



Eötvös Loránd Tudományegyetem

Informatikai Kar

Információs Rendszerek Tanszék

Többnyelvű szótár

Gombos Gergő
adjunktus

Réti Marcell
Programtervező Informatikus BSc

Budapest, 2019

Tartalomjegyzék

1. Bevezetés	1
1.1. A feladat leírása	1
1.2. A témaválasztás indoklása	2
2. Felhasználói dokumentáció	3
2.1. A program rövid megfogalmazása	3
2.2. Rendszerkövetelmények	3
2.3. Szükséges telepítések	3
2.4. A program futtatása és használata	5
2.4.1. Bejelentkezés és regisztráció	5
2.4.2. Szótár	6
2.4.3. Kategóriák	8
2.4.4. Profil	8
2.4.5. Szintlépés	9
2.4.6. Kihívások	9
2.4.7. Segítségek	10
2.4.8. Kikérdezés	10
2.4.9. Kiértékelés	13
3. Fejlesztői dokumentáció	15
3.1. Az alkalmazás felépítése	15
3.2. A back-end architektúrája	16
3.2.1. A webserver	16
3.2.2. A back-end alapja	18
3.2.3. Globális objektumok	19
3.2.4. Felhasználói adatobjektumok	20
3.2.5. A kikérdezés előkészítése	23
3.2.6. A kikérdezés kielemezése	27
3.3. A front-end architektúrája	28
3.3.1. A Szótár keret	29
3.3.2. A Kategória keret	30

3.3.3. A Kikérdezés keret	31
3.3.4. A Profil keret	33
3.4. Tesztelés	34
3.4.1. Lintelés	34
3.4.2. Felhasználói történetek	35
4. Összegzés	37
5. Irodalomjegyzék	38

1. fejezet

Bevezetés

1.1. A feladat leírása

Ezen dokumentáció egy asztali alkalmazás felhasználói- és fejlesztői dokumentációját tartalmazza.

A program egy olyan többnyelvű szótárfüzetet valósít meg, ahol külön felhasználók külön fiókokon tudják nyomon követni a tanult szavaikat, melyeket a beépített kikérdező funkcióval gyakorolhatnak és tesztelhetnek.

Egy felhasználó felvihet új szavakat különböző kategóriákkal ellátva, módosíthatja és törölheti ezeket. Új nyelv felvételével dinamikusan bővül a szótár is, így nincs szükség újra felvenni már meglévő szavakat.

A kikérdezés módban a felhasználó próbára teheti tudását különböző funkciókkal, mint például kategória/nyelv szerinti szűrés, könnyített és nehezített módok. A kikérdezés eredményeit a program elmenti, hogy azokból rövid és hosszútávú eredményeket tudjon generálni.

Minden felhasználó különféle szintjelzőkkel tudja nyomon követni haladását – külön szintjelző mutatja az egész és a nyelvekre lebontott haladást. Minden szónak minden nyelven külön van egy szint mutató, amit a felhasználó nem lát – ezzel mérlegelve hogy milyen gyakran jelenjen meg kikérdezésnél.

A felhasználók kikérdezés során különböző segítségeket használhatnak fel elakadásnál, ezeket két helyről szerezhetik: szintlépésenként vagy napi/heti kihívásokból. A kihívások egy nagy méretű halmazból vannak kiválasztva véletlenszerűen, amik különböző specifikus feladatot adnak a felhasználóknak, ezzel sarkallva őket a folyamatos használatra, és nehézségük és jutalmuk a kihívás jelzőjétől függ.

1.2. A témaválasztás indoklása

A témát három barátom ihlette. Mindannyiuk egyszerre tanul több nyelvet, és szerettek volna egy olyan alkalmazást, amivel a kigyűjtött szavaikat effektíven tudják tárolni, és gyakorolni. Hosszútávú terveim közé tartozik nekem is egy harmadik nyelv, így a későbbiekben számomra is hasznosnak fog bizonyosulni.

Kezdetől fogva Python-ban szerettem volna programozni. A szakmai gyakorlatom során lehetőségem volt megbarátkozni vele, és szerettem volna jobban belemélyedni a nyelv trükkjeibe - ötvözni az objektum-orientált és az imperatív programozási paradigmákat, amihez a Python egy ideális nyelv. Grafikus felhasználói felület fejlesztésével eddig nem foglalkoztam, ezért ideálisnak éreztem, hogy a Python de facto GUI framework segítségével, a Tkinter-rel próbáljam fejleszteni ezen a téren a tudásomat.

2. fejezet

Felhasználói dokumentáció

2.1. A program rövid megfogalmazása

A Többnyelvű szótár egy olyan szótárfüzetet megvalósító asztali alkalmazás, amely azon emberek részét célozza meg, akik egyszerre több idegen nyelvet is szeretnének tanulni párhuzamosan. A szótár bármennyi nyelvet képes tárolni, ahova egyénileg vihetünk fel szavakat, melyeket egyéni kategóriákkal láthatunk el tetszés szerint. A program fő funkciója a kikérdezés, mellyel gyakorolni lehet a felvett lexémákat, amik konzisztens sikeres megválaszolás esetén kevésbé fognak előkerülni, így fókuszot állítva a kevésbé megtanult szavakra. A program ösztönözve a haladást és folyamatos használatot, szintekkel látja el a felhasználót és annak nyelveit, és napi kihívásokkal ösztönzi a legalább napi használatot.

2.2. Rendszerkövetelmények

A program támogatott operációs rendszere a **Windows 7**, **Windows 8** és **Windows 10**. Linux és MacOS rendszereken az elvárt működés nincsen garantálva, mivel a grafikus interface beállításai a Windows Tcl parancsaihoz lettek szabva. A program teljes megjelenítéséhez minimum 1200x600-as felbontású monitor kell.

A futtatáshoz **Python 3.6.X**[1] verzióra van szükség. Az online adatbázishoz folyamatos internetelérés, lokális adatbázis esetén pedig egyéni **MariaDB**[2] adatbázis szükséges.

2.3. Szükséges telepítések

A **Python** letöltését a "<https://www.python.org/downloads/release/python-363/>" linken tehetjük meg, az ajánlott és demonstrált módszer a futtatható telepítő (executable installer), melyet a 2.1. ábrán láthatunk pirossal bekeretezve.

Files					
Version	Operating System	Description	MD5 Sum	File Size	GPB
Gzipped source tarball	Source release		e9180c69ed9a878a4a8a3ab221e32fa9	22673115	SIG
XZ compressed source tarball	Source release		b9c2c36c33fb89bda1fed37ad5af9be	16974296	SIG
Mac OS X 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	ce31f17c952c657244a5cd0ccae34ad	27696231	SIG
Windows help file	Windows		a82270d7193f9fb8554687e7ca342df1	8020197	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64, not Itanium processors	b1daa2a41589d7504117991104b96fe5	7145844	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64, not Itanium processors	89044fb577636803bf49f36371dca09c	31619840	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64, not Itanium processors	b6d61642327f25a5ebd1a7f11a6d3707	1312480	SIG
Windows x86 embeddable zip file	Windows		cf1c75ad7ccf9dec57ba7269198fd56b	6388018	SIG
Windows x86 executable installer	Windows		3811c6d3203358e0c6b6677ae980d3	30584520	SIG
Windows x86 web-based installer	Windows		39c2879cecf252d4c935e4f8c3087aa2	1287056	SIG

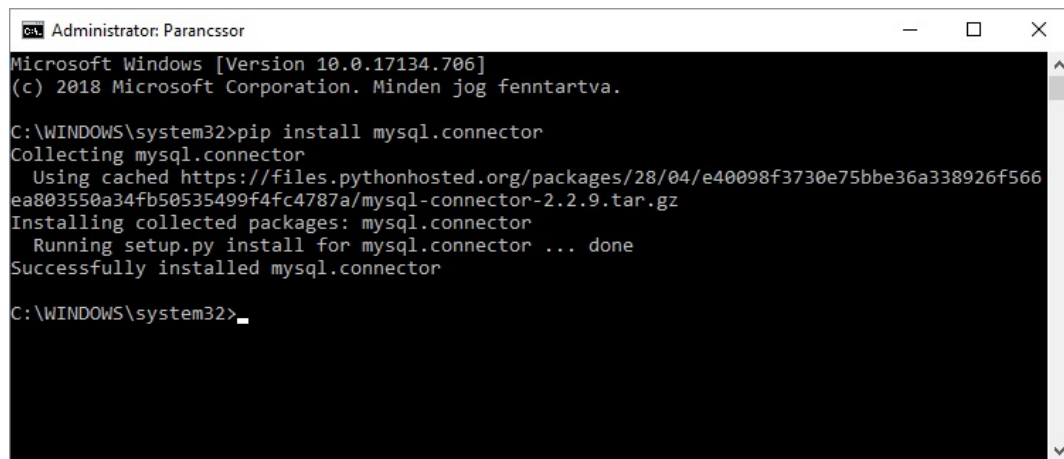
2.1. ábra. A Python letöltési fájlljai

A letöltött fájl elindításával a telepítő első oldalán figyeljük arra, hogy az "Add Python 3.6 to PATH" opció be legyen pipálva, melyet a 2.2. ábrán láthatunk (ehhez rendszergazdai engedély szükséges). Ezek után egy egyszerű telepítési folyamat után végeztünk.



2.2. ábra. A Python telepítő első oldala

Szükséges feltelepítenünk még a **MySQL.Connector**[3] könyvtárat is a Python-on belül, mivel az nem egy alapértelmezett framework. Ehhez rendszergazdaként indítsunk egy parancssort, majd pedig futtassuk a "pip install mysql.connector" parancsot. A 2.3.-mas ábrán láthatjuk milyen a sikeres telepítés.



```
Administrator: Parancssor
Microsoft Windows [Version 10.0.17134.706]
(c) 2018 Microsoft Corporation. Minden jog fenntartva.

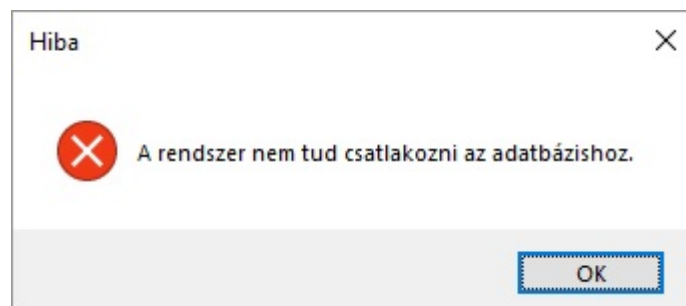
C:\WINDOWS\system32>pip install mysql.connector
Collecting mysql.connector
  Using cached https://files.pythonhosted.org/packages/28/04/e40098f3730e75bbe36a338926f566ea803550a34fb50535499f4fc4787a/mysql-connector-2.2.9.tar.gz
Installing collected packages: mysql.connector
  Running setup.py install for mysql.connector ... done
Successfully installed mysql.connector

C:\WINDOWS\system32>
```

2.3. ábra. A mysql.connector telepítése

2.4. A program futtatása és használata

Indításhoz a mappaszerkezet gyökerében lévő MultilanguageDictionary.exe fájl megnyitására van szükség. Ha lokális adatbázist használunk, indítsuk el a program előtt. Ha a megadott adatbázist használjuk, akkor internet kapcsolat szükséges, ellenben az applikáció nem tud elindulni, melyről értesíti a felhasználót egy felugró ablakban.

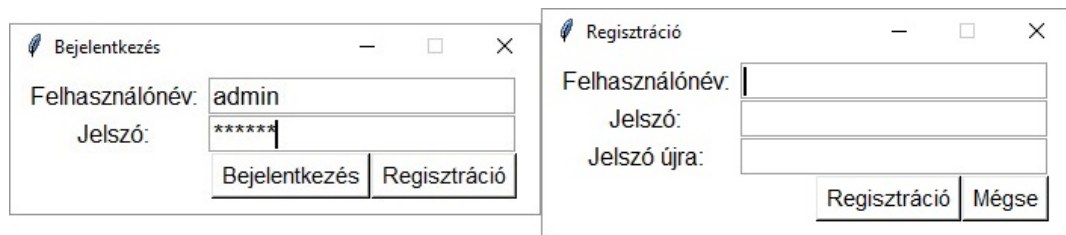


2.4. ábra. Adatbázis kapcsolat meghiúsulása

2.4.1. Bejelentkezés és regisztráció

A program elindulásakor a bejelentkezés ablak jelenik meg előttünk. Első futtatásnál készítenünk kell egy felhasználót, amit a jobb alsó sarokban lévő "Regisztráció" gombbal előhozott oldalon tehetünk meg. A regisztrációhoz minden mezőt ki kell tölteni, a felhasználónév és jelszó az angol ABC kis- és nagybetűit és számokat tartalmazhatja, és a két jelszónak meg kell egyeznie.

