

FiveFile

Vizsgaremek dokumentáció

Csapat: **Pap András, Farkas Péter, Pardi Polett**

Gyakorlatvezető tanár neve: **Várkonyi Tibor**

Feladat megnevezése: **Vizsgaremek**

**Tartalom**

Szerző 1

Tartalom 2

Felhasználói dokumentáció 3

Feladat 3

Környezet 3

Használata 3

Alkalmazások 3

Bejelentkezés / Regisztráció kezdőlap 3

Menüpontok 3

Főoldal 3

Saját fájlok 3

Velem megosztva 3

Felhasználók 3

Beállítások 4

Kijelentkezés 4

Fejlesztői dokumentáció 5

**Felhasználói dokumentáció**

# Feladat

Egy online fájl rendszerező szoftvert készítünk, célja, hogy a felhasználó egy helyen el tudja érni a számára fontos fájlokat. Valamint akik regisztráltak már nálunk egy helyen tudják kezelni, átnevezni, kedvencnek jelölni, akár megosztani vagy kitörölni az adott fájljait és bárhol bármikor elérhetik a fájlaikat.

# Környezet

Az oldal bármilyen böngészőben futtatható.

# Használata

## Alkalmazások

Alkalmazások: Docker Desktop, Visual Studio Code, Php Storm, GitHub Desktop

Letöltjük a GitHub-ról a projektünket, megnyitjuk a Visual Studio Code-ban. A menüpontokból kiválasztjuk a Terminalt majd New Terminalt. Eközben elindítjuk a docker programot. Ha ez megtörtént, visszalépünk a backend mappába: nyitunk egy cmd-t, majd beírjuk, hogy install.bat. Ahhoz, hogy sikerüljön kell a projekt mappánkban lennie egy conf egy www mappának, egy .env, egy docker-compose.yml valamint egy Dockerfile-nak. Ha lefutott, visszalépünk a visual studio code-ra és beírjuk a következőket: npm install, npm run dev.

## Bejelentkezés / Regisztráció kezdőlap

Ha még nem regisztráltunk az oldalra, akkor egyértelműen ezt kell elsőként megtennünk. Meg kell adnunk egy felhasználónevet, emailcímet, jelszót majd egy jelszó ismétlést. Ha az email címet például @ nélkül töltjük ki, akkor kapni fogunk egy figyelmeztetést. A regisztrálok gomb megnyomása után, átirányít minket a bejelentkezés felülethez.

A bejelentkezésnél írjuk be a már korábban regisztrált felhasználónevet, jelszót. Ha elfelejtettük a jelszót, tudunk kérni az Elfelejtett jelszó gombra kattintva egy új jelszót. Be kell írnunk az email címünket, majd kapunk egy linket, amin keresztül tudunk új jelszót rögzíteni.

## Menüpontok

### Főoldal

Az oldal bal szélén egy menüsáv található, ahol az alábbi menüpontok vannak: Főoldal, Saját fájlok, Velem megosztva, Felhasználók, Beállítások, Kijelentkezés.

A felhasználó, az oldalon láthatja, a legutóbbi fájljait, mennyi helye maradt még, valamint egy fájl feltöltés részt.

### Saját fájlok

Az oldalon a felhasználó saját fájljait listázzuk ki, ahol letudja tölteni, megosztani valamint törölni őket.

### Velem megosztva

Ha tagja vagy már egy csoportnak akkor itt láthatod a veled megosztott fájlokat.

### Felhasználók

Az oldal tetején találunk egy keresés mezőt, itt rákereshetünk másik felhasználókra, majd be tudjuk őket jelölni. Tovább 4 kis dobozt látunk az oldalon: Barátaid, függőben lévő jelöléseid, csoportjaid, bejelöltek barátnak. A csoportjaid dobozon belül tudsz létrehozni új csoportot.

### Beállítások

A gombra kattintva, megjelenik az adatok beállítások menüpontja. Itt a felhasználó tudja a profilképét de az emailcímét is módosítani, ezt a jelszó beírásával tudja elmenteni.

A biztonság gombra kattintva a jelszavát tudja módosítani, valamint a fiókját tudja véglegesen törölni.

### Kijelentkezés

A gombra kattintva a felhasználó kijelentkezik a weboldalról.

**Fejlesztői dokumentáció**

# Megvalósítás

## Backend

A projekt backendjéül egy Laravel alkalmazás szolgál.

## Frontend

A projekt frontendje VUE JS keretrendszerrel lett megvalósítva.

## Android

A mobil alkalmazás a Google által fejlesztett Flutterrel lett elkészítve

# Forráskód

A projekthez tartozó teljes fejlesztői anyag megtalálható az alábbi linken:

* <https://github.com/papandras/NJIT-2021-2022-Vizsgaremek>

A projektet letöltve a további almappákban érhetőek el a projekt részegységei.

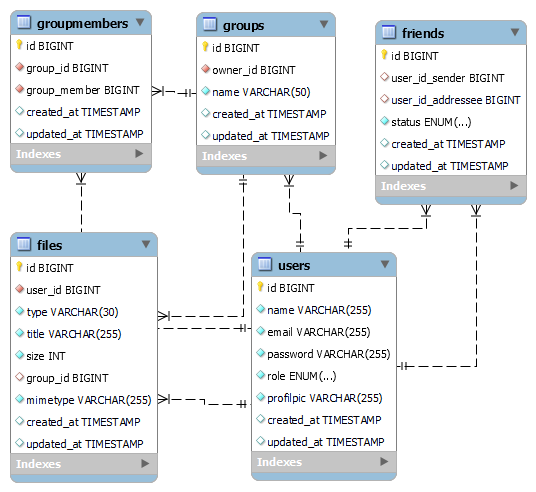
Futtatáshoz szükséges információk:

* + Backend: install.bat / install.sh / install.ps1 fájl futtatára.
    - Ezt követően az alkalmazás elérhető a localhost:8881-en, az adatbázis pedig a localhost:8882-n
  + Frontend:
    - a következő parancsok kiadása szükséges:
      * npm install
      * npm run dev / npm run build
    - Ezt követően az alkalmazás elérhető a CLI által jelzett címen (localhost:3000)
  + Android: android studio ide-ből emulátorral futtatható
* A parancsok futtatásakor az adott projekt főmappája tekintendő root könyvtárnak!

**Backend**

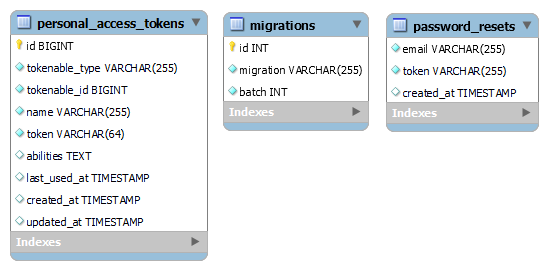
# Adatbázis modell

A fájlok láthatóságának esetében lényeges feladat volt az adatbázis megtervezése. A modell elkészítését követően, a modell alapján írt migrációk segítségével le tudtam kérdezni a megfelelő rekordokat. Ezt követően a felhasználó azokat a fájlokat kapta vissza, amelyeket kizárólag Ő töltött fel, illetve egy másik felhasználó által meg lettek osztva vele. A tervezés alapja a felhasználó adatainak tárolása, amik az *users* táblában szerepelnek. A *users* táblához 3 tábla kapcsolódik közvetlenül, és 1 közvetetten. Minden felhasználóhoz tartoznak a saját fájljai, a barátai, és az általa létrehozott csoportjai. Ezek az adatok rendre a *files*, *friends* és *groups* táblákban találhatóak. A csoportokhoz természetesen lehet tagokat is hozzárendelni. Ezek a tagok a *groupmembers* táblában vannak eltárolva. A csoport rendszer közvetlen összefüggésben áll a barát rendszerrel, mivel a csoport tagjai csak olyan felhasználók lehetnek, akik az adott felhasználó barát listáján szerepelnek. A csoport rendszer pedig a fájlok megosztását létesíti meg. Fájlokat nem közvetlenül lehet a többi felhasználóval megosztani, hanem egy csoporton keresztül.



