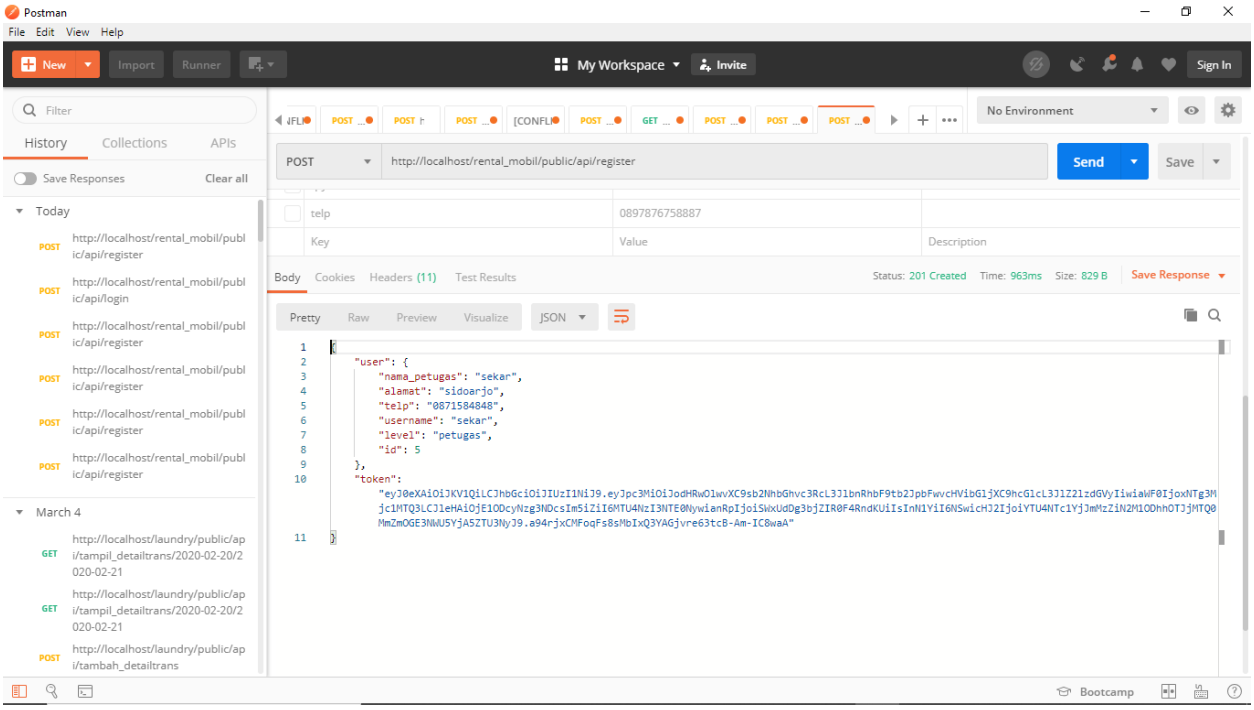
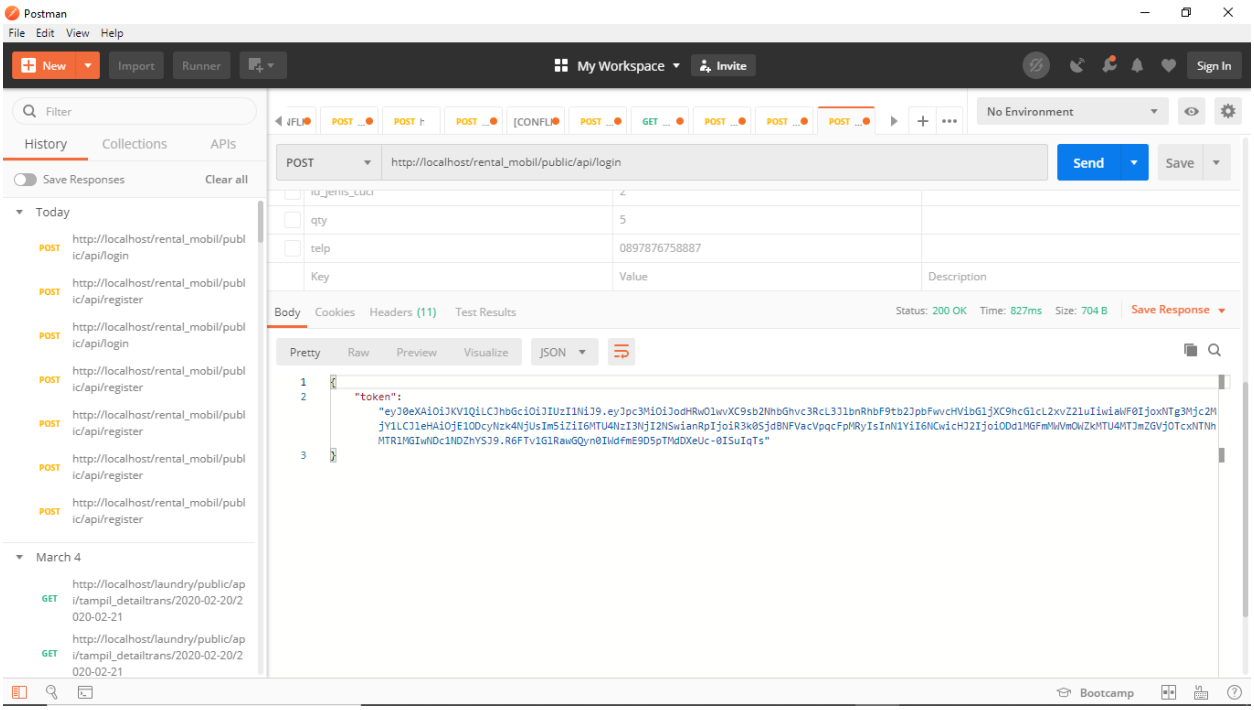
API register



