NotNeptun Dokumentáció

Előszó

A program Apache XAMP-ot használ, mysql adatbázissal, nimble programozási nyelvben irva. Elméletileg teljesen cross platform (Windows, Linux, macOS), viszont Ubuntu alapó rendszeren való teszelés után kiderült egy openssl hiba, emiatt Windows rendszeren ajánlott a futtatása.

Inditás

Inditás lehetséges egyenesen a NotNeptun mappájában lévő bináris fájlal, de ha valamiért nem működne akkor szükségünk lesz nim-re és nimble-re: nim

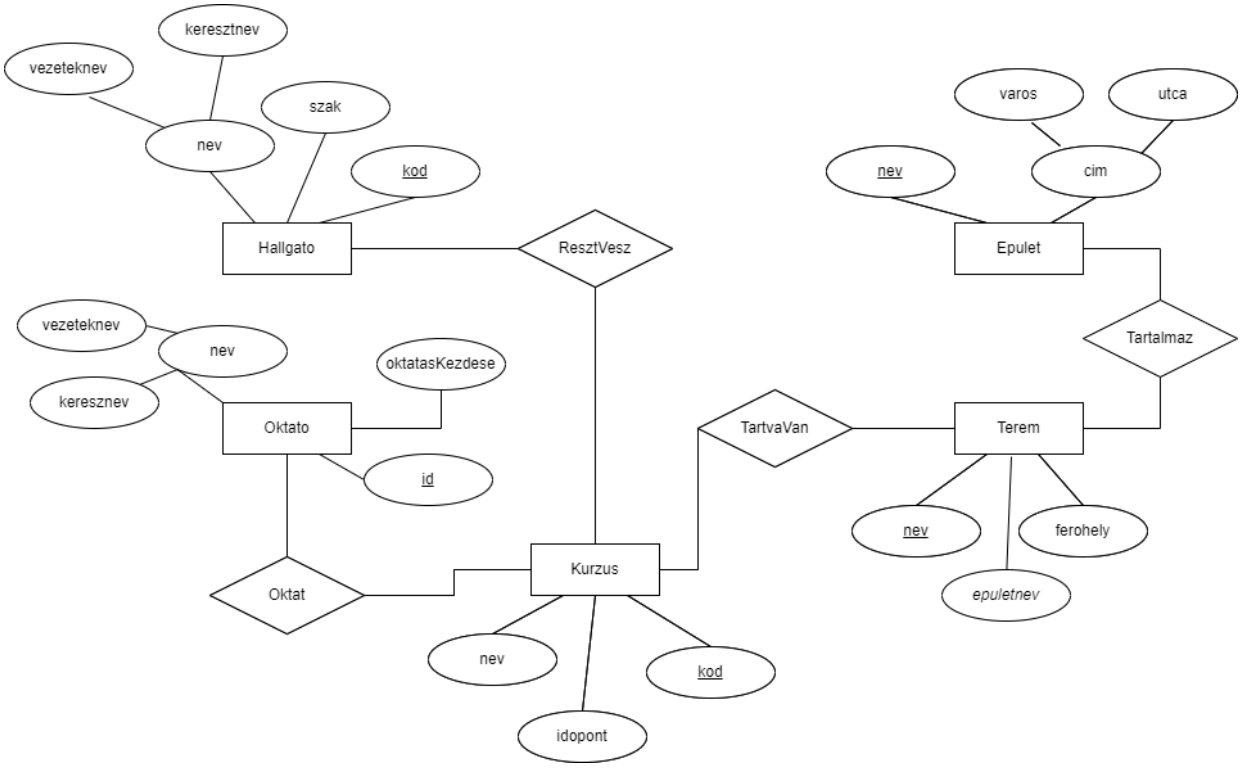
Emellett a fidget gui csomagra, amelyet `nimble install fidget` parancsal lehet telepiteni.

Forditani és inditani egyszerre a `nim c -r ./NotNeptun.nim -d:release --threads:on` paranccsal lehet. Utána pedig a létrehozott bináris fájlal.