Ábra: Adatbázis EER modell 1

Az adatbázishoz tartoznak a fent említetteken kívül további táblák is. A *migrations* táblára a Laravel keretrendszernek van szüksége. Ebben vannak a lefuttatott migrációk tárolva. A *personal\_access\_tokens* táblában a felhasználók bejelentkeztetéséért felelős *token*-ek kapnak helyet. A felhasználóknak továbbá lehetőségük van a jelszó visszaállítására, amennyiben nem tudnak bejelentkezni az oldalra. Ebben az esetben az email címükre küldött linken keresztül képesek új jelszót beállítani.



Ábra: Adatbázis EER modell 2

Controller-ek

Az alábbiakban ismertetésre kerülnek az alkalmazás által használt kontrollerek, illetve az ezek elérésére használt útvonalak.

Az alkalmazás az alábbi táblázatokban található útvonalak elérését biztosítja. A Method oszlopban az útvonal metódusa, az URI oszlopban az útvonal, Action oszlopban az útvonalhoz tartozó @ jellel ellátott controller metódus, a Name oszlopban az útvonal neve, a Middleware oszlopban pedig az útvonalhoz tartozó szűrő található. Minden metódus a fejlécben jelzett controller osztályhoz tartozik.

A táblázatok alatt található listában az adott táblázatban szereplő metódusok rendeltetésére lehet kapható magyarázat.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| App\Http\Controllers\AuthController | | | | |
| **Method** | **URI** | **Action** | **Name** | **Middleware** |
| POST | api/register | @register | register | - |
| POST | api/login | @login | login | - |
| POST | api/ forgot-password | @forgotpassword | password.request | - |
| GET | api/user | @user | user | auth:sanctum |
| GET | api/getNewCookie | @getNewCookie | getNewCookie | auth:sanctum |
| POST | api/logout | @logout | logout | auth:sanctum |
| GET | resetpassword/{token} | @resetpassword | password.reset | guest |
| POST | update-password | @updatepassword | password.update | guest |

* @register: Ez a metódus a felhasználói fiók elkészítésére szolgál.
* @login: Ez a metódus a felhasználónév – jelszó párossal történő bejelentkezést biztosítja.
* @logout: Kijelentkezteti a felhasználót.
* @forgotpassword: Sikeres email cím megadása esetén a megadott email címre küld üzenetet, melyen keresztül elérhető számára a jelszó visszaállítására szolgáló felület.
* @resetpassword: A kiküldött emailen talál található linket biztosítja. Erre kattintva érhető el a jelszó visszaállítására szolgáló felület.
* @updatepassword: A megadott új jelszót tárolja el az adatbázisban.
* @user: A bejelentkezett felhasználó adatait adja vissza.
* @getNewCookie: A bejelentkezett felhasználó számára új sütit biztosít. A felhasználói felület újra töltésekor lép életbe. A sütik élettartama 30 perc, és ennek meghosszabbítását szolgálja.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| App\Http\Controllers\UserController | | | | |
| **Method** | **URI** | **Action** | **Name** | **Middleware** |
| GET | api/users | @getAllUsers | getAllUsers | auth:sanctum |
| GET | api/users/{username} | @getUser | getUser | auth:sanctum |
| GET | api/users/letter/{startletter} | @getUserByStartLetter | getUserByStartLetter | auth:sanctum |
| GET | api/user/{id}/groups | @getgroups | user.groups | auth:sanctum |
| GET | api/user/groups | @getownedgroups | user.owngroups | auth:sanctum |
| GET | api/group/{id}/files | @getFilesInGroup | group.files | auth:sanctum |

* @getAllUsers: Visszatér az összes regisztrált felhasználó adataival a UserResource osztályban meghatározott szabályok szerint.
* @getUser: Visszatér a *username* paraméterben megkapott felhasználó adataival a UserResource osztályban meghatározott szabályok szerint.
* @getUserByStartLetter: Visszatér az összes olyan felhasználóval, akiknek felhasználóneve a *startletter* paraméterben megkapott betűvel kezdődik.
* @getgroups: Visszatér az összes olyan csoport adataival, amelynek tagjaihoz hozzá van rendelve az adott felhasználó.
* @getownedgroups: Visszatér a felhasználó összes saját csoportjával.
* @getFilesInGroup: Visszatér a paraméterben megkapott csoportban található összes fájllal a GroupFileResource osztályban meghatározott szabályok szerint.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| App\Http\Controllers\FriendController | | | | |
| **Method** | **URI** | **Action** | **Name** | **Middleware** |
| GET | api/user/friends | @getFriends | friends | auth:sanctum |
| POST | api/user/friends/add | @addFriend | addfriend | auth:sanctum |
| POST | api/user/friends/remove | @removeFriend | removeFriend | auth:sanctum |
| GET | api/user/friends/friendrequests | @friendrequests | friendrequests | auth:sanctum |
| POST | api/user/friends/accept | @acceptrequest | acceptrequest | auth:sanctum |
| GET | api/user/friends/sentrequests | @sentrequests | sentrequests | auth:sanctum |
| POST | api/user/friends/cancelrequest | @cancelrequest | cancelrequest | auth:sanctum |
| POST | api/user/friends/rejectrequest | @rejectrequest | rejectrequest | auth:sanctum |

* @getFriends: Visszatér a felhasználó összes megerősített barátjával.
* @addFriend: Barát jelölést küld az adott felhasználónak. Requestben kötelező elküldeni a bejelölendő barát nevét!
* @removeFriend: Megszűnteti a barát kapcsolatot az adott felhasználóval. Requestben kötelező elküldeni a törlentő barát azonosítóját!
* @friendrequests: Visszatér az adott felhasználó összes olyan bejövő barát kérésével, amely még nem lett elfogadva.
* @acceptrequest: Az adott barát kérés megerősítésére szolgál.
* @sentrequests: Visszatér az adott felhasználó összes olyan elküldött barát kérésével, amely még nem lett elfogadva.
* @cancelrequest: Adott elküldött és még nem visszaigazolt barát kérés visszavonása. Requestben kötelező elküldeni a visszavonandó kérés azonosítóját!
* @rejectrequest: Bejövő barát kérés elutasítása. Requestben kötelező elküldeni az elutasítandó kérés azonosítóját!

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| App\Http\Controllers\GroupController | | | | |
| **Method** | **URI** | **Action** | **Name** | **Middleware** |
| POST | api/group/create | @create | group.create | auth:sanctum |
| POST | api/group/addmember | @addmember | group.addmember | auth:sanctum |
| GET | api/group/{id}/members | @getmembers | group.members | auth:sanctum |
| DELETE | api/group/{id}/delete | @deletegroup | group.delete | auth:sanctum |
| DELETE | api/group/{group}/member/{user}/delete | @deletememberfromgroup | deletememberfromgroup | auth:sanctum |
| DELETE | api/group/{id}/leave | @leavegroup | group.leave | auth:sanctum |

* @create: Csoport létrehozása. A requestben kötelező szerepelnie a *name* paraméternek!
* @addmember: Barát hozzáadása adott csoporthoz. A requestben kötelező elküldeni a barát és a csoport azonosítóját!
* @getmembers: Az *id* paraméterben átadott csoport tagjait adja vissza.
* @deletegroup: Az *id* paraméterben átadott csoportot törli az adatbázisból.
* @deletememberfromgroup: A *group* paraméterben megadott csoportból törli a *user* paraméterben megadott felhasználót.
* @leavegroup: Az id paraméterben megadott csoportból törli a requestet küldő felhasználót.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| App\Http\Controllers\FileController | | | | |
| **Method** | **URI** | **Action** | **Name** | **Middleware** |
| POST | api/file/upload | @upload | file.upload | auth:sanctum |
| GET | api/file/get/{limit?} | @getFiles | file.get | auth:sanctum |
| GET | api/file/download/{filename} | @download | file.download | auth:sanctum |
| DELETE | api/file/delete/{filename} | @delete | file.delete | auth:sanctum |
| PUT | api/file/{id} | @sharewithgroup | file.share | auth:sanctum |
| PUT | api/file/{id}/unshare | @unshare | file.unshare | auth:sanctum |
| GET | api/file/stat | @getstat | file.stat | auth:sanctum |

* @upload: A requestben érkező fájlokat eltárolja a projekt *storage/app/storage* mappájában a következő néven: *felhasználónév-fájlnév.kiterjesztés*. Ugyan ezen a néven eltárolja az adatbázis *files* táblájába az Adatbázis modellben látható adatokkal együtt.
* @getFiles: Visszatér a felhasználó fájljaival. A *limit* paraméter opcionális. Ha nincs megadva akkor az összes fájlt visszaadja, ha meg van adva akkor annyi darabszámú lesz csak visszaadva.
* @download: A *filename* paraméterben megkapott fájl letöltését teszi elérhetővé.
* @delete: A *filename* paraméterben megkapott fájl törlését teszi lehetővé.
* @sharewithgroup: Az *id* paraméterben átadott fájlt osztja meg a requestben megkapott *group\_id* id-jű csoporttal.
* @unshare: Az *id* paraméterű fájl megosztását vonja vissza.
* @getstat: Az adott felhasználó tárolt fájljairól készít statisztikát fájl típusonként és összességében.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| App\Http\Controllers\AdminController | | | | |
| **Method** | **URI** | **Action** | **Name** | **Middleware** |
| GET | api/admin/users/{user?} | @getUsers | admin.finduser | auth:sanctum |
| PUT | api/admin/users/setrole/{user}/{role} | @setrole | admin.setrole | auth:sanctum |
| DELETE | api/admin/users/delete/{user} | @deleteuser | admin.deleteuser | auth:sanctum |
| POST | api/admin/files | @getFiles | admin.getfiles | auth:sanctum |

* @getUsers: Adott felhasználó keresésére szolgál. A *user* paraméter opcionális. Ha nincs kitöltve az összes felhasználót visszaadja, ha ki van töltve akkor csak a paraméterbe illő nevű felhasználókat adja vissza.
* @setrole: A *user* paraméterű felhasználó szerepkörét állítja a *role* paraméterben megkapott szerepkörre. Két fajta szerepkör van: admin, user.
* @deleteuser: A *user* paraméterben megkapott felhasználót törli.
* @getFiles: A neki küldött requestnek 2 paraméter adható át: *user* és *file*. Egyik paraméter sem kötelező. Ha egyik sincs kitöltve akkor minden fájlt visszaad. Ha csak a *user* van kitöltve akkor az adott felhasználó összes fájlját visszaadja. Ha a *file* is ki van töltve akkor az adott felhasználó olyan fájljait adja vissza, amelyek illenek a *file* paraméterű kulcsszóra. (szabad szavas keresés). Ha csak a *file* van kitöltve, akkor felhasználótól függetlenül adja vissza az illeszkedő nevű fájlokat.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| App\Http\Controllers\SettingsController | | | | |
| **Method** | **URI** | **Action** | **Name** | **Middleware** |
| POST | api/user/settings | @settings | user.settings | auth:sanctum |
| DELETE | api/user/delete | @removeaccount | user.delete | auth:sanctum |

* @settings: A következő adatok megváltoztatását teszi lehetővé: (a hozzá szükséges request paraméter)
  + - Email cím (*email*)
    - Jelszó (*password*)
    - Profil kép (*profilpic* / *removepic*)
  + A requestben minden esetben kötelező elküldeni a *confirm* paramétert, amely a jelenlegi jelszó megerősítésére szolgál.
* @removeaccount: A kérést küldő felhasználó profilját törli, majd kijelentkezteti az oldalról.

Request-ek

A Laravel Request osztálya segítségével hozzá tudunk férni a bejövő kérések adataihoz, többek között a sütikhez, és egyéb fájlokhoz. Továbbá lehetőségünk van egyéni Request osztályok létrehozására, melyekben le tudjuk kezelni, hogy az aktuális felhasználó végre tudja-e hajtani az adott kérést, és ha igen, a bejövő adatokra tudunk szabályokat kiszabni. Továbbá lehetőségünk van az alapértelmezett angol üzeneteket felülírni. Ezt az osztályok messages metódusában tudjuk majd megtenni, ahogy majd látható a RegisterRequest és a PasswordResetRequest osztályokban.

A FiveFile alkalmazásban 3 egyéni request osztály található.

1. LoginRequest

Alkalmazott szabályok:

* name: kötelező kitölteni, minimun 3 és maximum 20 karakter hozzú.
* password: kötelező kitölteni, minimun 8 és maximum 20 karakter hozzú.



Ábra: LoginRequest

1. RegisterRequest

Alkalmazott szabályok:

* name: kötelező kitölteni, minimun 3 és maximum 20 karakter hozzú.
* email: kötelező kitölteni, nem szerepelhet már a *users* tábla *email* oszlopában. Hossza 10 és 50 karakter között lehet.
* password: kötelező kitölteni, minimun 8 és maximum 20 karakter hozzú.



Ábra: RegisterRequest

1. PasswordResetRequest

Alkalmazott szabályok:

* name: kötelező kitölteni, minimun 3 és maximum 20 karakter hozzú.
* password: kötelező kitölteni, minimun 8 és maximum 20 karakter hozzú, meg van erősítve.
* password\_confirmation: kötelező kitölteni



Ábra: PasswordResetRequest

Resources

A Laravel Resource osztályaival könnyedén tudjuk az adatbázis lekérdezések eredményeit átalakítani JSON formátumba, továbbá meg tudjuk adni, hogy milyen formában kapjuk vissza az adatokat. Ez akkor előnyös, ha különböző felhasználói csoportoknak nem akarunk minden egyes adatot megjeleníteni.

Resource osztályt a php artisan segítségével tudunk generálni a következő parancs kiadásával:

php artisan make:resource NameResource

A generált osztályok az app/Http/Resources mappába fognak létrejönni.

A FiveFile alkalmazásban 6 resource osztály található.

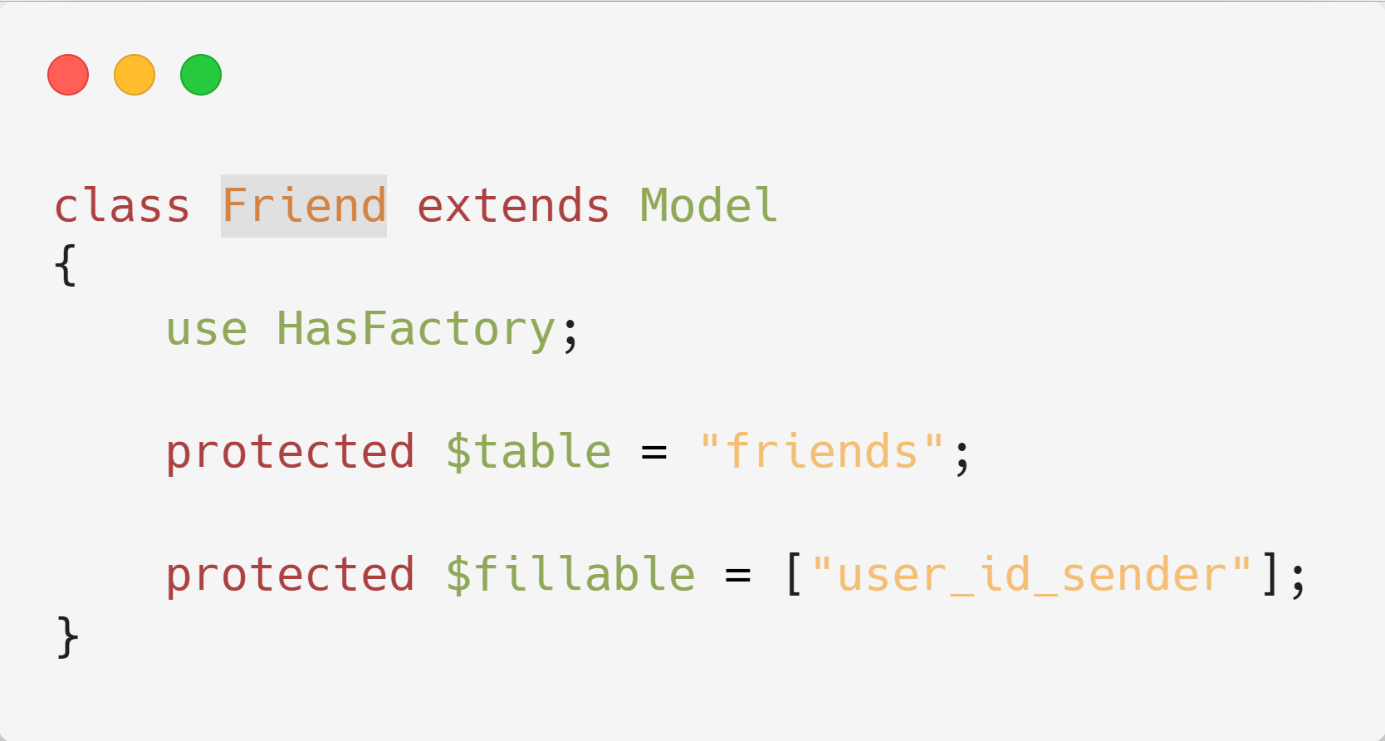
1. AdminFileResource
2. AdminUserResource
3. FileResource
4. FriendResource
5. GroupFileResource
6. UserResource

Modelek

A Laravel projektben a model osztályok az app/Models mappában helyezkednek el. Az eddigiekhez hasonlóan a modeleket is a php artisan-nal lehet generálni:

php artisan make:model NameModel

Az alkalmazásban 5 model találtó.

1. User
2. File
3. Friend
4. Group



1. GroupMember

**Frontend**

A FiveFile alkalmazás frontend része Vue JS keretrendszerrel lett megvalósítva. Maga az alkalmazás egy SPA (Single Page Application), ami annyit tesz, hogy minden adat az index.html-re renderelődik ki, és az router segítségével lehet az oldalakon mozogni. Az oldal forráskódja az src mappában található. Az src-n belül 5 almappába vannak csoportosítva a fájlok. Az assets mappában van az összes megjelenő ikon - a kategóriák ikonjai és fájl kiterjesztés ikonjai. Továbbá adott komponensek js és css fájljai az ezekre kijelölt további almappákban. Következő mappa a components. Ebben a mappában találhatóak az oldalon használt komponensek. Működésük a későbbiekben kerül ismertetésre. A router mappában található index.js felel az alkalmazásban található útvonalakért. Itt vannak meghatározva, hogy melyik útvonal melyik view fájlt töltse be, és vizsgálja, hogy a bejelentkezett felhasználónak van-e jogosultsága megtekinteni azt. Amennyiben nincsen bejelentkezett felhasználó, akkor nem tudja megtekinteni az oldalakat, mivel nincsen vendég mód, a használat regisztrációhoz kötött. A router gondoskodik még arról is, hogy ha a felhasználó nem létező útvonalat próbál elérni, hibaüzenetet kap. A router mappát követően a store mappában lévő auth.js felel az authentikáció lebonyolításáért. Megvizsgálja, hogy az oldalon elérhető a backend által küldött süti. Amennyiben elérhető akkor a felhasználó be van jelentkezve, ha pedig nem, akkor a router átirányítja a bejelentkező oldalra. Ha pedig be van jelentkezve, akkor a bejelentkező oldalról irányít át az index oldalra. Maga a store a pinia használatával került megvalósításra. Ennek hivatkozása megtalálható a dokumentáció végén. A következő views mappában az oldal nézet fájljai találhatóak. Mind a 11 útvonalhoz 1-1 nézet tartozik. Az első nézet amivel a felhasználó találkozik, az az LoginView.vue fájlban találkozik. Itt van a bejelentkező felület, illetve a regisztrációs oldalra is innen lehet tovább lépni. A továbbiakban részletezve lesz, hogy milyen útvonalak találhatóak az alkalmazásban, és a komponensek, amelyekből felépül.

Path: útvonal, Title: a menüpont neve, View: a betöltődő nézet fájl neve, RequiresAuth: szükséges-e hozzá bejelentkezés

Útvonalak

|  |  |  |  |
| --- | --- | --- | --- |
| **Path** | **Title** | **View** | **RequiresAuth** |
| / | Bejelentkezés | LoginView.vue | false |
| /register | Regisztráció | RegisterView.vue | false |
| /index | Főoldal | IndexView.vue | true |
| /myfiles | Saját fájlok | MyFilesView.vue | true |
| /shared | Velem megosztott fájlok | SharedView.vue | true |
| /users | Felhasználók | UsersView.vue | true |
| /admin | Admin | AdminView.vue | true + admin role |
| /settings | Beállítások | SettingView.vue | true |
| /unauthorized | Nem jelenzkeztél be | UnAuthorized.vue | false |
| /logout | Kijelentkezés | LogoutPage.vue | true |
| /:pathMatch(.\*)\* | 404 | NotFoundPage.vue | true |

Nézet fájlok

# LoginView

# 

Használt komponensek: Login (Login.vue)

# RegisterView



Használt komponensek: Register (Register.vue)

# IndexView



Használt komponensek: IndexPage (Index.vue)

# MyFilesView

Használt komponensek

Menu (LeftSideMenu.vue)

* Ez a komponens alkotja magát az oldal menüjét.
* Az oldalak fő szerkezete CSS grid-del van megvalósítva. Egy sor 1:9 arányban van mindig felosztva. A bal oldali oszlopban a menüsor, a jobb oldaliban pedig az oldal további tartalma.

FileTable (LastFilesTable.vue)

* Ebben a táblázatban vannak megjelenítve a felhasználó fájljai.

Állapotok

data

files

* Kezdőértéke: null
* Ide fognak betöltődni a felhasználó fájljainak az adatai, majd át lesz adva a FileTable komponensnek, amely megjeleníti őket.

methods

getfiles

* Lekéri a bejelentkezett felhasználó fájljait a backend api/file/get útvonaláról, majd átadja őket a files állapotnak.

# SharedView



Ez az oldal jeleníti meg azokat a csoportokat, amelyekhez az adott felhasználó hozzá lett adva. Amennyiben egy ilyen csoport sincs, kiírja, hogy „Nem vagy tagja egy csoportnak sem!”. Ha vannak csoportok, akkor a v-for direktíva segítségével kilistázza őket, és a benne lévő fájlokat. Ha nincsen fájl a csoportban akkor is kap a felhasználó üzenetet. Továbbá lehetőség van kilépni az adott csoportból.

Használt komponensek

Menu (LeftSideMenu.vue)

TableRow (GroupFilesRow.vue)

* Átadásra kerül az adott csoport id-je, ami alapján majd kilistázza a fájlokat. Átadunk még egy függvényit is nofilemessage nélkül, amit majd visszaküld ha nincsen fájl a csoportban, és kiírjuk hogy „A csoportban nem található fájl!”

Állapotok

data

store

* A bejelentkezett felhasználó lekéréséhez szükséges

groups

* Ebben lesznek a felhasználó csoportjainak adatai

nofilemessage

* Ez felel a „A csoportban nem található fájl” üzenet megjelenítéséért.

methods

loadgroups

* Ez a metódus tölti be a csoport adatait, majd adja át a *groups*-nak.

leavegroup

* Ez a metódus fut le, ha a felhasználó kilép a csoportból
* Paramétere az adott csoport id-je, melyet elküld a backend api/group/${id}/leave útvonalára.

# UsersView

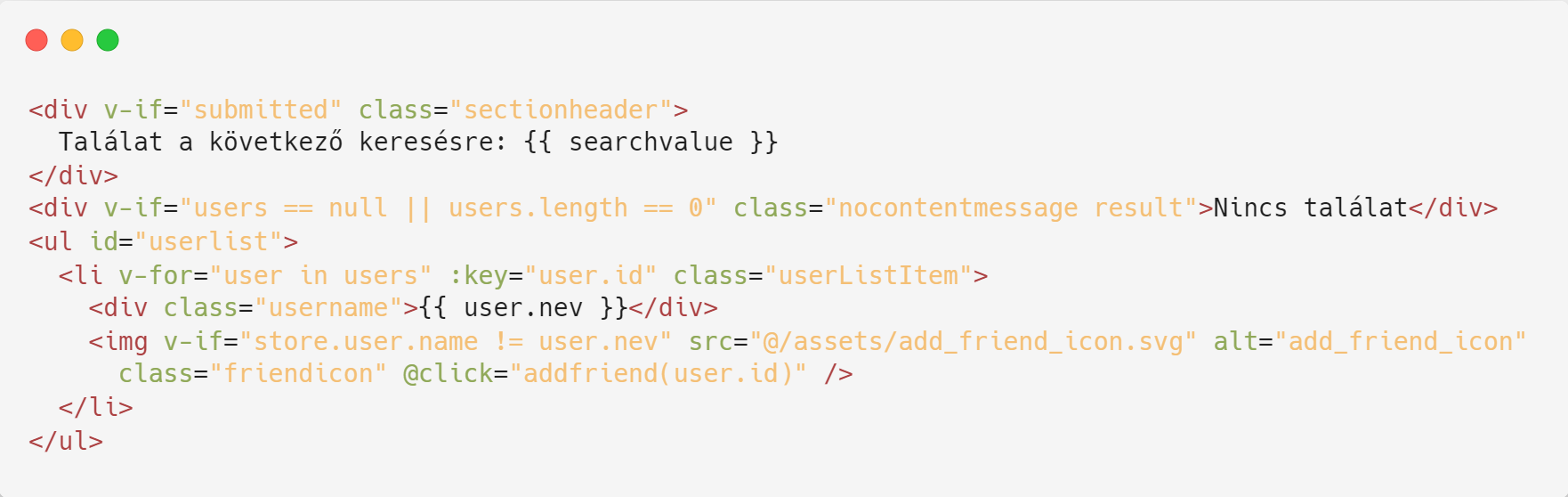
Ez a nézet 6 fő részből áll: Felhasználó kereső, Keresés eredménye, Barátok, Csoportok, Elküldött barát jelölések, Bejövő barát jelölések.

Ezen az oldalon lehet lebonyolítani a többi felhasználóval végzett lehetséges interakciókat és a csoportok létrehozását.

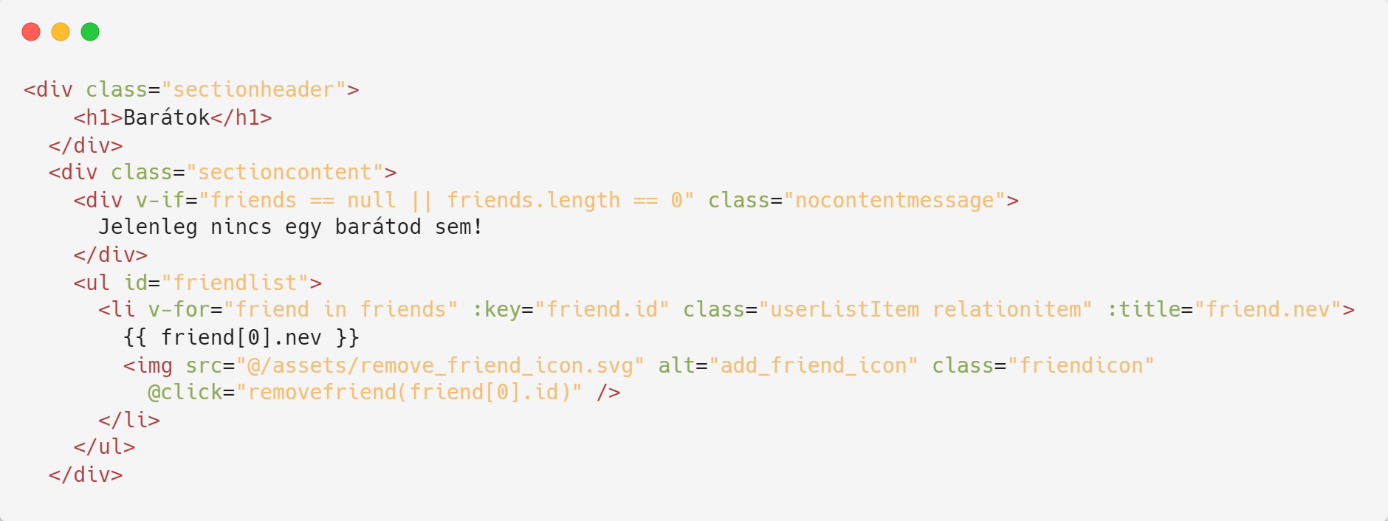
# Felhasználó kereső



# Keresés eredménye



# Barátok



# Csoportok



# Elküldött barát jelölések



# Bejövő barát jelölések.



Használt komponensek

Menu (LeftSideMenu.vue)

Groups (Groups.vue)

* Ez a komponens jeleníti meg csoportok menüt. Ebben lehet létrehozni, törölni és feltölteni csoportokat.

Állapotok

data

users

* A felhasználó keresés eredményét tárolja.

friends

* A felhasználó barátait tárolja.

friendrequests

* A felhasználó bejövő barát kéréseit tárolja.

searchvalue

* A felhasználó kereső kulcsszavát tárolja.

sentrequests

* A felhasználó elküldött barát jelöléseit tárolja.

submitted

* Tárolja, hogy a keresés el lett-e indítva. Ez alapján jelenik meg a keresés eredménye mező.

store

* A bejelentkezett felhasználó lekéréséhez szükséges.

letters

* Az ABC betűi vannak benne. A kezdőbetűs kereséshez szükséges.

methods

getUsers

* A backend api/users útvonaláról betölti az összes felhasználót.

getFriends

* A backend api/user/friends útvonaláról betölti az aktuális felhasználó barátait.

getFriendRequests

* A backend api/user/friends/friendrequests útvonaláról betölti az aktuális felhasználó elküldött barát kéréseit.

findUsers

* A backend api/users/{username} útvonaláról betölti a felhasználó keresés eredményét.
* Paramétere a *searchvalue*

addfriend

* Elküldi a barát kérést a backend api/user/friends/add útvonalára.
* Paraméter a bejelölendő felhasználó id-je.

removefriend

* Elküldi a barát eltávolítása kérést a backend api/user/friends/remove útvonalára.
* Paraméter a törlendő felhasználó id-je.

acceptfriend

* Elküldi a barát elfogadása kérést a backend api/user/friends/accept útvonalára.
* Paraméter az elfogadandó jelölés id-je.

getSentrequests

* A backend api/user/friends/sentrequests útvonaláról betölti az adott felhasználó elküldött barát jelöléseit.

cancelrequest

* A backend api/user/friends/cancelrequest útvonalára elküldi a barátjelölés visszavonása kérést.
* Paraméter az visszavonása jelölés id-je.

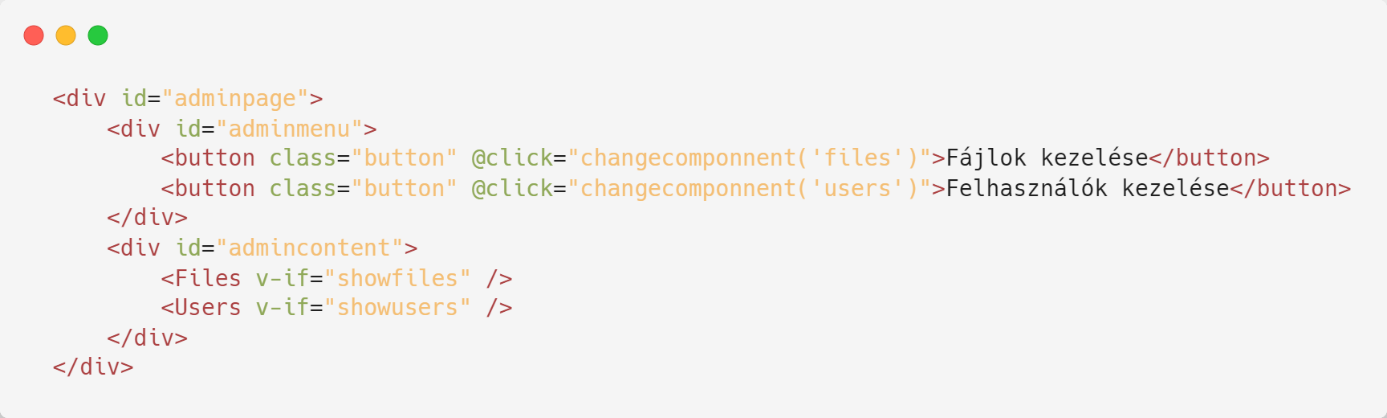
reject

* A backend api/user/friends/rejectrequest útvonalára elküldi a barátjelölés elutasítása kérést.
* Paraméter az elutasítandó jelölés id-je.

searchbyletter

* Felhasználó keresés kezdőbetű alapján.
* A backend api/users/letter/${ startletter} útfonalára küldi a kérést.

# AdminView



Ez a menüpont csak admin szerepkörű felhasználóknak jelenik- és tekinthető meg. Ha a szerepkör nem egyezik akkor a felhasználó visszairányításra kerül.

Használt komponensek

Menu (LeftSideMenu.vue)

Files (admin/Files.vue)

Users (admin/Users.vue)

Állapotok

data

showfiles

* Ez alapján jelenik meg a Files komponens.

showusers

* Ez alapján jelenik meg a Users komponens.

methods

changecomponnent

* Paramétere a megjelentetni kívánt komponens.
* A paraméter alapján beállítja a *showfiles* és a *showusers* értékeit.

# SettingView

Használt komponensek

Menu (LeftSideMenu.vue)

Datas (settings/setDatas.vue)

A felhasználó adatainak beállításai – email cím, profil kép

Secutiry (settings/setSecurity.vue)

Biztonsági beállítások – jelszó, fiók törlése

Állapotok

data

showdata

* Ez alapján jelenik meg a Datas komponens.

showsecurity

* Ez alapján jelenik meg a Security komponens.

settings

* FormData típusú objektum. Ehhez lesznek hozzáadva a módosítandó beállítások.

errors

* A mentés gomb megnyomásakor, ha jön a backendről hibaüzenet, ebben lesz eltárolva.

confirm

* A jelszó megerősítése értékét tárolja.

store

* A bejelentkezett felhasználó azonosításához szükséges.

methods

displayDatas, displaySecurity

* Beállítják a showdata és a showsecurity változókat aszerint, melyik komponensnek kell megjelennie.

shavechanges

* A változtatott beállításokat elküldi a backend api/user/settings útvonalára.

# UnAuthorized



Az oldal akkor jelenik meg, ha a felhasználó bejelentkezés nélkül akarja megtekinteni a weboldalt.

Állapotok

data

counter

* A visszaszámláló aktuális értékét tárolja

mounted

* 5-től kezdve számol visszafelé másodpercenkénet, és amit eléri a nullát, átdob a bejelentkezés oldalra.

# LogoutPage



Az oldal akkor jelenik meg, ha a felhasználó a kijelentkezés gombra kattint, vagy a bejelentkezve töltött ideje elérte a 30 percet aktivitás nélkül.

Állapotok

data

counter

* A visszaszámláló aktuális értékét tárolja

mounted

* 5-től kezdve számol visszafelé másodpercenkénet, és amit eléri a nullát, átdob a bejelentkezés oldalra.

# NotFoundPage



Az oldal akkor jelenik meg, ha a felhasználó olyan linket próbál elérni, amely nem létezik.

Nem tartalmaz javacsript kódot, kizázólag html+css-ből épül fel.

Komponensek

# Index

Az Index komponens a főoldal megjelenítéséért felelős. A tartalom része 3 elemből áll.

1. Legutóbbi fájlok
   1. Taralma 1 komponensből áll: LastFilesTable (LastFilesTable.vue)
   2. Ez ugyan az a komponens, mint ami a *Saját fájlok* menüpontban található, azzal a különbséggel, hogy csak a 3 legutóbb módosított fájlt jeleníti meg.
   3. A komponensnek a *lastfilesobject* paraméterébe átadásra kerül a *lastfiles* nevű objektumban tárolt adatok, valamit a *refresh* paraméterbe az adatok újratöltéséért felelős függvény.
2. Statisztika
   1. Elemei *StatElement* komponensek. Megjelenítik hogy fájltípusonkét (Zene, Videó, Kép, Dokumentum) és összesen mekkora méretben találhatók a fájlok az adott felhasználói fiókban.
3. Fájl felöltése
   1. Ebben a menüpontban lehet fájlt feltölteni az adatbázisba.
   2. A fájlok feltöltését az *uploadfile* metódus végzi.
   3. A form-ban az enctype biztosítja, hogy fájlokat lehessen küldeni.

Állapotok

data

lastfiles

* Ebbe a változóba fogja betölteni a *getlastfiles* metódus a 3 legutóbb módosított fájlt.

stat

* Ebbe a változóba tölti be a getStat metódus a statisztika adatait.

methods

uploadfile

* Egy FormData objektumba összegyűjti a Fájl feltöltése menüpont inputjából a betallózott fájlokat, majd a FormData objektummal együtt továbbítja a backendnek őket. Sikeres feltöltés esetén betöltődnek újra a legutóbbi fájlok és a statisztika.

getlastfiles

* A backend api/file/get/{limit?} útvonaláról betölti az adott legutóbbi 3 fájlját, majd átadja a *lastfiles* változónak.

getStat

* A backend api/file/stat útvonaláról betölti az adott felhasználó statisztikáját, majd átadja a kapott adatokat a *stat* változónak.

# Login

A Login komponens a bejelentkező felület megjelenítéséért felelős. Itt van elhelyezve a bejelentkező form. Az input mezők a html adottságait kihasználva validálják a bemeneti adatokat. A bejelentkezés alatt lehetőség van jelszó emlékeztető email küldésére is.

Használt komponensek

LoginOrRegister (LoginOrRegister.vue)

* Ez a komponens 2 gombot tartalmaz, amelyek megjelenítik a kiválasztott felületet. (Bejelentkezés/Regisztráció)

Változók

data

* Ebben tárolódnak a bejelentkezéshez szükséges adatok. (felhasználónév-jelszó)

store

* Ebbe tárolódnak a bejelentkezett felhasználó adatai.

passwordreminder

* Ebbe van eltárolva az email cím, amire a jelszóemlékeztető el lesz küldve.

mailsent

* Felelős az email sikeresen elküldve üzenet megjelenítéséért.

remindershow

* Felelős a jelszó emlékeztető mező megjelenítéséért.

Függvények

submit

* A bejelentkezés form küldés eseménye. Elküldi a backendnek a bekért felhasználónév-jelszó párost. Sikeres bejelentkezés esetén a store-ba eltárolja a bejelentkezett felhasználó adatait, majd átirányít a főoldalra.

displayreminder

* Megjeleníti a jelszó emlékeztető mezőt

reminder

* Elküldi a backendnek a jelszó emlékeztető kérést a megadott email címmel.

# Register

A Register komponens a regisztrációs felület megjelenítéséért felelős. Itt van elhelyezve a regisztrációs form. Az input mezők a html adottságait kihasználva validálják a bemeneti adatokat. Amennyiben mégis hibás adatok kerülnének elküldésre, a backend visszaküldött hibaüzenetei az egyes input mezők alatt jelennek meg. Sikeres regisztráció esetén „Sikeres regisztráció” üzenetet jelenít meg, és 5 másodperc után automatikusan átirányít a bejelentkező felületre.

Használt komponensek

LoginOrRegister (LoginOrRegister.vue)

Állapotok

data

errors

* Ebbe a változóba fogja betöltődni a hibaüzenetek, ha a regisztrációs űrlap hibásan lenne elküldve.

data

* Ebbe az objektumban vannak tárolva a regisztrációs űrlap adatai.

methods

submit

* Ez a metódus küldi el a backendnek a regisztrációs űrlap adatait. Megnézi, hogy a jelszó megerősítése mező egyezik-e a megadott jelszóval. Amennyiben nem, erről tájékoztatja a felhasználót. Sikeres elküldés esetén átirányít a bejelentkező felületre.

# LastFilesTable

Ez a tábla jeleníti meg a felhasználók saját fájljait. Lehetőséget biztosít több fájl egyszerre kijelölésére és rajta műveletek végrehajtására.

Használt komponensek

LastFilesTableRow (LastFilesTableRow.vue)

Állapotok

data

lastfiles

* Benne tárolódnak a megjelenítésre szánt fájlok. Tartalmát a szülő komponensétől kapja a lastfilesobject paraméteren keresztül. Tartalma automatikusan frissül a watch állapotban.

nofilemessage

* Ez a változó felelős a „Nincs feltöltött fájlod” üzenet megjelenítéséért.

checked

* Ez alapján lesz beállítva, hogy a táblázat összes eleme ki van-e jelölve vagy nincs kijelölve.

checkedBoxesData

* A kijelölt jelölőnégyzetek adatait tárolja. (Pontosabban az abban a sorban lévő fájl adatait)

store

* A bejelentkezett felhasználó eléréséhez szükséges.

groups

* A csoportos megosztáshoz tárolja a felhasználó csoportjainak adatait.

methods

getgroups

* A felhasználó csoportjainak adatait kéri el a backendtől.

checkall

* Ki- és bejelöli az összes táblázatban lévő fájlt.

download

* Letölt egy adott fájlt. A downloadselected metódusban van felhasználva.

deletefile

* Kitöröl egy adott fájlt. A deleteselected metódusban van felhasználva.

share

* Megoszt egy adott fájlt. A shareselected metódusban van felhasználva.

downloadselected

* Letölti a táblázatban kijelölt fájlokat.

deleteselected

* Kitörli a táblázatban kijelölt fájlokat.

shareselected

* Megosztja a táblázatban kijelölt fájlokat.

# LastFilesTableRow

Ez a komponens adja a LastFilesTable komponens sorait. Megjeleníti az egyes fájlok adatait, és a lenyíló menüjében lehetőséget biztosít a fájlokat törölni, letölteni és megosztani.

# LeftSideMenu

Ez a komponens adja az oldal menüjét. Az éppen használt menüpont gombját minden oldalon kijelöli. Megjelenik benne a profilkép, és egy számláló, amely az oldalon tétlenül biztosított 30 perctől kezdve számol vissza, és a végén kijelentkezik.

# LogOutCounter

A LeftSideMenu komponensnél említett számlálót valósítja meg.

# StatElement

A főoldali statisztika elemeinek komponens. Meg lehet adni neki milyen kategória nevet jelenítsen meg, az ikont hozzá, a foglalt tárhely méretét, az össz tárhely méretét, és hogy milyen színnel jelenjen meg.

# ProgressBar

Mivel a HTML által biztosított progress elem nem volt megfelelő a célra, szükséges volt saját komponens létrehozása. Be lehet állítani a szélességét, a tárhely maximum méretét, a megtelt tárhely méretét és a kitöltött rész színét. A két méret adat és a maximum szélesség alapján százalékosan kiszámolja hogy mennyi rész legyen színes, és mennyi üres.

# Groups

Ez a komponens a *Felhasználók* menüpontban van felhasználva. Itt található a csoportok létrehozása menü, és a létrehozott csoportok kilistázása, illetve tájékoztat, ha nincsen létrehozott csoportunk. Alkomponense a Groupmembers (Groupmembers.vue)

# Groupmembers

Ez a komponens listázza ki az adott csoport tagjait. Lehetőség van a tagokat eltávolítani a csoportból.

# GroupFilesRow

Ez a komponens a *Velem megosztva* aloldalon kapott helyet. Ebben vannak megjelenítve azon csoportban lévő fájlok adatai, amelyekhez hozzá van adva az adott felhasználó.

# settings/setDatas

A beállítások Adatok menüpontja található benne. Ebben lehet kiválasztani/kitörölni a profilképet, és az email címet beállítani.

# settings/setSecurity

A beállítások Biztonság menüpontja található benne. Itt lehet megváltoztatni a jelszót, és a felhasználói fiókot törölni. (A fiók törlés admin szerepkörrel nem lehetséges)

# admin/Files

Az admin menü Fájlok kezelése menüpontja. Itt található egy teljes adatbázisra kiterjedő fájlkereső. Lehet keresni fájlnévre, és tulajdonos névre. Ezek kombinálhatóak is, illetve üres adatok esetén minden fájlt megjelenít.

# admin/Users

Az admin menü Felhasználók kezelése menüpont. Található benne egy felhasználó kereső, ahol a felhasználó nevére / nevének részletére lehet keresni. Itt lehet felhasználókat törölni, és nekik jogosultságot adni.

**Android alkalmazás**

A FiveFile-hoz készült Androidon futtatható alkalmazás is. Itt kizárólag korlátozott funkciók érhetőek el, hiszen a fő profil a webalkalmazás lefejlesztése volt. Ezek a funkciók a feltöltött fájlok kilistázása, letöltése, törlése, beállításokban email cím és jelszó megváltoztatása, valamit a felhasználói fiók törlése. Az alkalmazás Flutter-rel készült. Ennek előnye, hogy a kód mind webre, Androidra, és IOS-re is lefordítható.

Az alkalmazásban 4 útvonal – 4 menüpont található.

Az útvonalak definiálása a *main.dart* fájlban történt.

1. /login: Bejelentkező oldal (pages/login.dart)
2. /register: Regisztrációs oldal (pages/register.dart)
3. /home: Főoldal (pages/home.dart)
4. /files: Saját fájlok (pages/files.dart)
5. /settings: Beállítások (pages/settings.dart)
6. /profil: Profil (pages/profil.dart)

login.dart

Az oldalon elhelyezett komponens osztályok:

* LoginTitle: az oldal címét jeleníti meg egy Text widgetben.
* UserName: A felhasználónév beviteli mezője. Tartalma bekerül a *username* kontrollerbe.
* Password: A jelszó beviteli mezője. Tartalma bekerül a *password* kontrollerbe.
* Bejelentkezés gomb: A szükséges adatok kitöltése esetén meghívja a *UserController* *login* metódusát, paraméterben átadva a felhasználónév-jelszó párost.
* Regisztráció gomb: Átirányít a /register útvonalra.

register.dart

Az oldalon elhelyezett komponens osztályok:

* RegisterTitle: az oldal címét jeleníti meg egy Text widgetben.
* UserName: A felhasználónév beviteli mezője. Tartalma bekerül a *username* kontrollerbe.
* Email: Az email cím beviteli mezője. Tartalma bekerül az *email* kontrollerbe.
* Password: A jelszó beviteli mezője. Tartalma bekerül az *password* kontrollerbe.
* PasswordConfirm: A jelszó megerősítés beviteli mezője. Tartalma bekerül az *passwordConfirm* kontrollerbe.
* Regisztrálok gomb: a fenti kontrollerek adatait elküldi a *UserController* register metódusának, majd átirányít a /login útvonalra.

home.dart

* Ez az oldal tájékoztató jellegű. Közli a felhasználóval hogy korlátozott funkciókkal elérhető az app.

files.dart

* A backendet futtatva minden esetben látni, hogy egy töltő képernyő várakoztatja a felhasználót, amíg az adatai betöltődnek. Ha nincsen mit betölteni akkor kiírjuk a felhasználónak, hogy nincsenek feltöltött fájljai. Mindez a FutureBuilder felhasználásával történt. A FutureBuildernek a future attribútumába megadtuk A *UserController* *loadFiles* metódusát, amely betölti a felhasználó fájljait. Amit megérkeztek az adatok, kilistázásra kerülnek. A kilistázás *ListView widgetbe* történt, ezen belül mindegyik elem be van ágyazva egy *Slidable widgetbe*. A Slidable widget *startActionPane* attribútumában van megadva, hogy milyen effekttel történjen a widget mozgatása, és hogy milyen widgetek jelenjenek meg elmozgatás esetén. DeleteFileButton és DownloadButton. Ezek a childred attribútumban vannak elhelyezve. Mindkét gombnak átadásra kerül a backendtől várt fájl név, amely szerint tudja meg törölni, illetve letölteni az adott fájlokat. Az oldal frissíthetőségéről a RefreshIndicator widget gondoskodik.

settings.dart

Az oldalon elhelyezett komponens osztályok:

* Email: Az email cím beviteli mezője. Tartalma bekerül az *email* kontrollerbe. A hinttext attribútumában megjelenik a bejelentkezett felhasználó aktuális email címe.
* Password: A jelszó beviteli mezője. Tartalma bekerül az *password* kontrollerbe.
* Confirm: A jelszó megerősítés beviteli mezője. Tartalma bekerül a c*onfirm* kontrollerbe.
* Mentés gomb: a fenti kontrollerek adatait elküldi a *UserController* settings metódusának.

profil.dart

Az oldalon elhelyezett komponensek:

* Köszöntő mező: Üdvözli a belépett felhasználót.
* Felhasználónév mező: Megjeleníti a belépett felhasználó felhasználónevét.
* Email cím mező: Megjeleníti a belépett felhasználó email címét.
* Regisztráció dátuma mező: Megjeleníti mikor regisztrált a felhasználó.
* Profil törlés jelölőnégyzet: Ha be van pipálva, kijelentkezéskor törlődik a felhasználói profil.
* Kijelentkezés gomb: Kijelentkezik az oldalról.

file\_model.dart

* A felhasználó fájljainak modelje. Tárolja a fájlról az azonosítóját, a tulajdonos azonosítóját, típusát, nevét, méretét, a megosztott csoport azonosítóját, feltöltés és módosítás dátumát.

file\_model.g.dart

* Generált fájl.
* A beérkező fájlok átalakítását végzi.

urlprefix.dart

* A backend requestek egyszerű átírását könnyíti meg.

user\_model.dart

* A bejelentkezett felhasználóról tárolja az azonosítóját, nevét, email címét, szerepkörét, profilképét és regisztrációjának dátumát.

user\_controller.dart

Az alábbi kéréseket küldi a backend felé:

* login: Bejelentkezés
* register: Regisztráció
* logout: Kijelentkezés
* settings: Beállítások mentése
* loadFiles: Fájlok betöltése
* deleteFile: Kiválasztott fájl törlése

menu.dart

* Ebben a fájlban található az oldal menü komponense.

DeleteFileButton.dart

* Ebben a fájlban található a kiválasztott fájl törléséért felelős gomb. Paramétere a törlendő fájl neve, és azt tovább küldi a *UserController* *deleteFile* metódusának.

DeleteFileButton.dart

* Ebben a fájlban található a kiválasztott fájl letöltéséért felelős gomb. A letöltött fájl a Letöltések mappába fog kerülni.

**A programozáshoz használt források jegyzéke**

1. HTML: HyperText Markup Language
   1. https://developer.mozilla.org/en-US/docs/Web/HTML
2. CSS: Cascading Style Sheets
   1. https://developer.mozilla.org/en-US/docs/Web/CSS
3. JavaScript
   1. https://developer.mozilla.org/en-US/docs/Web/JavaScript
4. W3Schools
   1. https://www.w3schools.com/
5. Laravel
   1. https://laravel.com/docs/8.x/installation
6. Laravel Collective
   1. <https://laravelcollective.com/docs/6.x/html>
7. Vue JS
   1. <https://vuejs.org/>
8. Pinia
   1. <https://pinia.vuejs.org/>
9. Flutter
   1. <https://flutter.dev/>
10. Stack Overflow
    1. <https://stackoverflow.com/>