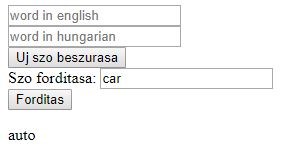
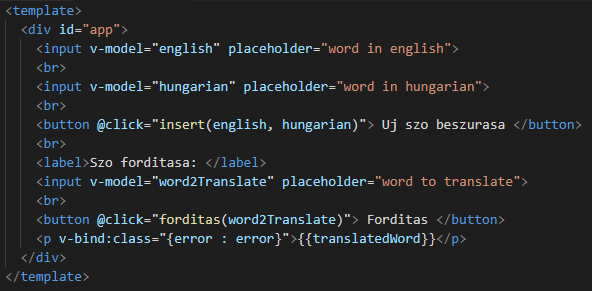
Dokumentáció a beadandó feladathoz

Az én feladatom egy fordító program elkészítése volt. A program nem túl bonyolult, megadhatunk szavakat, magyarul és angolul, ezeket beszúrhatjuk az adatbázisba, majd a már adatbázisban lévő szavakat lefordíthatjuk (ha beírjuk az adott szavat magyarul, kidobja az angol megfelelőjét, és fordítva), ha esetleg egy olyan szót adunk meg, ami még nincs az adatbázisban, akkor vörös betűkkel kiíródik, hogy nincs ilyen szó az adatbázisban.

A program elkészítéséhez az előadáson elkészített programot vettem alapul, majd azt alakítottam át. A képen látható a program. A legfelső mezőbe kell beírni angolul a szavat, az alatta lévőbe magyarul, majd klikk az Új szó beszúrása gombra, ez beviszi az adatbázisba a szavat. Ezek alatt van a szó fordítása mező, ide beírjuk a fordítandó szavat, majd a Fordítás gombra kattintva lefordíthatjuk az adott szót.

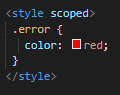
Frontend--template:

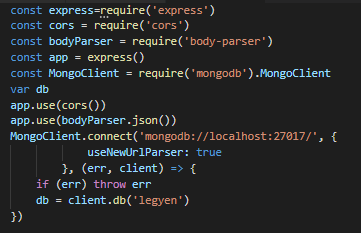
Itt található a html kód, ami megjelenítésre kerül az oldalon. Amint látható a textfieldekhez input tag-eket használtam, mivel ez tűnt a legegyszerűbbnek. A v-model attribútum tartalmazza magukat a szavakat, miután beírtuk őket, hogy később átadhassuk az insert, vagy forditas függvényeknek paraméterül. A gomboknak van egy @click esemény kezelőjük, mely meghívja a megfelelő függvényt. A placeholder tulajdonság szolgál arra, hogy a beviteli mezőkben kezdetben legyen egy tájékoztató jellegű szöveg. Ami még érdekes lehet az az utolsó paragrafus, aminek van egy css class-a, ami akkor aktiválódik, ha az error boolean változó igaz, hogy ez a változó mikor igaz, a későbbiekben láthatjuk.

Script:

A script tagek közt van az oldal logikája, vagyis hogy mi hogyan működik (függvények, adattagok, stb.). A data() tartalmazza a változókat: a szavak változóit, és az error-t. A methods property tartalmazza a metódusokat. Kezdjük az insert fügvénnyel, mivel az egyszerűbb. Paraméterül várja a szó párost, amit be kell szúrni az adatbázisba. Axios JavaScript függvénykönyvtár segítségével (mely HTTP kéréseket képes készíteni böngészőben és node.js környezetben is) egy beszúrási kérést küldünk a szerverre, mely a localhost:8082-es port-on fut, a paraméterül kapott szavakkal. Ez után az error változót hamisra állítjuk, a translatedWord változót pedig alaphelyzetbe állítjuk (ebben a változóban tároljuk a lefordított szót).

A fordítás metódus 1 paramétert vár, a szót, amit le kell fordítani. Axios segítségével újra egy kérést küldünk a szervernek, de most fordítási kérést. Ez után, ha a visszakapott adat 404-es hibakód, akkor az error boolean változót igazra állítjuk, kiíratjuk, hogy nincs ilyen szó és visszatérünk. Egyébként, ha nincs hiba, kiíratjuk a lefordított szót.

-Ez a kis kódrészlet pedig azért felel, hogy a hibaüzenetet pirosra színezzük.

Backend:

A fenti kódrészleten látszanak a változók melyekbe eltároljuk a felhasznált modulokat (importok).

Express: egy node.js szerver keretrendszer, ami egyszerűvé teszi a web applikációk készítését. Különböző HTTP kéréseket tud kezelni, pl: GET, PUT, POST és DELETE, nekünk ez miatt van rá szükségünk.

Cors (Cross Origin Resource Sharing): segítségével a szerver megadhatja, hogy ki férhet hozzá a szolgáltatásaihoz (erőforrásaihoz) és azt is, hogy hogyan lehet hozzáférni – mely HTTP kérések engedélyezettek külső forrásokból.

Body-Parser: HTTP post kérések kezelésére szolgál. Kicsomagolja a kérés body részét és betölti a req.body-ba.

MongoClient: MongoDB adatbázis kliens. Az adatbázissal kapcsolatos műveletek kezelésére szolgál.

Ezek után megadjuk az appunknak, hogy használja a corsot, bodyparsert, majd csatalakozunk az adatbázisra, mely a localhost:27017 porton fut, db változónkat beállítjuk a legyen adatbázisra.

A backend metódusai:

Az első post metódus az adatbázisba való beszúrás. A “szavak” kollekcióba beszúrjuk a kérés body részét, majd visszaküldünk egy 201-es status kódot, ami a hibátlan működést jelzi a kliensnek. A második metódus a fordításra szolgál. Először deklarálunk egy változót „actualWord” arra, hogy a kérésből kinyerjük a szót, amit fordítani kell. Ez után az adatbázisban a findOne metódussal (azért findOne, mert ha esetleg több van, akkor csak 1-et adjunk vissza) rákeresünk a szóra a magyar szavak közt. Az eredményt elmentjük a toAngolba. Ha volt ilyen szó, akkor az eredményt vissza is küldjük egyből (az angol megfelelőt). Viszont, ha toAngol null (!toAngol), akkor nem találtunk ilyen magyar szavat, ami jelentheti azt, hogy angolról akarunk magyarra fordítani. Ilyenkor az adatbázisban újra rákeresünk a szóra, immáron az angol szóra. Ha most is null a szó amit találtunk, vagyis nincs ilyen szó (!toMagyar), akkor visszaküldünk egy 404-es hibakódot. Ellenkező esetben pedig a szó magyar megfelelőjét.

-Végül ez a kis kódrészlet megadja, hogy melyik portot figyeli a szerver, az erre a portra érkező kéréseket kezeli.