A képen clipart, macska, illusztráció, rajzfilm látható

Automatikusan generált leírás

**Széchenyi István Katolikus Technikum és Gimnázium**

**Szoftverfejlesztő és -tesztelő projektfeladat**

**„Nonprofit adományozó webalkalmazás”**

Készítették:

Trefiman Lili Nóra,

Ivák Panna,

Berecz Balázs

**Ózd, 2026**

Tartalomjegyzék

[1. Bevezetés 3](#_Toc222139146)

[2. Adatbázis 4](#_Toc222139147)

[3. Backend 9](#_Toc222139148)

[a. Alkalmazás backend részének részletes leírása + képernyőfotók 9](#_Toc222139149)

[i. index(): 9](#_Toc222139150)

[ii. store(): 9](#_Toc222139151)

[iii. update(): 10](#_Toc222139152)

[iv. destroy(): 10](#_Toc222139153)

[4. Tesztelés(Thunder Client) 12](#_Toc222139154)

[b. Az alkalmazás sikeres tesztjeinek képernyőfotói 12](#_Toc222139155)

[c. Az alkalmazás sikertelen tesztjeinek képernyőfotói 14](#_Toc222139156)

## Bevezetés

A webalkalmazásunk egy olyan nonprofit szervezet online platformja, amely segíteni szeretne azoknak, akiknek a legnagyobb szükségük van támogatásra. A szervezet célja, hogy rászoruló emberekhez juttasson el ruhákat, így hozzájárulva a mindennapi életük megkönnyítéséhez. Ez az online felület azért készült, hogy megkönnyítse és bárki számára elérhetővé tegye az adományozást. Az oldal segítségével bárki, bárhonnan könnyedén feltöltheti a megunt vagy már nem használt ruháit, amelyeket mások közvetlenül elérhetnek. A civil szervezetek, akik ténylegesen segítenek a rászorulókon, itt kiválaszthatják azokat az adományokat. Így az adományozók biztosak lehetnek abban, hogy a ruhák valóban jó helyre kerülnek, miközben az online rendszer egyszerűvé, gyorssá és átláthatóvá teszi az egész folyamatot.

A projekt GitHub hivatkozása: <https://github.com/BBazsi2006/Projekt.git>

## Adatbázis

Ez az adatbázis egy olyan rendszerhez készült, ahol adományozók különböző alapítványokat támogatnak különféle adományokkal. A cél az volt, hogy egy egyszerű, átlátható és logikus adatmodell készüljön, amiben nyomon lehet követni, hogy ki adományoz, kinek, milyen adományt, és az pontosan mikor kerül kiszállításra.

Az adatbázis összesen 7 táblából épül fel:

1. ember
2. alapitvany
3. adomanyozo
4. adomanynev
5. anyag
6. adomany
7. kiszallitas

1. emberek tábla

Ez a tábla tulajdonképpen egy „közös szülő” tábla, ahol minden szereplő (alapítványok és

adományozók) külön kategóriát kap. Így egy helyen nyilvántarthatók a rendszer felhasználói.

|  |  |  |
| --- | --- | --- |
| **Mező neve** | **Típus** | **Leírás** |
| id | int | Az ember egyedi azonosítója. Ez az elsődleges kulcs (PK). |
| kategoria | varchar(20) | Megmondja, hogy az adott ember „alapitvany” vagy „adomanyozo”. |

2. alapitvanyok tábla

Itt vannak tárolva azok az alapítványok, akik fogadják az adományokat. Minden alapítvány hozzá van kötve egy ember rekordhoz.

|  |  |  |
| --- | --- | --- |
| **Mező neve** | **Típus** | **Leírás** |
| alapitvanyId | ind | Az alapítvány egyedi azonosítója. Ez az elsődleges kulcs. |
| cegnev | varchar(50) | Az alapítvány neve. |
| helyszin | varchar(50) | Város vagy település, ahol az alapítvány működik. |
| telszam | int | Telefonszám számként tárolva. |
| email | varchar(50) | Kapcsolattartási e-mail cím. |
| emberId | int | Hivatkozás az ember tábla id mezőjére. |

3. adomanyozok tábla

Ebben tároljuk az adományozókat: nevüket, elérhetőségüket és a jelszavukat.

|  |  |  |
| --- | --- | --- |
| **Mező** **neve** | **Típus** | **Leírás** |
| adomanyozoId | int | Az adományozó egyedi azonosítója. Elsődleges kulcs. |
| nev | varchar(50) | Az adományozó teljes neve. |
| email | varchar(50) | Kapcsolattartási cím. |
| jelszo | varchar(50) | A bejelentkezéshez használható jelszó. |
| emberId | int | Az ember tábla megfelelő rekordjára mutat. |

4. adomanynevek tábla

Itt vannak felsorolva a különböző adományfajták (pl. élelmiszer, takaró stb.).

|  |  |  |
| --- | --- | --- |
| **Mező neve** | **Típus** | **Leírás** |
| adomanynevId | int | Az adomány típusának egyedi azonosítója. Elsődleges kulcs. |
| nev | varchar(50) | Az adomány neve / típusa. |

5. anyagok tábla

Ez a tábla azt mondja meg, hogy egy adott adomány milyen anyagból készült (például: fa, textil, műanyag stb.).

|  |  |  |
| --- | --- | --- |
| **Mező neve** | **Típus** | **Leírás** |
| anyagId | int | Anyag típus egyedi azonosítója. Elsődleges kulcs. |
| anyagnev | varchar(50) | Anyag megnevezése. |

6. adomanyok tábla

Itt tároljuk a konkrét adományokat: mit, mennyit és milyen anyagból.

|  |  |  |
| --- | --- | --- |
| **Mező neve** | **Típus** | **Leírás** |
| adomanyId | int | Egyedi azonosító. Elsődleges kulcs. |
| darabszam | int | Hány darab az adott adományból. |
| nevId | int | Hivatkozás az adomanynev tábla egyik adománytípusára. |
| adomanyanyagId | int | Hivatkozás az anyag tábla egyik anyagára. |

