# Alkalmazott Mesterséges Intelligencia a gyakorlatban

Házi feladat Dokumentáció

VoxControll

Harsányi Zsolt - XORZVV

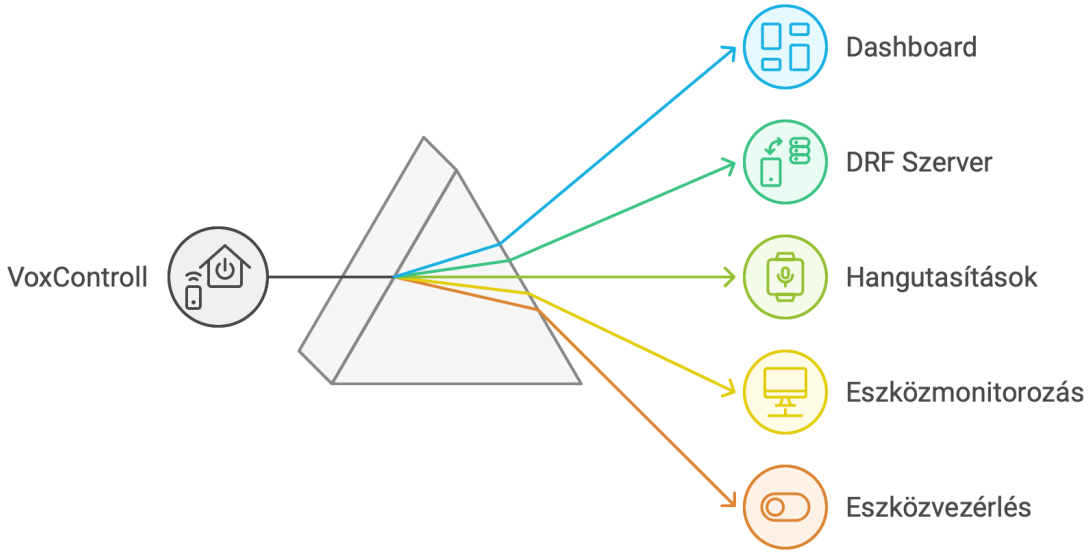
Választott feladat: A35 - Hangalapú vezérlőrendszer fejlesztése

Készíts egy olyan hangalapú vezérlőrendszert, amelyet különböző gépek és eszközök kezelésére lehet használni. A rendszernek képesnek kell lennie a felhasználók által adott parancsok valós idejű feldolgozására és végrehajtására

Github repository: https://github.com/harsnyi/VoxControl

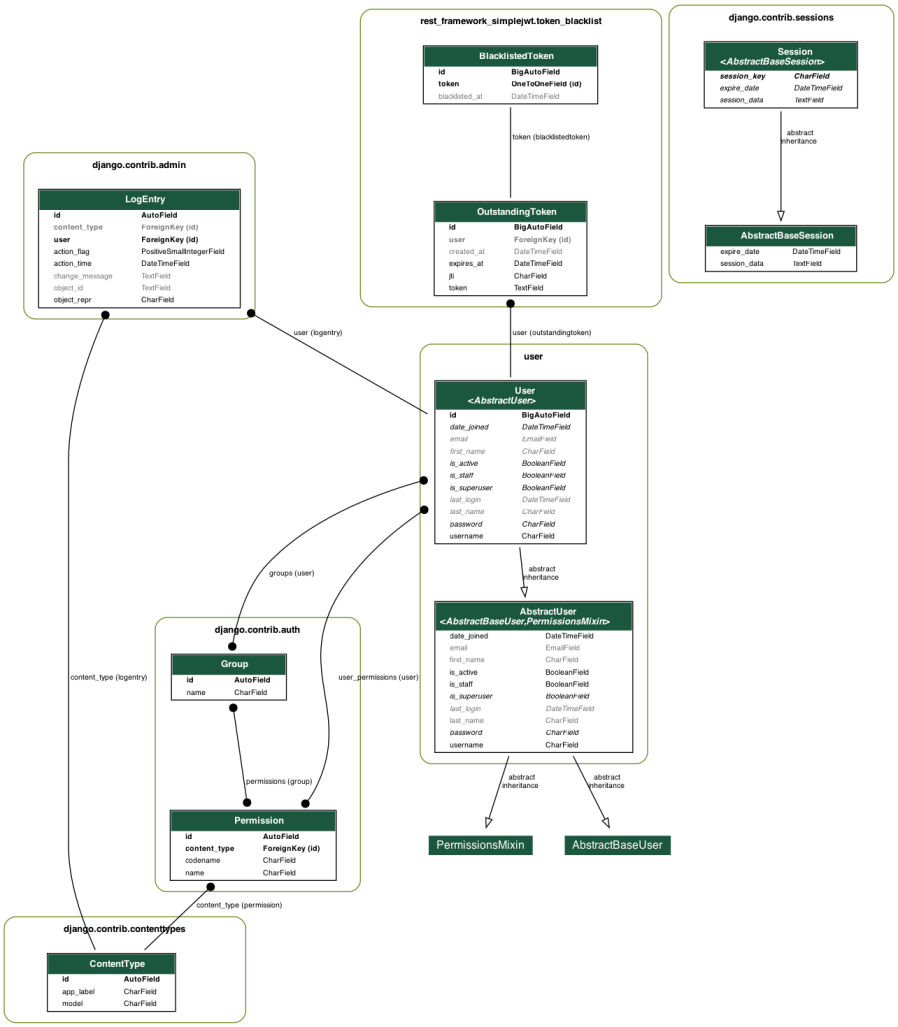
Az elkészített applikációm neve a VoxControll amelynek lényege hogy különböző háztartásbeli, okos eszközt tudjunk irányítani, azokat monitorozni, fel és bekapcsolni. Azért választottam ezt a feladatot mert érdekesnek találtam hogyan lehet ebbe a rendszerbe hangvezérlést beépíteni és hogy miképp tudok esetleg a jövőben valós okoseszközöket a szerveremhez csatolni és azokat hangvezérléssel irányítani.

