Unit API Authentication with Sanctum in Laravel

Create new larvael project name API Authentication and select mySql as Database.

Open project in visual studio code and go to terminal

Php artisan install:api

Status Codes

Response Status code	Used for
200	OK
201	Data Created with POST
204	Data deleted
400	Bad Request
401	Unauthorized
403	Forbidden
404	Resource not found
500	Internal server error

Changes in migration files

Remove email_varified_at from users migration table

New file

php artisan make:migration create_students

```
Schema::create('students', function (Blueprint $table) {
            $table->id();
            $table->string("fname", 20);
            $table->string("lname", 20);
            $table->string("city", 20);
            $table->string("email", 64);
            $table->string("phone", 10);
            $table->string("gender",10);
            $table->tinyInteger("age");
            $table->timestamps();
        });
```

Run following command

php artisan migrate:refresh

open users model and update as following

```
use Laravel\Sanctum\HasApiTokens;

class User extends Authenticatable
{

    use HasFactory, Notifiable, HasApiTokens;
}
```

Create required models and controllers

php artisan make:model student

php artisan make:controller studentsController --api

php artisan make:controller usersController

usersController file

```php
<?php

namespace App\Http\Controllers;

use App\Models\User;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Validator;

class usersController extends Controller
{
    // Signup function
    public function signup(Request $request){
        //return $request->all();
        $validatedUser = validator::make($request->all(),[
            "name" => "required",
            "email" => "required|email|unique:users,email",
            "password" => "required"
        ]);


        if(!$validatedUser->fails()){
            $user = User::create([
                "name" => $request->name,
                "email" => $request->email,
                "password" => $request->password,
            ]);
```

```
            return response()->json([
                "status" => true,
                "message" => "New User Created",
                "user" => $user
            ], 200);

        }else{
            return response()->json([
                "status" => false,
                "message" => "Validation Failed",
                "errors" => $validatedUser->errors()->all()
            ], 401);
        }
    }

    // Signup function
    public function login(Request $request){
        $validatedUser = validator::make($request->all(),[
            "email" => "required",
            "password" => "required"
        ]);
```

```
        if(!$validatedUser->fails()){
            if(Auth::attempt(["email" => $request->email,
"password" => $request->password])){
                $authenticatedUser = Auth::user();
                return response()->json([
                    "status" => true,
                    "message" => "Login Successfully",
                    "token" => $authenticatedUser-
>createToken("API-Token")->plainTextToken,
                    "tokenType" => "bearer",
                    "authentcatedUser" => $authenticatedUser
                ], 401);
            }else{
                return response()->json([
                    "status" => false,
                    "message" => "Invalid Username or Password",
                    "errors" => $validatedUser->errors()->all()
                ], 401);
            }
        }else{
            return response()->json([
                "status" => false,
                "message" => "Validation Failed",
                "errors" => $validatedUser->errors()->all()
            ], 401);
        }
    }

    // Signup function
    public function logout(Request $request){
        $user = $request->user();
        $user->tokens()->delete();
```

```
        return response()->json([
            "status" => true,
            "message" => "Logout Successfully",
            "user" => $user
        ], 200);
    }
}
```

student model

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class student extends Model
{
    use HasFactory;

    protected $fillable = [
        "fname", "lname", "city", "email", "phone", "gender",
"age"
    ];

    protected function casts(): array
    {
        return [
            'created_at' => 'datetime',
            'updated_at' => 'datetime',
        ];
    }
}
```

usersController

```php
<?php

namespace App\Http\Controllers;

use App\Models\User;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Validator;

class usersController extends Controller
{
    // Signup function
    public function signup(Request $request){
        //return $request->all();
        $validatedUser = validator::make($request->all(),[
            "name" => "required",
            "email" => "required|email|unique:users,email",
            "password" => "required"
        ]);


        if(!$validatedUser->fails()){
            $user = User::create([
                "name" => $request->name,
                "email" => $request->email,
                "password" => $request->password,
            ]);

            return response()->json([
                "status" => true,
```

```
                "message" => "New User Created",
                "user" => $user
            ], 200);

        }else{
            return response()->json([
                "status" => false,
                "message" => "Validation Failed",
                "errors" => $validatedUser->errors()->all()
            ], 401);
        }
    }

    // Signup function
    public function login(Request $request){
        $validatedUser = validator::make($request->all(),[
            "email" => "required",
            "password" => "required"
        ]);

        if(!$validatedUser->fails()){
            if(Auth::attempt(["email" => $request->email,
"password" => $request->password])){
                $authenticatedUser = Auth::user();
                return response()->json([
                    "status" => true,
                    "message" => "Login Successfully",
                    "token" => $authenticatedUser-
>createToken("API-Token")->plainTextToken,
                    "tokenType" => "bearer",
                    "authentcatedUser" => $authenticatedUser
                ], 401);
            }else{
```