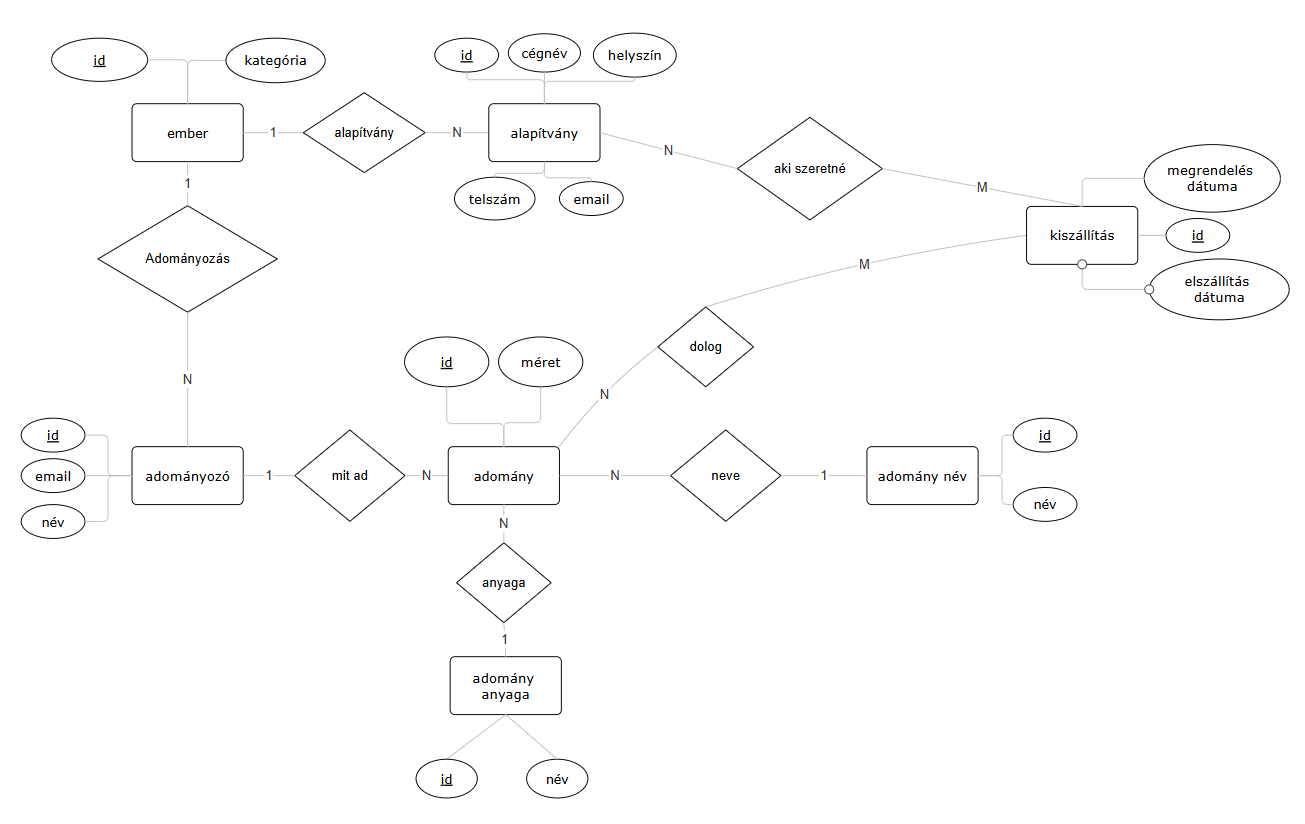
7. kiszallitasok tábla

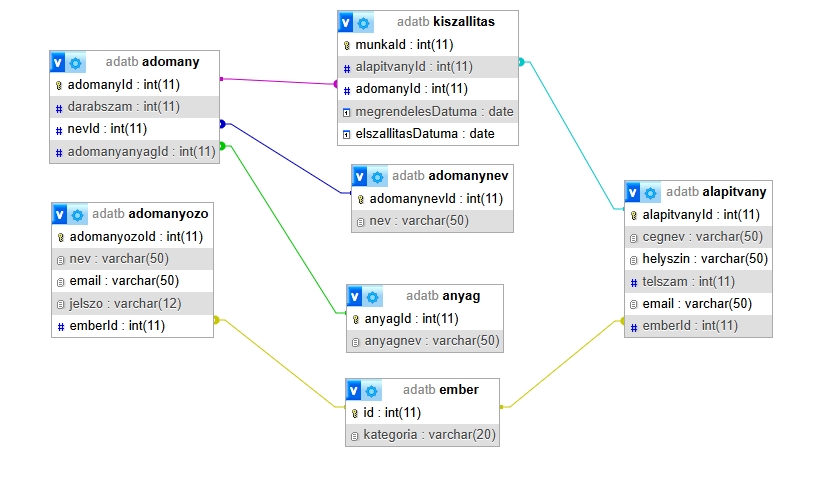
Ez tárolja a kiszállítási „feladatokat”: melyik alapítvány rendelt, melyik adományról van szó, és mik a dátumok.

|  |  |  |
| --- | --- | --- |
| **Mező neve** | **Típus** | **Leírás** |
| munkaId | int | A munka azonosítója. Elsődleges kulcs. |
| alapitvanyId | int | Annak az alapítványnak az ID-je, aki kapja az adományt. |
| adomanyId | int | A kiszállított adomány ID-je. |
| megrendelesDatuma | date | Mikor adták le a rendelést. |
| elszallitasDatuma | date | Mikor vitték el az adományt. |

Ez az adatbázis egy adományos rendszerhez készült, ahol követni lehet, hogy ki adományoz, melyik alapítvány kapja, mit kapnak, és mikor szállítják ki. Az *ember* tábla csak arra van, hogy el lehessen dönteni, ki adományozó és ki alapítvány. Az adományoknál külön van kezelve a típus és az anyag, hogy ne legyen minden összekeverve. A *munka* tábla köti össze az alapítványokat a konkrét adományokkal és a dátumokkal. A lényege az, hogy az adatok átláthatók legyenek és vissza lehessen keresni, mi hova került.