Csatlakozni az adatbázisba a programon belül lehet. A kiexportált adatbázis alap adatai:

- host: localhost
- user: root
- password: root
- database: etr

Egyed-Kapcsolati Diagram



Táblák

Hallgato(kód, vezeteknev, keresztnev, szak)

- kód: a hallgató monogrammjából, a lista hosszából és véletlen számokból generált egyedi azonositó, legfeljebb 10 karakter, elsődleges kulcs

- vezeteknev: a hallgato vezetékneve, legfeljebb 255 karakter
- keresztnev: a hallgato keresztnneve, legfeljebb 255 karakter
- szak: hallgató szakja, legfeljebb 10 karakter

Oktato(kód, vezeteknev, keresztnev, kezdes)

- kód: az oktató monogrammjából, a lista hosszából és véletlen számokból generált egyedi azonositó, legfeljebb 10 karakter, elsődleges kulcs
- vezeteknev: az oktató vezetékneve, legfeljebb 255 karakter
- keresztnev: az oktató keresztnneve, legfeljebb 255 karakter
- szak: az oktató szakja, legfeljebb 10 karakter
- kezdes: mikor kezdett oktatni, évszám

Kurzus(kod, nev, idopont)

- kod: kurzus kódja, amely a nevének a röviditése, legfeljebb 10 karakter, elsődleges kulcs
- nev: kurzus neve, legfeljebb 50 karakter
- idopont: nap amikor van a kurzus, datetime

Terem(nev, ferohely)

- nev: terem neve, legfeljebb 255 karakter, elsődleges kulcs
- ferohely: terem befogadó képessége, 11 hosszú szám

Epulet(nev, varos, utca)

- nev: épület neve, legfeljebb 50 karakter, elsődleges kulcs
- varos: város ahol az épület található, legfeljebb 255 karakter
- utva: az utca a városban ahol található az épület, legfeljebb 255 karakter

ResztVesz(Hallgato.kod, Kurzus.kod): N:N kapcsolat
Oktat(Oktato.kod, Kurzus.kod) N:N kapcsolat
TartvaVan(Kurzus.kod, Terem.epuletnev) N:1 kapcsolat
Tartalmaz(epulet.nev, Terem.epuletnev) 1:N kapcsolat

Relációs Adatbáziséma

Hallgato(kód, vezeteknev, keresztnev, szak)
Oktato(kód, vezeteknev, keresztnev, kezdes)
Kurzus(kod, *terem.nev* ,nev, idopont)
Terem(nev, *epuletnev*, ferohely)
Epulet(nev, varos, utca)
ResztVesz(Hallgato.kod, Kurzus.kod)
Oktat(Oktato.kod, Kurzus.kod)

Funkcionális függőségek

Hallgato(kód, vezeteknev, keresztnev, szak)
{kod} -> { vezeteknev, keresztnev, szak }

Oktato(kód, vezeteknev, keresztnev, szak, kezdes)
{kod} -> { vezeteknev, keresztnev, szak, kezdes}

Kurzus(kod, *terem.nev* ,nev, idopont)
{kod} -> { terem.nev, nev, idopont }

Terem(nev, *epulet.nev*, ferohely)
{ nev } -> { epulet.nev, ferohely }

Epulet(nev, varos, utca)
{ nev } -> { varos, utca }

2NF és 3NF

Hallgato(kód, vezeteknev, keresztnev, szak)
Oktato(kód, vezeteknev, keresztnev, szak, kezdes)
Kurzus(kod, *terem.nev*, nev, idopont)
Terem(nev, *epulet.nev*, ferohely)
Epulet(nev, varos, utca)
ResztVesz(Hallgato.kod, Kurzus.kod)
Oktat(Oktato.kod, Kurzus.kod)

Használt lekérdezések:

- SELECT * FROM hallgato;
- SELECT * FROM terem;
- SELECT * FROM kurzus;
- SELECT * FROM epulet;
- SELECT * FROM oktato;
- SHOW TABLES FROM etr;
- SHOW COLUMNS FROM [table] IN etr; ahol table egy változó
- INSERT INTO [table] VALUES ...; ahol table a különböző táblák az adatbázisban, viszont felsorolásuk nagyon hosszú lenne
- DELETE FROM [table] WHERE hallgato.kod = [key]; ahol table a különböző táblák, a key pedig a kulcs ami alapján törlünk
- SELECT * FROM [table] WHERE hallgato.kod = [key]; ahol table a különböző táblák, a kulcs pedig ami alapján keresünk
- UPDATE [table] SET ... WHERE [key]; ahol table a tábla amit frissitünk, ... az adatok, key pedig a kulcs, ami alapján frissitünk

Megvalósitás

A szoftver nim-ben volt irva, és mivel a fidget ui library nem támogatja erősen az objektum orientált programozást, ezért teljesen imperativ módon van megirva.

Kettő fontos fájl van, a NotNeptun.nim és a db.nim. A NotNeptun a megjelenitésért és adatok felviteléért felel, a db pedig az adatbázis műveletekért.

Különböző funkciók, fájlonként:
NotNeptun.nim:

- viewHallgatok
  A hallgato tábla megjelenését biztositja
- viewKurzusok
  A kurzus tábla megjelenését biztositja

- viewTermek
  A terem tábla megjelenését biztositja
- viewEpuletek
  Az epulet tábla megjelenését biztositja
- viewOktatok
  Az oktato tábla megjelenését biztositja
- hozzaAd
  A táblákhoz való hozzáadást biztositja
- torles
  A táblákból való törlést biztositja
- modositas
  A táblákon a modositást biztositja
- connect
  Az adatbázis felé a csatlakozást továbbitja
- drawMain
  A meglenitésért felelős főleg, menű és háttér rajzolásáért
- startFidget
  Ez inditja el az egész applikációt, kommunikál a fidget gui könyvtárral

db.nim:

- listHallgato
  SQL üzenetet küld a szerver fele, nem vár adatot, visszadja az összes hallgató adatait
- listTermek
  SQL üzenetet küld a szerver fele, nem vár adatot, visszadja az összes terem adatait
- listKurzusok
  SQL üzenetet küld a szerver fele, nem vár adatot, visszadja az összes kurzus adatait
- listEpulet
  SQL üzenetet küld a szerver fele, nem vár adatot, visszadja az összes épület adatait
- listOktato
  SQL üzenetet küld a szerver fele, nem vár adatot, visszadja az összes oktató adatait
- getTableNames
  SQL üzenetet küld a szerver fele, nem vár adatot, visszadja az összes tábla nevét ami az adatbázisban van
- getColumnsTable
  SQL üzenetet küld a szerver fele, egy tábla nevet vár, visszadja az összes oszlop nevét egy táblában
- insertData
  SQL üzenetet küld a szerver fele, egy tábla nevet és egy kulcsot vár, beilleszt adatot egy táblába
- deleteData
  SQL üzenetet küld a szerver fele és töröl egy adott adatot a táblába
- changeData
  SQL üzenetet küld a szerver fele, egy tábla nevet, egy kulcsot és az új adatokat várja, megváltoztat egy adott adatot a táblába
- searchKey
  SQL üzenetet küld a szerver fele, egy tábla nevet és egy kulcsot vár, visszadja azt a sort amely