A regisztrációt követve, vagy már meglévő felhasználó esetén helyes bejelentkezési adatpáros esetén a "Bejelentkezés" gombra kattintva léphetünk be a Többnyelvű szótárba.

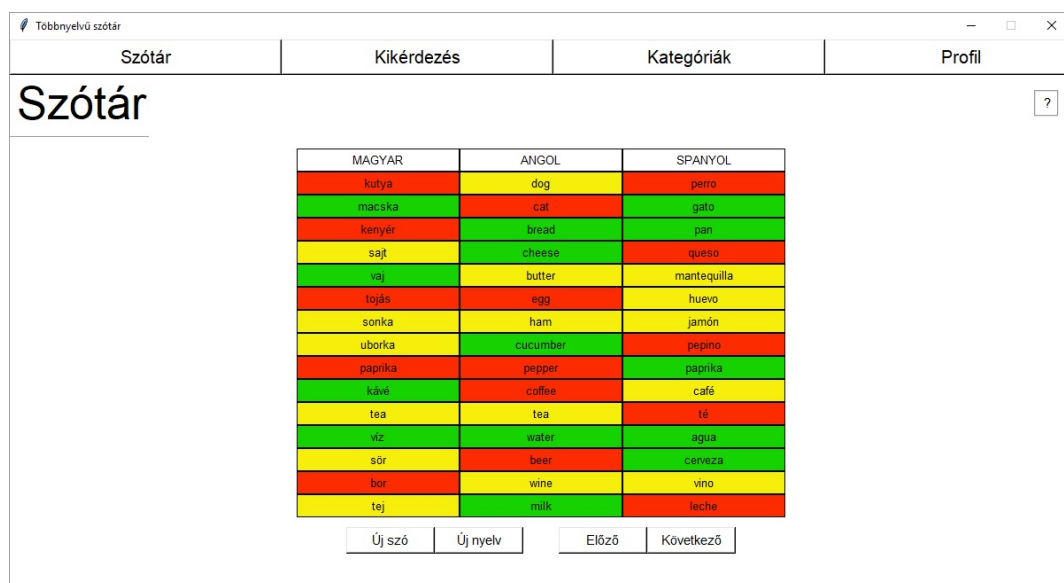


2.5. ábra. A bejelentkezés és a regisztráció ablak

2.4.2. Szótár

Sikeres bejelentkezés után vagy a felső menüpontból a "Szótárt" kiválasztva a Szótár földre kerülünk. Itt láthatjuk a nyelveinkhez tartozó szavakat. Minden szó 3 féle háttérszínnel rendelkezhet: piros, sárga, zöld - ezek jelölik a haladást az adott szóra. A jobb felső sarokban lévő kérdőjellel ellátott mezőre mozgatva az egeret láthatjuk ezt az információt.

Egy oldalon egyszerre 15 szót jelenít meg a program, a lapozást alul a program közepétől jobbra elhelyezkedő "Előző" és "Következő" gombokkal végezhetjük el.



2.6. ábra. A szótár oldal kinézete

Hozzáadás

Nyelv és szó hozzáadáshoz alul a program ablakának közepétől balra találjuk az "Új Nyelv" és az "Új Szó" gombokat. Új nyelv esetén a felugró ablakban kell megadni a bemenetet, míg új szó esetén minden nyelvhez adhatunk meg lexémát, azonban csak egy nyelvhez szükséges. Mindkét esetben ha végeztünk, az "Ok" gombbal menthetjük el a bevitt adatot, a "Mégse" gombra pedig megszakíthatjuk a hozzáadást, melyet meg kell erősíteni.



2.7. ábra. Szó hozzáadás

Törlés

A táblázatban jobb klikkelve egy nyelvre, vagy egy szóra láthatjuk az opciókat amik között láthatjuk a "Törlés" opciót. Nyelv törlése esetén a nyelvhez tartozó összes szó is, míg szó törlésénél az adott szó minden nyelvnél elveszik. A törléshez is megerősítés szükséges.

Szó módosítása

Egy szóra jobb klikkelve a "Szerkesztés" opciót választva jelenik meg felugró ablakban a változtatás ablaka, melyet a 2.8. ábrán láthatunk. Itt egyesével megváltoztathatjuk az adott nyelvekhez tartozó szavakat. Hogy ezt tegyük, a szóval egy sorban lévő mezőbe kell beírunk az új lexémát, majd a mellette lévő "Megváltoztat" gombbal cseréljük le a régi szót. Az ablak alján találhatjuk a kategóriák legördülő menüt - itt állíthatjuk be, hogy mely csoportokkal szeretnénk ellátni. Ha végeztünk, alul a "Véglegesítés" gombbal fejezhetjük be a módosítást. Ha szót is változtattunk, akkor megerősítés szükséges, ugyanis ilyenkor a szó adott nyelvhez tartozó haladásmérője visszaáll alap helyzetre.

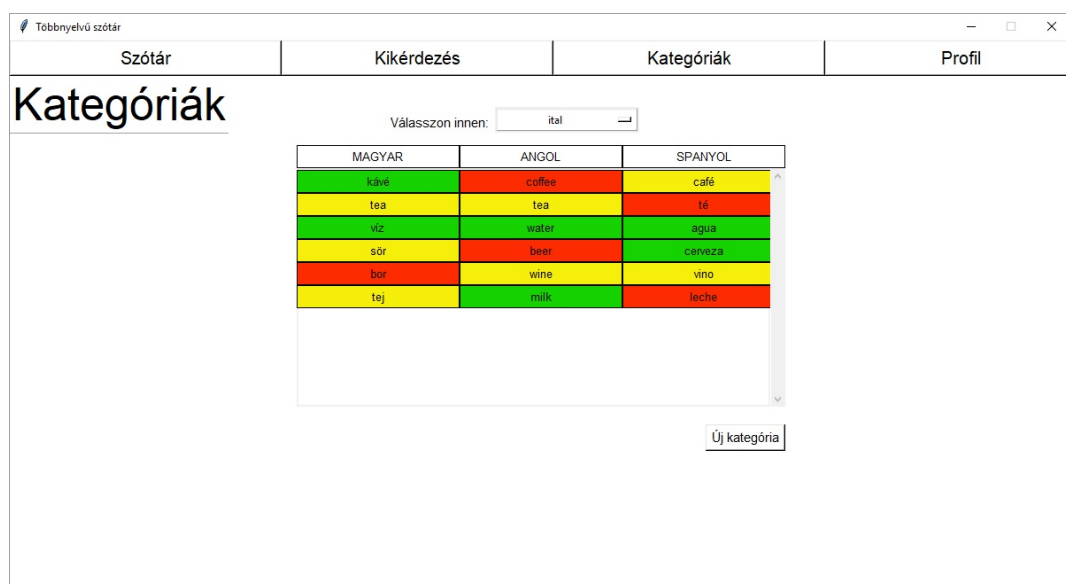


2.8. ábra. Szó módosítás

2.4.3. Kategóriák

A "Kategóriák" fülre lépve a felhasználó megtekintheti az adott kategóriáihoz tartozó szavakat, és vehet fel új kategóriát. Az ablak közepén felül lévő legördülő menüből kiválasztva a csoporthoz tartozó szavak jelennek meg a táblázatban, a szótár lappal ellentétben itt lapozás nélkül láthatjuk a szavakat, tekintve arra hogy egy kategóriához jelentősen kevesebb szó fog társulni az összeshez képest rövid idő után. Ha így nem férne ki, a listázott lexémákat görgővel tudjuk végignézni.

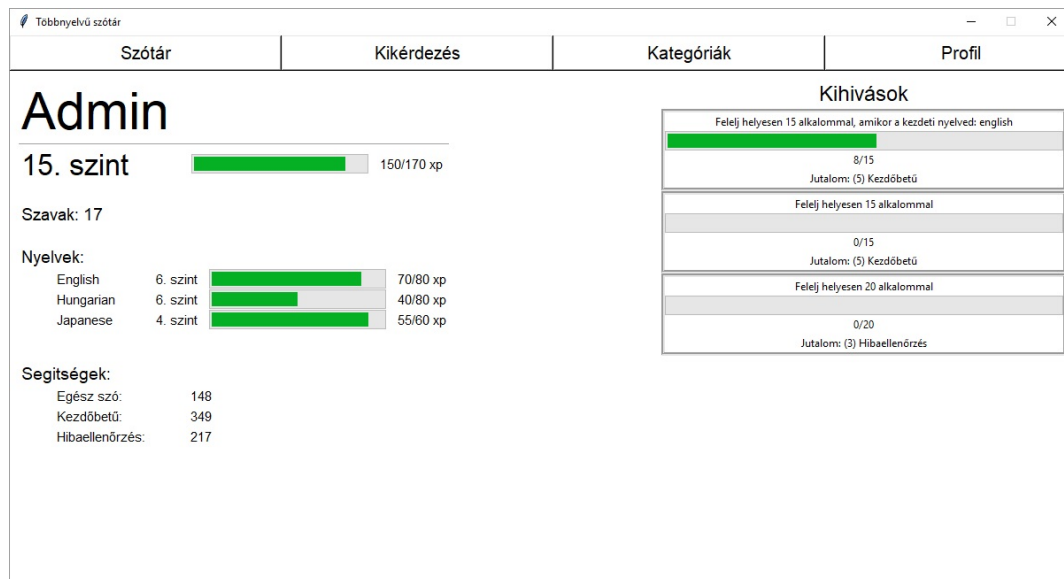
A táblázat alján látható "Új kategória" gombra kattintván a nyelv hozzáadása mintáján már ismerős felugró ablak bemeneti mezőjébe beírt, nem üres szót menthetjük el a kategóriáink közé.



2.9. ábra. A kategóriák

2.4.4. Profil

A felső menüpontok legvégső elemére, a "Profilra" kattintva tekinthető meg a felhasználó fiókjához tartozó adatok. Ez az oldal is két részre tagolható - bal oldalon a statisztikai adatok, míg jobb oldalon az aktuális kihívások szerepelnek.



2.10. ábra. A profil

A fenti 2.10.-es ábrán látható, hogy az adatok a felhasználónévvel kezdődnek, mely alatt látható a fiók szintje, mellette a következő szinthez való tapasztalati pont mérővel. Ezt követően látható a szavak mennyisége. Ezután a nyelvek sorakoznak, melyek soronként helyezkednek el egymást alatt, és jelezve a nevük mellett a szintjüket, és a szükséges tapasztalati pontok haladását a következő szinthez. Mind a nyelvekhez és a felhasználóhoz tartozó szintek előre haladva egyre több pontot igényelnek a szintlépéshez.

2.4.5. Szintlépés

A felhasználónak és nyelveinek szintmérői vannak, melyek vizuális megerősítést adnak a használatról. A szintek tapasztalati pontoktól függenek, melyeket helyes válasz után kaphatunk. Egy helyes válasz 5 pontot jelent, mely a felhasználó szintjéhez hozzáadódik, nyelvi szinthez pedig csak az adott megválaszolandó szó nyelvéhez. Minden szint eléréséhez folyamatosan növekedő pont szükséges.

2.4.6. Kihívások

A profil jobb oldalán lévő részben láthatjuk a kihívásokat. Egyszerre maximum 3 kihívás lehet egy felhasználónál. Sikeres teljesítés után a hozzá tartozó jutalmat hozzáadja a megfelelő segítséget a felhasználó részére, és törlődik. Minden nap az első belépés folyamán egy előre meghatározott sablon listából véletlenszerűen frissíti fel a felhasználóhoz tartozó kihívásokat, duplikátum kerülésével.

A kihívások teljesítendő kritériuma mind helyes választ igényel, mely lehet bármely nyelven, vagy meghatározott kezdő- vagy célnyelvvel, és meghatározott mennyi-

séggel a teljesítéshez. Két kihívás a nem egyezhet, legalább eltér az egyik a következőkből:

- Kezdő-, cél-, vagy meghatározatlan nyelv
- A megválaszolandó szavak számossága
- A jutalomban megfogalmazott segítség vagy mennyiség

2.4.7. Segítségek

A felhasználói adatok a segítségekkel zárulnak, melyek mellett elhelyezkednek a hozzájuk tartozó mennyiségek. Három fajtát különböztetünk meg:

- **Egész szó:** Ezt elhasználva az egész szót be tudjuk helyettesíteni a bemeneti mezőbe
- **Kezdőbetű:** A fentivel egyezően a kezdő betűt tudjuk beilleszteni.
- **Hibaellenőrzés:** Erre kattintva a bemeneti mezőben lévő szó zöld vagy pirosra színeződik, attól függően, hogy helyes-e a beírt szó.

A segítségek elhasználás után maguktól nem gyarapodnak, helyette a felhasználó két módszerrel tudja emelni a számukat:

- Szintlépések: Felhasználó vagy nyelv szintlépése esetén a program véletlenszerűen sorsol segítség típust és hozzá egytől ötig mennyiséget.
- Kihívások: Minden kihívás egy előre megszabott és látható segítséget ad megszabott számossággal.

2.4.8. Kikérdezés

A kikérdezés egy három fázisú művelet: a szűrők beállítása után a kikérdezés, melyet a kiértékelés zár le - így a "Kikérdezés" fülre kattintva a "Beállítások" alrészhez navigálunk első lépésben. Kikérdezést hagyhatunk félbe, azonban ez esetben a haladások elvésznek.

A kikérdezés beállítása

Négy opciót tudunk állítani tetszés szerint ezen a részen:

- **Kategória:** Itt állíthatjuk be, hogy mely kategóriákból szeretnénk látni kérdéseket. Ha semmit nem választunk, akkor alapértelmezetten nem szűri le kategóriák alapján a szavakat, és kategória nélküli szavak is megjelenhetnek.

- **Nyelv:** A fenti mintája alapján ez is szűrőként szolgál a szavak célnyelveire, mely alapértelmezett esetben nem filterezi le a szavakat.
- **Kezdő nyelv:** Ezen opciónál választhatjuk ki, hogy a lefordítandó szó milyen nyelvből jöjjön.
- **Nehézség:** Itt választhatunk a három nehézségi fokozat közül:
 - Könnyített: Maximum 10 szóból áll a kikérdezés.
 - Normál: Maximum 15 szóból áll a kikérdezés.
 - Neheztett: Maximum 30 szóból áll a kikérdezés, és a lefordítandó szó nyelve nem jelenik meg.



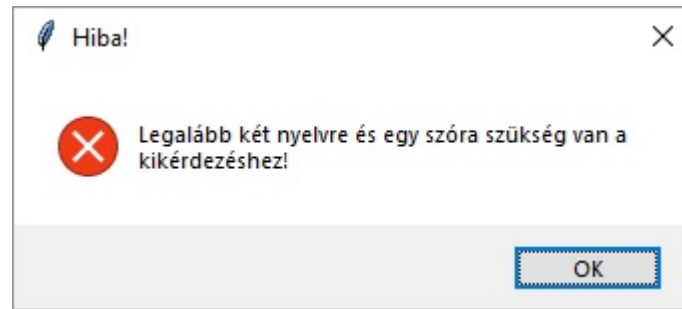
2.11. ábra. Kikérdezés beállítása

Ha nem lennénk biztosak mi mit jelent, a jobb felső sarokban lévő kérdőjellel ellátott mezőre víve a kurzort ismét felvilágosíthatjuk a tudásunkat az adott elemről.

Az opciók beállítása után a program véletlenszerűen válogat ki szavakat, figyelve arra, hogy kevesebbszer jelenjenek meg azok, melyek már nagyobb haladásmérőn állnak, ügyelve, hogy ne legyen ismétlés szavak között. Ha az opciók során olyan beállításokat jelölünk be, amellyel nem tud a program elég szót adni a kikérdezéshez, akkor kevesebbre kell csak válaszolni.

Például: Állatok kategóriában csak 2 szó van felvíve, ezért a Normál kikérdezés során is csak 2 kérdésre kell felelni.

Kikérdezéshez szükség van legalább 2 nyelvre és 1 szóra, különben a program elutasítja az elindítást, melyről értesíti a felhasználót, ez látható az alábbi 2.12.-es ábrán.



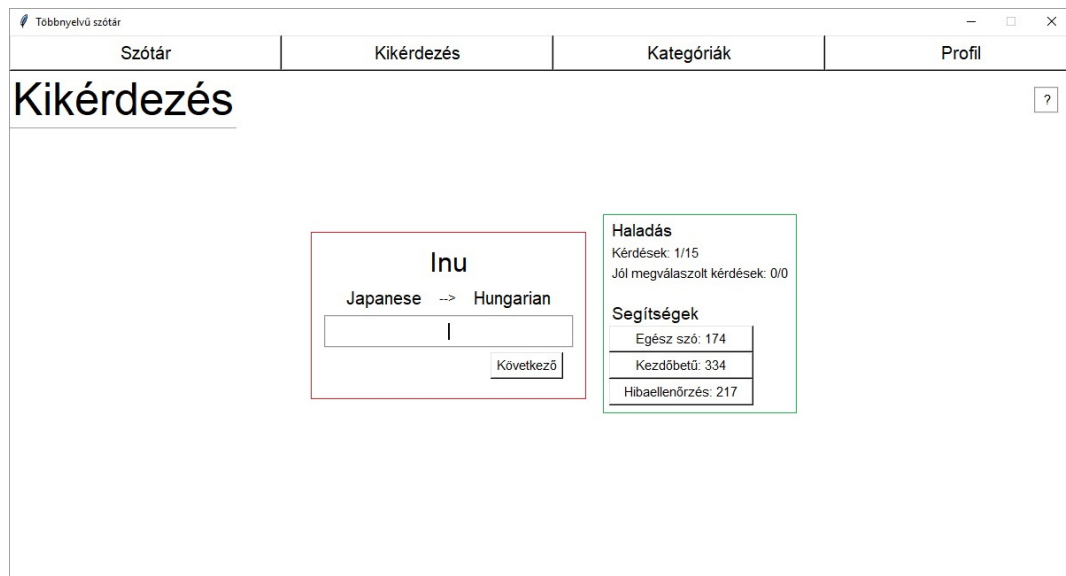
2.12. ábra. Hiba a kikérdezés indításakor

A kikérdezés folyamata

A beállítás után amint a program összegyűjtötte a szavakat, a kikérdezés ablakra vált át. Ezt az ablakot két részre lehet bontani: a baloldali jelenlegi tesztre adott szóelemre és a jobboldali haladásmérőre és segítségekre.

A tesztre adott szóra egy elem van létrehozva, melyet a 2.13. ábra pirossal bekeretezett részén láthatunk: felül középen a lefordítandó szó, alatta a kezdő- és cél nyelv, melyet követ a bemenetet váró felület, végül a "Következő" gomb, mellyel haladhatunk előre.

2.13. ábrán látható a zölddel határolt részen belül, hogy hányadik kérdésnél tartunk, és hány kérdésre adtunk eddig jó választ. Alatta helyezkednek el a segítségék.



2.13. ábra. A kikérdezés

A segítségék nem töltődnek újra maguktól, azonban újakat lehet szerezni, melyre a **2.4.5/Segítség** résznél található leírás.

2.4.9. Kiértékelés

A kikérdezés után az adatokat feldolgozza a program, a szavak haladásjelzőjét, a felhasználói és nyelvek szintjeinek előrelépését, a kihívások változását és az újonnan szerzett segítségeket elmenti, végül készít belőlük egy jelentést, amely a következő részeket tartalmazhatja:

- Helyes szavak mennyisége, és mellette egy "Megtekintés" feliratú gomb, mellyel kilistázhadjuk az összes szót ami a kikérdezésben volt, helyes és adott válasszal együtt.
- Megszerzett tapasztalati pontok, és esetleges szintlépés esetén az új szint - először a felhasználói, majd külön a nyelvekhez tartozó haladáshoz.
- Megszerzett segítségek, melyek szintlépés vagy teljesített kihívás után nyert a felhasználó a kikérdezés során.
- Kihívások állása, mely egyesével mutatja a leírást, és a hozzá tartozó haladást.

Egy lehetséges eredményt láthatunk a 2.14.-es ábrán egy kikérdezésről.



2.14. ábra. A kikérdezés eredmény

A helyes válaszok száma melletti "Megtekintés" gomb segítségével hozhatjuk fel egy felugró ablak keretében a megválaszolt szavakat, és láthatjuk mellette a megválaszolandó és helyes szót, a nyelvekkel kiegészítve.

Itt az alábbi, 2.15.-ös ábrán illusztrálva középen helyezkedik el a megválaszolt szavunk, mely zöld vagy piros színű az alapján, hogy sikeresen vagy sikertelenül választottunk, és tőle jobbra helyezkedik el a kiindulási szó, balra pedig zárójelben a helyes válasz.

hungarian	japanese			
F	-->	"F"	(F)	
hungarian	japanese			
I	-->	"I"	(I)	
japanese	english			
Hyou	-->	"Table"	(Table)	
japanese	hungarian			
Neko	-->	"Macska"	(Macska)	
japanese	english			
A	-->	"A"	(A)	
hungarian	japanese			
Szám	-->	"K"	(Kazu)	
hungarian	english			
Kutya	-->	""	(Dog)	
hungarian	japanese			

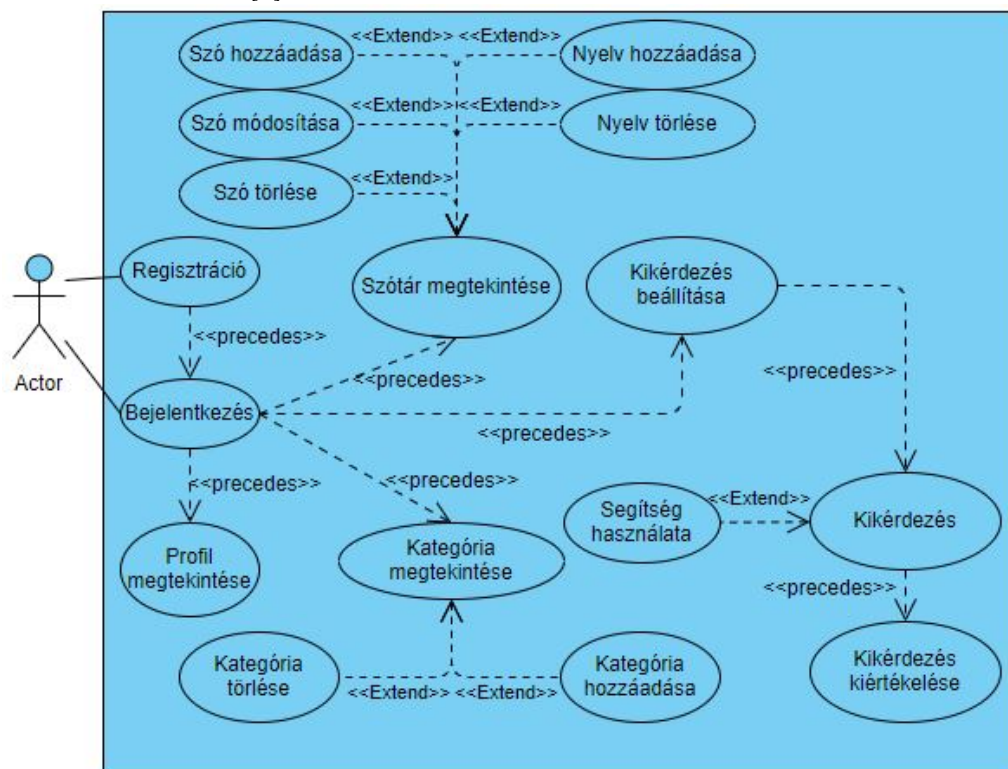
2.15. ábra. A kikérdezés szavaihoz tartozó adatok felugró ablaka

3. fejezet

Fejlesztői dokumentáció

3.1. Az alkalmazás felépítése

A Többnyelvű szótár kliensoldalon működteti mind a front- és back-end-et, az adatok bárholi elérése érdekében pedig egy adatbázis szerveren tárolja el az adatokat, így biztosítva a kényelmesebb felhasználói élményt. Lehetőség van lokális adatbázis használatra is, de ezt a program nem tartalmazza. A kommunikációt a **MySQL.Connector**[3] Python modul biztosítja.



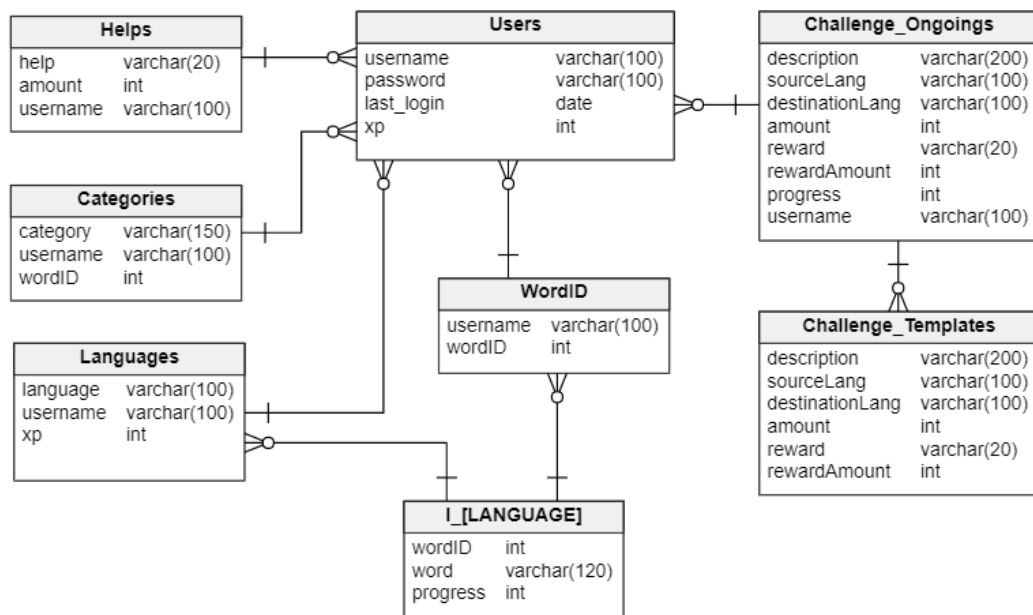
3.1. ábra. Használati eset diagram

3.2. A back-end architektúrája

3.2.1. A webszerver

Az adatbázis server alapja a **MariaDB Platform**[2]. Ez egy nyílt forráskódú, SQL-alapú relációsadatbázis-kezelő server, amely a *MySQL* egy fork-ja. Az erősségei közé tartozik a kiváló teljesítmény és az igény szerinti skálázhatóság, amely a valós idejű kommunikációban jelentős szerepet játszanak. A *MariaDB*-hez nincsen modul a *Python* könyvtárai között, de mivel célja, hogy a *MySQL*-lél a kompatibilitása magas legyen, ezért a **MySQL Connector** framework-öt használjuk a kommunikáció kiépítésére és vezérlésére.

A *Database/config.ini* fájlban adhatóak meg az adatbázis-kapcsolathoz szükséges adatok. MariaDB esetén csak egy adatbázis-kezelő serverhez kapcsolódunk, melyhez helyes host, felhasználónév, jelszó, port és adatbázis megadásával tudunk kapcsolódni, ezeket a *Database/config.ini* fájlban tudjuk beállítani. Mivel a MariaDB a MySQL egy forkja, és egyik fő irányelve a magas kompatibilitása a fent említett adatbáziskezelőnek, ezért lokális adatbázis esetén a kettő közül bármelyiket használhatjuk. A 3.2.-es ábrán látható az adatbázis táblák relációinak vizualizációja.



3.2. ábra. Az adatbázis felépítése

Users

A *Users* tábla áll a középpontban, melyek oszlopai a következők:

- **username**: A felhasználóneveket tartalmazó oszlop.

- **password:** A jelszavak tárolására használt oszlop.
- **last_login:** Az utolsó belépést tároló dátum típusú oszlop, mely a kihívások feltöltésére használt.
- **xp:** A felhasználóhoz tartozó tapasztalati pontok száma.

Helps

A Users táblával áll relációban, a *username* oszlop alapján. A *help* és *amount* oszlopok elmentik a felhasználóhoz tartozó segélytípusokat, és a mennyiségeiket. Minden felhasználónak három sora van, a három segélyhez egy-egy. Ezek a regisztrációt követő első belépés után kerülnek be az adatbázisba.

Challenge_Templates

A kihívások készítéséhez tartozó sablonokat tartalmazza. A mérete konstans, a program futása során innen csak lekérdezés történik.

- **description:** A kihívás leírása, ebből tudja a felhasználó, hogy mit kell megoldania.
- **sourceLang:** "-" vagy "Y" értékkel rendelkezik, ahol az utóbbi jelzi, hogy kezdő nyelv szerepel a kihívásban.
- **destinationLang:** A sourceLang mintáján szintén "-" vagy "Y" értéket tárol.
- **amount:** A mennyiség, amely szükséges a teljesítéshez.
- **reward:** A jutalom segély típusa.
- **rewardAmount:** A jutalom mennyisége.

Challenge_Ongoings

Ide kerülnek az épp folyamatban lévő kihívások. A *Challenge_Templates* elemeit tartalmazza, immáron a sourceLang vagy destinationLang helyére behelyettesítve az aktuális felhasználóhoz tartozó nyelvvel. Két új mezőt is tartalmaz, mely a *progress*, amiben a teljesítési haladást tárolja, míg a *username* a felhasználóhoz tartozó nevet raktározza.

WordID

A *username* oszlop tárolja a felhasználóneveket, és a *wordID* oszlop rendeli hozzá a szavak azonosítóját.

Languages

A *username* oszlopban található a felhasználónév, mellette lévő *language* oszlopban pedig a nyelv. Az *xp* oszlop tárolja az adott felhasználó adott nyelvét tartozó tapasztalati pontokat.

1_ [LANGUAGE]

Minden a *Languages* táblában szereplő nyelvhez létezik egy külön tábla, melyben a *wordID* oszlopban található azonosító mellett helyezkedik el a hozzá tartozó szó, és annak a haladásmérője, melyek rendre a *word* és *progress* oszlopban vannak. A tábla nevében a szögletes zárójel helyére kerül a nyelv, tehát Spanyol nyelv esetén a hozzá tartozó tábla "1_spanyol".

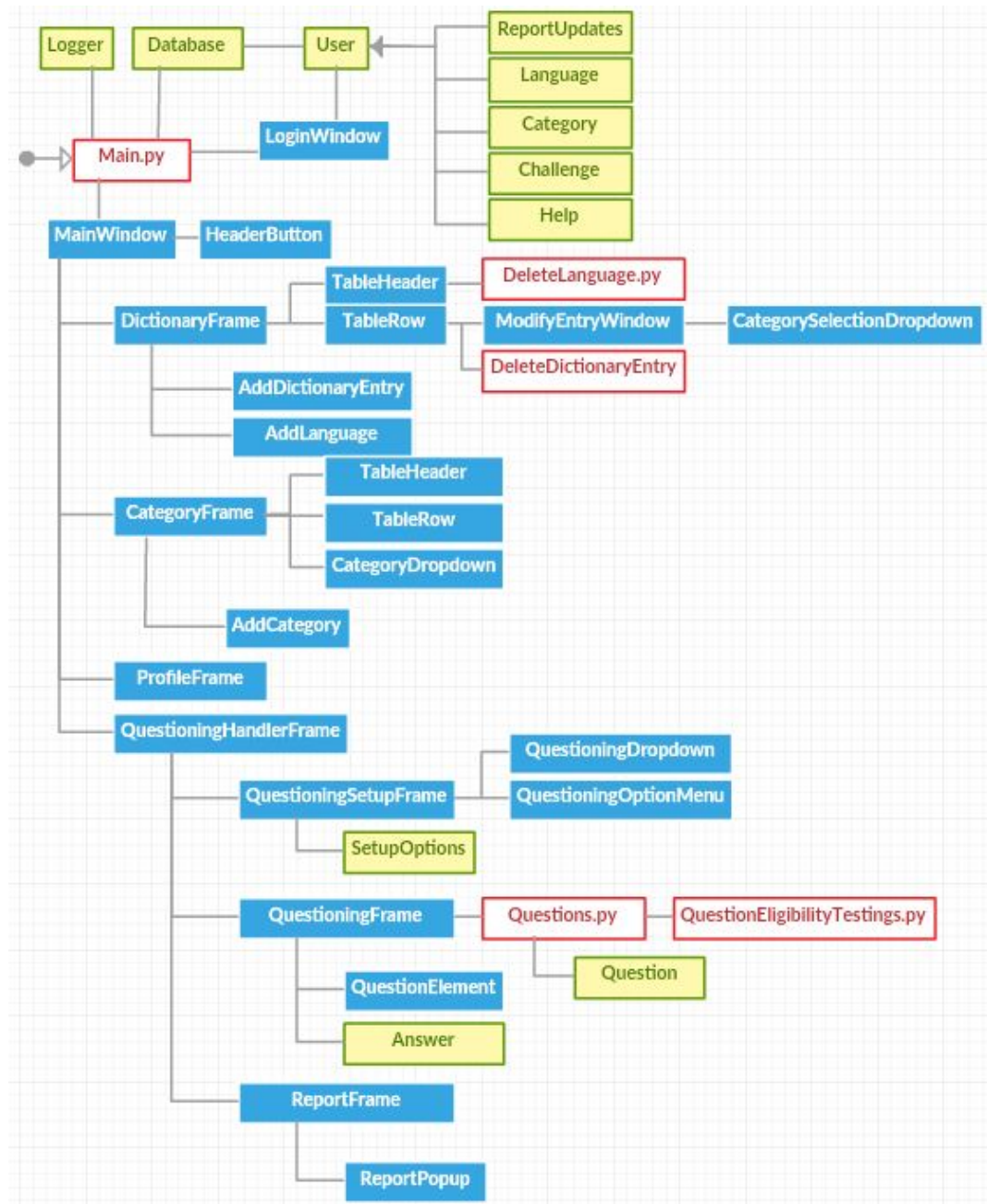
Categories

A kategóriák nyilvántartására fenntartott tábla, melyben a *category* oszlopban helyezkedik el a kategória neve, majd pedig a *username* jelzi az adott felhasználót, és a *wordID* az adott szó azonosítóját.

3.2.2. A back-end alapja

A szoftver **Python 3.6.3**[4]-ben íródott, mely egy nagyon magas szintű, általános célú, interpretált nyelv. Egyik nagy előnye, hogy egyszerre támogatja a funkcionális, az objektumorientált, az imperatív és procedurális programozási paradigmákat, melyekből az első kettőt ötvözve épül fel a program implementációja. Ugyanakkor a nyelv tartalmaz két darab "kétélű kardot" is - ezek a privát és nem típusdefiniálható adattagok hiánya, és a futási időben való típusellenőrzés. Mindkettő célja, hogy megkönnyítse a programozó dolgát, de ugyanakkor felelős és átgondolt kódolást igényel - ha ez nincs rendesen vezényelve, a kódot elárasztja a lokális típusellenőrzés.

A futtatáshoz a MultilanguageDictionary.exe egy elrejtett parancssorban hívja meg a Python fordítót a Main.py fájlra, amely a program belépési pontja.



3.3. ábra. A program komponens diagramja

3.2.3. Globális objektumok

A program indulásakor kettő objektum jön létre, melyek a *Logger* és a *Database* osztályok példányosított változatai.

Logger

A *Logging/logger.py* fájl tartalmazza. A program indulásakor ez az osztály jön létre legelőször, létrehozza a **LOGS/** és **Reports/** mappákat, ha még nem léteznek, és az előbbibe létrehoz egy *.log* kiterjesztésű log fájlt

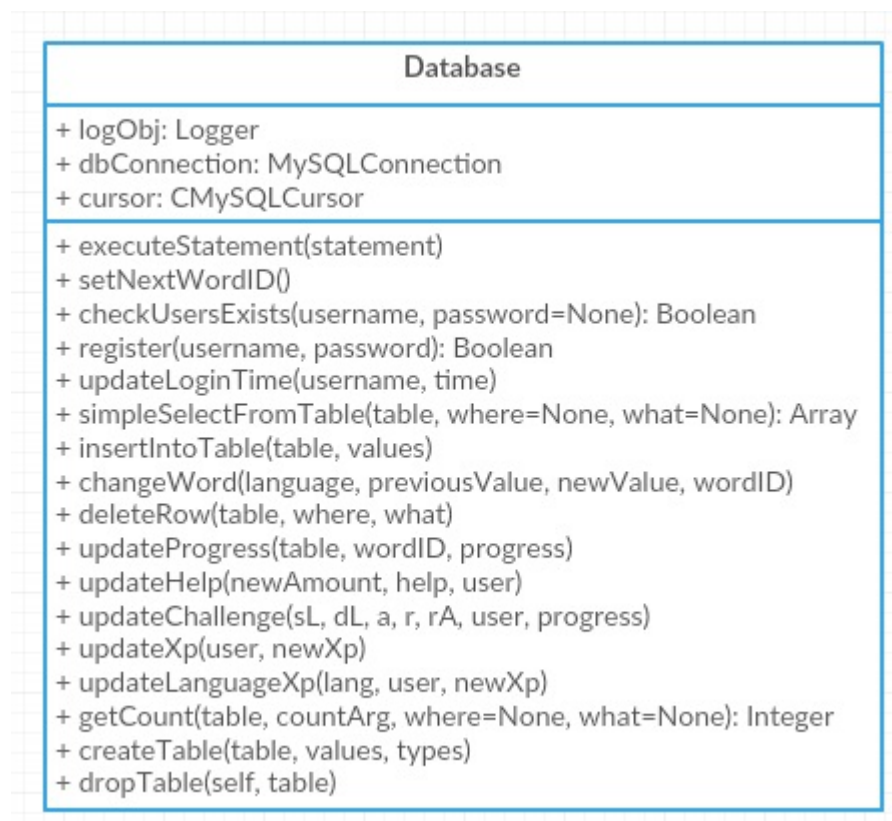
"YYYY_MM_DD_HH_mm_SSlog.log"

néven. A program menetével ebbe a fájlba helyezi el a futás során elért mérföldköveket hiba eseti backtrace érdekében. A kikérdezés befejeztével a *Reports/* mappába kerülnek szöveges formátumba a kiértékelés eredményei, mely mind backtrace-re, mind felhasználói visszatekintésre alkalmas.

Database

A *Database/DatabaseMain.py* írja le. Az adatbázis kapcsolódáshoz az adatokat a **Database/config.ini** fájlból a **configparser**[7] konfigurációs fájlfeldolgozáshoz alkalmazható könyvtár segítségével végezzük. A kapcsolatot a program indulásakor alakítjuk ki, és kilépés után zárjuk le. Két fő változója a "dbConnection" és a "cursor", melyekkel az adatbázis lekérdezéseket és utasítások végzi el, olvassa vissza a kimenetet és menti el az állapotot.

A 3.4.-es ábrán látható a *Database* osztály adattagjai és a metódusai a hozzá készített osztálydiagramon.



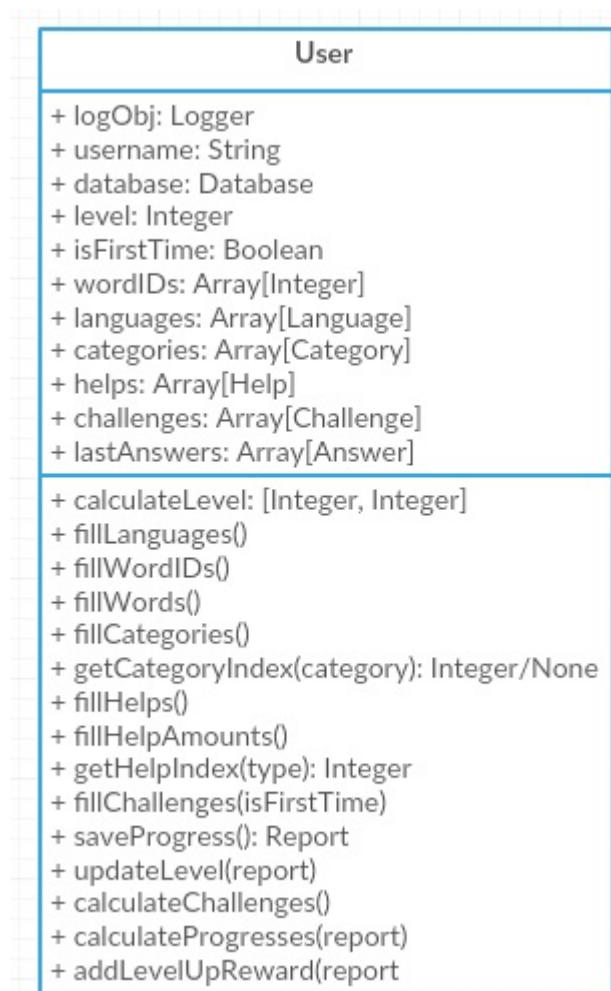
3.4. ábra. Az adatbázis modul

3.2.4. Felhasználói adatobjektumok

User

Sikeres bejelentkezés esetén létrehozuk egy a felhasználó adatait reprezentáló objektumot, mely *User/User.py* fájlban lévő osztály példányosítása. A konstruktorának átadjuk a belépés alatt elmentett felhasználónevet, egy logikai változót, mely reprezentálja, hogy ez-e az első bejelentkezés adott felhasználóhoz az adott napon, és az adatbázis objektumot, utóbbit a program innen éri el. Az adatbázisból kinyeri a Users táblából a felhasználói szintű tapasztalati pontját, majd pedig sorban feltölti a tömb változóit, melyek a következő adatokat mentik el:

- **wordIDs:** A felhasználó szavaihoz tartozó azonosítók
- **languages:** A nyelveket reprezentáló Language objektumok listája
- **categories:** A Category osztály példányainak tömbje, melyek a kategóriákat képviselik
- **helps:** A segítségeket tároló Help objektumok összessége
- **challenges:** A kihívásokat megtestesítő Challenge osztály példányainak nyilvántartására létrehozott lista



3.5. ábra. A felhasználó modul

Language

A definiálását a *User/Language.py* fájlban találjuk. Az osztály nyilván tartja a nyelvet, a hozzá tartozó szavakat és azonosítójukat, és a nyelvhez tartozó szintet. Metódusai szimpla lekérdezések, adatmódosítások.

Category

A *User/Category.py* tartalmazza. A fent említett nyelv mintáján működik, tartalmazza a kategória nevét, a szavait és azonosítóit, egyszerű lekérdezésekkel és adatmódosításokkal.

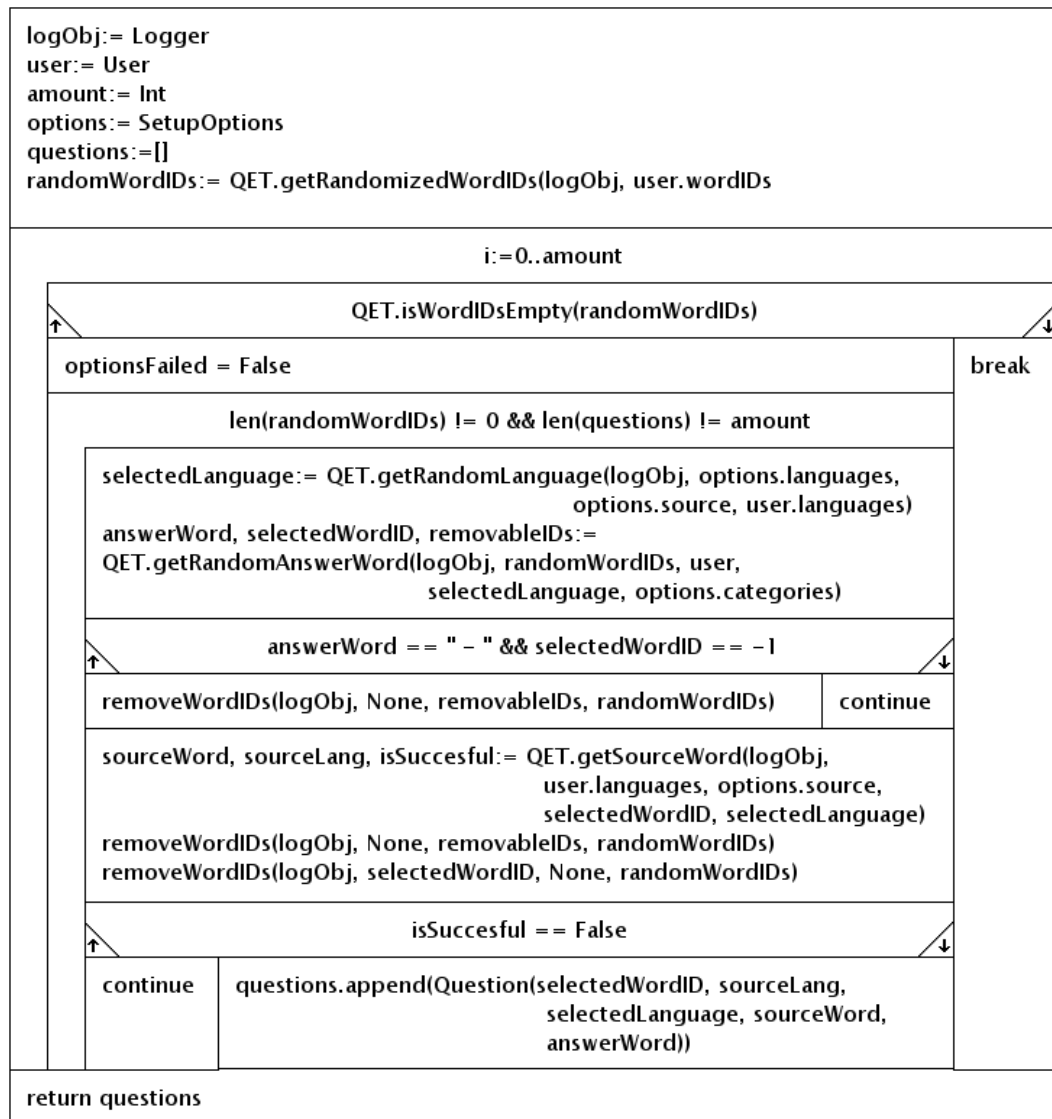
Help

A *User/Help.py* írja le. Egy egyszerű adatrepresentatív osztály, mely tartalmazza a segítség kulcsszavát, szövegét, és mennyiségét.

Challenge

A *User/Challenge.py* fájlban találhatjuk. Szintén egy adatrepresentatív osztály, mely a kihívás adatait tartalmazza, egyszerű lekérdező és adatmódosító metódussal.

3.2.5. A kikérdezés előkészítése



3.6. ábra. A szavak begyűjtésének struktogramja

A felhasználó által elvégzett szűrések beállítása után az "Indítás" gomb megnyomásával létrejövő **QuestioningFrame** első lépése, hogy begyűjtse a felteendő kérdésekhez az adatokat. Ehhez a konfigurációval létrejövő **SetupOptions** osztály ad segítséget, mely a *MainFrame/Questioning/SetupOptions.py* fájlban található, mely tárolja a beállított opciókat. Átadva a nehézségi szint által deklarált maximum mennyiséget, a *MainFrame/Questioning/Objects/Questions.py* *getQuestions* metódusa vezényli le a folyamatot és ad vissza egy **Question** objektumokkal megtelített tömböt, ezt szemlélteti a 3.6. ábra.

A begyűjtéshez használva vannak a *MainFrame/Questioning/QuestionEligibilityTesting.py* fájlban található metódusok, melyekre a 3.5. ábrán a QET mozaikszóval referálunk. Ebben olyan metódusok vannak, amelyek a *Question* objektumok különböző adatait generálják. Minden me-

tódus első paramétere egy **logObj** *Logger* típusú változó, ez az alábbiakban nincs feltüntetve a redundancia elkerülése érdekében.

getRandomizedWordIDs

Bemeneti paramétere egy **array** tömb változó, melyből egy másolatot készít, és megkeverve visszaadja azt.

isWordIDsEmpty

A paraméteréül kapott wordIDs tömböt leellenőrzi, hogy üres-e, és az alapján egy logikai értéket küld vissza.

getRandomLanguage

Bemenetei a következők:

- **languages:** A beállítási opciókban kiválasztott nyelvek.
- **sourceLang:** A beállítási opciókban kiválasztott kezdő nyelv.
- **userLanguages:** A felhasználó Languages objektumokat tároló tömbje.

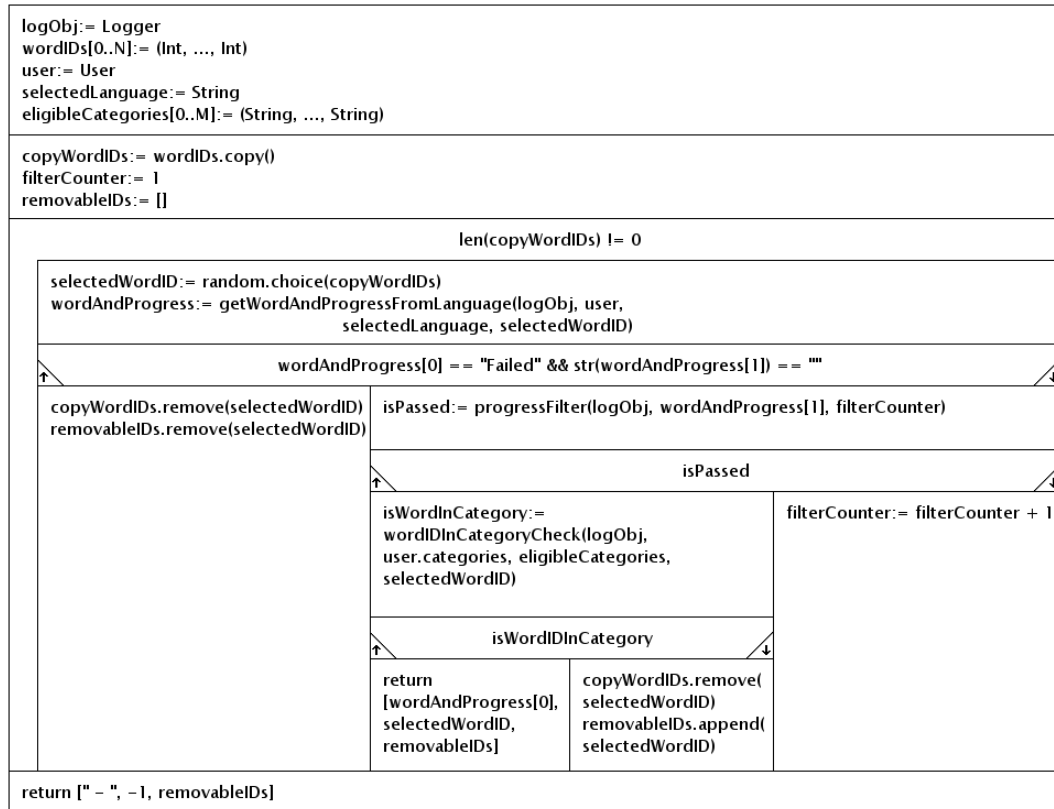
A metódus kiválasztja a lehetséges nyelvek közül véletlenszerűen a megválaszolandó szóhoz az egyiket, figyelve arra, hogy esetleges kezdő nyelv esetén mást válasszon.

getRandomAnswerWord

Paraméterei:

- **wordIDs:** A szavak azonosítóinak tömbjei.
- **user:** A felhasználót szimbolizáló User objektum.
- **selectedLanguage:** A kiválasztott cél nyelv.
- **eligibleCategories:** A felhasználó által beállított kategóriák.

Működése a bemeneti paraméterek figyelembe vevése alapján vezényli le a cél szó kiválasztásának folyamatát. Sikeres találat esetén a szót és azonosítóját, míg sikertelen esetben " - " és -1 értékeket ad vissza. Mindkét esetben visszatérési értéke egy removableIDs tömb, melyben a tesztelés során elbukott azonosítókat adja vissza az azonosítók közüli törlésre. A folyamat látható az alábbi struktogramon, amelyet a 3.7. ábra tartalmaz.



3.7. ábra. A getRandomAnswerWord metódus struktogramja

getWordAndProgressFromLanguage

A paraméteréül kapott **user**, **lang** és **wordID** változókkal kikeresi és visszaadja az adott azonosítóhoz tartozó szót és haladásmérőt a *lang* Language objektumból. Ha nem talál ilyet, "Failed" szöveggel tér vissza.

progressFilter

Paraméteréül kapja a **progress** és **counter** egész típusú változókat. Sorsol egy véletlen számot 1 és 10 között egy *randint* változóba, majd azzal egyezteteti a *progress* értékét. A következő esetekben igaz értéket ad vissza, ellenben hamisat:

- A randint ≤ 5 és a progress ≤ 20
- A randint $\in (5, 8]$ és a progress $\in (20, 40]$
- A randint > 8 és a progress > 40
- A counter == 6

wordIDInCategoryCheck

Paraméterei a következők:

- **userCategories:** A felhasználó kategóriái.
- **eligibleCategories:** A kiválasztott kategóriák.
- **wordID:** A *getWordAndProgressFromLanguage* metódusban kiválasztott azonosító.

Leellenőrzi, hogy az azonosító szerepel-e a kiválasztott kategóriák között, és ez alapján logikai értékkel tér vissza.

getSourceWord

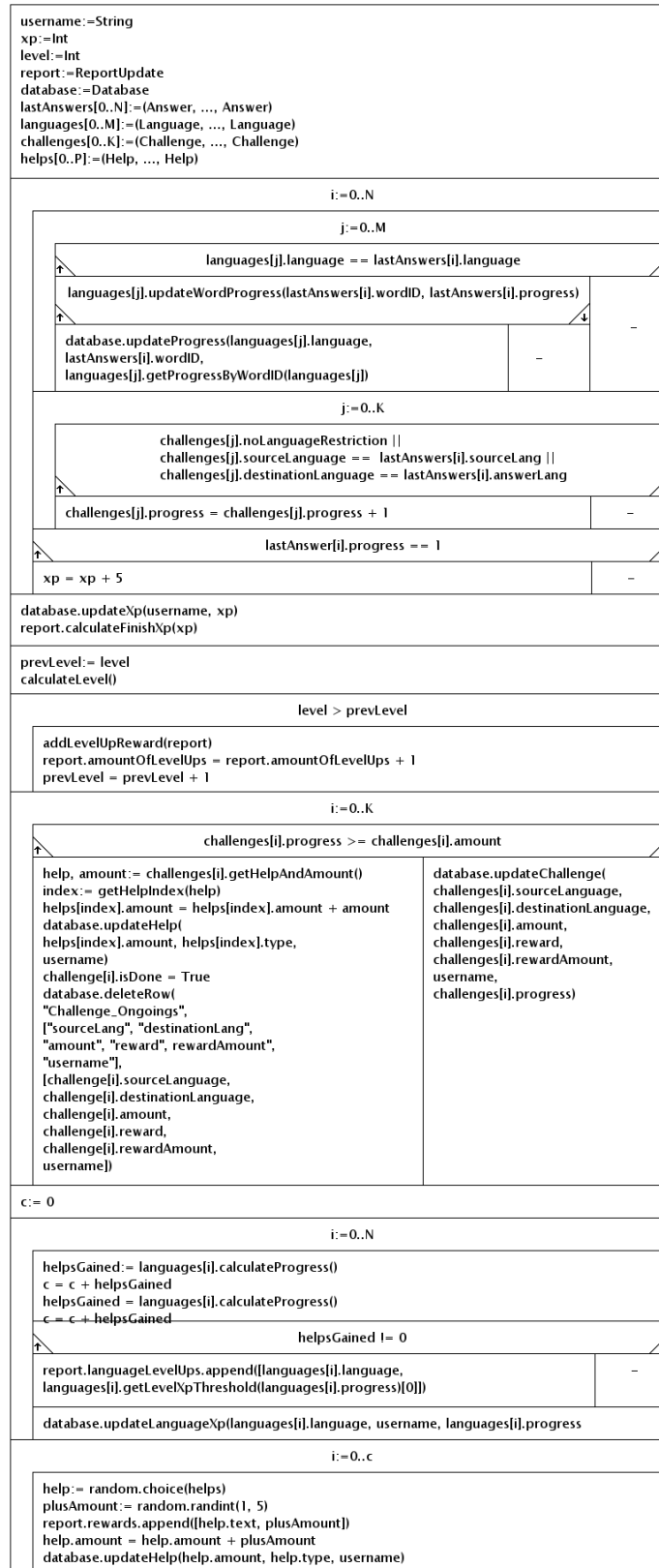
Visszaadja a lefordítandó szót. Paraméterek:

- **languages:** A felhasználó nyelvei.
- **sourceLang:** A kiindulási nyelv.
- **wordID:** A kiválasztott szóazonosító.
- **selectedLanguage:** A kiválasztott cél nyelv.

createReturnArrayForSourceWord

A *getSourceWord* metódus használja fel, egy nyelv objektumból a szóazonosító alapján visszaad egy tömböt, melynek elemei az azonosító, a szó és a nyelv, végezetül pedig egy logikai érték, mely hamisat vesz fel, ha a szó " - ".

3.2.6. A kikérdezés kielemezése



3.8. ábra. A kielemezés struktogramja

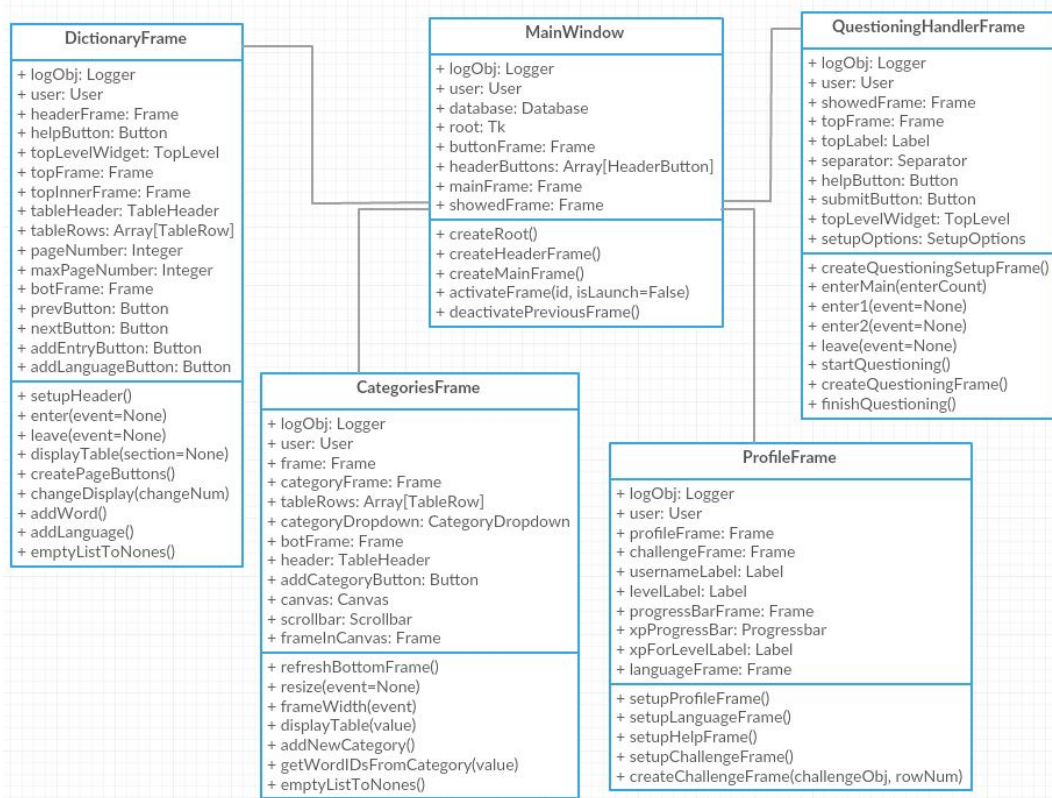
Minden bevitt válasz után a program elmenti az adott kérdéshez az összes adatot egy **Answer** objektumba, mely a *MainFrame/Questioning/Answer.py* tartalmaz.

Ezek az answer objektumok egy listába vannak befűzve, melyeket a kikérdezés után bemásol a *User* objektum *lastAnswers* adatmezejébe. Ezután a jelentés előtt elmenti az összes haladást a *User* osztály metódusait használva, mely a 3.8. ábra szemléltet egy stuktogrammal.

3.3. A front-end architektúrája

A de facto Python GUI framework segítségével valósul meg a grafikus interface, a **Tkinter**[5]-rel és alkönyvtáraival, a **ttk**[6]-val és a **messagebox**-al. Ezek a Python telepítésével együtt érkező alapértelmezett könyvtárak, melyek Tcl parancsá alakítja át a hívásait.

A program egy fő és számos felugró ablakból áll, melyek két részre különíthetők. A bejelentkezési és a fő ablak, és az adatbevitelt kérő felugró ablakok a *tkinter.Tk* metódusával kreált ablakok, míg a megerősítést kérő, az információt és hibát mutató ablakok pedig a *tkinter.messagebox* metódusban definiált sablonjaival jönnek létre.



3.9. ábra. A MainWindow és a fő Frame elemek viszonyának osztály diagramja

A *Tkinter* **Frame** objektumokkal tud egy területen belül elhelyezést vezényelni, akár egymásba illesztve is, bármely mélyen. A fő ablak - melyet a *Windows/MainWindow.py* fájlban található **MainWindow** osztály valósít meg - így a felső menü alatt lefoglalja a teljes területet egy *Frame* objektummal a *showedFrame* változóba, melyeket a menügombokkal változtatva az adott oldalhoz tartozó

Frame osztályból származtatott objektumát példányosítja a fent említett változóban. Váltás esetén az előző *Frame*, és vele együtt minden elem törlődik, és az új betöltődik. A kapcsolatokat láthatjuk a 3.9.-as ábrán.

HeaderButton

A *GUIAssets/HeaderButton.py* tartalmazza a **HeaderButton** osztályt, mely a **Tkinter.Button** osztályt implementálja. Az alap opciók mellett lekezezi az eseményeket, mikor a kurzor belép vagy elhagyja a területét, és az alapján színezi.

3.3.1. A Szótár keret

A szótár fő frame-je a *MainFrame/Dictionary/DictionaryHandler.py* mappában található, mely a *Frame* osztályból származtatott osztályt valósít meg. Ez fel van osztva három részre, tetején az oldal címe bal oldalt, és a segítség gomb jobb oldalt, középen a szótár táblázat, alul pedig a hozzáadásra és a lapozásra való gombok. A jobb oldali segítség mező a *Tkinter.Button* osztály egy példányosítása, mely a "groove" domborzati opciót alkalmazva kelti egy sima mező hatását. Ha a kurzor belép a területére, aktiválódik az "Enter" esemény, mellyel létrejön egy **Tkinter.TopLevel** modul, mely egy lebegő mező tartalmazva a segítség információkat.

A Tkinter nem tartalmaz beépített táblázat elemet, így meglévő elemekből kellett azt felépíteni, melyhez a legkézenfekvőbb opció a *Tkinter.Button* osztály példányosítása, és formázása, hogy ismétlésével egy rácsszerkezet álljon össze.

TableHeader

A *GUIAssets/DictionaryTableRow.py* fájl tartalmazza. A fő változója egy tömb, mely a fent említett *Tkinter.Button* elemek egy listáját tárolja. Minden gomb "solid" domborzati opcióval jön létre, szegély bekapcsolásával és a kattintás eseményt lekezelve fedi el eredeti valóját. Az elemek egy nyelv feliratával jönnek létre; így reprezentálja a **TableHeader** osztály a táblázat fejlécét. Az összes gomb jobb kattintási eseményére kötve van egy **Tkinter.Menu** elem, melyben a "Törlés" opció van elhelyezve.

TableRow

A *GUIAssets/DictionaryTableRow.py* fájl tartalmazza. A *TableHeader* mintájára *Tkinter.Button*-ök segítségével reprezentál egy sort a táblázatban, mely megfelelő nyelvhez megfelelő szót társít. A szó haladásával megegyezően a mező háttérszínét piros, sárga vagy zöld színűre állítja be. Szintén jobb egérgombhoz társít egy *Tkinter.Menu*-t, mely tartalmazza a "Szerkesztés" és "Törlés" opciókat.

A Szótár feugró ablakai

AddEntry

A *MainFrame/Dictionary/EntryModifications/AddDictionaryEntry.py* fájl tartalmazza. A *Tkinter.Tk* osztállyal létrehoz egy új ablakot, mely soronként egy **Tkinter.Label** szövegelemmel - amely a nyelvet jelzi - és **Tkinter.Entry** bemeneti mezővel rendelkezik. A mérete dinamikusan változik az elemek méretével, és az alján található "Hozzáadás" és "Mégse" gombokkal adhatjuk hozzá, vagy vethetjük el a beírt szavakat.

ModifyEntryWindow

A *MainFrame/Dictionary/EntryModifications/ModifyDictionaryEntry.py* írja le. Az első *n* sorban, ahol *n* a nyelvek száma balról jobbra haladva soronként tartalmazza az éppen elmentett szót és a hozzá tartozó nyelvet, egy bemeneti mezőt és egy "Megváltoztat" felíratú gombot, mellyel a bal oldali *Tkinter.Label* megváltozik. Alattuk található a kategóriák beállításához tartozó legördülű menü, melyet a **CategorySelectionDropdown** osztály példányosításával valósít meg. Alján a "Véglegesítés" és "Mégse" gombokkal fejezhetjük be a módosítást.

CategorySelectionDropdown

A *GUIAssets/CategoryDropdown.py* tartalmazza. A **Tkinter.OptionMenu** osztályt implementálva jön létre, mely a kategóriákat tartja számon. Hozzáadva egy kategóriát alatta megjelenik a "Kiválasztott kategóriák" sorában, újra rányomva törlődik onnan.

AddLanguage

A *MainFrame/Dictionary/LanguageModifications/AddLanguage.py* tartalmazza. A *Tkinter.Tk* példányosításával létrehoz egy ablakot, melyben egy *Tkinter.Entry* bemeneti mezőt tartalmaz, alján a "Hozzáadás" és "Mégse" gombokkal végezhetünk.

3.3.2. A Kategória keret

A *MainFrame/Categories/CategoriesHandler.py* fájlban leírt **CategoriesFrame** osztály valósítja meg, mely a *Tkinter.Frame* osztályt implementálja. Bal felső sarokban jelzi az oldal címét, és a felső sorban középen található kategória választásra használható legördülő menü található, melyet a **CategoryDropdown** osztályból példányosítunk. A kategória keret a szavak listázáshoz a 3.5.1-ben említett *TableHeader*

és *TableRow* elemeket használja fel, azonban letiltja a jobb klikkel előhozható opciókat. A táblázat szavait egy **Tkinter.Canvas** objektum tartalmazza, mely a **Tkinter.Scrollbar** modul alkalmazása miatt használt, így teremtve görgőt a szavakhoz. Az alján elhelyezkedő "Új kategória" gombbal adhatunk hozzá egy új csoportot.

CategoryDropdown

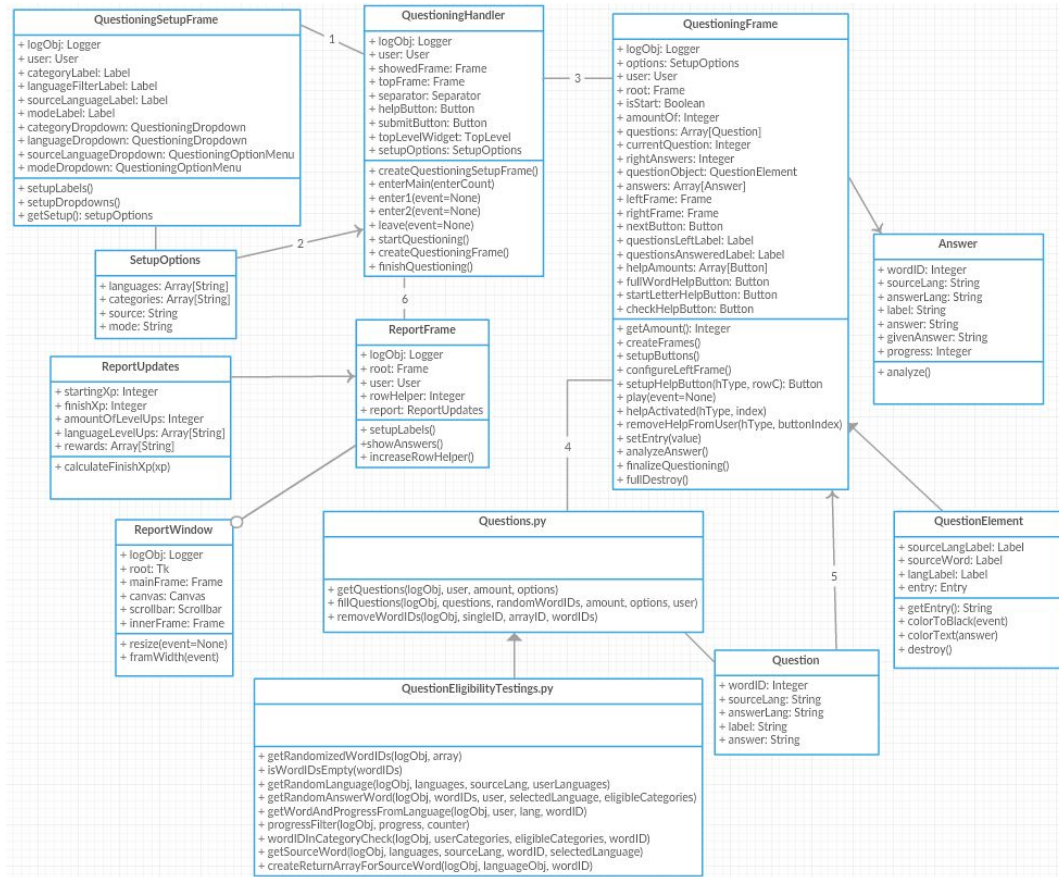
A *GUIAssets/CategoryDropdown.py* fájl tartalmazza. A **Tkinter.OptionMenu** osztályból származtatva jön létre, és a kategóriákat tartalmazza listában. Egy elemre kattintva változtatja meg a kilistázott szavakat a táblázatában.

A Kategória felugró ablaka

AddCategory

A *MainFrame/Categories/AddCategory.py* írja le. A nyelv hozzáadásával megegyezően egy új *Tkinter.Tk* ablakban egy *Tkinter.Entry* mezőbe írva adható meg, és fogadható vagy utasítható el az alul elhelyezkedő "Hozzáadás" vagy "Mégse" gombokkal.

3.3.3. A Kikérdezés keret



3.10. ábra. A kikérdezés menetének osztály diagramja

A kikérdezés menetét a *MainFrame/Questioning/QuestioningHandler.py*-ban található **QuestioningHandlerFrame** vezényli, mely egy *Tkinter.Frame* osztályból származtatott modul. Felső sorjában tartalmaz egy címet leíró *Tkinter.Label*-t, és jobb oldalt a 3.5.1-ben látható segítség mezőt, mely működésben azonos. Ezen sor alá helyez el egy újabb *Tkinter.Frame*-et, melyet a három állapotnak tart fenn. Kezdetben tartalmaz egy *Tkinter.Button* elemet, mely az opciók elfogadására, és a kikérdezés elindítására használatos. A 3.10. ábrán szemléltetve van a kikérdezés menete egy osztály diagrammal.

A kikérdezés beállításának kerete

A *MainFrame/Questioning/QuestioningSetup.py*-ban található

QuestioningSetupFrame osztály valósítja meg, mely a *Tkinter.Frame* osztályt implementálja. Négy fő eleme van, melyek az opciók beállítására használatosak: a kategória és nyelv szűrőre használt **QuestioningDropdown** osztály, és a kezdő nyelvre és nehézségre alkalmazott **QuestioningOptionsMenu** osztály.

QuestioningDropdown

A *GUIAssets/QuestioningSetupDropdown.py* tartalmazza, és a *Tkinter.Menubutton* osztályt implementálja. Opciói a kategóriák vagy a nyelvek lehetnek, melyekhez 0 és 1 értéket rendel a kiválasztás utáni lekérdezésre.

QuestioningOptionsMenu

Szintén a *GUIAssets/QuestioningSetupDropdown.py* fájlban található, a *Tkinter.OptionMenu* osztályból származik. Opciói a nyelvek lehetnek, és a három módifikáció. Működési eltérése a *QuestioningDropdown*-tól, hogy itt csak egy opciót lehet választani.

A kikérdezés kerete

A *MainFrame/Questioning/QuestioningFrame.py* tartalmazza a

QuestioningFrame osztályt, mely a *Tkinter.Frame* osztályból származik. A *QuestioningHandlerFrame* "Indítás" gombjával aktiválódik, és vele együtt a lap címe és segítség mezője is frissül. A keret két részre osztható, bal oldalt elhelyezkedő kérdés elemből - melyet a **QuestionElement** osztály alkot -, és egy "Következő" feliratú léptető gombból áll, míg jobb oldalon helyezkedik el a haladást mérő *Tkinter.Label*-ek, és segítséget igénybe vételhez használatos *Tkinter.Button* gomb elemek.

QuestionElement

A *GUIAssets/QuestionElement.py* tartalmazza. Felülről lefelé helyez el egy a lefordítandó szót leíró szöveget, alatta pedig a kezdő és cél nyelvet reprezentáló *Tkinter.Label* elemeket, melyet egy *Tkinter.Entry* bemeneti mező követ. Gombnyomás esemény esetén a bemeneti mező feketére színezi a benne lévő bemenetet.

Segítség gombok

A meglévő *QuestionElement* objektum bemeneti mezőjét változtatja, vagy az értékét változtatják, vagy pedig a szín opcióját állítják át.

A kikérdezés jelentés kerete

A *MainFrame/Questioning/QuestioningReportFrame.py* fájlban található **ReportFrame** osztály valósítja meg, mely a *Tkinter.Frame* osztályt implementálja. Ha az összes kérdésre válasz lett adva, a *QuestioningHandlerFrame* lebontja az összes elemét, és betölti a *ReportFrame* egy példányát, mely *Tkinter.Label*-ek segítségével írja ki az összes haladásmérési adatot, társítva a kihívás mennyiségéhez a **ttk.Progressbar** elemeket használva. A szavak helyessége mellett található *Tkinter.Button* elemmel nyithatjuk meg az adott kikérdezés szavaira adott választ egy felugró ablakban, melyet a **ReportWindow** osztály kezel.

ReportWindow

A *MainFrame/Questioning/ReportPopup.py* fájl tartalmazza. Egy új *Tkinter.Tk* ablakot nyitva jeleníti meg az adatokat, minden szó elemet egy sorban mutat *Tkinter.Label* elemekkel, és a szavak között **ttk.Separator** elemmel húz elválasztó vonalat.

3.3.4. A Profil keret

A **ProfileFrame** osztály valósítja meg, mely a *Tkinter.Frame* osztályból származik, és a *MainFrame/Profile/ProfileHandler.py* fájlban található. Felosztja a keretet két *Tkinter.Frame* modulra, bal oldalra helyezi a profil adatait, jobb oldalra a kihívásokat.

Az adatok *Tkinter.Label* szövegelemekkel jelennek meg, bal oldalt felülről lefelé haladva a profilnévvel, felhasználói szinttel, szavak számával, nyelvekkel és a szintjükkel, végül pedig a segítségekkel és mennyiségükkel. A *ttk.Progressbar* elem vizualizálja a következő szintekhez való megszerzett és még megszerzendő tapasztalati pontok arányát.

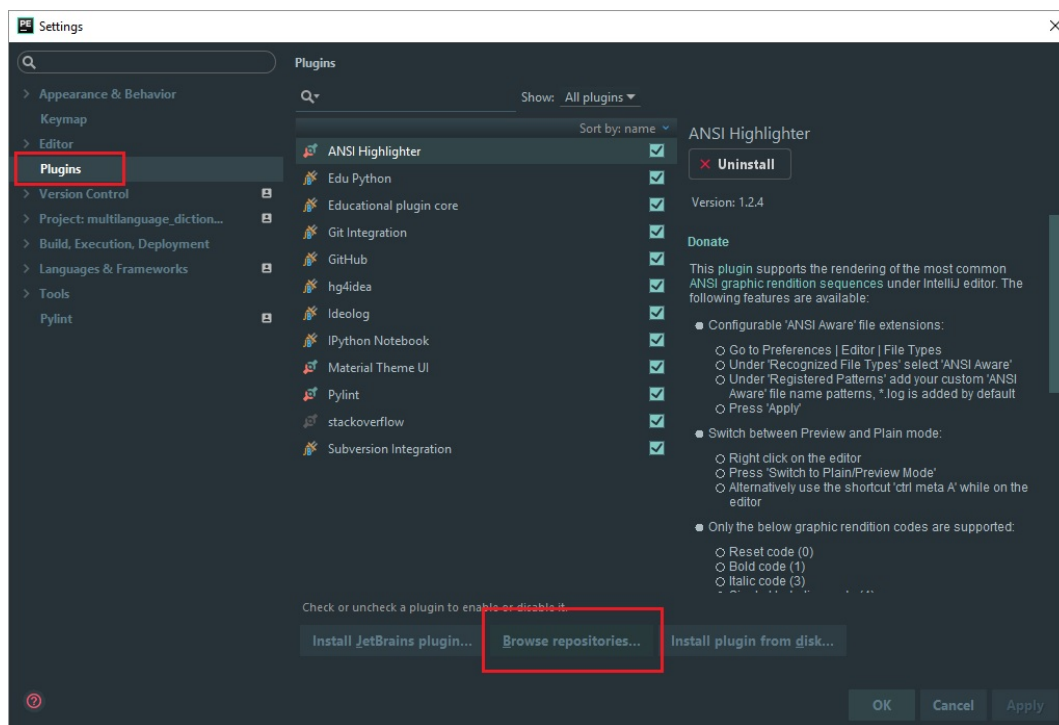
A jobb oldalon elhelyezkedő kihívásokat tároló keret egy *Tkinter.Label* címmel rendelkezik, ami alatt helyezkednek el a kihívások, melyek mindegyike egy *Frame* objektum, amiben *Tkinter.Label* jelzi a követelményt, a haladást és a jutalmat, és egy *ttk.Progressbar* vizualizálja a haladás/szükség hányadost.

3.4. Tesztelés

A helyes működés tesztelése két módszerrel történt. A kód írása során *linter*t használtam, és a működéshez manuális teszteket végeztem felhasználói történetek alapján.

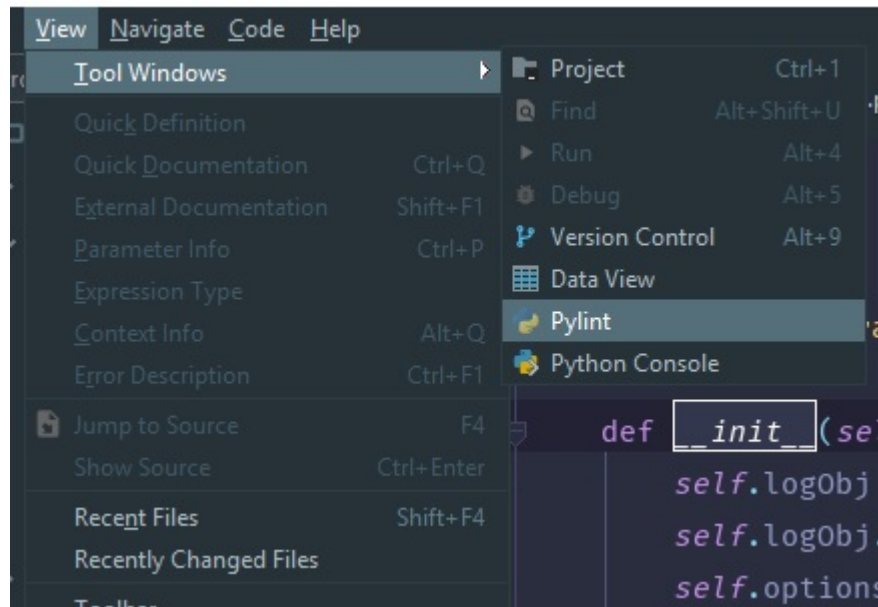
3.4.1. Lintelés

A **Pylint**[8] végzi el ezt, mely egy Java-ban íródott eszköz. Ezt a **Pycharm**[9] azonos nevű bővítményével[10] végeztem. A program írása közben ellenőrzi a kód helyességét, figyelmeztet a hibákra, a tiszta kód elvekre és a **PEP 8**[11] stíluselvekre. Esetemben a figyelem a hibákat ellenőrizte. A lintelés elvégzésére a Pycharm "Setting->Plugins->Browse Repositories..." úton érhetőek el a bővítmények, melyet az alábbi 3.11. ábra pirossal bekeretezett részein követhetünk.



3.11. ábra. A Pycharm bővítményeit kezelő ablak

Telepítés után újraindítás szükséges, majd pedig a View -> Tool Windows -> Pylint (3.12. ábra) kiválasztásával hozhatjuk elő a scan felületet. Itt választhatunk, hogy teljes projectet, adott modult vagy metódust akarunk ellenőrizni.



3.12. ábra. A Pylint aktiválása

3.4.2. Felhasználói történetek

Manuális teszteket végeztem a **Windows 7, 8 és 10** operációs rendszereken. A tesztek kiterjedtek az egész program használatára, új, meglévő, és legacy felhasználói fiókokkal, különböző számítógépeken. A kritikusabb eseteket fekete doboz módszerrel teszteltem és jegyeztem fel:

- Ha az adatbázis elérés megghiúsul, a program jelzi a felhasználónak, majd kilép.
- Ha meglévő felhasználónevet választunk, üresen hagyunk egy kitöltendő mezőt, nem a megengedett karaktereket használjuk, vagy nem megegyező a két begépett jelszó, a regisztráció megghiúsul.
- Ha rossz belépési adatokat adunk meg, a bejelentkezés meg lesz akadályozva.
- Ha új nyelvet adunk hozzá a szótárhoz, nem lehet üres bemenettel elfogadtatni a bevitelt.
- Ha új szót akarunk felvenni, akkor először szükség van nyelvre, amiről a program értesítést ad.
- Ha új szót akarunk felvenni, legalább egy nyelvi mezőt ki kell töltenünk, máskülönben a hozzáadás megghiúsul.
- Ha lapozni szeretnénk a szótáron olyan irányba, mely felé már nincs több megjeleníthető szó, akkor a lapozást nem próbálja megtenni a program.

- Ha megváltoztatunk egy szót, a haladása elveszik, azonban más nyelven a szó párja változatlanul marad.
- Ha két nyelv és/vagy egy szó hiánya nélkül akarunk elindítani egy kikérdezést, a program megakadályozza és értesít bennünket.
- Ha nincs elég szó az adott nehézségi szinthez a kikérdezésnél, a program kevesebbet mutat.
- Ha nincs kikérdezhető szó az adott beállításokhoz, egy felugró ablak értesíti a felhasználót.
- Ha csak egy választott nyelv van, és az megegyezik a kiválasztott kezdő nyelvvel, akkor a kikérdezést felfüggeszti a program.
- Ha negatívba vagy harminc fölé menne egy szó haladásmérője, akkor nem történik változás.
- Ha nem UTF-8 belí karaktert veszünk fel a szavak közé, az adatbázis rendesen el tudja menteni.

4. fejezet

Összegzés

A program kódja tökéletesen megtestesíti a Python erősségeit, és egyben hátrányait. A munkát elején megnehezítette a típus helyesség hibalehetősége, azonban nagyon gyorsan rutinszerű tudássá vált, melyet bármely nyelvben lehet jövedelmeztetni. A nyelvhez tartozó kód stílisztikai követelményei könnyen megszokhatóak, és élvezetes írási élményt biztosít rövid időn belül. A Python egy remek nyelv, mely mind kezdőknek és tapasztalt programozóknak egyaránt ajánlom.

A grafikai megvalósításért felelős Tkinter a gyakorlatban nem teljesített jól. Egyszerű, kis ablakok gyors felépítésére megfelelő, mivel könnyen tanulhatóak az alapjai. Nagyobb, bonyolultabb projekteknél a limitált elhelyezési metódusai nem tudnak lépést tartani a napokban elvárt igényeknek: legyen gyors és egyszerű a kódolása magas szinten is. A Tkinter talán legjobb tulajdonsága a gyorsasága, mely nagy projektek esetén is szinte egyből fel tudott állni.

Végső soron azokat a következtetéseket vontam le, hogy a Python nagyon jól és egyszerűen tud működni, ha a programozó nyitott a kezdeti más nyelvektől eltérő tulajdonságokra, azonban GUI programozásra inkább gondolnám alkalmasnak egy C vagy Java framework-öt.

5. fejezet

Irodalomjegyzék

- [1] Python: <https://www.python.org/> (2019.05.07)
- [2] MariaDB: <https://mariadb.org/> (2019.05.07)
- [3] MySQL.Connector: <https://dev.mysql.com/downloads/connector/python/> (2019.05.07)
- [4] Python 3.6.3: <https://www.python.org/downloads/release/python-363/> (2019.05.07)
- [5] Tkinter: <https://docs.python.org/2/library/tkinter.html> (2019.05.07)
- [6] ttk: <https://docs.python.org/2/library/ttk.html> (2019.05.07)
- [7] configparser: <https://docs.python.org/3/library/configparser.html> (2019.05.07)
- [8] Pylint: <https://www.pylint.org/> (2019.05.07)
- [9] Pycharm: <https://www.jetbrains.com/pycharm/> (2019.05.07)
- [10] Pylint JetBrains plugin: <https://plugins.jetbrains.com/plugin/11084-pylint> (2019.05.07)
- [11] PEP 8: <https://www.python.org/dev/peps/pep-0008/> (2019.05.07)