EK-model:





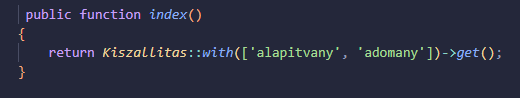
## Backend

### Alkalmazás backend részének részletes leírása + képernyőfotók

A backend feladata az adatok kezelése, tárolása és kiszolgálása a frontend felé egy REST API-n keresztül. A rendszer Laravel keretrendszerben készült, és különböző controllereken keresztül valósítja meg az egyes funkciókat: adatok lekérdezése, új adatok létrehozása, meglévő adatok módosítása, adatok törlése. Minden controller metódus felelős egy adott művelet végrehajtásáért, és szabványos HTTP kéréseken (GET, POST, PUT, DELETE) keresztül érhető el. A metódusok feldolgozzák a beérkező kéréseket, majd megfelelő választ adnak vissza JSON formátumban.A válaszok tartalmazzák az adott művelet eredményét, valamint a hozzá tartozó HTTP státuszkódot, amely jelzi a kérés sikerességét vagy sikertelenségét.

#### index():

Az index() metódus az összes alapítvány, illetve adomány adatát lekérdezi az adatbázisból. Ez biztosítja, hogy a frontend egy listában meg tudja jeleníteni az összes alapítványt és adományt a kapcsolódó adatokkal együtt. Ezek a kérések nem módosítják az adatokat, csak visszaadják a kért rekordokat vagy egy adott azonosítóhoz tartozó elemet.



#### store():

A store() metódus új rekord létrehozását végzi, például új adomány, adományozó vagy kiszállítás rögzítését. A kliens elküldi a szükséges mezőket (alapitvanyId, adomanyId stb.), a backend validálja az adatokat, majd elmenti az adatbázisba. Sikeres létrehozás után a rendszer visszaadja az új erőforrás adatait. A képen a kiszállítás store() metódusa látható.



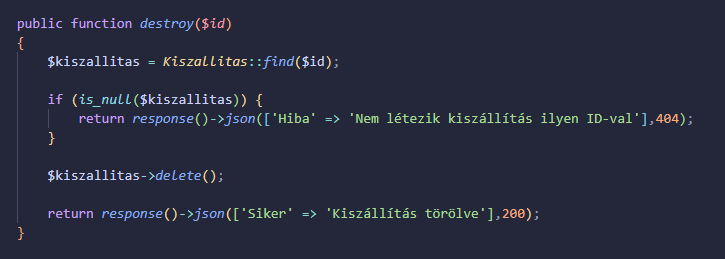
#### update():

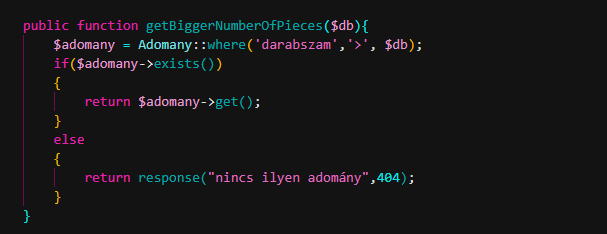
Az update() metódus meglévő adatok frissítésére használható, például egy kiszállítás adatai vagy egy adomány darabszáma módosítható vele. A kérés tartalmazza a frissített mezőket, amelyeket a backend az adott azonosítóhoz tartozó rekordban felülír. Laravelben ez az update() metódusban történik validáció és mentés után.

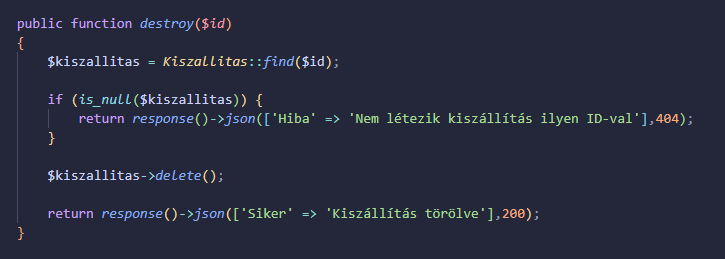


#### destroy():

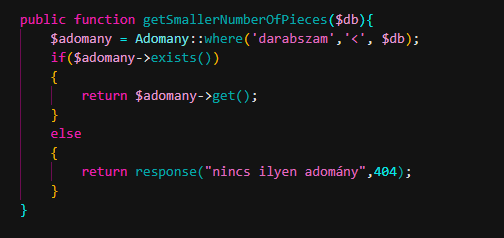
A destroy() végpont egy adott rekord törlésére szolgál, például egy kiszállítás eltávolítására az adatbázisból. A backend az ID alapján megkeresi a megfelelő sort, majd végrehajtja a törlést, figyelembe véve a kapcsolódó táblák közti relációkat. A válasz általában egy sikeres státusz vagy visszaigazoló üzenet.

