**Eötvös Loránd Tudományegyetem**

**Informatikai Kar**

**Informatikatudományi Intézet**

**Programozáselmélet és Szoftvertechnológia Tanszék**

Wireless Tools Analytics

Szerző: Témavezető:

Bezzeg András Benjamin Dr. Nikovits Tibor

Programtervező informatikus BSc. mesteroktató

**Budapest , 2023**

Ide kerül a hivatalos témabejelentő lap.

Tartalomjegyzék

[1. Bevezetés 1](#_Toc154085327)

[1.1. Rövid kontextus: 1](#_Toc154085328)

[1.2. Motiváció: 1](#_Toc154085329)

[1.3. Az alkalmazás: 2](#_Toc154085330)

[2. Felhasználói dokumentáció 3](#_Toc154085331)

[2.1. Projekt struktúra 3](#_Toc154085332)

[2.2. Futtatási lépések: 3](#_Toc154085333)

[2.2.1. Leírás: 3](#_Toc154085334)

[2.2.2. run-script.js 4](#_Toc154085335)

[2.2.3. Extra tudni valók: 5](#_Toc154085336)

[2.3. Felhasználói felület: 5](#_Toc154085337)

[2.3.1. Beállítások 5](#_Toc154085338)

[2.3.2. Navigálás 5](#_Toc154085339)

[2.3.3. Landing page 5](#_Toc154085340)

[2.3.4. Total page 5](#_Toc154085341)

[2.3.5. Per Tool page 6](#_Toc154085342)

[2.3.6. Compare page 6](#_Toc154085343)

[3. Fejlesztői dokumentáció 8](#_Toc154085344)

[3.1. Leírás 8](#_Toc154085345)

[3.2. Fejlesztési terv- és folyamatok: 8](#_Toc154085346)

[3.2.1. Technológiák 8](#_Toc154085347)

[3.3. Projektstruktúra leírása: 9](#_Toc154085348)

[3.4. Adatbázisrétegek 9](#_Toc154085349)

[3.4.1. Docker file 9](#_Toc154085350)

[3.5. Szerver 9](#_Toc154085351)

[3.6. Kliens 10](#_Toc154085352)

[3.7. Futtatási lépések fejlesztőknek: 10](#_Toc154085353)

[3.8. Adatbiztonság és Teljesítmény 10](#_Toc154085354)

[3.9. Összegzés 10](#_Toc154085355)

[4. Összefoglalás és további fejlesztési lehetőségek 11](#_Toc154085356)

[5. Irodalomjegyzék 12](#_Toc154085357)

[6. Melléklet 13](#_Toc154085358)

# Bevezetés

## Rövid kontextus:

A cégünk vezetéknélküli kommunikációs chipek tervezésével , szabadalmaztatásával és gyártásával foglalkozik. Mindemellett biztosit egy integrált fejlesztési környezetet ahol a felhasználó tesztelheti ,megfigyelheti, konfigurálhatja vagy egyéni kódot is írhat ezekre a hardware-ekre. (Továbbiakban ***Studio*** néven erre referálok). Ebben a Studio-ban számos saját és globális vezetéknélküli kommunikációs protokol támogatására van lehetőség, azonban nem mindenki képes hosszú és bonyolult C kód írására, a cég ezeket a felhasználókat is figyelembe véve hozta létre a *Wireless* *Tooling* csapatot aminek énis a tagja vagyok. A célunk átlagos vagy kevésbé hozzáértő felhasználók részére is lehetővé tenni, hogy egyszerűen generálhassanak hardware kódot. Ehhez különböző webes applikációkat hozunk létre ami az adott technológiához tartozó kódot képesek generálni egy SDK segítségével. Ezeket a webes applikációkat hívjuk ***Tool-***oknak.

## Motiváció:

Ezek a **Tool**-ok tehát egy külső alkalmazásba integrált (**Studio**) webes alkalmazások, aminek a használatáról logolás készül egy felhőben lévő adatbázisba. Ezeket a logokat fogom **Esemény**-nek hívni a továbbiakban

A képen szöveg, képernyőkép, menü, monokróm látható

Automatikusan generált leírás

. ábra: "Esemény" minták

Mivel ezekenk a Tool-oknak a száma amivel a csapatunk foglalkozik mára bőven kétszámjegyűvé nőtt, elérkezett az igény, hogy ezeket a használati szokásokat feldolgozva, feltérképezhessük és láthatóvá tegyük melyik projekt milyen súlyú hasznot hoz a belefektetett munka arányában.

Ennek a kivitelezésével lettem megbízva a gyakornoki időszakom alatt és ez ihlette a szakdolgozat témáját.

## Az alkalmazás:

A célom tehát egy olyan alkalmazás elkészítése ami ezeket az adatokat felhasználva segíti és információval látja el először a csapatatot majd a menedzsmentet. Ehhez szintén egy webes alkalmazást fogok elkészíteni ami egy belső hálózaton való elérésre lesz alkalmas.

Tehát felhasználók kizárólag cég alkalmazottak lesznek.

