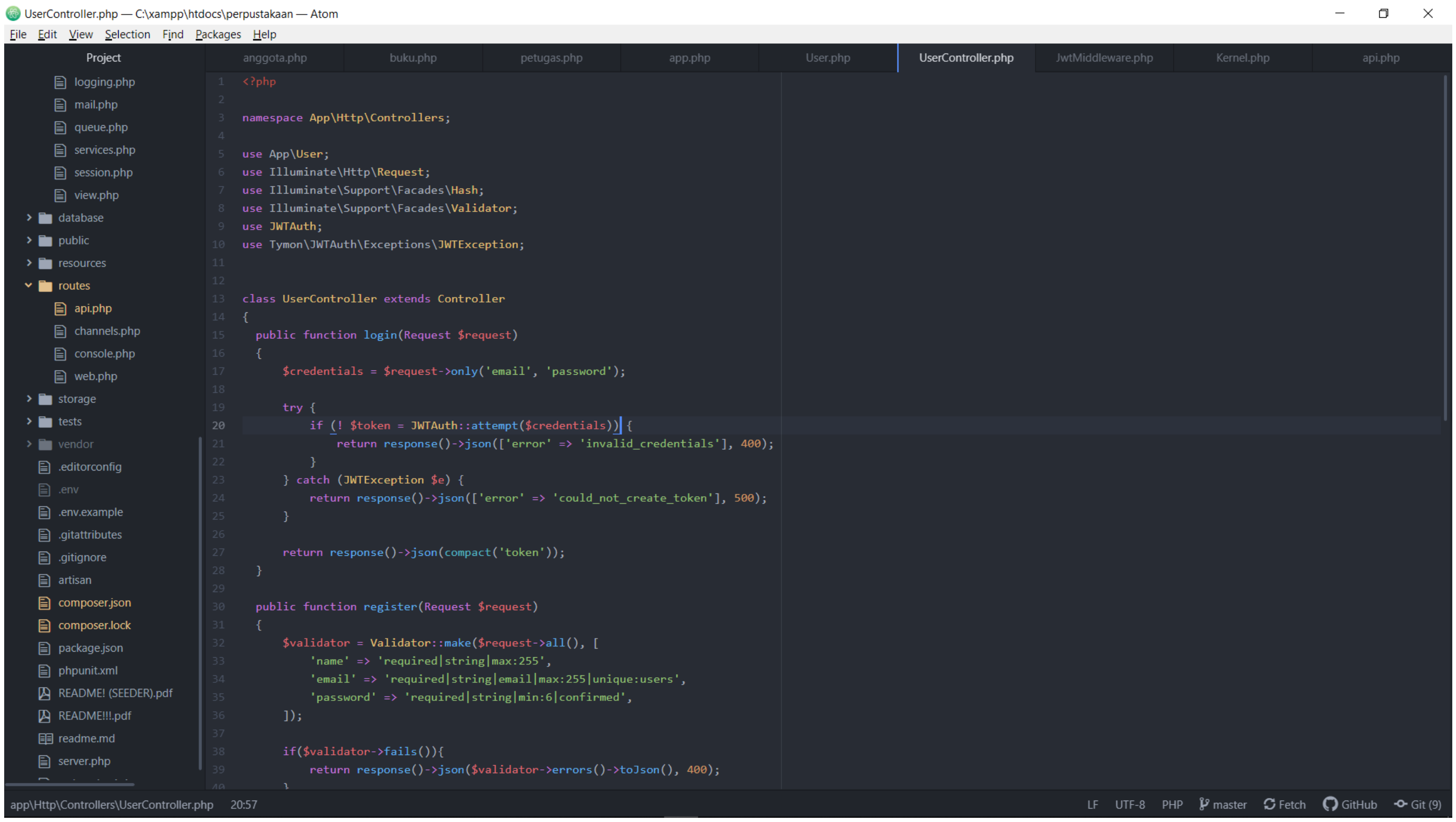
Nama: Rafi Raihan F

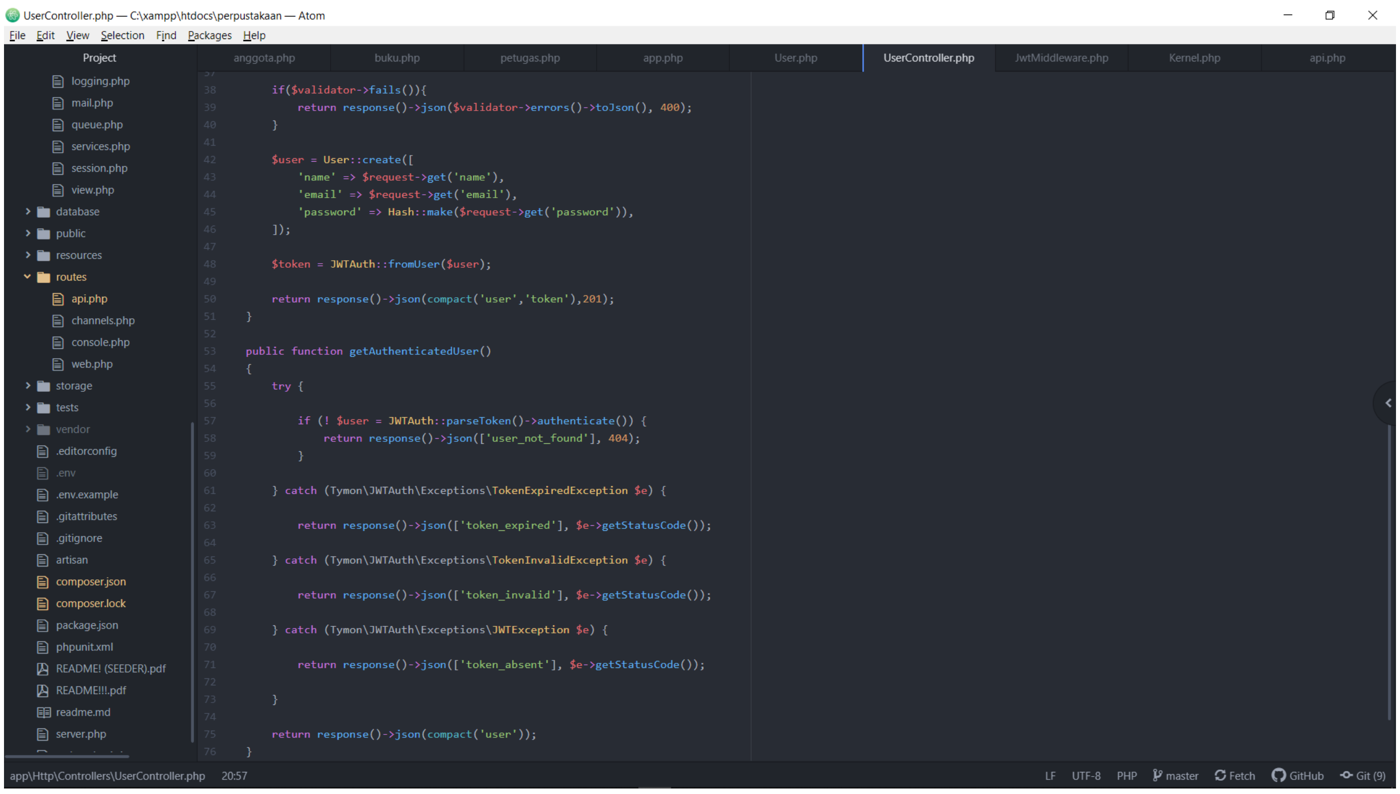
Kelas: XI RPL 6

No.absen: 30

Script UserController

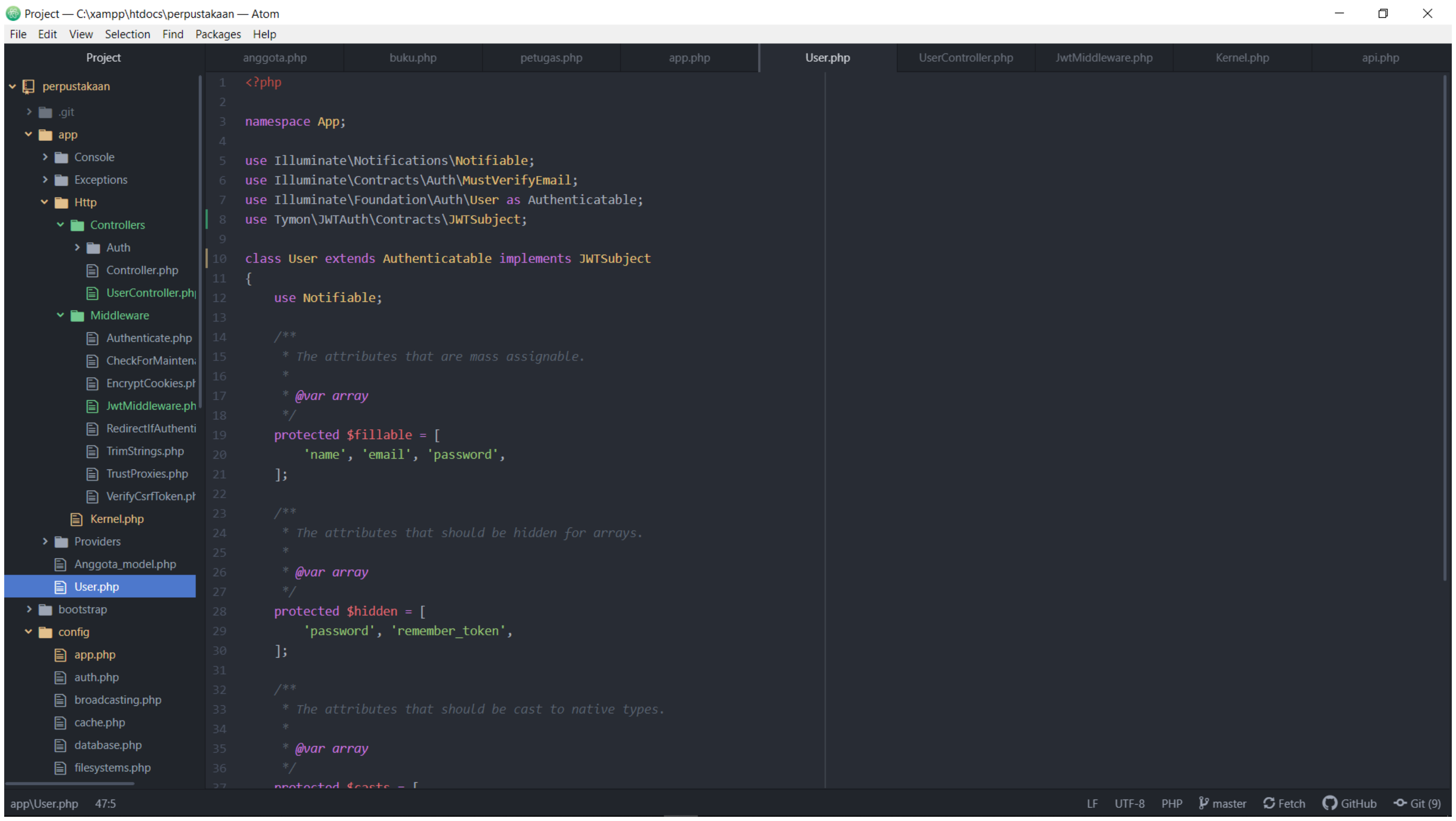

UserController.php — C:\xampp\htdocs\perpustakaan — Atom
File Edit View Selection Find Packages Help
Project
anggota.php buku.php petugas.php app.php User.php UserController.php JwtMiddleware.php Kernel.php api.php
logging.php mail.php queue.php services.php session.php view.php database public resources routes api.php channels.php console.php web.php