**getBiggerNumberOfPieces()**

**.**

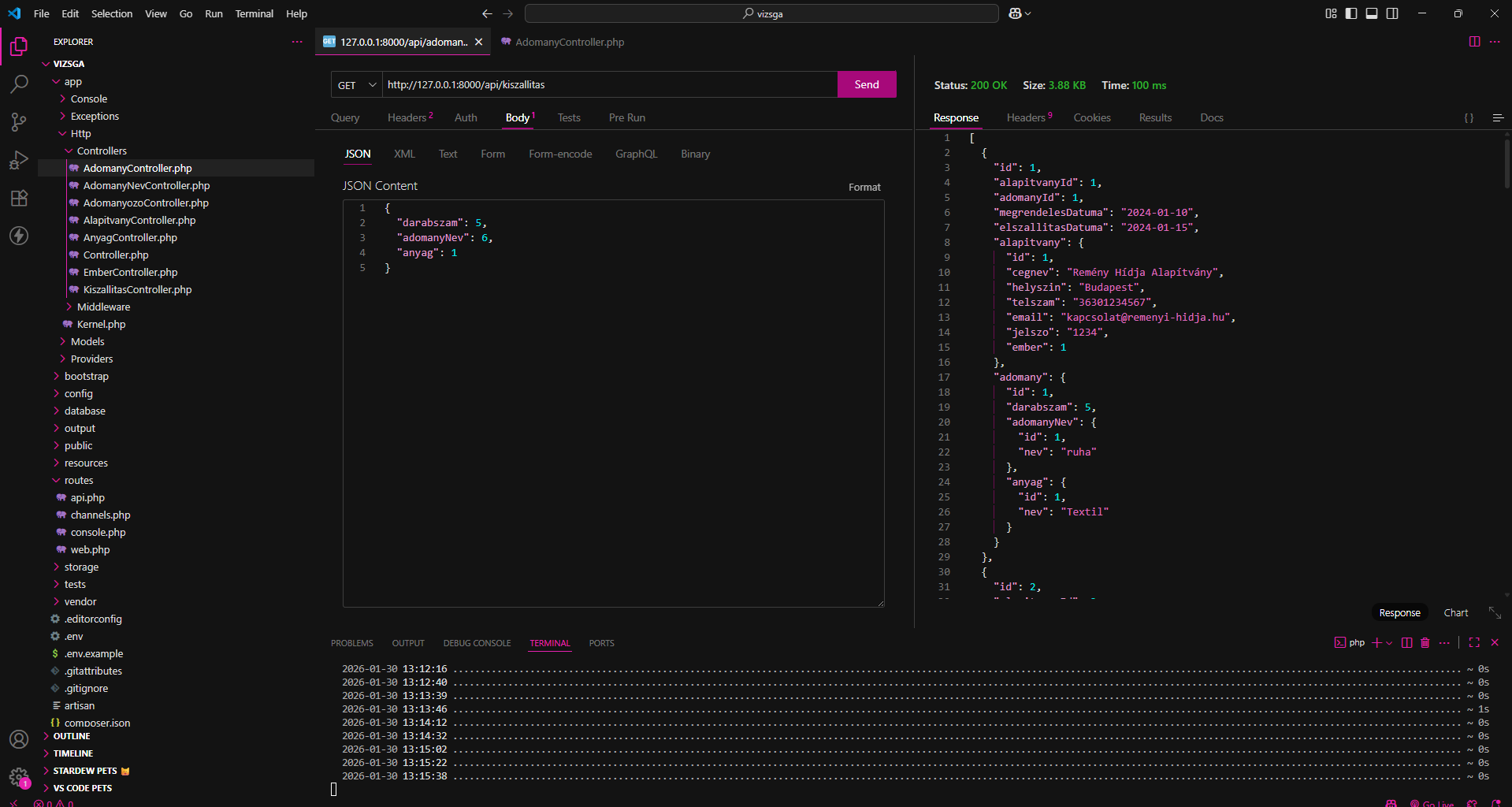
**getSmallerNumberOfPieces()**

**.**

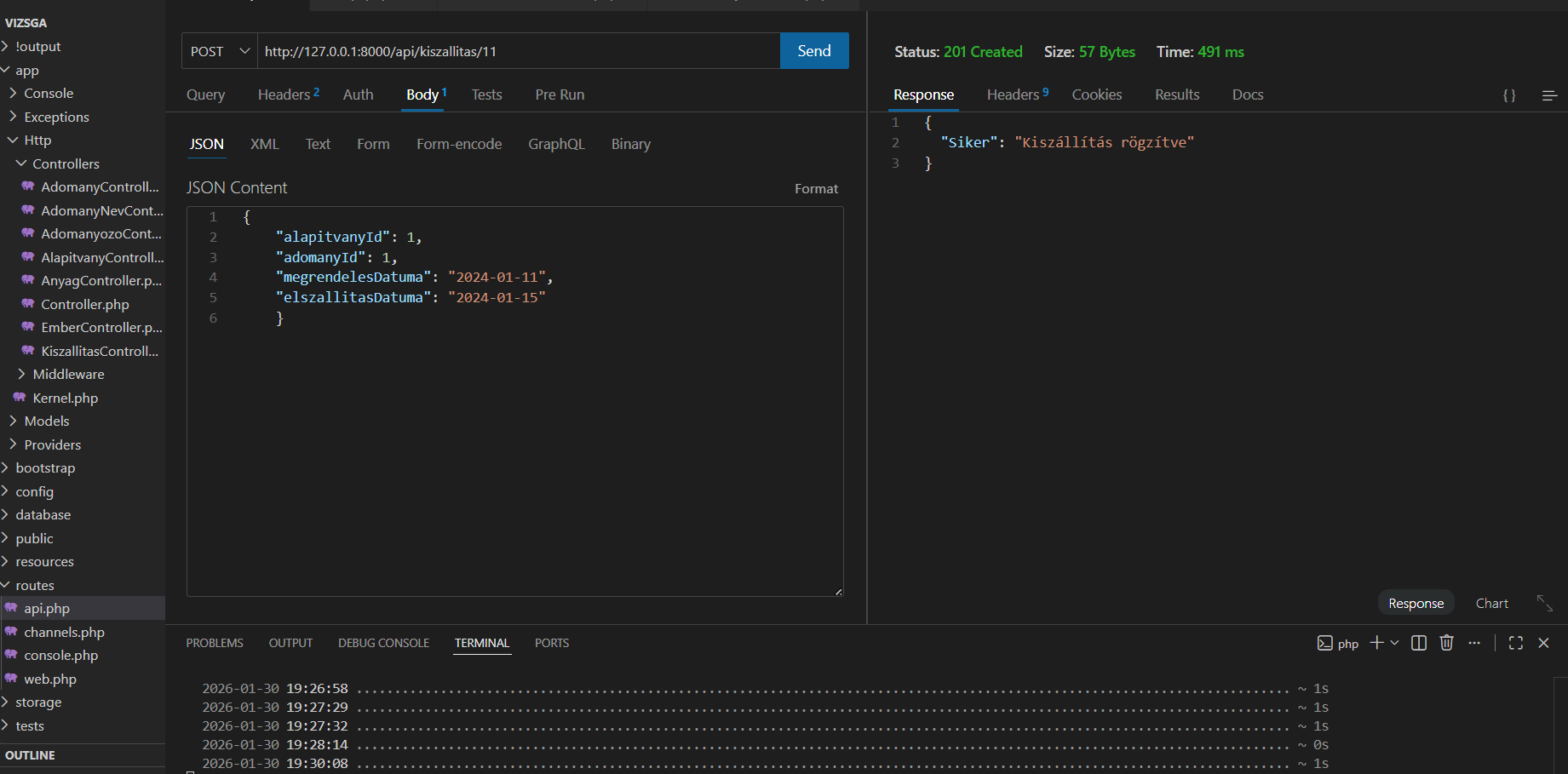
