Studal

Fejlesztői dokumentáció

2021

Minta feladat

# Cél

Egy tanuló nyilvántartás készítése, amely biztosítja egy iskola számára a tanulók nyilvántartását osztályok szerint csoportosítva.

A backend-t Laravel php keretrendszer, az adatbázist MariaDB szerver biztosítja. Frontendnek egy asztali, egy mobil (Android) és egy webes alkalmazás áll rendelkezésre.

A mobil alkalmazásokról böngészhetők az elérhető tanulók. Az asztali alkalmazásból adminisztrálhatók a tanulók. A webes felületen kezelhetők a tanulók és lekérdezhetők osztályok szerint.

Felhasznált technológiák

Backend

Laravel API ( PHP keretrendszer )

* laravel - sanctum ( Autentikációs kiegészítő csomag )

Mariadb server ( Adatbázis kiszolgáló )

Készítéshez használt programok

dia

Visual studio code 1.63.0

Isomnia 2021.5.3

Mariadb server

phpmyadmin

Frontend

Asztali alkalmazás

Mobil alkalmazás

# Kódolási konvenciók

A kódot git verziókezelővel használjuk.

## Alapkönyvtárak

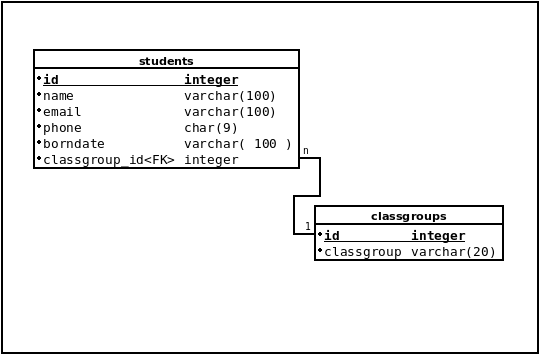
* api
* assets
* database
* desktop
* doc
* mobil
* web

Az api backend, a web a webes frontend.