```
                return response()->json([
                    "status" => false,
                    "message" => "Invalid Username or Password",
                    "errors" => $validatedUser->errors()->all()
                ], 401);
            }
        }else{
            return response()->json([
                "status" => false,
                "message" => "Validation Failed",
                "errors" => $validatedUser->errors()->all()
            ], 401);
        }
    }

    // Signup function
    public function logout(Request $request){
        $user = $request->user();
        $user->tokens()->delete();

        return response()->json([
            "status" => true,
            "message" => "Logout Successfully",
            "user" => $user
        ], 200);
    }
}
```

Studentscontroller

```php
<?php

namespace App\Http\Controllers;

use App\Models\student;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Validator;


class studentsController extends Controller
{
    /**
     * Display a listing of the resource.
     */
    public function index()
    {
        //
        $studentData = student::all();

        return response()->json([
            "status" => true,
            "message" => "Students Data",
            "students" => $studentData
        ], 200);
    }

    /**
     * Store a newly created resource in storage.
     */
    public function store(Request $request)
```

```
    {
        //
        $validationRules = [
            "fname" => "required",
            "lname" => "required",
            "city" => "required",
            "email" => "required|email|unique:students,email",
            "phone" => "required|numeric|digits:10",
            "gender" => "required|in:male,female",
            "age" => "required|numeric|max:100|min:0",
        ];

        $validationResult = validator::make($request->all(),
$validationRules);

        if (!$validationResult->fails()) {
            $student = new student();

            $student->fname = $request->fname;
            $student->lname = $request->lname;
            $student->city = $request->city;
            $student->email = $request->email;
            $student->phone = $request->phone;
            $student->gender = $request->gender;
            $student->age = $request->age;

            $student->save();

            return response([
                "message" => "New Student Created"
            ], 200);
        } else {
            return response([
```

```
                "message" => "Validation Failed",
                "errors" => $validationResult->errors()
            ], 202);
        }
    }


    /**
     * Display the specified resource.
     */
    public function show(string $id)
    {
        //
        $studentData = student::find($id);

        if (!$studentData == null) {
            return response([
                "message" => "Student Data Found",
                "student" => $studentData
            ], 200);
        } else {
            return response([
                "message" => "No Students Data with $id ID",
            ], 404);
        }
    }

    /**
     * Update the specified resource in storage.
     */
    public function update(Request $request, string $id)
    {
        //
```

```
        $studentData = student::find($id);
        if (!$studentData == null) {
            $validationRules = [
                "fname" => "required",
                "lname" => "required",
                "city" => "required",
                "email" =>
"required|email|unique:students,email",
                "phone" => "required|numeric|digits:10",
                "gender" => "required|in:male,female",
                "age" => "required|numeric|max:100|min:0",
            ];

            $validationResult = validator::make($request->all(),
$validationRules);

            if (!$validationResult->fails()) {


                $studentData->fname = $request->fname;
                $studentData->lname = $request->lname;
                $studentData->city = $request->city;
                $studentData->email = $request->email;
                $studentData->phone = $request->phone;
                $studentData->gender = $request->gender;
                $studentData->age = $request->age;

                $studentData->save();

                return response([
                    "message" => "Student Updated with $id ID"
                ], 200);
            } else {
```

```
                return response([
                    "message" => "Validation Failed",
                    "errors" => $validationResult->errors()
                ], 202);
            }
        } else {
            return response([
                "message" => "No Students Data found for Update
with $id ID",
            ], 404);
        }
    }

    /**
     * Remove the specified resource from storage.
     */
    public function destroy(string $id)
    {
        //
        $studentData = student::find($id);

        if (!$studentData == null) {
            $studentData->delete();
            return response([
                "message" => "Student Data Deleted With $id ID",
            ], 200);
        } else {
            return response([
                "message" => "No Students Data found for Delete
with $id ID",
            ], 404);
        }
    }
```

```
}
```

Api.php

```
<?php

use App\Http\Controllers\studentsController;
use App\Http\Controllers\usersController;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Route;

/*Route::get('/user', function (Request $request) {
    return $request->user();
})->middleware('auth:sanctum');*/

Route::post("/signup", [usersController::class, "signup"]);
Route::post("/login", [usersController::class, "login"]);
//Route::post("/logout", [usersController::class, "logout"])-
>middleware("auth:sanctum");
//Route::apiResource("/students", studentsController::class)-
>middleware("auth:sanctum");

Route::middleware(['auth:sanctum'])->group(function () {
    Route::post("/logout", [usersController::class, "logout"]);
    Route::apiResource("/students", studentsController::class);
});
```