Fejlesztői dokumentáció

Laravel

UserController

public function index()

API hívás esetén visszaadja az összes regisztrált felhasználót.

public function show($id)

API hívás esetén kikeresi ID alapján a felhasználót, és visszaadja

public function destroy($id)

API hívás esetén kikeresi ID alapján a felhasználót, és kitörli az adatbázisból

public function SignupUser(SignupRequest $request)

SignupRequest alapján validálja a request-et, majd kap jelszót, role-t, szintet, pontszámot, és profilképet. Végül ezen tulajdonságok alapján be lesz regisztrálva a felhasználó, és a fiókjához fog társulni egy UserSettings rekord is az adatbázisban, ahol a személyes preferenciáit tároljuk.

public function LoginUser(LoginRequest $request)

LoginRequest alapján validálja a request-et. Ha a felhasználó rossz email címet írt be, akkor egy “Az adatbázisban nem található ilyen email” üzenetet kap, ha pedig rossz jelszót, akkor “Hibás adatok” üzenetet. Egyéb esetben pedig sikeresen bejelentkezik.

public function ModifyUser(ModifyUserRequest $request)

ModifyUserRequest alapján validálja a request-et, majd adatbázisban módosulnak a felhasználó adatai.

public function DeleteUser(Request $request)

Meghívja a destroy metódust, és törli a felhasználót. Amennyiben a felhasználó csapatvezető, kap egy tájékoztatást, hogy előbb a csapatát kell törölnie, és csak utána tudja a saját profilját.

public function LogoutUser(Request $request)

Kijelentkezteti a felhasználót.

public function AcceptInvitation(AcceptInvitationRequest $request)

Elfogadja a bejövő baráti kérelmet, és ezután a barátok oldalon látható lesz az adott felhasználó.

public function RejectInvitation(Request $request)

Elutasítja a baráti kérelmet.

public function DeleteFriend(Request $request)

Törli az adott barátot.

public function DeleteSentRequest(Request $request)

Törli a baráti kérelmet.

public function ReleaseBlockedUser(Request $request)

Ha egy felhasználó blokkolt egy másik felhasználót, akkor ez a metódus megszünteti a tiltást.

public function InviteFriend(Request $request)

Baráti kérelmet küld egy másik felhasználónak

public function GetCountryLeaderboard()

API hívás esetén visszaadja az ország TOP 10 felhasználóját.

public function GetNonFriends(Request $request)

Visszaadja azokat a felhasználókat, akik az adott felhasználónak nem a barátai.

public function SaveSettings(Request $request)

A felhasználónak a beállítások oldalon van lehetősége saját preferenciákat beállítani, és ezeket eltárolni.

TeamController

public function index()

API hívás esetén visszaadja az összes csapatot.

public function show($id)

API hívás esetén kikeresi ID alapján a csapatot, és visszaadja

public function update(TeamRequest, $id)

API hívás esetén kikeresi ID alapján a csapatot, validálja a requestet, és módosítja a csapatot az adatbázisban.

public function CreateTeam(Request $request)

Egy 18 év feletti felhasználó létrehoz egy csapatot. A role-ja átváltozik ‘user’-ről ‘admin’-ra, és a csapat vezetőjének ő lesz beállatva.

public function DeleteTeam(Request $request)

A csapat ID-t null-ra változtatja a vezetőnél is, és a csapattagoknál is, majd törli a csapatot.

public function AddMember(Request $request)

Hozzáad egy új tagot a csapathoz.

TaskController

public function index()

API hívás esetén visszaadja az összes feladatot.

public function show($id)

API hívás esetén kikeresi ID alapján a feladatot, és visszaadja

public function store(TaskRequest $request)

API hívás esetén feladathoz társított képet eltárolja storage-ban, majd elmenti a feladatot.

public function update(TaskRequest $request, $id)

API hívás esetén kikeresi ID alapján a feladatot, validálja a requestet, és módosítja a feladatot az adatbázisban.

public function AcceptTask()

A felhasználó el tud vállalni a szintjén lévő feladatok közül valamit.

public function FinishTask()

A felhasználó amennyiben úgy gondolja, hogy végzett a feladattal, le tudja adni a vezetőjének ellenőrzésre. Azonban ha nincsen csapata (ezzel együtt vezetője), akkor egy session üzenet tájékoztatni fogja őt erről, és nem megy végbe a leadás.

public function ConfirmTask()

A vezetők el tudják fogadni a csapattagoktól beérkezett feladatok megoldásait, és ekkor a felhasználó sikeresen elvégezte a feladatot.

public function RejectTask()

A vezetők el tudják utasítani a csapattagoktól beérkezett feladatok megoldásait, és ekkor a felhasználó újra tud próbálkozni a megoldással.

public function ViewTask()

Átirányítja a felhasználót a szintek-feladatok oldalra, és egy szürke jelzéssel látni fogja a feladatot, amit éppen meg szeretne nézni.

LevelController

public function index()

API hívás esetén visszaadja az összes szintet.

public function store(LevelRequest $request)

API hívás esetén validálja a requestet, majd az új szintet eltárolja adatbázisban.

public function update(LevelRequest $request, $id)

API hívás esetén kikeresi ID alapján a szintet, validálja a requestet, majd módosítja adatbázisban.