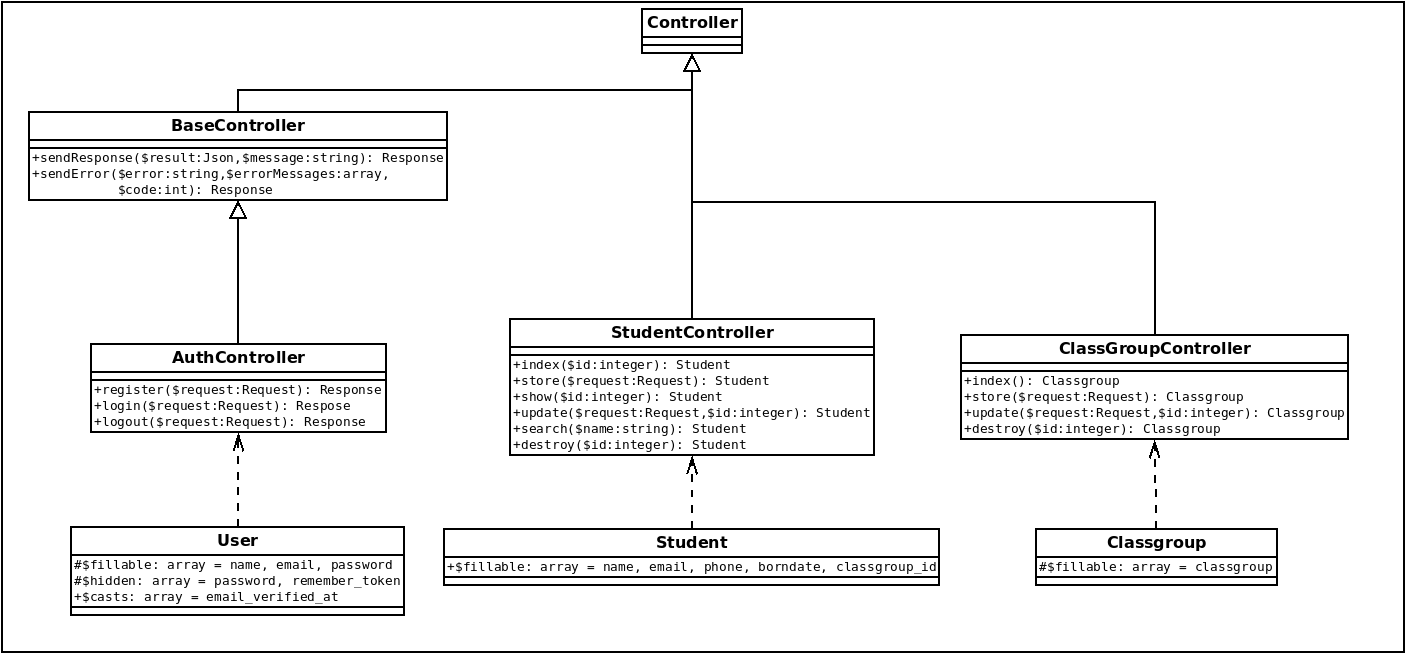
# 

# Felülettervek

## Adatmodell



API UML terv



## Végpontok

|  |  |  |  |
| --- | --- | --- | --- |
| **Végpont** | **Metódus** | **Azonosítás** | **Leírás** |
| */*students/{id} | get | nem | tanulók lekérdezése ( csoportok szerint ) |
| /students | post | igen | tanuló felvétele |
| /students/{id} | get | nem | egy tanuló lekérdezése |
| /students/search/{name} | get | nem | Tanulók keresése név vagy név részlet szerint |
| /students/{id} | put | igen | tanuló módosítása |
| /students/{id} | delete | igen | tanuló törlése |
| /groups | get | nem | csoportok (osztályok) lekérdezése |
| /groups | post | igen | csoport felvétele |
| /groups/{id} | put | igen | csoport módosítása |
| /groups/{id} | delete | igen | csoport törlése |
| /register | post | nem | Felhasználó regisztrációja |
| /login | post | nem | felhasználó bejelentkezése |
| /logout | post | igen | felhasználó kijelentkezése |

Általános működés:

A REST API http kéréseket fogad, melyek tartalmazzák a műveletekhez szükséges megfelelő adatokat. A kényes műveletek végpontjai védettek, autentikáció szükséges a használatukhoz. Ilyenek a kijelentkezés, az új diák felvétele, diák adatainak módosítása, diák törlése, új csoport felvétele, csoport módosítása, csoport törlése. A regisztráció, a bejelentkezés, a diákok és csoportok lekérdezésének végpontjai publikusak, ezek használatához autentikáció nem szükséges.

Az adatokat Json formátumban fogadja és feldolgozza. A vezérlést kontrollerek valósítják meg, minden adatkezelési csoportnak ( autentikáció, diák, csoport ) külön kontrollere van, itt történik az adatfeldolgozás.

A kontrollerek modellekkel vannak kapcsolatban amelyek az adatkezelésért felelősek. Minden adatkezelési csoportnak ( felhasználó, diák, csoport ) külön modellje van, itt történik az adatok adatbázisból kiolvasása, illetve a az adatok kiírása adatbázisba.

A modellek adatbázis táblákkal vannak kapcsolatban, melyek az adatok tárolásáért felelősek. Az adatbázis táblák adatait a modellek kezelik.

## Osztályok

**BaseController**

Feladata a bejelentkezési műveletek válaszainak küldése. Sikeres művelet esetén a generált azonosító tokent visszaküldi, sikertelen művelet esetén hibaüzenetet küld.

Metódusok:

sendResponse()

- bejövő paraméterek: $result ( A generált token ), $message ( A felhasználónak szánt saját üzenet )

- kimenő adatok: Response ( sikeres bejelentkezés esetén a token ).

sendError()

-bejövő paraméterek: $error ( A php által generált hibaüzenet ), $errorMessages ( Saját hibaüzenet ), $code ( A válaszban küldendő http kód ).

-kimenő adatok: Response ( sikertelen bejelentkezés esetén hibaüzenet és a hiba kódja ).

**AuthController**

Feladata egy új felhasználó felvétele, felhasználók autentikációja, felhasználók kijelentkeztetése..

Metódusok:

register()

- bejövő paraméterek: $request ( az regisztrációhoz szükséges adatok a kérésben, name, email, password, confirm\_password )

Az adatokat validálja, majd sikeres érvényesítés után bejegyzi az users adatbázis tábla megfelelő mezőibe.

- kimenő adatok: saját üzenet.

login()

Feladata a felhasználó azonosítása név és jelszó alapján. Sikeres autentikáció esetén generál egy tokent a felhasználó számára és bejegyzi a personal\_access\_tokens adatbázis tábla megfelelő mezőjébe, majd átadja a BaseController sendResponse() metódusának a saját üzenettel együtt.

- bejövő paraméterek: $request ( a bejelentkezéshez szükséges adatok a kérésben, name, password ).

- kimenő adatok: token, saját üzenet.

logout()

Feladata a felhasználó kijelentkeztetése és a token törlése az adatbázis táblából.

- bejövő paraméterek: $request ( a kijelentkezéshez szükségek adatok a kérésben, token )

- kimenő adatok: saját üzenet.

**StudentController**

Feladata egy új diák felvétele, meglévő diák adatainak módosítása, meglévő diák keresése, meglévő diák törlése.