****

## Tesztelés(Thunder Client)

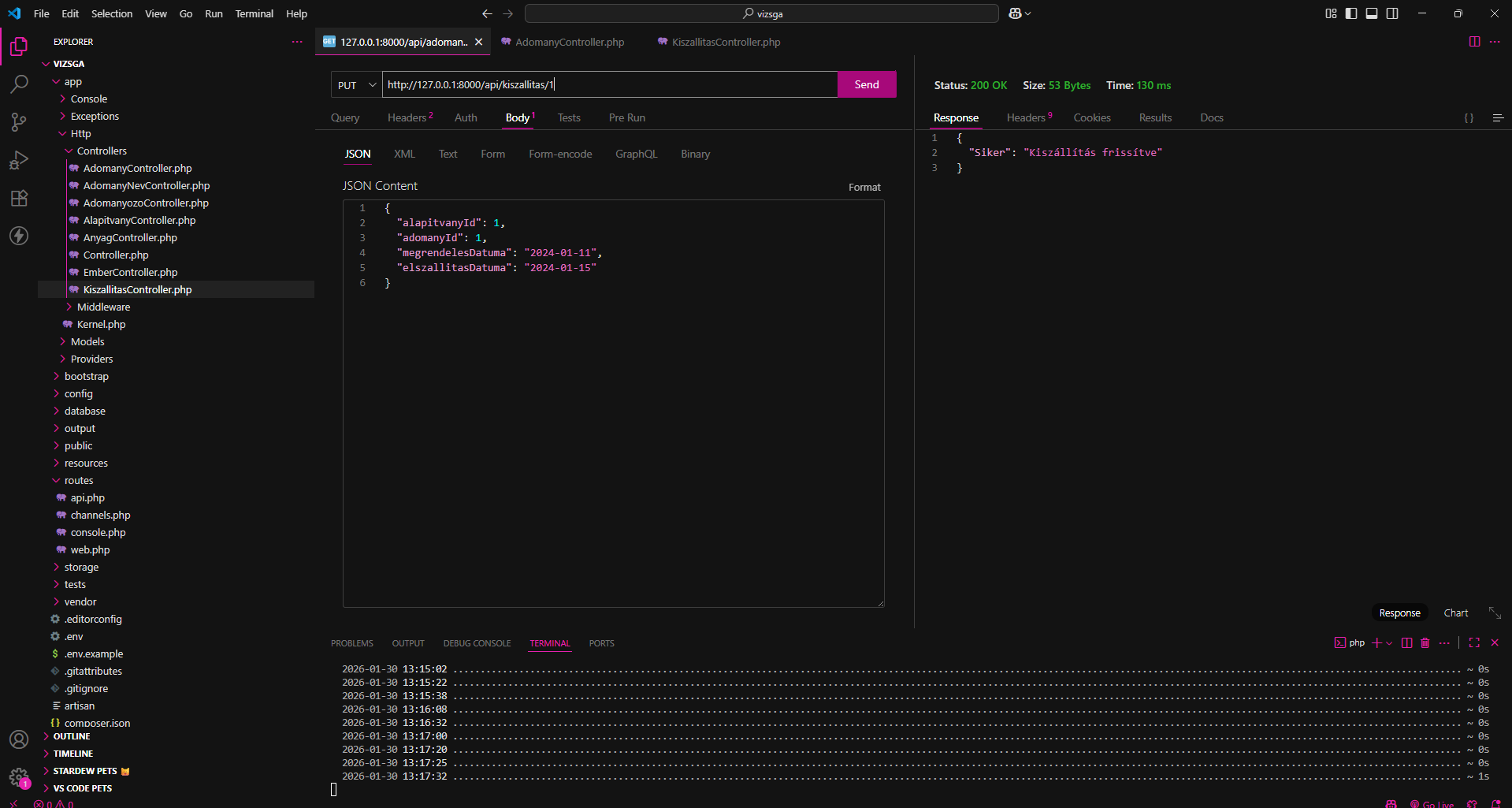
### Az alkalmazás sikeres tesztjeinek képernyőfotói

