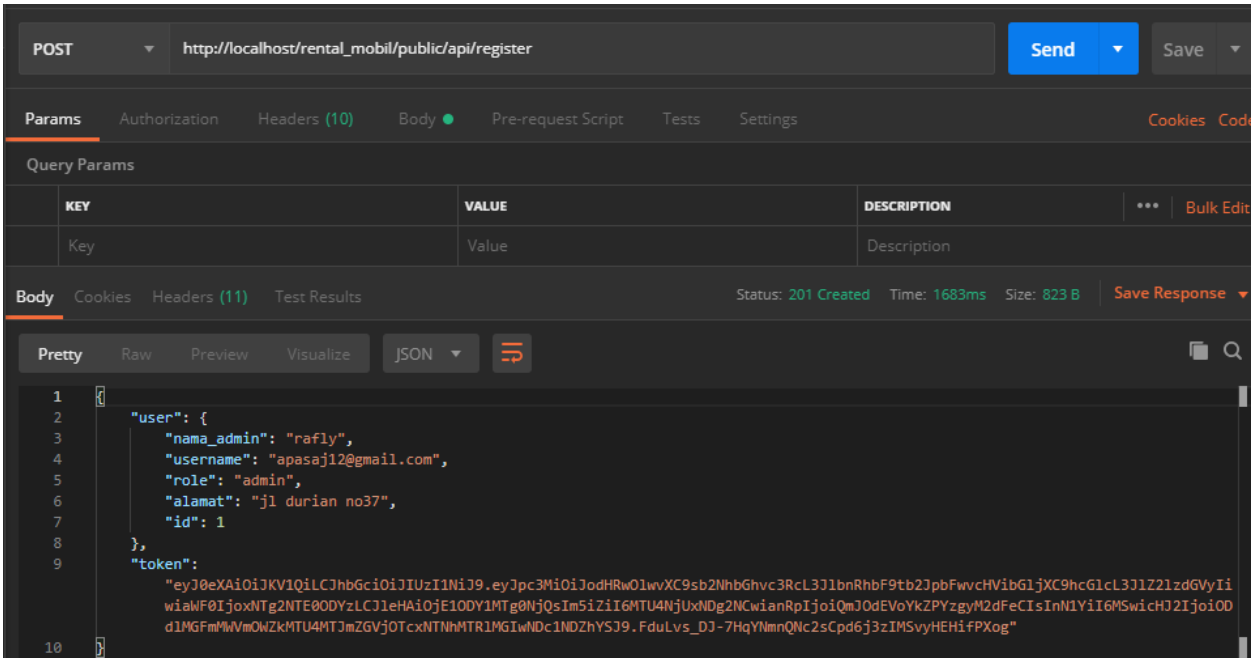
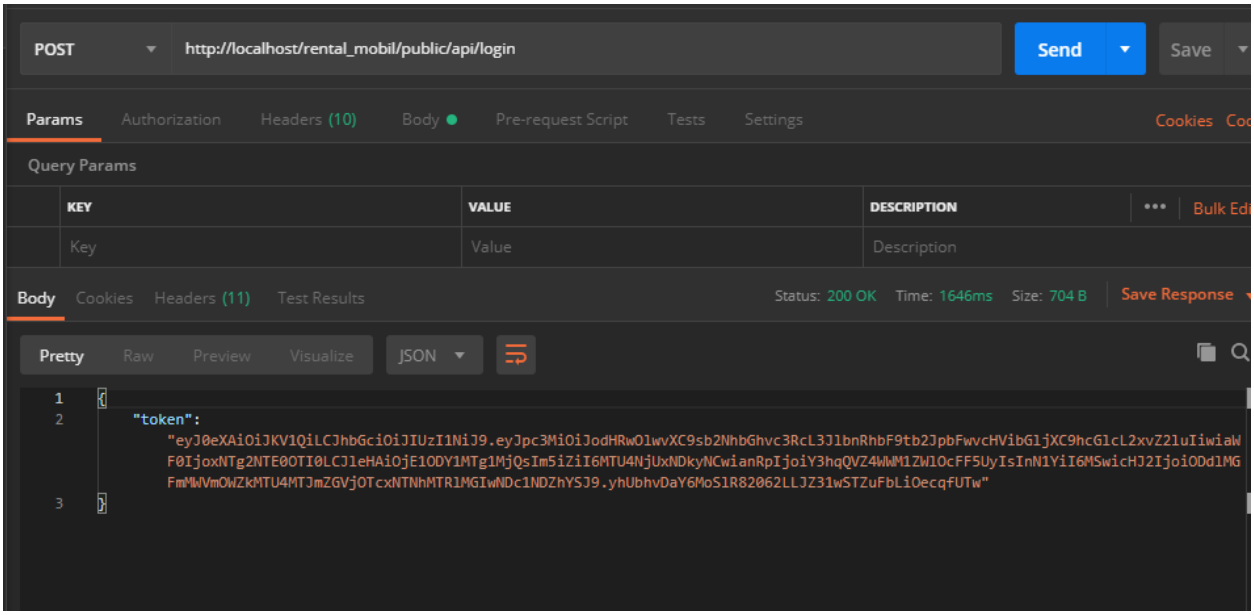
Nama : Mukhamad Rico Ramada