storage tests vendor .editorconfig .env .env.example .gitattributes .gitignore artisan composer.json composer.lock package.json phpunit.xml README! (SEEDER).pdf README!!!.pdf readme.md server.php
app\Http\Controllers\UserController.php 20:57
LF UTF-8 PHP master Fetch GitHub Git (9)

```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('email', '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(), [
          'name' => 'required|string|max:255',
          'email' => 'required|string|email|max:255|unique:users',
          'password' => 'required|string|min:6|confirmed',
      ]);

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

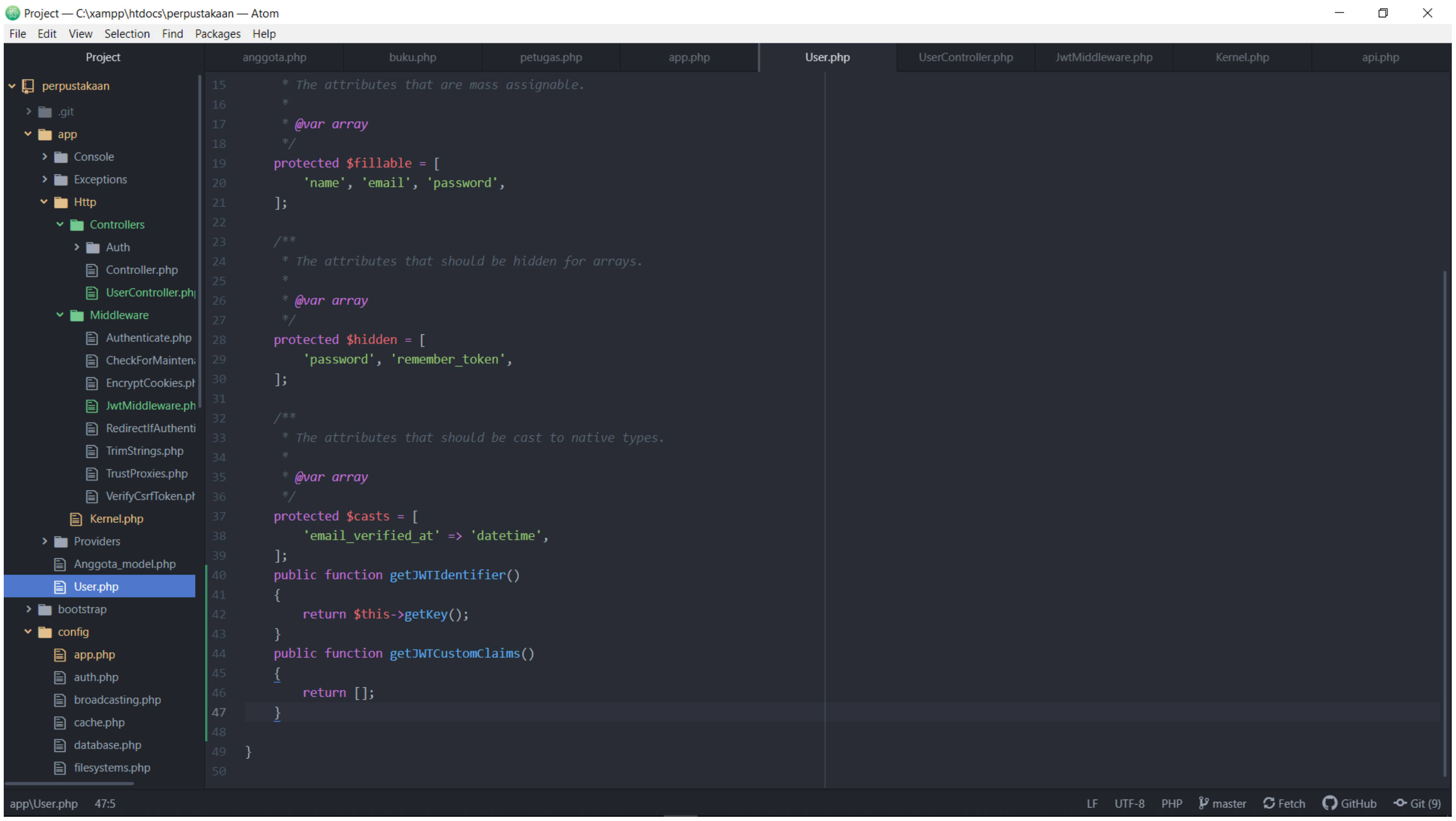

UserController.php — C:\xampp\htdocs\perpustakaan — Atom
File Edit View Selection Find Packages Help
Project
anggota.php buku.php petugas.php app.php User.php UserController.php JwtMiddleware.php Kernel.php api.php
logging.php mail.php queue.php services.php session.php view.php database public resources routes api.php channels.php console.php web.php storage tests vendor .editorconfig .env .env.example .gitattributes .gitignore artisan composer.json composer.lock package.json phpunit.xml README! (SEEDER).pdf README!!!.pdf readme.md server.php
app\Http\Controllers\UserController.php 20:57
LF UTF-8 PHP master Fetch GitHub Git (9)

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

      $user = User::create([
          'name' => $request->get('name'),
          'email' => $request->get('email'),
          'password' => Hash::make($request->get('password')),
      ]);

      $token = JWTAuth::fromUser($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(['token_invalid'], $e->getStatusCode());

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

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

      }

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

Script User.php

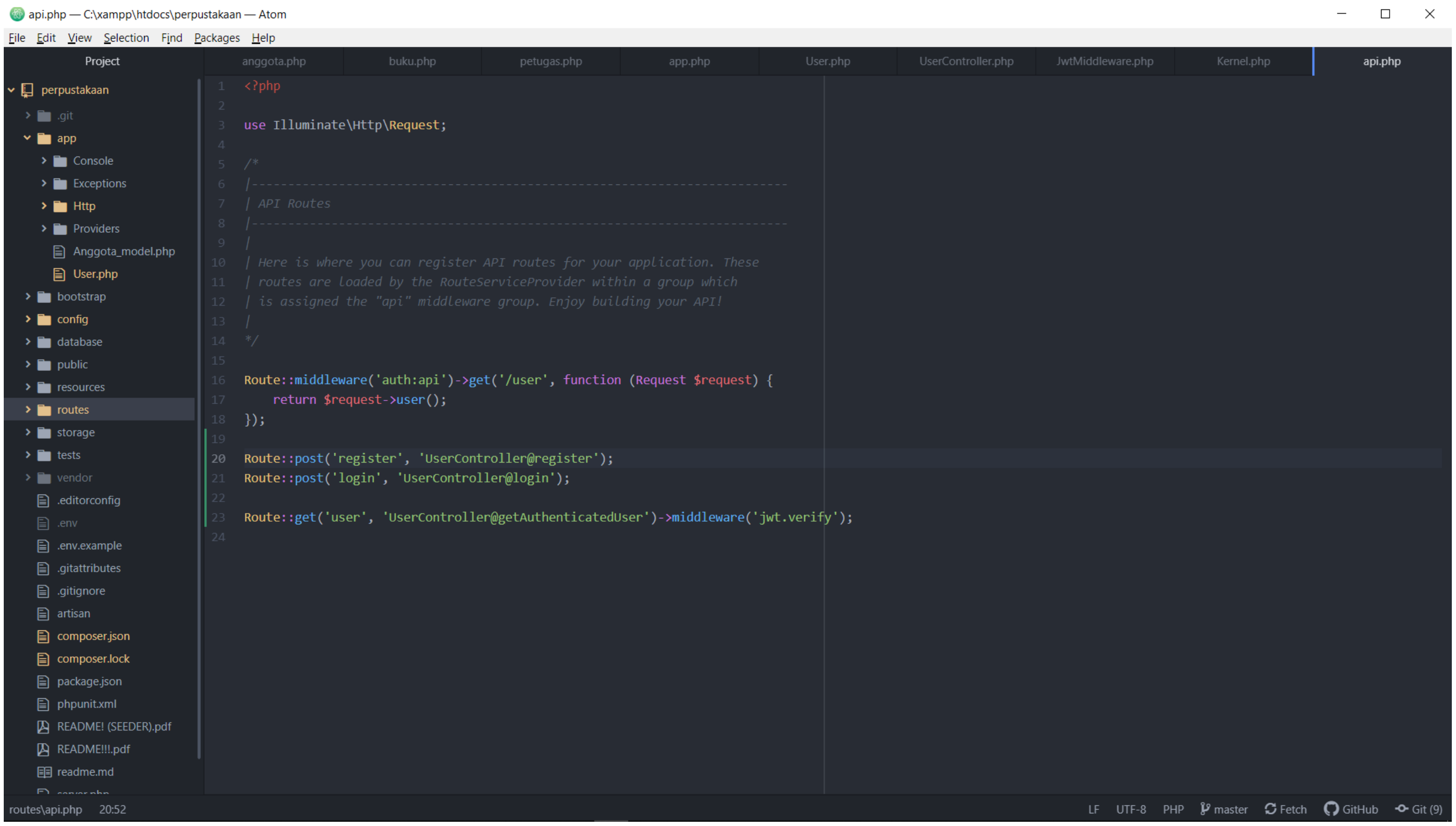

Project — C:\xampp\htdocs\perpustakaan — Atom
File Edit View Selection Find Packages Help
anggota.php buku.php petugas.php app.php User.php UserController.php JwtMiddleware.php Kernel.php api.php
app\User.php 47:5
LF UTF-8 PHP master Fetch GitHub Git (9)