Metódusok:

index()

Visszaadja az adatbázisban szereplő diákok listáját csoportonként.

- bejövő paraméterek: NINCS

- kimenő adatai: Student ( diák modell amely tartalmazza a diákok adatbázisból lekért adatait.

store()

Eltárol az adatbázisban egy új diákot.

- bejövő paraméterek: request ( A diák adatait tartalmazó kérés ).

- kimenő adatok: Student ( A diák adatait tartalmazó modell ).

show()

Visszaadja egy kiválasztott diák adatait.

- bejövő paraméterek: $id ( a kiválasztott diák azonosítója )

- kimenő adatok: Student ( a kiválasztott diák adatait tartalmazó modell ).

update()

Frissíti az adatbázisban tárolt egy diák adatait.

- bejövő paraméterek: $request, $id ( a diák adatait tartalmazó kérés és a diák azonosítója ).

- kimenő adatok: Student ( a diák adatait tartalmazó modell ).

search()

Név vagy név részlet alapján kikeres egy vagy több diákot az adatbázisból.

- bejövő paraméterek: $name ( a diák neve vagy nevének részlete ).

- kimenő adatok: Student ( az egyezést mutató diákok adatait tartalmazó modell ).

destroy()

Feladata a kiválasztott diák törlése az adatbázisból.

- bejövő paraméterek: $id ( a kiválasztott diák azonosítója ).

- kimenő adatok: Student ( a kiválasztott diák adatait tartalmazó modell ).

**ClassGroupController**

Feladata a csoportok kezelése, új csoport felvétele, csoport módosítása, csoport törlése.

Metódusok:

index()

Visszaadja az adatbázisban szereplő csoportok listáját.

- bejövő paraméterek: NINCS

- kimenő adatai: Classgroup ( csoport modell amely tartalmazza a csoportok adatbázisból lekért adatait.

store()

Eltárol az adatbázisban egy új csoportot.

- bejövő paraméterek: request ( A csoport nevét tartalmazó kérés ).

- kimenő adatok: Student ( A csoport nevét tartalmazó modell ).

update()

Frissíti az adatbázisban tárolt egy csoport nevét.

- bejövő paraméterek: $request, $id ( a csoport nevét tartalmazó kérés és a csoport azonosítója ).

- kimenő adatok: ClassGroup ( a csoport nevét tartalmazó modell ).

destroy()

Feladata a kiválasztott csoport törlése az adatbázisból.

- bejövő paraméterek: $id ( a kiválasztott csoport azonosítója ).

- kimenő adatok: ClassGroup ( a kiválasztott csoport nevét tartalmazó modell ).

**User**

A regisztrált felhasználók adatait illetve az azonosító tokent amelynek a generálása a felhasználóhoz kapcsolódik írja az adatbázis megfelelő táblájába, valamint onnan szolgáltatja szükség szerint. A User modell a users adatbázis táblával van kapcsolatban az adatokat onnan olvassa és oda írja. Az osztályt a Laravel projekt generálja, használata a Sanctum autentikációs kiegészítő csomagon keresztül történik.

**Student**

Az adatbázisban felvett tanulók adatait kezeli, a StudentController megfelelő metódusai által kért adatot adja vissza az adatbázis megfelelő táblájából , illetve a kontrollerből érkező adatokat írja ki. Az adatbázis students táblájával van kapcsolatban, az adatokat onnan olvassa és oda írja.

Mezők:

$fillable ( array ) elemei: name, email, phone, borndate, classgrop\_id

A fillable mező elemei egy – egy mező a students adatbázis táblában. Biztonsági okokból csak a felsorolt mezőket írhatja a modell az adatbázisban.

**Classgroup**

Az adatbázisban felvett csoportok adatait kezeli, a ClassGroupController megfelelő metódusai által kért adatot adja vissza az adatbázis megfelelő táblájából , illetve a kontrollerből érkező adatokat írja ki. Az adatbázis classgroups táblájával van kapcsolatban, az adatokat onnan olvassa és oda írja.

Mezők:

$fillable ( array ) elemei: classgroup

A fillable mező eleme egy mező a classgroups adatbázis táblában. Biztonsági okokból csak a felsorolt mezőket írhatja a modell az adatbázisban.

## Mobil alkalmazás felülete

# Fejlesztői környezet

* XAMPP
  + Adatbázis kezelése a mysql nevű klienssel.
  + PHP
* Visual Studio Code
* React Native Expo
* Laravel
* Angular
* Java FX

# Szerkezetek

* employeeList
  + dolgozók listája

# Algoritmusok

???

# Tesztelés

# Ismert hibák

Fejlesztési lehetőség