Kelas : XIRPL6

1. Register



2. Login

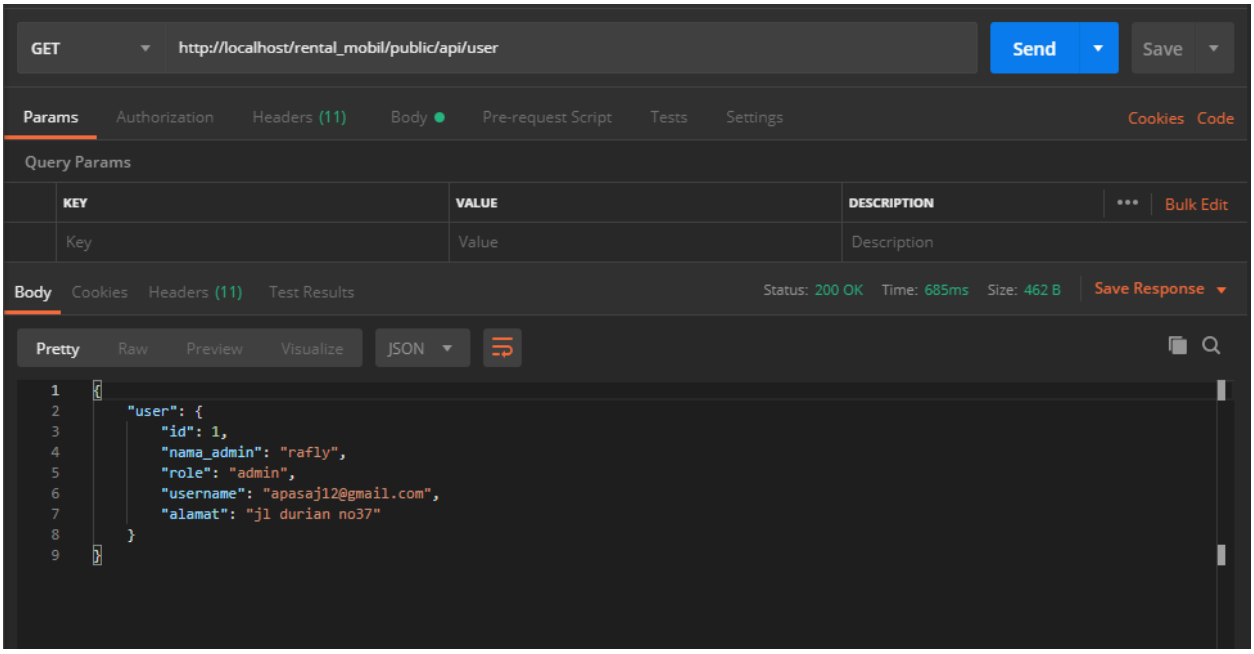


```php
<?php

namespace App\Http\Controllers;

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

class UserController 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_admin' => 'required|string|max:255',
            'role' => 'required|string|max:255',
            'username' => 'required|string|email|max:255|unique:admin',
            'password' => 'required|string|min:8|confirmed',
            'alamat' => 'required|string|max:255',
        ]);

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

        $user = User::create([
            'nama_admin' => $request->get('nama_admin'),
            'username' => $request->get('username'),
            'password' => Hash::make($request->get('password')),
            'role' => $request->get('role'),
            'alamat' => $request->get('alamat'),
        ]);

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

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

```

API

```php
<?php

use Illuminate\Http\Request;

/*
|--------------------------------------------------------------------------
| 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:api')->get('/user', function (Request $request) {
    return $request->user();
});
Route::post('register', 'UserController@register');
Route::post('login', 'UserController@login');
Route::get('book', 'BookController@book');

Route::get('bookall', 'BookController@bookAuth')->middleware('jwt.verify');
Route::get('user', 'UserController@getAuthenticatedUser')->middleware('jwt.verify');

```


GET
http://localhost/rental_mobil/public/api/user
Send
Save
Params
Authorization
Headers (11)
Body
Pre-request Script
Tests
Settings
Cookies
Code
Query Params
KEY
VALUE
DESCRIPTION
Bulk Edit
Key
Value
Description
Body
Cookies
Headers (11)
Test Results
Status: 200 OK
Time: 685ms
Size: 462 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"user": {
"id": 1,
"nama_admin": "rafly",
"role": "admin",
"username": "apasaj12@gmail.com",
"alamat": "jl durian no37"
}
}

```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 User extends Authenticatable implements JWTSubject
{
    use Notifiable;
    protected $table = 'admin';
    protected $primaryKey = 'id';
    public $timestamps =false;
    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = [
        'nama_admin','role','username','password','alamat',
    ];

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

    /**
     * The attributes that should be cast to native types.
     *
     * @var array
     */
    protected $casts = [
        'email_verified_at' => 'datetime',
    ];
    public function getJWTIdentifier()
    {
        return $this->getKey();
    }
    public function getJWTCustomClaims()
    {
        return [];
    }
}
```