```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;

    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = [
        'name', 'email', 'password',
    ];

    /**
     * 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
     */
```


Project — C:\xampp\htdocs\perpustakaan — Atom
File Edit View Selection Find Packages Help
anggota.php buku.php petugas.php app.php User.php UserController.php JwtMiddleware.php Kernel.php api.php
app\User.php 47:5
LF UTF-8 PHP master Fetch GitHub Git (9)

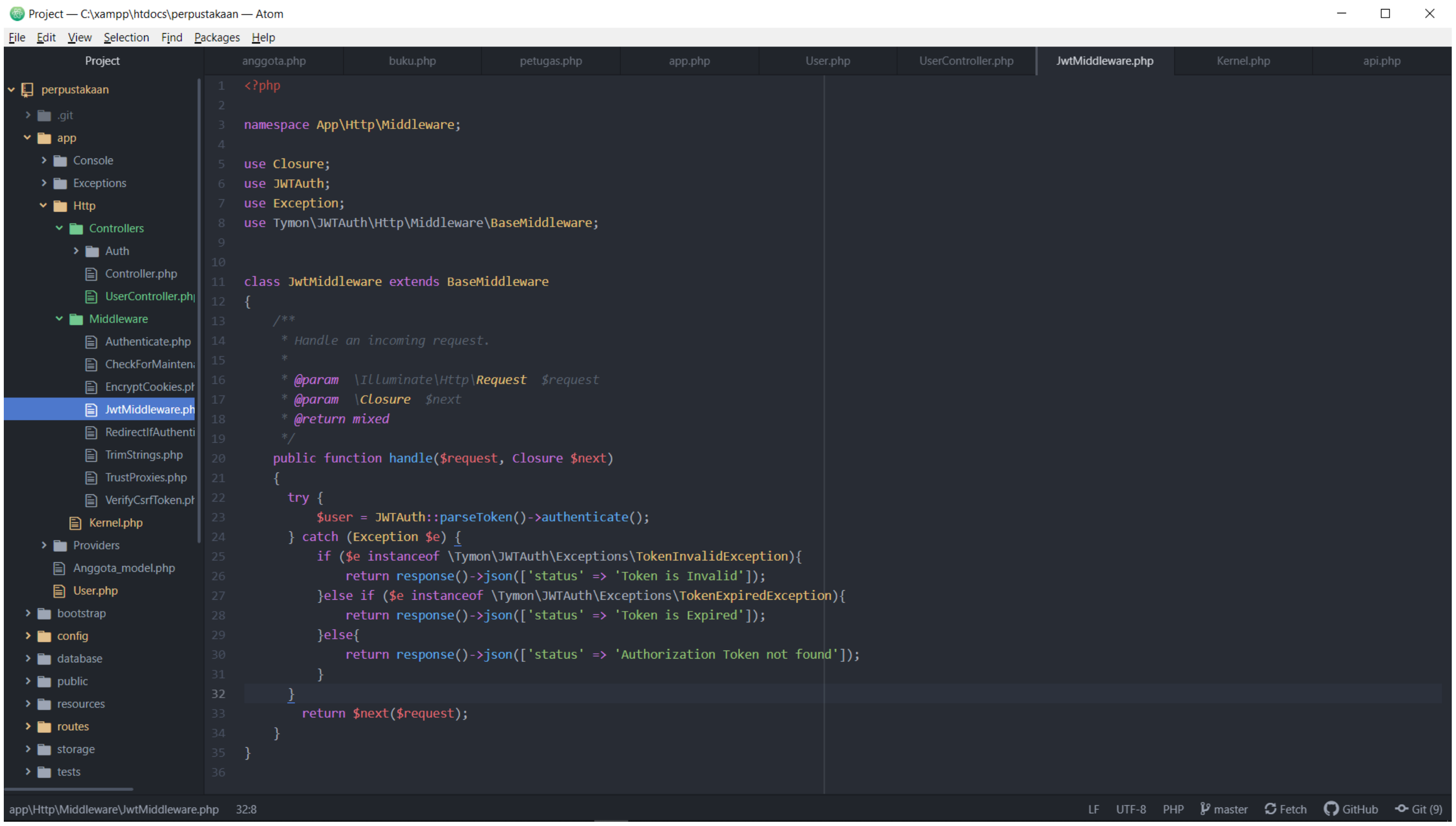
```php
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = [
        'name', 'email', 'password',
    ];