megegyezik a várt kulccsal

- dbConn

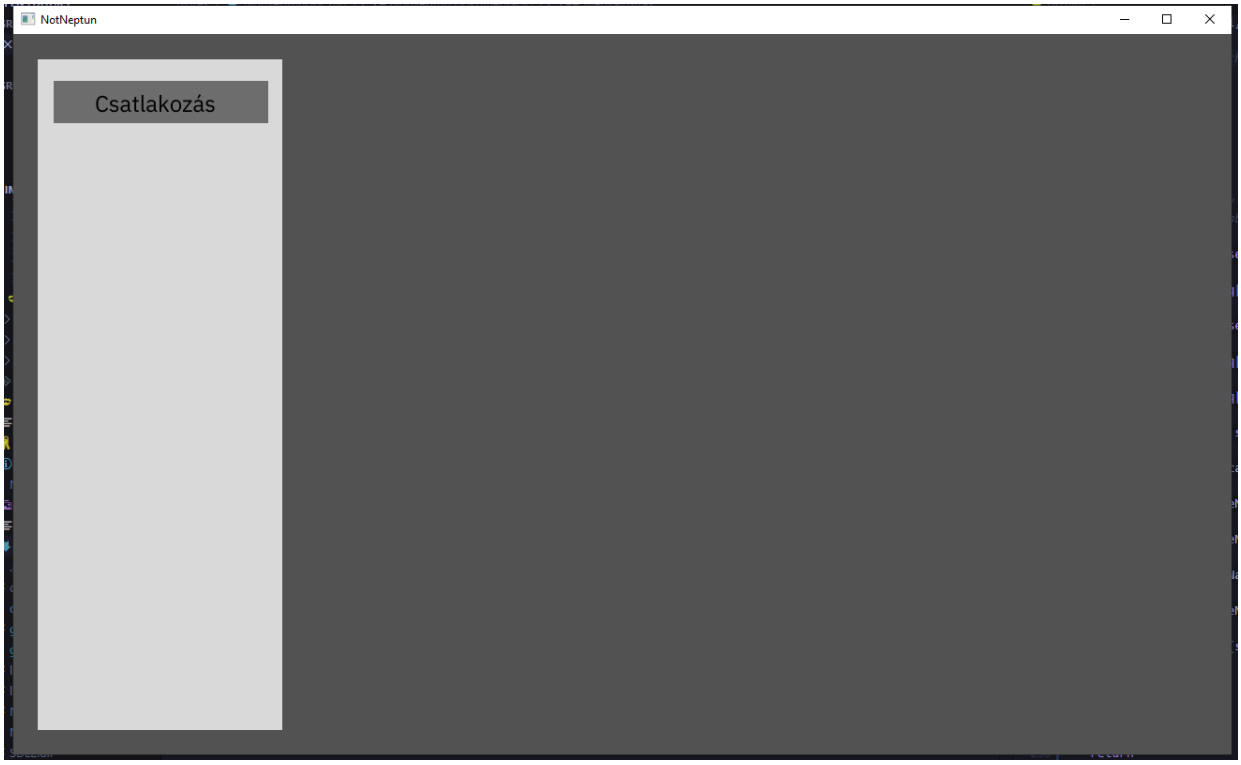
  A szerverhez való csatlakozási adatokat állitja be, a csatlakozás adatait várja

Funkciók

A programban különböző menűkön keresztül lehetőségünk van csatlakozni az adatbázishoz, adatokat megnézni és módositani. Ezekre a képernyőképeket lentebb láthatjuk.

Felhasználói útmutató

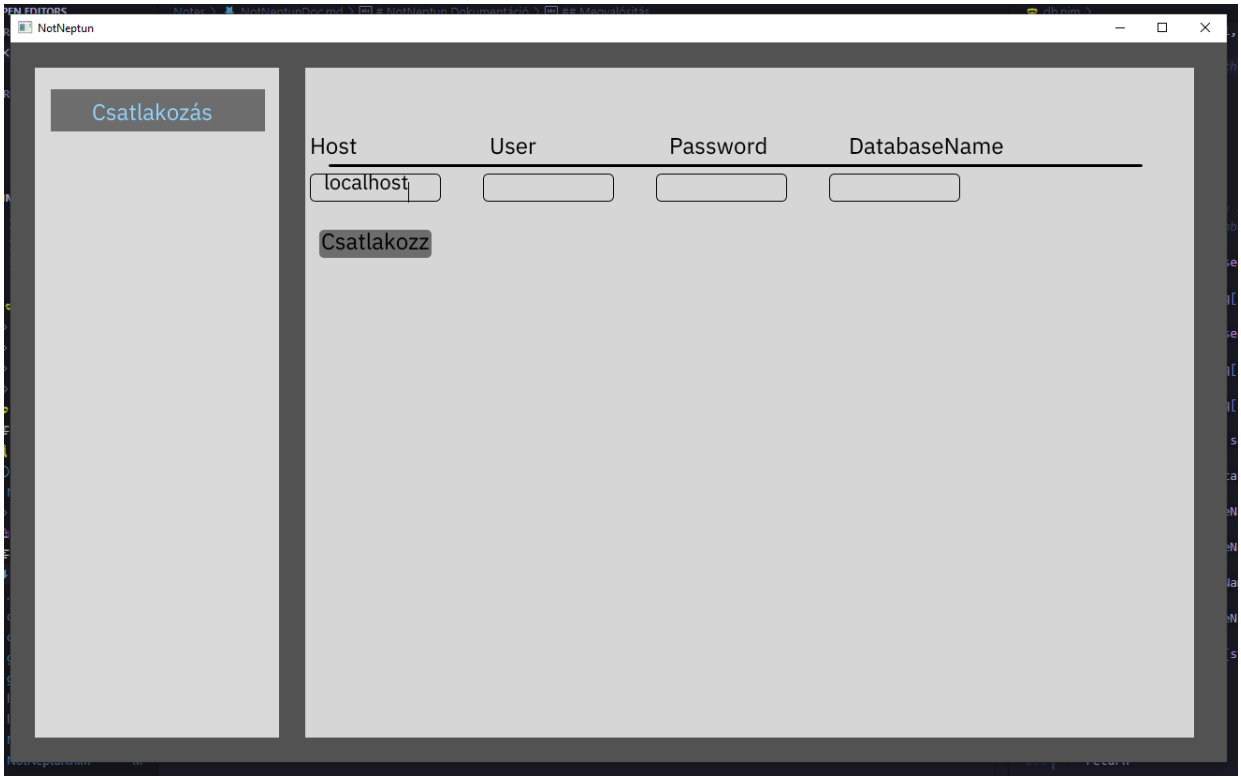
Miután elinditottuk a programot, csakis egy lehetőségünk lesz, kiválasztani az adatbázishoz való csatlakozás menűt:



Onnan tudjuk hogy kiválasztottunk egy menűt, hogy kékre fog váltani a betű szine a gombban.

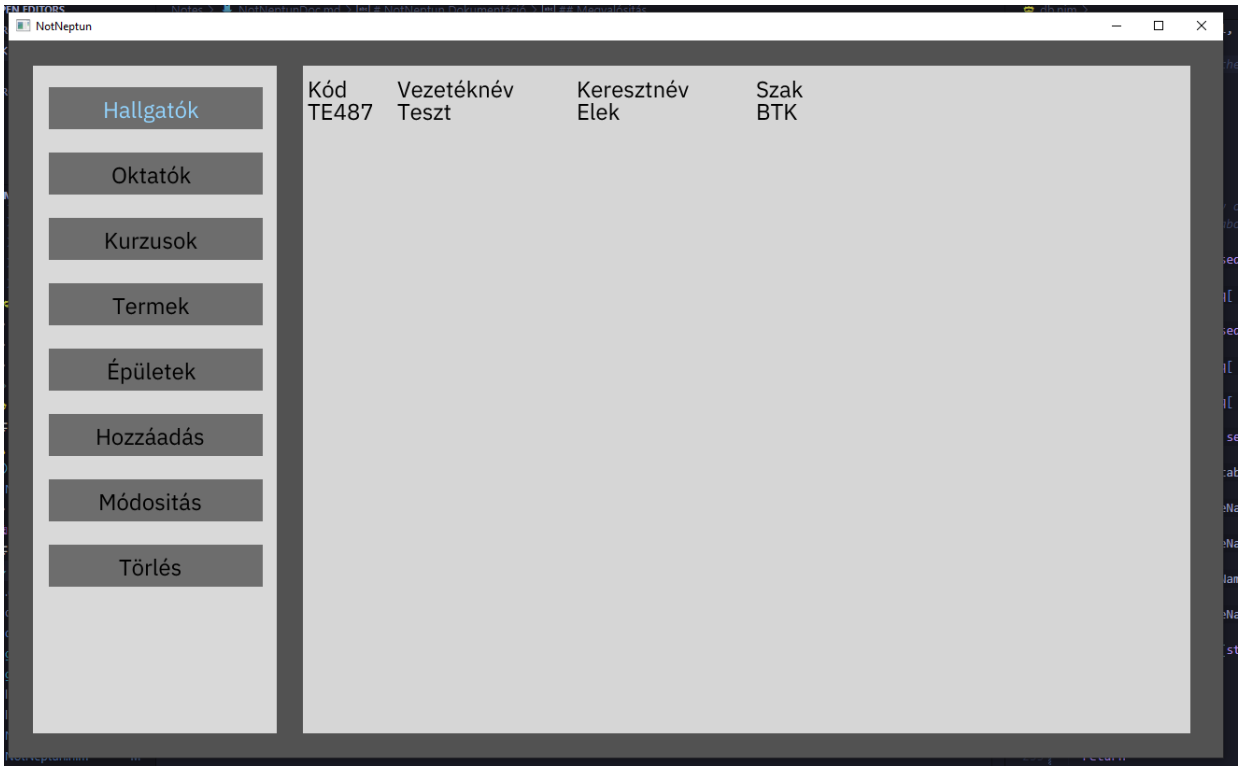
Ha rámentünk a csatlakozás menűre akkor meg kell adni az adatbázishoz való csatlakozási adatokat. Ezek után a csatlakozás gombra kattintva tovább tudunk lépni. A mezők **nem** lehetnek

üresek!

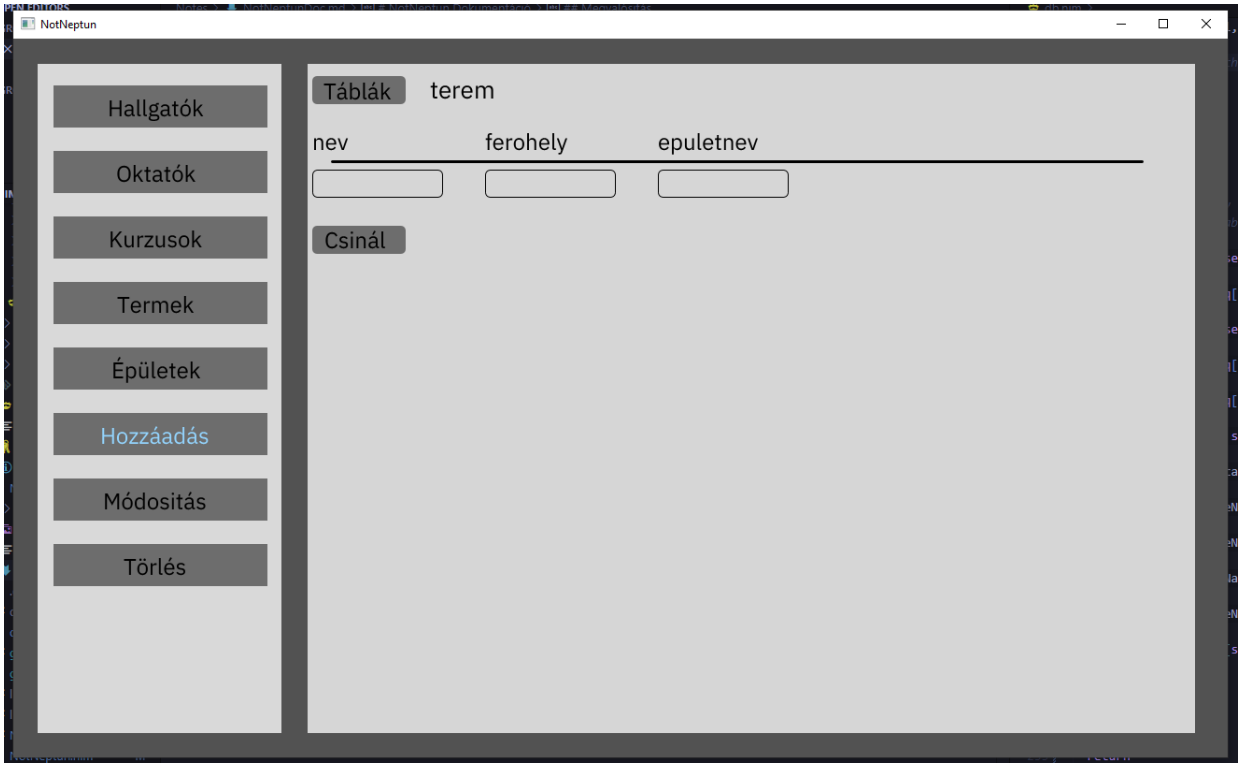


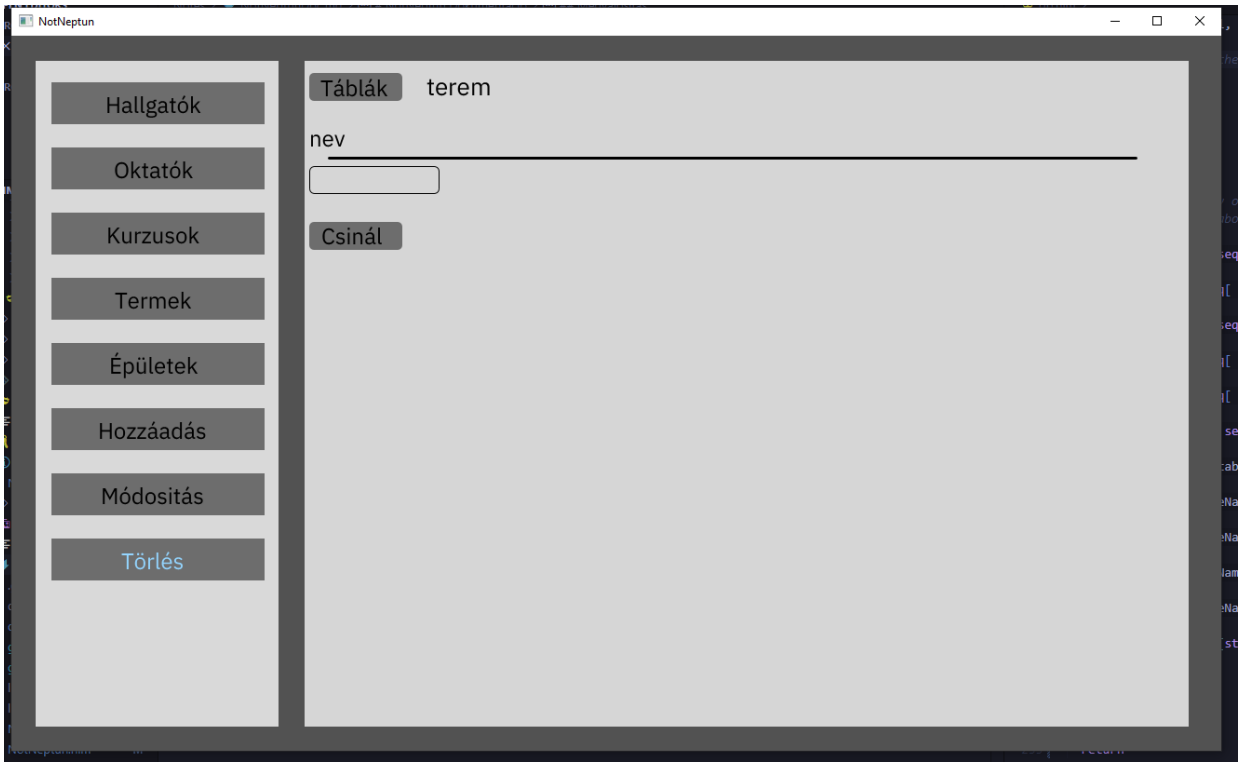
Továbblépésnél a főmenűben találjuk magunkat. Itt egy opciót kiválasztva léphetünk tovább különböző részekre.

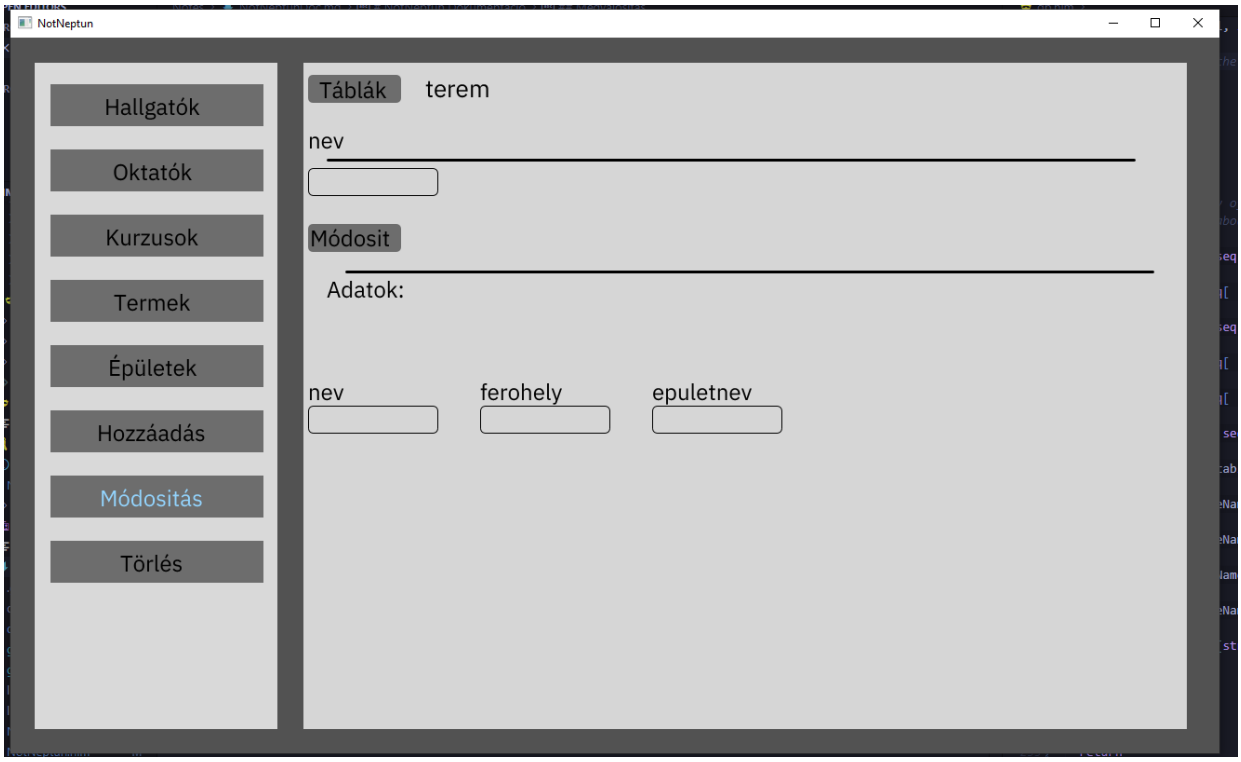
Ha kiválasztottunk egy menűt akkor meg fog jelenni a hozzá tartozó funkciók, példa az adatok megjelnitésére és a különböző funkciókra:



A különböző adat manipulációs mezőknél ki kell választani egy leeső menün keresztül a megváltoztatni kivánt táblát, fel kell vinni az kért adatokat, majd a gomb megnyomására az adatok változtatásra kerülnek, amiket a listázó menűkben megtekinthetünk.




NotNeptun
Hallgatók
Oktatók
Kurzusok
Termek
Épületek
Hozzáadás
Módositás
Törlés
Táblák
terem
nev
Csinál


NotNeptun
Hallgatók
Oktatók
Kurzusok
Termek
Épületek
Hozzáadás
Módositás
Törlés
Táblák
terem
nev
Módosit
Adatok:
nev
ferohely
epuletnev