API login



Api.php

```
////////////PETUGAS

Route::post('register', 'PetugasController@register');
Route::post('login', 'petugasController@login');
Route::put('/ubah_petugas/{id}','PetugasController@updat
e');
Route::delete('/hapus_petugas/{id}','PetugasController@h
apus');
Route::get('/tampil_petugas','PetugasController@tampil_p
etugas');
Route::get('/',function(){
    return Auth::user()->level;
});
```

petugasModel.php

```
<?php

namespace App;

use Illuminate\Notifications\Notifiable;
use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Tymon\JWTAuth\Contracts\JWTSubject;

class PetugasModel extends Authenticatable implements JW
TSubject
{
    use Notifiable;
    protected $table="petugas";
    public $timestamps=false;
```

```
    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = [
        'nama_petugas','alamat','telp', 'username', 'pas
sword','level',
    ];

    /**
     * The attributes that should be hidden for arrays.
     *
     * @var array
     */
    protected $hidden = [
        'password', 'remember_token',
    ];

    public function getJWTIdentifier()
    {
        return $this->getKey();
    }
    public function getJWTCustomClaims()
    {
        return [];
    }
}
```

PetugasController.php

```php
<?php

namespace App\Http\Controllers;

use App\PetugasModel;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Validator;
use JWTAuth;
use Tymon\JWTAuth\Exceptions\JWTException;
use Auth;

class PetugasController extends Controller
{
    public function login(Request $request)
    {
        $credentials = $request->only('username','password');

        try {
            if (! $token = JWTAuth::attempt($credentials)) {
                return response()->json(['error' => 'invalid_credentials'], 400);
            }
        } catch (JWTException $e) {
            return response()->json(['error' => 'could_not_create_token'], 500);
        }

        return response()->json(compact('token'));
    }
```

```
    public function register(Request $request)
    {
        $validator = Validator::make($request->all(), [
            'nama_petugas' => 'required|string|max:255',
            'alamat' => 'required|string|max:255',
            'telp' => 'required',
            'username' => 'required|string|max:255',
            'password' => 'required|string|min:6|confirm
ed',
            'level' => 'required',
        ]);

        if($validator->fails()){
            return response()->json($validator-
>errors()->toJson(), 400);
        }

        $user = PetugasModel::create([
            'nama_petugas' => $request-
>get('nama_petugas'),
            'alamat' => $request->get('alamat'),
            'telp' => $request->get('telp'),
            'username' => $request->get('username'),
            'password' => Hash::make($request-
>get('password')),
            'level' => $request->get('level'),
        ]);

        $token = JWTAuth::fromUser($user);
```

```
        $update = PetugasModel::where('nama_petugas', $r
equest->nama_petugas)->update([
            'nama_petugas' => $request-
>get('nama_petugas'),
            'alamat' => $request->get('alamat'),
            'telp' => $request->get('telp'),
            'username' => $request->get('username'),
            'password' => Hash::make($request-
>get('password')),
            'level' => $request->get('level'),
        ]);

        return response()-
>json(compact('user','token'),201);
    }

    public function update($id,Request $req)
    {
        $validator=Validator::make($req->all(),
        [
            'nama_petugas'=>'required',
            'alamat' =>'required',
            'telp' =>'required',
            'username'=>'required',
            'password'=>'required',
            'level'=>'required'
        ]);

        if($validator->fails()){
            return Response()->json($validator-
>errors());
        }
```

```
        $ubah=PetugasModel::where('id', $id)->update([
            'nama_petugas'=>$req->nama_petugas,
            'alamat' =>$req->alamat,
            'telp'=>$req->telp,
            'username'=>$req->username,
            'password'=>$req->password,
            'level'=>$req->level
        ]);
        if($ubah){
            return Response()->json(['status'=>1]);
        }else{
            return Response()->json(['status'=>0]);
        }
    }

    public function hapus($id)
    {
        $hapus=PetugasModel::where('id',$id)->delete();
        if($hapus){
            return Response()->json(['status'=>1]);
        }else{
            return Response()->json(['status'=>0]);
        }
    }

    public function tampil_petugas()
    {
        $data_petugas=PetugasModel::get();
        return Response()->json($data_petugas);
    }

    public function getAuthenticatedUser()
```

```
    {
        try {

            if (! $user = JWTAuth::parseToken()->authenticate()) {
                return response()->json(['user_not_found'], 404);
            }

        } catch (Tymon\JWTAuth\Exceptions\TokenExpiredException $e) {

            return response()->json(['token_expired'], $e->getStatusCode());

        } catch (Tymon\JWTAuth\Exceptions\TokenInvalidException $e) {

            return response()->json(['token_invalid'], $e->getStatusCode());

        } catch (Tymon\JWTAuth\Exceptions\JWTException $e) {

            return response()->json(['token_absent'], $e->getStatusCode());

        }

        return response()->json(compact('user'));
    }
}
```