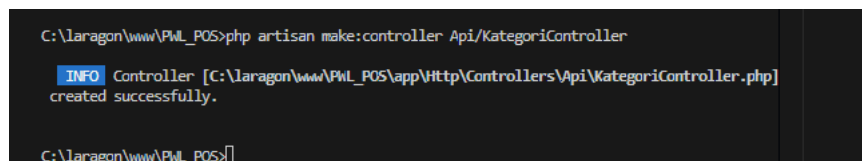


DELETE



3

Tabel m_kategori



KategoriController.php



```
api.php M KategoriController.php U X UserController.php U LevelController.php
app > Http > Controllers > Api > KategoriController.php > KategoriController
1  <?php
2
3  namespace App\Http\Controllers\Api;
4
5  use App\Http\Controllers\Controller;
6  use Illuminate\Http\Request;
7  use App\Models\KategoriModel;
8
9  class KategoriController extends Controller
10 {
11     public function index()
12     {
13         return KategoriModel::all();
14     }
15
16     public function store(Request $request)
17     {
18         $kategori = KategoriModel::create($request->all());
19         return response()->json($kategori, 201);
20     }
21
22     public function show(KategoriModel $kategori)
23     {
24         return KategoriModel::find($kategori);
25     }
26
27     public function update(Request $request, KategoriModel $kategori)
28     {
29         $kategori->update($request->all());
30         return KategoriModel::find($kategori);
31     }
32
33     public function destroy(KategoriModel $kategori)
34     {
35         $kategori->delete();
36         return response()->json([
37             'success' => true,
38             'message' => "Data terhapus",
39         ]);
40     }
41 }
```

Api.php

```
use App\Http\Controllers\Api\KategoriController;
// Kategori
Route::get('kategoris', [KategoriController::class, 'index']);
Route::post('kategoris', [KategoriController::class, 'store']);
Route::get('kategoris/{kategori}', [KategoriController::class, 'show']);
Route::put('kategoris/{kategori}', [KategoriController::class, 'update']);
Route::delete('kategoris/{kategori}', [KategoriController::class, 'destroy']);
```

Postman

POST



localhost/PWL_POS/public/api/kategoris

POST localhost/PWL_POS/public/api/kategoris Send

Params Auth Headers (9) Body Pre-req. Tests Settings Cookies

form-data

	Key	Value	...	Bulk Edit
<input checked="" type="checkbox"/>	kategori_kode	KD006		
<input checked="" type="checkbox"/>	kategori_nama	Perhiasan		
	Key	Value		

Body 201 Created 884 ms 553 B Save Response

Pretty Raw Preview Visualize JSON

```
1
2  "kategori_kode": "KD006",
3  "kategori_nama": "Perhiasan",
4  "updated_at": "2024-11-06T09:21:02.000000Z",
5  "created_at": "2024-11-06T09:21:02.000000Z",
6  "kategori_id": 12
7
```

GET



localhost/PWL_POS/public/api/kategoris

GET localhost/PWL_POS/public/api/kategoris Send

Params Auth Headers (9) Body Pre-req. Tests Settings Cookies

form-data

	Key	Value	...	Bulk Edit
<input checked="" type="checkbox"/>	kategori_kode	KD006		
<input checked="" type="checkbox"/>	kategori_nama	Perhiasan		
	Key	Value		