    /**
     * 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 [];
    }

}
```

Script api.php


api.php — C:\xampp\htdocs\perpustakaan — Atom

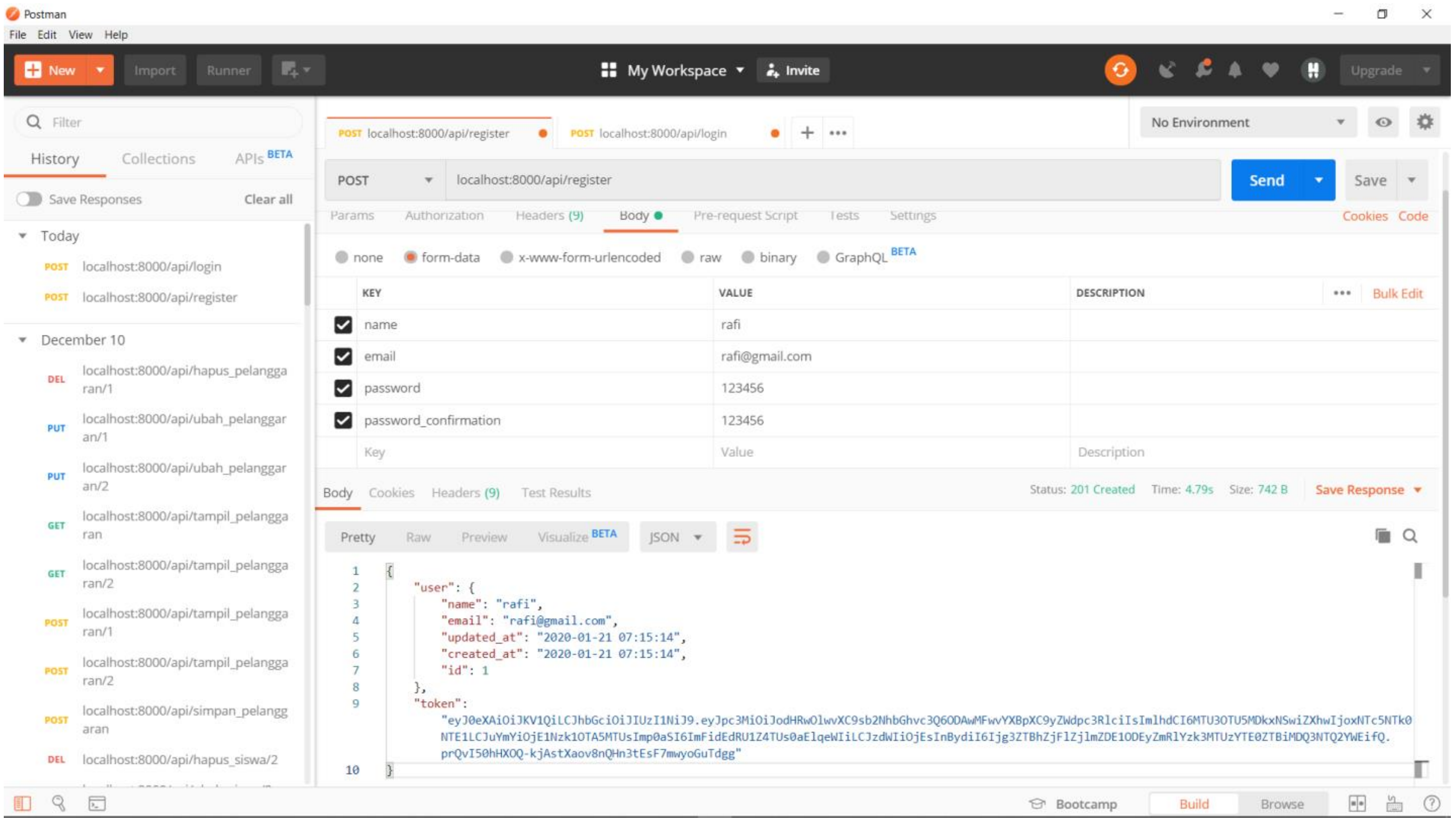
```php
<?php

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('user', 'UserController@getAuthenticatedUser')->middleware('jwt.verify');
```

Script JwtMiddleware


Project — C:\xampp\htdocs\perpustakaan — Atom

```php
<?php

namespace App\Http\Middleware;

use Closure;
use JWTAuth;
use Exception;
use Tymon\JWTAuth\Http\Middleware\BaseMiddleware;


class JwtMiddleware extends BaseMiddleware
{
    /**
     * Handle an incoming request.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \Closure  $next
     * @return mixed
     */
    public function handle($request, Closure $next)
    {
      try {
          $user = JWTAuth::parseToken()->authenticate();
      } catch (Exception $e) {
          if ($e instanceof \Tymon\JWTAuth\Exceptions\TokenInvalidException){
              return response()->json(['status' => 'Token is Invalid']);
          }else if ($e instanceof \Tymon\JWTAuth\Exceptions\TokenExpiredException){
              return response()->json(['status' => 'Token is Expired']);
          }else{
              return response()->json(['status' => 'Authorization Token not found']);
          }
      }
        return $next($request);
    }
}
```

Postman register



POST localhost:8000/api/register

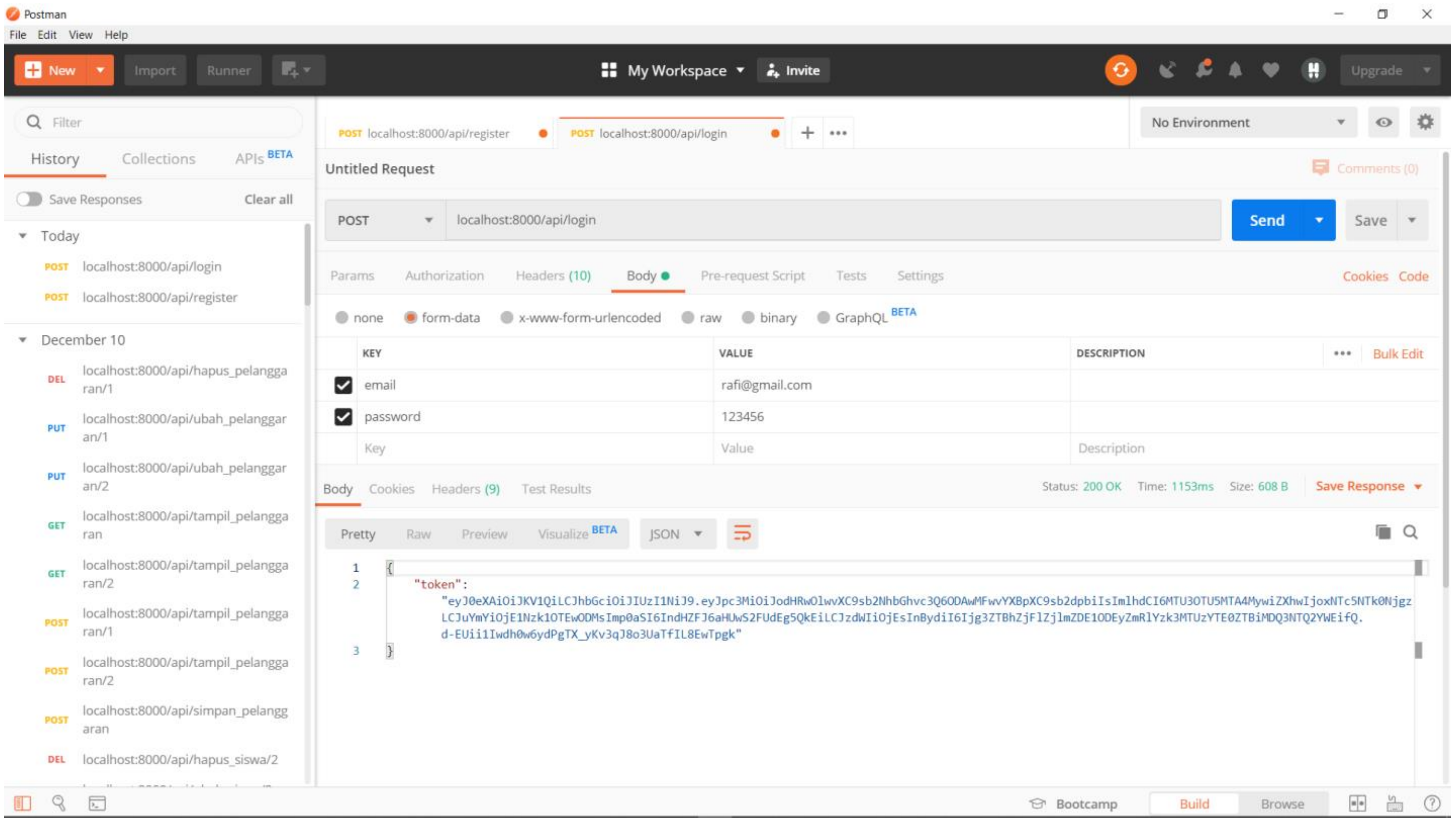
Body: form-data

KEY	VALUE	DESCRIPTION
name	rafi	
email	rafi@gmail.com	
password	123456	
password_confirmation	123456	

Status: 201 Created Time: 4.79s Size: 742 B

```
{
    "user": {
        "name": "rafi",
        "email": "rafi@gmail.com",
        "updated_at": "2020-01-21 07:15:14",
        "created_at": "2020-01-21 07:15:14",
        "id": 1
    },
    "token":
        "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwOlwvXC9sb2NhbGhvc3Q6ODAwMFwvYXBpXC9yZWdpc3RlciIsImlhdCI6MTU3OTU5MDkxNSwiZXhwIjoxNTc5NTk0NTE1LCJuYmYiOjE1Nzk1OTA5MTUsImp0aSI6ImFidEdRU1Z4TUs0aElqeWIiLCJzdWIiOjEsInByditI6Ijg3ZTBhZjFlZjlmZDE1ODEyZmRlYzk3MTUzYTE0ZTBiMDQ3NTQ2YWEifQ.prQvI50hHXOQ-kjAstXaov8nQHn3tEsF7mwyoGuTdgg"
}
```

Postman login



POST localhost:8000/api/login

Body: form-data

KEY	VALUE	DESCRIPTION
email	rafi@gmail.com	
password	123456	

Status: 200 OK Time: 1153ms Size: 608 B

```
{
    "token":
        "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwOlwvXC9sb2NhbGhvc3Q6ODAwMFwvYXBpXC9sb2dpbiIsImlhdCI6MTU3OTU5MTA4MywiZXhwIjoxNTc5NTk0NjgzLCJuYmYiOjE1Nzk1OTEwODMsImp0aSI6IndHZFJ6aHUwS2FUdEg5QkEiLCJzdWIiOjEsInBydiI6Ijg3ZTBhZjFlZjlmZDE1ODEyZmRlYzk3MTUzYTE0ZTBiMDQ3NTQ2YWEifQ.d-EUii1Iwdh0w6ydPgTX_yKv3qJ8o3UaTfIL8EwTpgk"
}
```