Rövid beszámoló

Pilot projekt

Az első két hétben mindenkinek egy önálló pilot projektet kellett megvalósítania. Ennek keretein belül kaptunk egy ~500.000 soros csv filet különböző társaságok adatairól, amire sémát kellett kitalálni, fel kellett tölteni, normalizálni kellett, a normalizált adatokból egy kalkulációs nézetet létrehozni, az így kapott nézetet egy OData service felhasználásával hozzáférhetővé tenni, majd valamiféle frontend funkcionalitást hozzákötni. A frontend végül egy Angular projekttel lett megvalósítva, mivel az OData által közétett JSON-t felhasználásánál a Cross-Origin Request Blocked problémába ütköztünk, amit Angularban egy proxy segítségével sikerült megoldani.

További hetek

Pilot projekt - folytatás

A pilot projekten belül többféle részfeladatot is végzett mindenki, azt ezt követő szakosodáshoz tapasztalatot adandó. Én magam részéről a modellezéshez jelentkeztem. Az elkövetkező időszakban túl sok útmutatást nem kaptunk, hogy mi lenne a további feladat, így sokáig a megkezdett pilot projekten dolgoztam tovább.

Az eredeti „végleges” állapotban az oldal megjelenítette a rekordokat, illetve egy sor végén levő gombra kattintva a Bing térkép API-ját felhasználva a rekordhoz tárolt címből megjelenített egy statikus térképet rajta a címhez tartozó gombostűvel. Először is módosítottam az OData hozzáférést, ami addig egy HTTPkliens alapú service volt, ehelyett egy o.js nevű NodeJS modult használtam fel. A térképen való megjelenítés eddig csak a címből történt, ehelyett az adatbázisban külön táblába szedtem a címet, és hozzá tároltam el a default null szélességi-hosszúsági koordinátákat, és amikor egy címet a térképet jelenítenénk meg, akkor először a tárolt koordinátákat vizsgáljuk meg, ha ezek null értékűek, akkor először az API segítségével geokódoljuk őket, az eredményt visszatöltjük a tárolt címhez, majd a tárolt koordinátákat megjelenítjük. Létrehoztam külön egy gombot, amivel a kódolás műveletét az összes tárolt címre végrehajtjuk és feltöltjük, amennyiben nem tartoznak hozzájuk koordináták.

A koordináták tárolásával megvizsgáltam az azonos koordinátákhoz tartozó címeket. Sajnos ez nem túl sok lehetőséget adott a címek algoritmikus egységesítésére, mert bár pár címnél működött volna, összeségében sok lett volna az adatveszteség a részletekben, mint pl. emelet/ajtószám. Ezen kívül az adatokat próbáltam a koordinátákból való visszakódolással tisztítani, szintén eredménytelenül, mert a részletek megint csak elvesztek, sőt, sokkal nagyobb eltérések is előjöttek, úgy, hogy az új címekből újrakódolt koordináták nem is egyeztek az eredeti címből kódoltakkal. Ennek eredményét őrzi az adatbázis CIM_REGI_UJ táblája.

Szenzorok

A negyedik hét környékén fel lett dobva egy új feladatötlet, amiben szenzoradatokkal kellett volna foglalkozni. Sok ötletem az előző projekt bővítésére ekkorra már nem volt, így ezzel mentem tovább. Kaptunk egy évről hónapok szerint 12 adatfájlt, amiben a hónap összes óráját jelölő sorokban az összes mért szenzor mérési adatai (hőmérséklet, légnyomás, páratartalom, porszemcse méretek) voltak. Ezt első körben egy c++ program segítségével (ennek eredménye az src\szenzor\WeatherParser) ezeket az adatokat szétbontottam, hogy a sorok időpont és szenzorazonosító szerint legyenek bontva, majd ezeket feltöltöttem az adatbázisba. Később volt egy rövid indítványozás a többi modullal való együttműködésre, név szerint azokkal, akik a sap prediktív analitikai könyvtárát (PAL) próbálgatták, ezért létre lett hozva egy olyan tábla is az adatokból, amelyik ezeknek az algoritmusaira illeszkedik, ami a gyakorlatban annyit jelent, hogy

tartalmaz egy Integer egyszerű kulcsot a többi mező mellett, illetve a bármelyik mezőben NULL értékéket tartalmazó rekordok kiszűrésre kerültek.

Ezekből a táblákból létre lett hozva egy-egy kalkulációs nézet, amik közzé lettek téve egy OData service-en keresztül, majd hozzájuk lett valamennyi funkcionalitás kötve egy újabb Angular projekttel. Ez egyrészt megjeleníti az adatokat, a NULL értékeket még a sablonban kezelve, a megjelenítésnél ezen felül lehet szenzorazonosító szerint szűrni, másrészt lehetőséget ad adatok feltöltésére az eredeti adattáblába (amelyik nem tartalmazza az egyszerű Integer kulcsot), ammenyiben GET paraméterekként megadjuk legalább a 'Time' és 'Sensor' értékeket. Az így feltöltendő rekord először egy átmeneti Uploads táblába kerül. Ez annyiban különbözik séma szintjén az adattáblától, hogy az időt String-ként tárolja a dátumformátum helyett. Ez azért szükséges, mert a kalkulációs nézet az adatok eredeti dátumot tartalmazó helyett egy számolt, Stringgé konvertált mezőt jelenít meg, mivel az OData a dátumot a háttérben tárolt Integerként jelenítette meg, így viszont a feltöltés nem tud egy az egyben további trükközés nélkül megtörténni. Az Upload táblába felkerült rekordot egy trigger feltölti az eredeti adattáblába is, majd egy újabb trigger az egyszerű kulcsos táblába is.