Body 200 OK 684 ms 1.07 KB Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "kategori_id": 1,
3   "kategori_kode": "KD001",
4   "kategori_nama": "Minuman",
5   "created_at": null,
6   "updated_at": "2024-09-30T10:11:19.000000Z"
7 },
8 {
9   "kategori_id": 2,
10  "kategori_kode": "KD002",
11  "kategori_nama": "Makanan",
12  "created_at": null,
```

PUT



localhost/PWL_POS/public/api/kategoris

PUT localhost/PWL_POS/public/api/kategoris Send

Params Auth Headers (9) Body Pre-req. Tests Settings Cookies

form-data

	Key	Value	...	Bulk Edit
<input checked="" type="checkbox"/>	kategori_kode	KD006		
<input type="checkbox"/>	kategori_nama	Perhiasan		
	Key	Value		

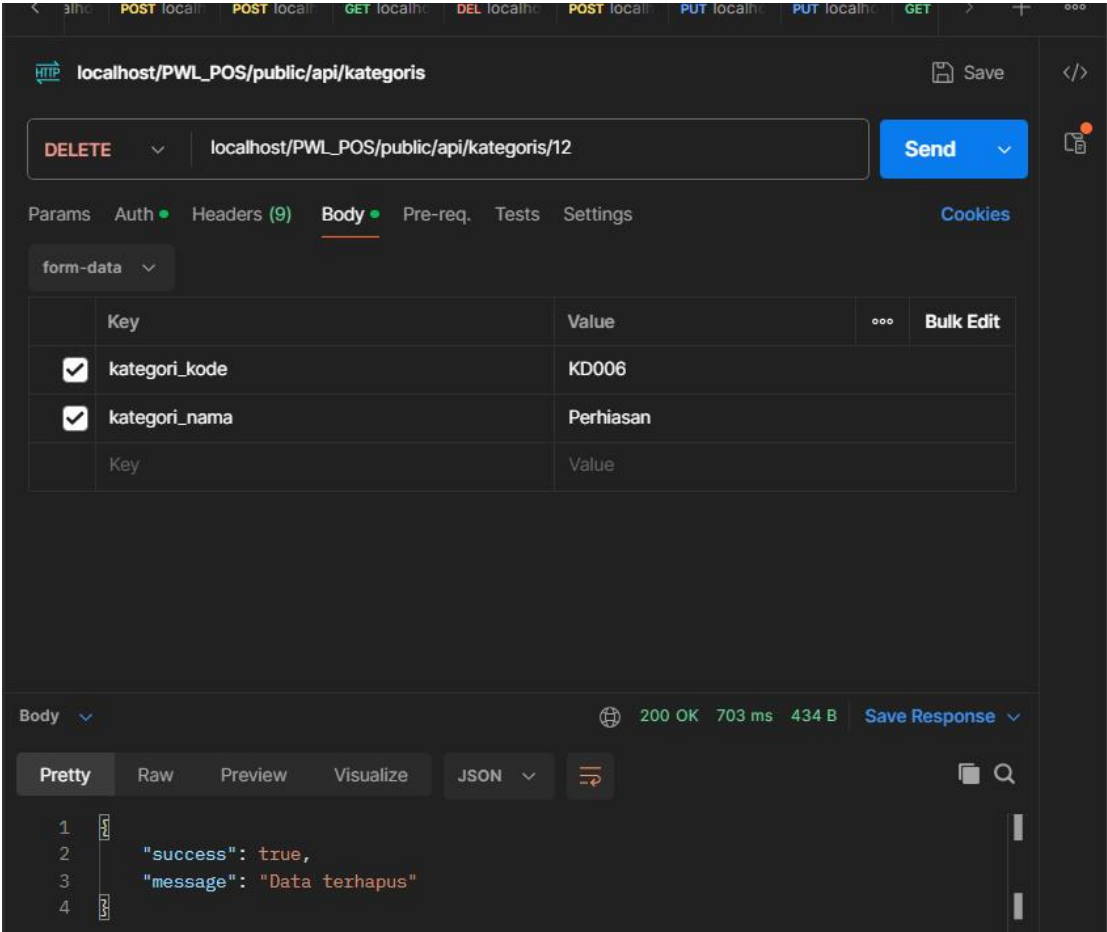
Body 405 Method Not Allowed 865 ms 1017.09 KB Save Response

Pretty Raw Preview Visualize HTML

```
1 <!DOCTYPE html>
2 <html lang="en" class="auto">
3 <!--
```

DELETE



	
4	<p>Tabel m_barang</p> <pre>C:\laragon\www\PWL_POS>php artisan make:controller Api/BarangController</pre> <pre>INFO Controller [C:\laragon\www\PWL_POS\app\Http\Controllers\Api\BarangController.php] created successfully.</pre> <pre>C:\laragon\www\PWL_POS></pre> <p>BarangController.php</p>



```
app > Http > Controllers > Api > BarangController.php > BarangController
1  <?php
2
3  namespace App\Http\Controllers\Api;
4
5  use App\Http\Controllers\Controller;
6  use Illuminate\Http\Request;
7  use App\Models\BarangModel;
8
9  class BarangController extends Controller
10 {
11     public function index()
12     {
13         return BarangModel::all();
14     }
15
16     public function store(Request $request)
17     {
18         $barang = BarangModel::create($request->all());
19         return response()->json($barang, 201);
20     }
21
22     public function show($id)
23     {
24         $barang = BarangModel::findOrFail($id); // findOrFail akan mencari data dengan ID dan mengembalikan 404 jika tidak ditemukan
25         return response()->json($barang, 200);
26     }
27
28     // Update data berdasarkan ID yang diberikan
29     public function update(Request $request, $id)
30     {
31         $barang = BarangModel::findOrFail($id);
32         $barang->update($request->all());
33         return response()->json($barang, 200); // Mengembalikan data setelah diupdate
34     }
35
36     public function destroy(BarangModel $barang)
37     {
38         $barang->delete();
39         return response()->json([
40             'success' => true,
41             'message' => "Data terhapus",
42         ]);
43     }
44 }
```

Api.php

```
//Barang
Route::get('barangs', [BarangController::class, 'index']);
Route::post('barangs', [BarangController::class, 'store']);
Route::get('barangs/{barang}', [BarangController::class, 'show']);
Route::put('barangs/{barang}', [BarangController::class, 'update']);
Route::delete('barangs/{barang}', [BarangController::class, 'destroy']);
```

Postman

POST



localhost/PWL_POS/public/api/barangs

POST localhost/PWL_POS/public/api/barangs Send

Params Auth Headers (9) Body Pre-req. Tests Settings Cookies

form-data

<input checked="" type="checkbox"/>	barang_kode	BRK006
<input checked="" type="checkbox"/>	barang_nama	Teh Kotak
<input checked="" type="checkbox"/>	harga_beli	4000
<input checked="" type="checkbox"/>	harga_jual	5000
	Key	Value

Body 201 Created 919 ms 606 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "kategori_id": "3",
3   "barang_kode": "BRK006",
4   "barang_nama": "Teh Kotak",
5   "harga_beli": "4000",
6   "harga_jual": "5000",
7   "updated_at": "2024-11-06T07:51:30.000000Z",
8   "created_at": "2024-11-06T07:51:30.000000Z",
9   "barang_id": 25
10 }
```

GET



localhost/PWL_POS/public/api/barangs

GET localhost/PWL_POS/public/api/barangs Send

Params Auth Headers (9) Body Pre-req. Tests Settings Cookies

form-data

	Key	Value	...	Bulk Edit
<input checked="" type="checkbox"/>	kategori_id	3		
<input checked="" type="checkbox"/>	barang_kode	BRK006		
<input checked="" type="checkbox"/>	barang_nama	Teh Kotak		
<input checked="" type="checkbox"/>	harga_beli	4000		

Body 200 OK 863 ms 3.88 KB Save Response

Pretty Raw Preview Visualize JSON

```
2 {
3   "barang_id": 1,
4   "kategori_id": 1,
5   "barang_kode": "BRK001",
6   "barang_nama": "Indomie Goreng",
7   "harga_beli": 2200,
8   "harga_jual": 2500,
9   "created_at": null,
10  "updated_at": null
11 },
12 {
13   "barang_id": 2,
14   "kategori_id": 2,
15   "barang_kode": "BRK002",
16   "barang_nama": "Roti Bakar",
17   "harga_beli": 2000,
18   "harga_jual": 3000,
19   "created_at": null,
20   "updated_at": "2024-09-30T10:38:46.000000Z"
```

PUT



HTTP client interface showing a PUT request to `localhost/PWL_POS/public/api/barangs`. The request body is form-data with the following fields:

Field	Value
<input type="checkbox"/> kategori_id	
<input type="checkbox"/> barang_kode	BRK006
<input type="checkbox"/> barang_nama	Teh Kotak
<input type="checkbox"/> harga_beli	4000
<input checked="" type="checkbox"/> harga_jual	5000

The response status is **405 Method Not Allowed**. The response body is HTML showing an error message:

```
1 <!DOCTYPE html>
2 <html lang="en" class="auto">
3 <!--
4 Symfony\Component\HttpKernel\Exception\MethodNotAllowedHttpException: The PUT method
  is not supported for route api/barangs. Supported methods: GET, HEAD, POST. in
  file
  C:\laragon\www\PWL_POS\vendor\laravel\framework\src\Illuminate\Routing\AbstractRo
  uteCollection.php on line 122
5
6 #0 C:\laragon\www\PWL_POS\vendor\laravel\framework\src\Illuminate\Routing\AbstractRou
  teCollection.php(107): Illuminate\Routing\AbstractRouteCollection->
  requestMethodNotAllowed(Object(Illuminate\Http\Request), Array, &#039;PUT&#039;);
7 #1 C:\laragon\www\PWL_POS\vendor\laravel\framework\src\Illuminate\Routing\AbstractRou
  teCollection.php(41): Illuminate\Routing\AbstractRouteCollection->
  getRouteForMethods(Object(Illuminate\Http\Request), Array)
8 #2 C:\laragon\www\PWL_POS\vendor\laravel\framework\src\Illuminate\Routing\RouteCollec
  tion.php(162): Illuminate\Routing\AbstractRouteCollection->handleMatchedRoute
  (Object(Illuminate\Http\Request), NULL)
9 #3 C:\laragon\www\PWL_POS\vendor\laravel\framework\src\Illuminate\Routing\Router.php
```

DELETE



localhost/PWL_POS/public/api/barangs/10

DELETElocalhost/PWL_POS/public/api/barangs/10Send

ParamsAuthHeaders (9)BodyPre-reqTestsSettingsCookies

form-data

	Key	Value	...	Bulk Edit
<input checked="" type="checkbox"/>	kategori_id	3		
<input checked="" type="checkbox"/>	barang_kode	BRK006		
<input checked="" type="checkbox"/>	barang_nama	Teh Kotak		
<input checked="" type="checkbox"/>	harga_beli	4000		

Body200 OK866 ms434 BSave Response

PrettyRawPreviewVisualizeJSON

```
1 {
2   "success": true,
3   "message": "Data terhapus"
4 }
```