Az alkalmazás két részből áll, az egyik az úgynevezett Dashboard PyQT5 Pythonos Desktop GUI, a másik egy Django DRF szerver, amelyhez lesznek csatolva a különböző okosotthoni eszközök. A Dashboard segítségével tudunk hangalapú kéréseket felvenni és azok alapján kéréseket küldeni a szerverünk felé, monitorozni az egyes eszközök állapotát, lekapcsolni őket, stb.



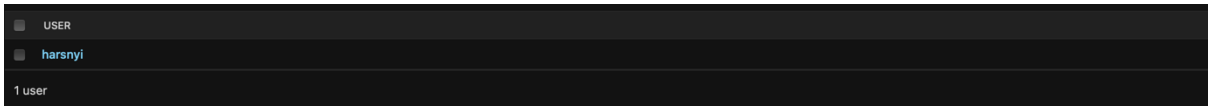

A feladat megoldását két részre bontottam, első körben a DRF szerver elkészítésével kezdtem. A DRF szerver azért ideális választás egy otthoni környezetbe, mert nagyon egyszerűen implementálható, moduláris, a jövőben igen könnyen párhuzamosítható, de ami a legfontosabb, hogy relatívan kevés kóddal lehet igen nagy eredményeket elérni, de ez általában a Python-os ökoszisztémára igaz. A DRF (Django Rest Framework) egyébként a mezei Django keretrendszernek a REST-es kiegészítője amit elsősorban API-k fejlesztésére terveztek. Ebben a fázisban létrehoztam egy felhasználó modelt és a hozzá

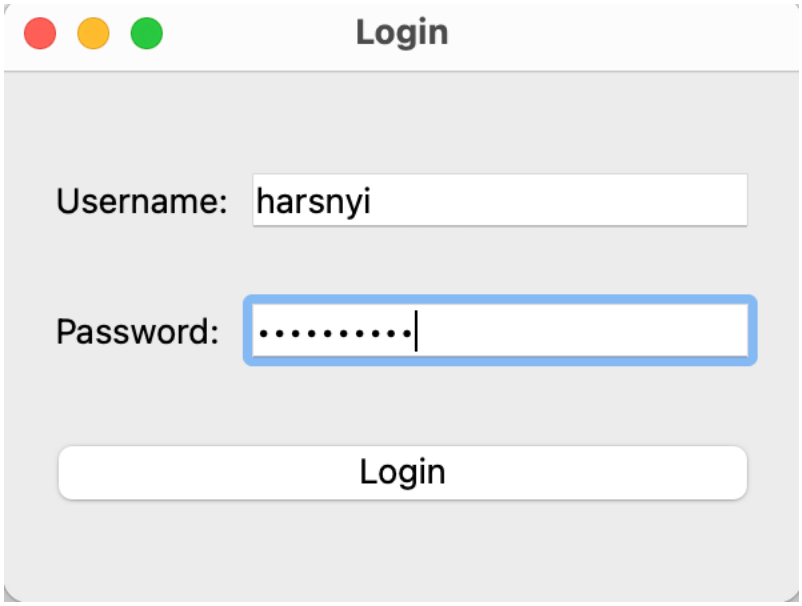
tartozó bejelentkezési végpontokat, stb, hogy a Dashboard-on tudjam majd azonosítani a felhasználókat és hogy véletlen se férjen mindenki hozzá az okosotthoni eszközökhöz. Ezeket az eszközöket manuálisan felvettem a DRF kódba amiknek az állapotát majd a végpontok segítségével lehet monitorozni és változtatni. Ehhez JWT-t használtam. A JWT segítségével sikeres bejelentkezésnél a Dashboard-on a felhasználó kap egy Access tokent aminek segítségével eltudja érni a védett végpontokat és aminek segítségével hozzáférhet az okosotthoni eszközökhöz.



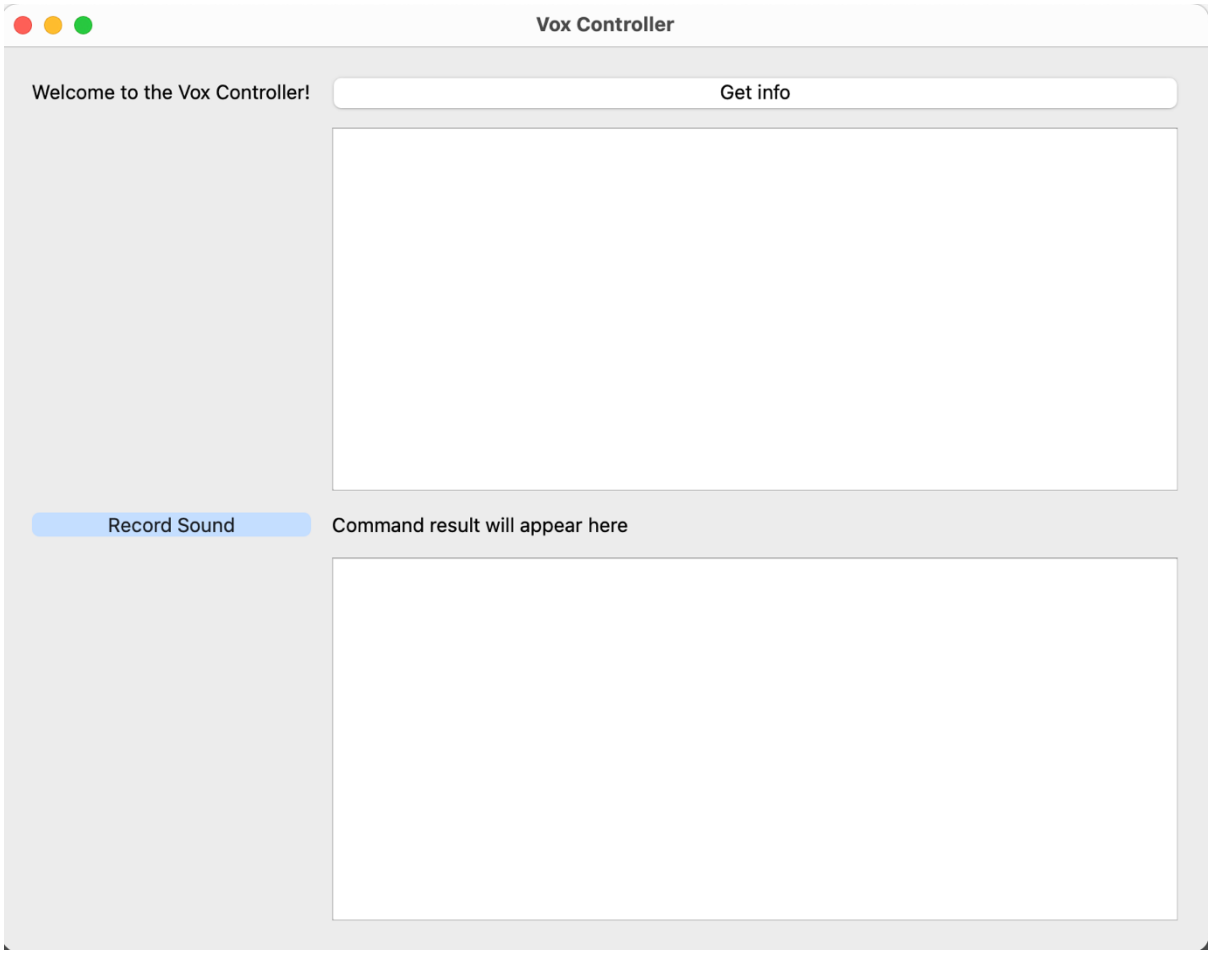

**ábra 1.** A DRF modellek egymáshoz való viszonya az alkalmazásban

Ezután kezdtem el foglalkozni a Dashboard implementálásával aminek feladata lesz az API-val történő kommunikáció megvalósítása. Ezt egy hangalapú vezérlőrendszer segítségével fogja megtenni, amihez először be kell lépni az alkalmazásba. Az authentikációhoz egyelőre csak egy szuperfelhasználót hoztam létre, akit a szerver tárol egy SQLite adatbázisban.

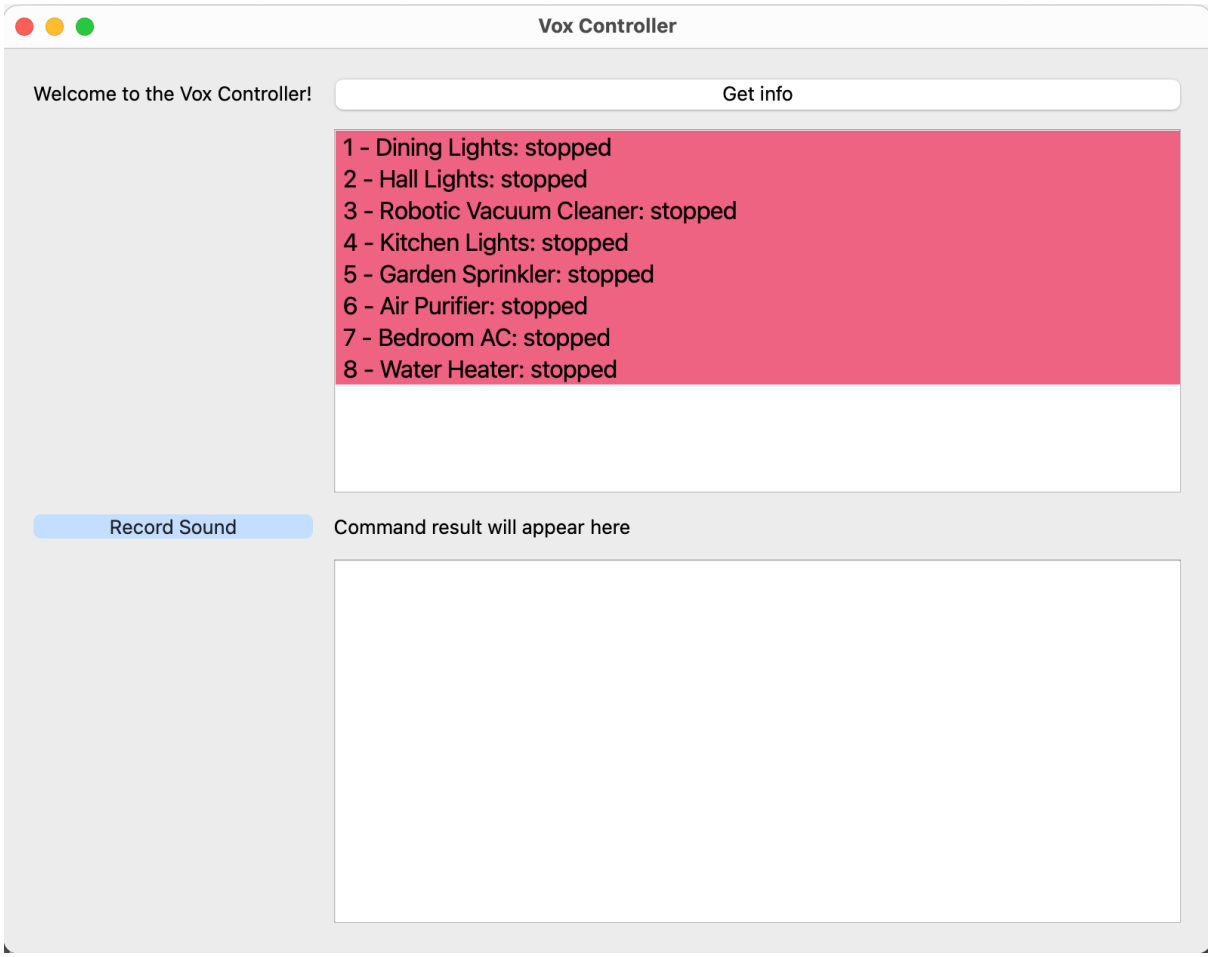



Miután sikeresen begépeltük a felhasználónevünket és jelszavunkat a PyQT alkalmazás átdob minket a Dashboard felületre.




A felület tartalmaz egy Record Sound és Get info gombot, mellette két úgynevezett ListView-ot ahol az akcióink alapján visszajelzést fogunk kapni. A Get info gomb segítségével lekérdezhetjük a szerverről a jelenleg elérhető okosotthoni eszközeinket feltéve ha elérhető a szerver.



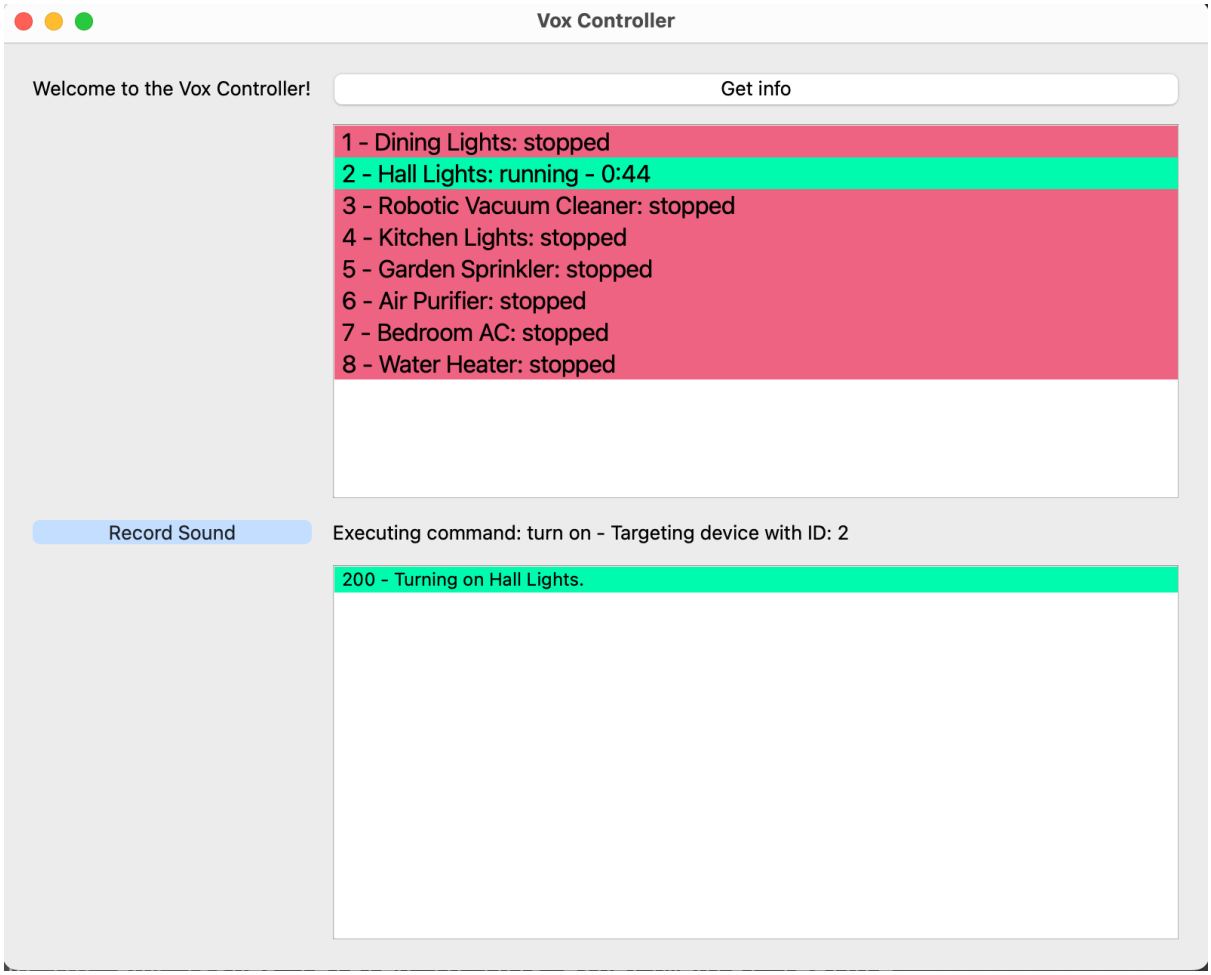

A példában több a normál okosotthonokban megjelenő eszközöket vettem fel a DRF kódjába hogy ezeknek a monitorozását és működtetését tudjam szimulálni. Látható hogy alapesetben mindegyik le van állítva.

Ennek befolyásolására van a másik gomb amelynek segítségével a hangunkat tudjuk rögzíteni és segítségével utasítások mentén fogjuk tudni irányítani az eszközeinket. Ehhez kapni fogunk egy 5 másodperces időablakot ami alatt fel tudjuk vetetni angolul az alkalmazással a hangunkat. Ennek a megoldásához a SoundDevice nevű csomagot használtam aminek segítségével könnyedén tudunk felvenni hangot Python alatt. Ezt többféle paraméter finomításával tudjuk megtenni, mint például a Sample Rate, audio csatornák száma, Bit Rate stb. Ezután a felvett hanganyagot a Speech_recognition segítségével formáltam szöveggé. A SR segítségével többféle, előre betanított Speech-to-text model áll rendelkezésünkre ahhoz hogy az audiót szöveggé konvertáljuk. Én is több modellel kísérleteztem, számomra a legpontosabbnak a Whisper model bizonyosult. Ezután nincs más dolgunk mint utasításokat adni a különböző azonosítóval ellátott eszközeinknek.

Mivel előtte még nem ismertem a PyQT nevű GUI-t, ennek megismeréséhez és több kódolási problémát is LLM, azonbelül ChatGPT-vel oldottam meg. A ChatGPT-nek különböző promptokat tettem fel többnyire a grafikus elemek elkészítéséhez, azok tájolásához és a folyamat során rengeteget tanultam segítségével. Megtudtam például

hogyan kell használni a PYQT Grid elrendezését a különböző komponensekre, hogyan tudok ListView-t definiálni és feltölteni.

Az alkalmazásban jelenleg hangvezérléssel tudjuk felkapcsolni és lekapcsolni a különböző eszközeinket, azonban a parancsok tetszőlegesen és rendkívül egyszerűen bővíthetők a modularitásuk miatt. Például ha sikeresen ismeri fel az algoritmus a Turn On 2 szóláncot a kettes azonosítóval ellátott eszközünk felé fog a Dashboard API kérést küldeni és azt a szerver minden reménnyel felkapcsolni.




Miután elküldte a Dashboard a kérést az API felé az visszatér egy üzenettel ami alapján látni fogjuk hogy sikeres volt-e a kérésünk. Ezt a második ListView-ban látjuk ahol a sikeres (200-as kódú) üzeneteket zölddel fogjuk látni a sikerteleneket (500) pedig piros háttérrel. Sikeres kérés után pedig ha újra lekérjük az eszközeinket akkor látni fogjuk hogy a kettes azonosítóval ellátott eszközünk felkapcsolt állapotba került és mellette pedig azt is monitorozhatjuk hogy mióta fog futni.

Összességében rengeteg mindent tanultam a feladattal és direkt egy olyan feladatot választottam ami számomra abszolút érdekes volt és otthoni környezetben Home Assistant segítségével tovább tudom fejleszteni, éles eszközökkel tudom tesztelni. A

megoldás során igyekeztem úgy definiálni a megoldást hogy az a lehető legkönnyebben átültethető és fejleszthető lehessen egy valós környezetben, vagyis ha például gazdái vagyunk egy Raspberry Pi-nak és az alkalmazás Dashboard részét becsomagoljuk .apk fájlként kisebb finomításokkal egy működő alkalmazást kapunk a mobilunkra amivel az otthoni eszközeinket fogjuk tudni monitorozni.