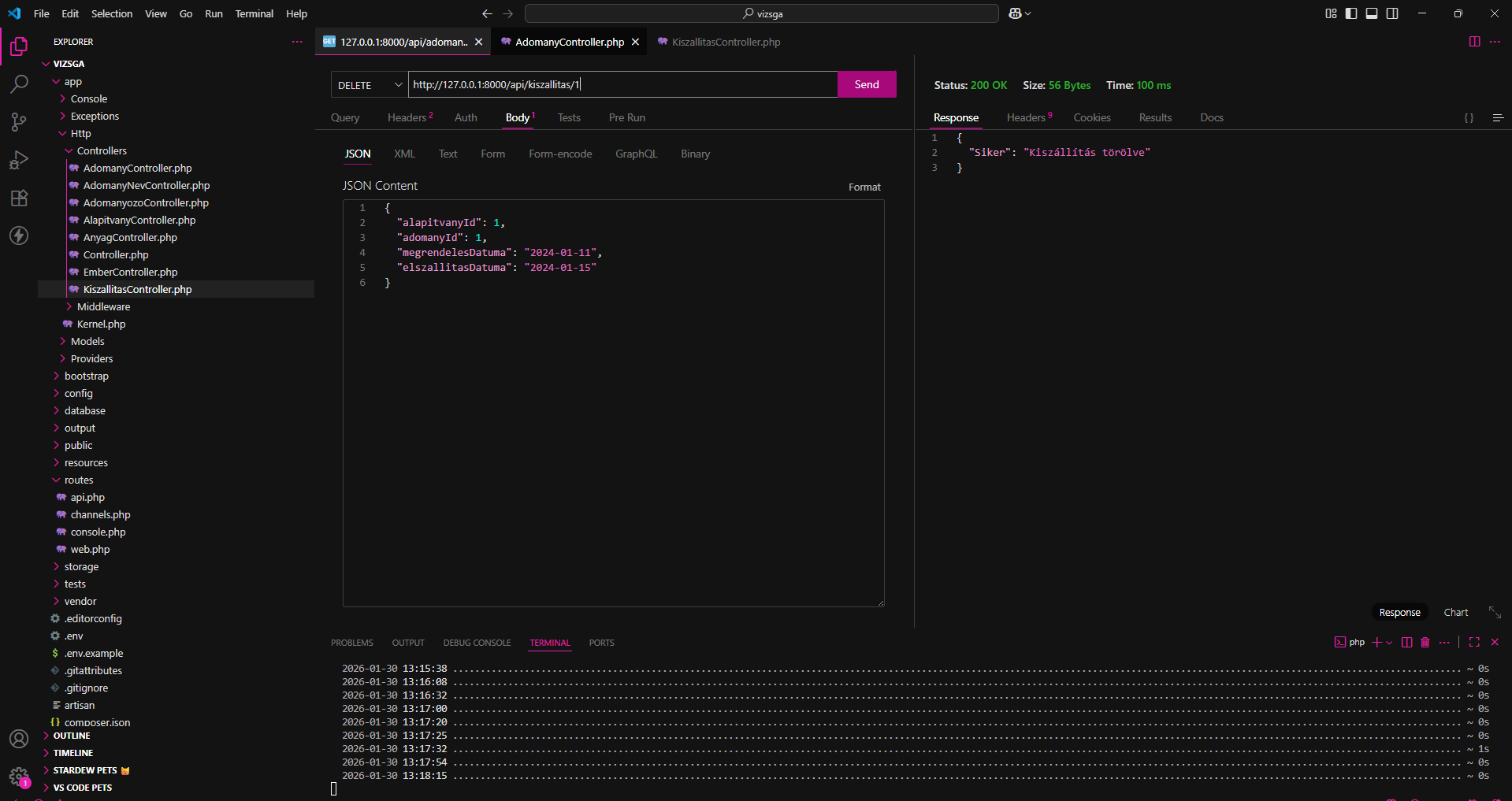
GET:

POST:

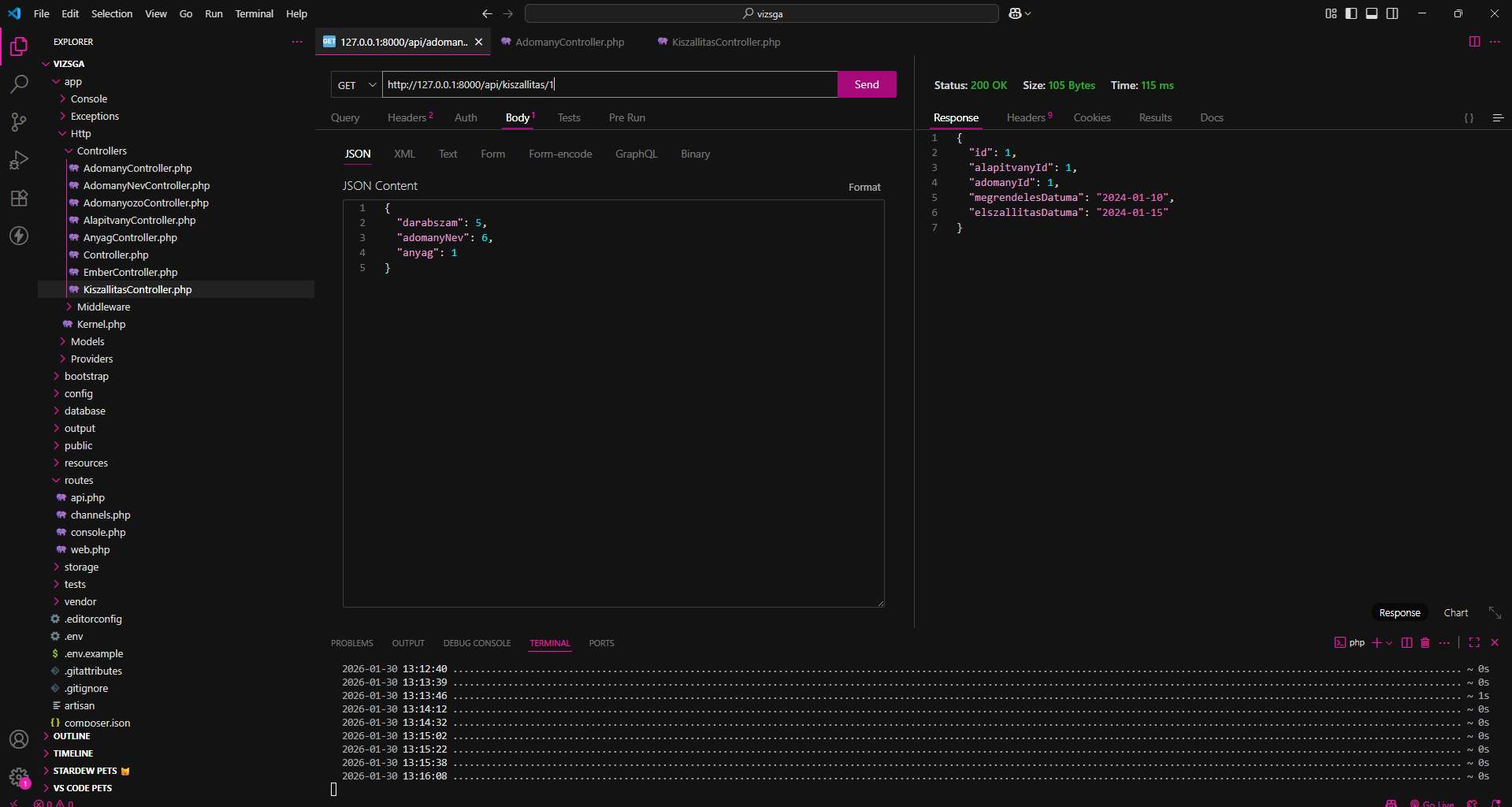


PUT:



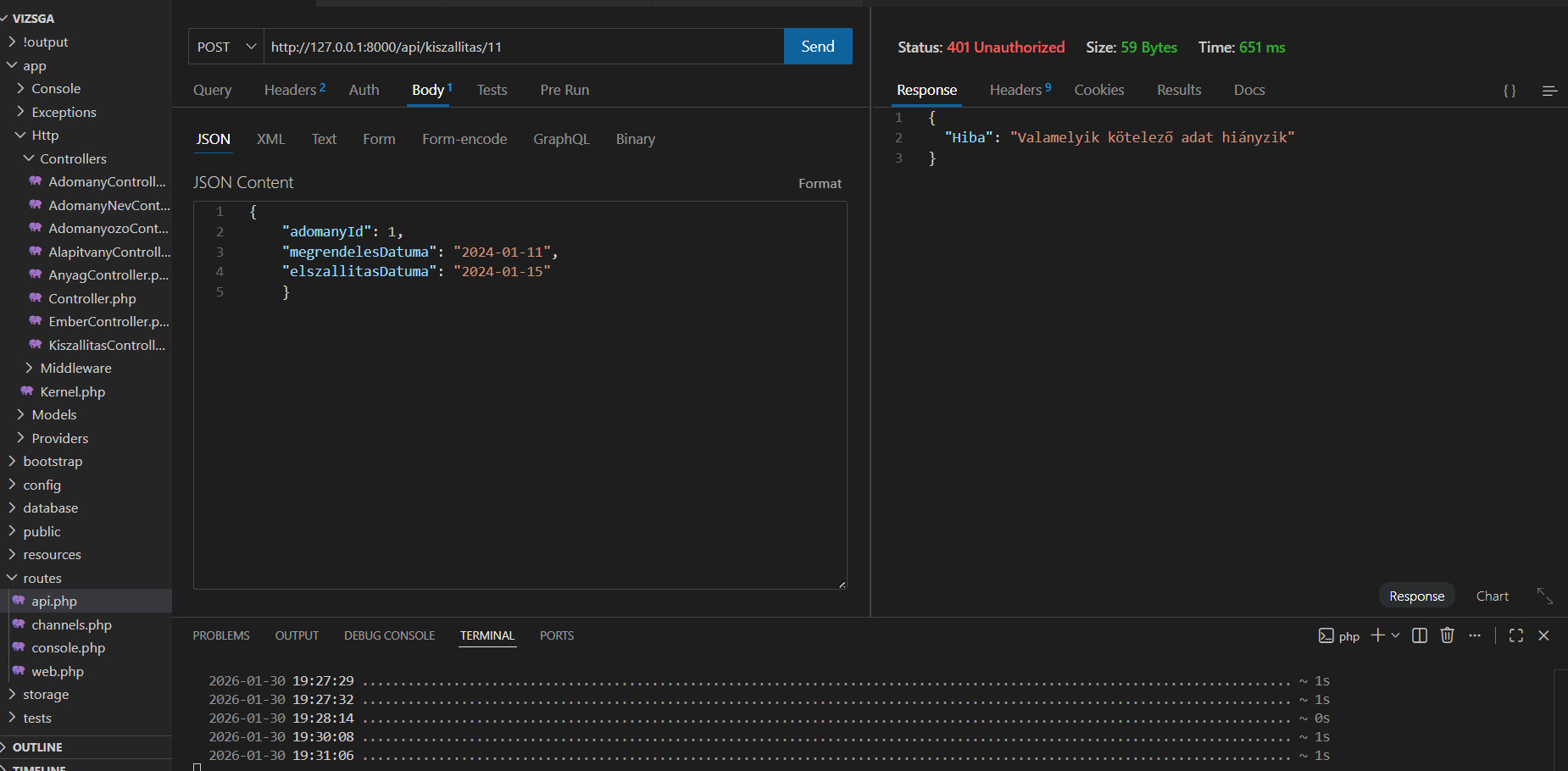
DELETE: 

GETBYID:

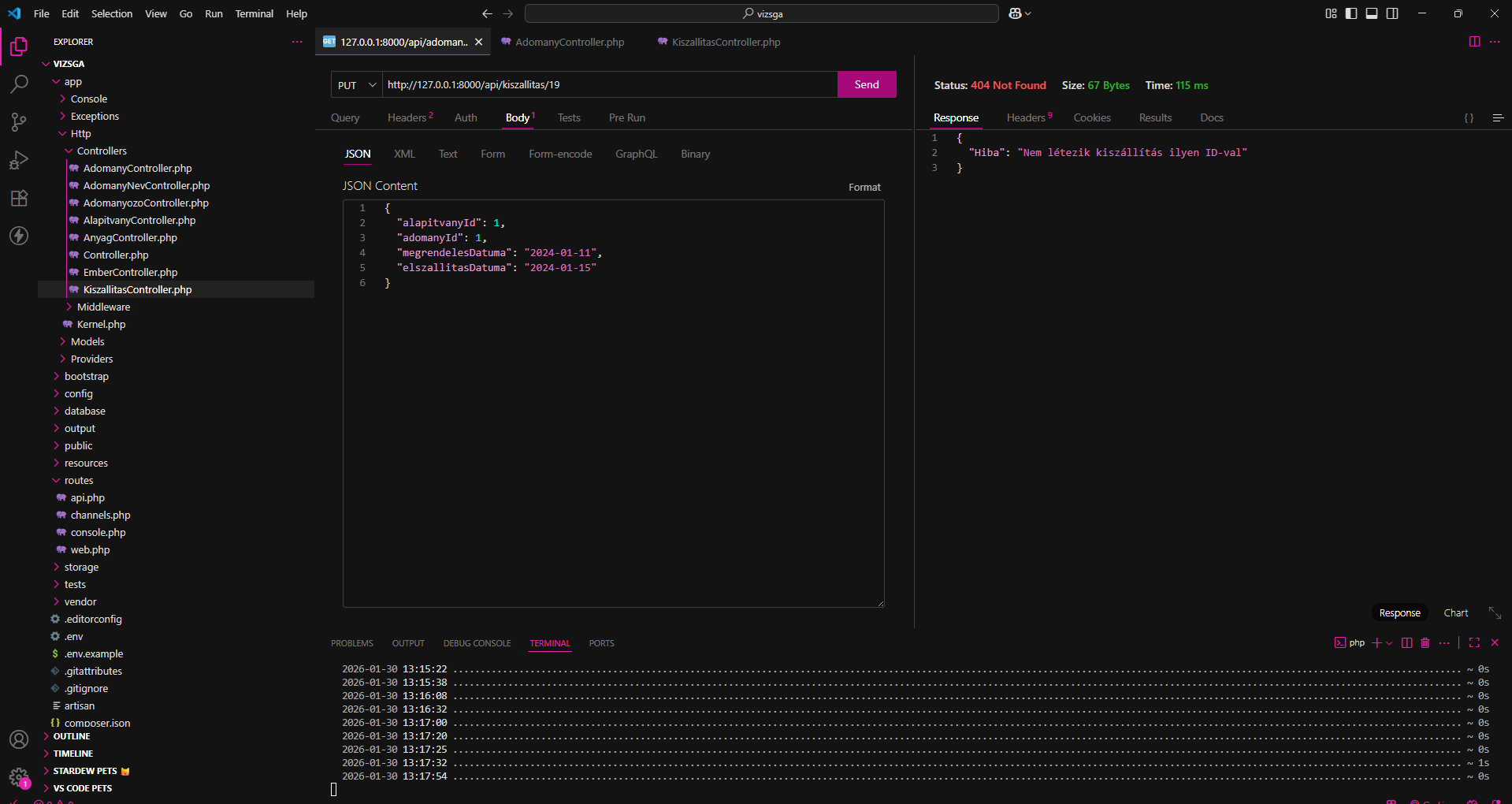


### Az alkalmazás sikertelen tesztjeinek képernyőfotói

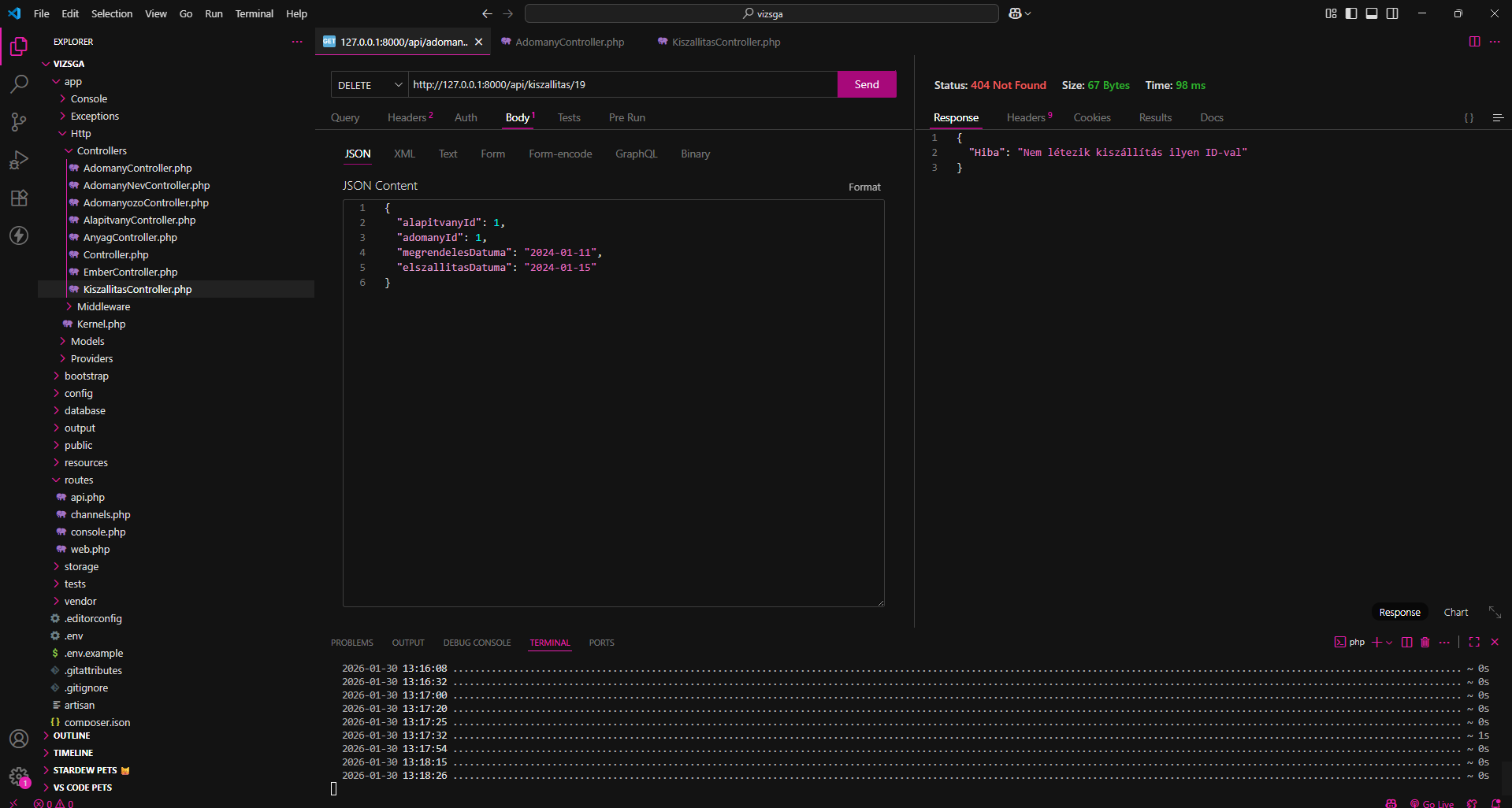
POST:



PUT:



DELETE:



GETBYID:

