Eötvös Lóránd Tudomány Egyetem

Informatika Kar

Programozási Nyelvek és Fordítóprogramok Tanszék

Tanulási keretrendszer a C++ programozáshoz

Témavezető:

**Pataki Norbert**

Adjunktus, PhD

Szerző:

**Török Richárd Dávid**

Programtervező informatikus, Bsc

Budapest, 2020

Tartalom:

1. Bevezetés
   1. Témaválasztás indoka
   2. Megoldandó feladat
2. Felhasználói dokumentáció
   1. Rendszerkövetelmények
   2. Telepítés
3. Fejlesztői dokumentáció
4. Összefoglalás
5. Hivatkozások

1. Bevezetés

1.1 Témaválasztás indoka

A koronavírus okozta helyzet rámutatott, hogy mekkora szükségünk van az online elérhető tananyagokra. A diplomamunkám célja, hogy egy olyan környezetet biztosítson a c++ programozási nyelvnek, hogy azt könnyen fogyasztható formában lehessen elsajátítani. A leckékhez feladatok is tartoznak, így a felhasználó egyből tudja gyakorolni a friss tudást. A leckék különböző nehézségi szintek szerint vannak beosztva, így a felhasználó a saját képességeihez mérten választhat a tananyagokból. A leckék egy diasor szerű felületen vannak prezentálva, amik között a felhasználó a navigációs gombok segítségével léphet. Az alkalmazás tartalmaz egy online fordítót is, amivel a lecke tanulása alatt kipróbálhatjuk az új ismereteinket, illetve a programozási feladatokban próbára tehetjük a tudásunk. Ennek működéséről a dokumentáció későbbi szakaszában lesz szó.

1.2 Megoldandó feladat

A projekt célja, hogy a végfelhasználónak ne kelljen semmilyen extra programot telepítenie. Egy modern böngészőre legyen szükség mindössze. A kódfordításohoz nem kell hogy c++ fordítóprogram legyen az eszközünkön, ez mind szerver oldalon történik. A kód fordítása és futtatása egy külön docker konténerben történik, szabályozható életidővel. Így ki tudjuk küszöbölni, hogy esetleges káros kód fusson le a szerveren és az kártékony hatásokat hajtson végre a szerver rendszerében. Az oldal irodalmi tartalmát egy headless cms (content management system) szolgáltatja. Ennek tartalma dinamikusan változtatható egy külön felhasználói felületen, ahol akár több szerkesztő felhasználót is létrehozhatunk. Itt tudunk új leckéket és feladatokat létrehozni. Ez a rendszer tartalmaz egy publikációs megoldást is, ami segítséget nyújt hogy csak az a tartalom jelenjen meg az alkalmazásban aminek már végeztünk a szerkesztésével.

2. Felhasználói dokumentáció

A teljes alkalmazás dockerizálva van, így ha lokálisan szeretnénk futtatni, nincs szükségünk nodeJS-re, vagy egyéb rendszerekre. Két alkamazást kell telepítenünk, [Docker Desktop](https://www.docker.com/products/docker-desktop), illetve [Postman](https://www.postman.com/downloads/). A docker biztosítja már azt a lehetőséget is hogy a virtualizációhoz Windows-os konténereket használjuk, de válasszuk a Linux alapú konténereket mert ezzel lett fejlesztve az alkalmazás, és nem várt problémák adódhatnak ellenkező esetben. A Postman az adatbázis inicializálásához lesz szükséges.

2.1 Rendszerkövetelmények

Docker Desktop for Windows esetén:

* Windows 10 (64-bit) Pro, Enterprise, vagy Education:
  + Build 16299 vagy nagyobb
  + Hyper-V és a Containers Windows funkció engedélyezve kell hogy legyen.
  + Hardveres követelmények:
    - 64 bit-es processzor SLAT támogatással
    - 4GB memória
    - BIOS-szintű hardwares virtualizáció engedélyezése, [bővebben](https://docs.docker.com/docker-for-windows/troubleshoot/#virtualization-must-be-enabled)
* Windows 10 (64-bit) Home:
  + 1903-as vagy nagyobb verzió
  + WSL 2 funkciók engedélyezése, [dokumentáció](https://docs.microsoft.com/en-us/windows/wsl/install-win10)
  + Hardveres követelmények:
    - 64 bit-es processzor SLAT támogatással
    - 4GB memória
    - BIOS-szintű hardwares virtualizáció engedélyezése, [bővebben](https://docs.docker.com/docker-for-windows/troubleshoot/#virtualization-must-be-enabled)

Illetve egy modern böngésző, az alábbi verzó számokkal vagy nagyobbal:

* Chrome (86.0.4240.198)
* Edge (87.0.664.55)
* Firefox (82.0.2)

2.2 Telepítés

Miután sikeresen feltelepítettük a Docker Desktop és Postman alkalmazást, a projektet a következő képpen tudjuk elindítani.

2.2.1 Első indítás

Az elsőindítás kicsit bonyolultabb, mivel inicializálnunk kell az adatbázist. Az adatbázis volume-ok segítségével perzisztálva van, így élettartama a konténerétől független. Tehát későbbi indításokkor nem kell a következő lépést végrehajtsuk.

Lépések:

* Service-ek elindítása és adatbázis inicializálása
  + docker-compose up --scale compiler=0 –d
  + docker-compose run --rm --entrypoint "/bin/sh -c" directus "/var/directus/bin/directus install:database"
* Admin felhasználó létrehozása
  + docker-compose run --rm --entrypoint "/bin/sh -c" directus "/var/directus/bin/directus install:install -e <email> -p <password> -t C++ -T <access-token>
  + Itt az <email> és <password> helyére tetszőleges értékeket írhatunk be.
  + Az <access-token> helyére diploma-2020 értéket írjunk, lentebb olvasható mi a tehendő ha más értéket szeretnénk

Saját access-token érték esetén:

Itt az access-token helyébe az egyszerűség kedvéért a diploma-2020 értéket írjuk. Ha mást szeretnénk megadni access token-nek, akkor a ./frontend/.env file-ban is átt kell írnunk a REACT\_APP\_ACCESS\_TOKEN kúlcs értékét, és újra buildelni a frontendet a docker-compose build frontend paranccsal, majd újraindítani a service-t, docker-compose restart frontend.

2.2.2 Már inicializált adatbázissal