SiteController

public function Country()

Visszaadja a site.country nézetet.

public function Friends()

Visszaadja a site.friends nézetet.

public function Index()

Visszaadja a site.index nézetet.

public function Levels()

Visszaadja a site.levels nézetet.

public function Login()

Visszaadja a site.login nézetet.

public function MyTasks()

Visszaadja a site.mytasks nézetet.

public function Signup()

Visszaadja a site.signup nézetet.

public function Settings()

Visszaadja a site.settings nézetet.

public function MyTeam()

Visszaadja a site.myteam nézetet.

public function Profile()

Visszaadja a site.profile nézetet.

JavaScript

Country.js

LoadCountryTop10(users)

Betölti az országban lévő top 10 felhasználókat

Fetch.js

GetCountryLeaderBoard()

API-ról lekéri az országban lévő top 10 felhasználókat

GetTasks()

API-ról lekéri az összes feladatot

GetUserByID(user\_id)

API-ról lekéri az adott ID-val rendelkező felhasználót.

NonFriendsFiltered()

API-ról lekéri összes felhasználót, aki az adott user-nek nem barátja, és megfelel a user keresési feltételeinek.

GetNonFriends()

API-ról lekéri az összes felhasználót, aki az adott user-nek nem barátja.

InviteFriend(id)

API-ra csatlakozva elküld egy baráti kérelmet az adott ID-val rendelkező usernek.

LoadingImage()

Létrehoz egy töltő ikont (amíg a felhasználó az API válaszára vár) DOM, illetve CSS segítségével.

displayLoading()

Megjeleníti a töltő ikont.

hideLoading()

Eltűnteti a töltő ikont.

Friends.js

LoadNonFriends()

Betölti az összes felhasználót, aki az adott usernek nem a barátja.

FiltersVisible()

A részletes szűrő gombra kattintva igaz-hamis lesz a szűrők megjelenésének értéke

Levels.js

LoadAllTasks(tasks)

Betölti az összes feladatot

ScrollToTask(taskID)

Szintek-feladatok oldalon legördül az adott ID-val rendelkező feladathoz

Modal.js

ShowProfile(user)

Megjelenít egy modal-t a paraméterként kapott user adataival

Myteam.js

CsapatomShow()

A csapatom gombra kattintva megjelennek a csapat tulajdonságai

CsapattagFelvetelShow()

A csapattal felvétele gombra kattintva megjelenik egy form, ahol a vezető fel tud venni új csapattagokat.

Setting.js

darkMode()

Meghívásra sötét hátteret ad az oldalnak.

Weather.js

showWeather(coordinates), fetchWeather(), appendCarouselItem(data)

Ezek a metódusok a főoldalon lévő időjárás jelentés megjelenítését szolgálják.

convertUnixTime(unix)

Unix idő formátumot konvertál alapértelmezett idő formátumra.

C#

RestApiHandler

Ez az osztály felelős a REST API kommunikációért. Itt csak generikus metódusok találhatóak, és csak meg kell hívni őket a megfelelő paraméterekkel, nem kell mindig lekódolni ugyanazt a REST API hívást.

T GetObject<T>(string path)

Ezt a metódust a GetAll, GetByID hívásoknál használjuk. Paraméterként bekéri az api útvonalat, és visszaadja a backend által szolgáltatott értékeket.

Task<bool> PostObject<T>(string path, T obj)

Ezzel a metódussal tudunk felvinni adatokat az adatbázisba. Paraméterként bekéri az api útvonalat, és a T típusú objectet, amit elküld az útvonalra. Így kerül fel az adatbázisba új érték.

Task<bool> PutObject<T>(string path, T obj)

Ezzel a metódussal tudunk módosítani adatokat az adatbázisban. Paraméterként bekéri az api útvonalat, és a T típusú objectet, amit elküld az útvonalra. Így módosulni fog az adott útvonalon a régi érték az új értékre.

Base64

Ez az osztály felelős a képek REST API-n keresztül történő közlekedéséért. Ha a backendről küldünk képet a C#-ra, akkor a laravelben egy resource fájlban base64 string-gé konvertáljuk a képet, majd amikor átjött REST API-n, akkor az itt található Decode() metódus segítségével vissza tudjuk alakítani kép formátumba, és meg tudjuk jeleníteni.

Amikor C#-ról küldünk képet backendre, az már egy picit bonyolultabb. Először is egy gombot megnyomva egy párbeszédpanel segítségével kiválasztjuk a kívánt képet. Majd ha sikeresen választottunk, akkor a gomb DataContext-je megkapja a kép teljes útvonalát. Amikor pedig rákattintunk a felvétel/módosítás gombra, akkor meghívódik ennek az osztálynak a Decode() metódusa, mely megkapja paraméterként a gombnak a DataContext-jét. Végül átalakítja base64 string-gé, és ez lesz átküldve a REST API-n laravelbe, ahol a controller dekódolja, és eltárolja a storage-ban.

Felhasznált segédanyagok / források:

**Laravel, PHP:**

* Rostagni Csaba tanár úr videói
* <https://laravel.com/docs/9.x>
* <https://www.php.net/manual/en/index.php>

**C#:**

* Répás Csaba tanár úr videói
* <https://stackoverflow.com/questions/21325661/convert-an-image-selected-by-path-to-base64-string>
* <https://github.com/ookii-dialogs/ookii-dialogs-wpf>

**HTML, CSS, JS:**

* <https://getbootstrap.com/docs/5.0/getting-started/introduction/>
* <https://www.codeinwp.com/snippets/convert-unix-time-to-date-with-javascript/>