Kisebb-nagyobb eltérések akadnak az eredeti verzióval szemben. Az adatokat véletlenszerűen generáltattam apró átalakításokkal és jelentősen kisebb mennyiségben.

A képen szöveg, képernyőkép, Diagram, szoftver látható

Automatikusan generált leírás

2. ábra: Alkalmazás minta

A projekt struktúrájának kialakításakor és a technológiák kiválasztásakor a standalone lokálisan futtathatóság és a teljes és egyszerű verziókezelés volt a fő szempont.

# Felhasználói dokumentáció

## Projekt struktúra

A képen szöveg, képernyőkép, Betűtípus, szoftver látható

Automatikusan generált leírás

. ábra: Forráskód mappa struktúra

/db/

Ebben a mappában találhatóak az adatbázis elindításához szükséges file-ok. A legfontosabb egyből a gyökérben lévő **docker-compose.yml**, ami a docker konténer leíró file-ja és a docker-comopose CLI parancs alapértelmezett inputja. Ennek tartalmát a fejlesztői dokumentációban részletezem. Ebben található egy mock nevezetű mappa is , itt találhatóak az inicializáló SQL szkriptek. Két almappával rendelekezik:

* *db/mock/data:* Ennek a mappának a tartalma automatikusan felkerül az elindított konténerbe és a benne lévő összes \*.sql kiterjesztésű file futtatásra kerül „abc” sorrendben. Ezek alkotják az „alap” tábláinkat.
* *db/mock/runtime-scripts:* Ennek a mappának a tartalma szintén felkerül egy előre definiált mappába a konténeren belül , azonban a futtatására manuálisan van szükség. Ezek alkotják a „kisegítő” tábláinkat.

A képen szöveg, képernyőkép, Betűtípus, tervezés látható

Automatikusan generált leírás

4. ábra: Db mappa lenyitva

/server/

Itt található a szerver futtatásához tartozó forráskód. Ezt is a fejlesztői dokumentációban fogom kellően részletezni. A legfontosabb file a gyökérben található main.ts, amit előszőr lekell fordítani tiszta javascript kódra , majd futtatásával indíthatjuk el a szerver szolgálltatást alapértelmezetten a 9000-es porton.

A képen szöveg, képernyőkép, szoftver, Betűtípus látható

Automatikusan generált leírás

. ábra: Lenyitott server mappa

/client/

Itt a tool\_analytics\_frontend mappában található egy quasar struktúrájó frontend projekt ami a kliensként szolgál. Részletezni itt is a fejlesztői dokumentációban fogok, de minden a framework struktúráját követi aminek a részletes dokumentációja elérhető a <https://quasar.dev/> című honlapon.

A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Automatikusan generált leírás

. ábra: Lenyitott client mappa

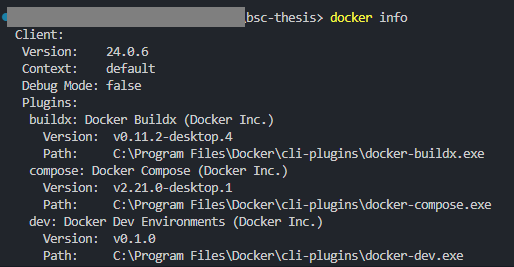
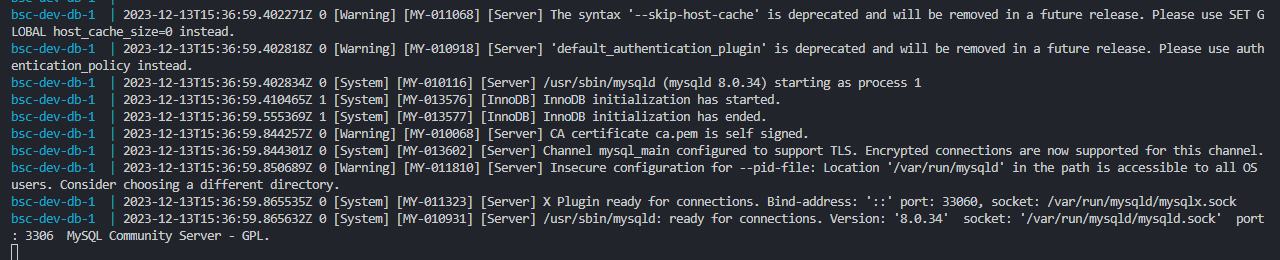
## Futtatási lépések:

A kicsomagolást követően a wireless\_tools\_analytics mappában találhatóak az alkalmazás forráskódjai. Három fontos dologgal kell rendelkeznie a felhasználónak:

* Node.js futtatási környezet
  + <https://nodejs.org/en/download>
* Npm package manager
  + <https://docs.npmjs.com/downloading-and-installing-node-js-and-npm>
* Futó docker daemon
  + Docker-Desktop
    - <https://www.docker.com/products/docker-desktop/>
  + Rancher Desktop , ha gond a liszenszelés
    - <https://rancherdesktop.io/>

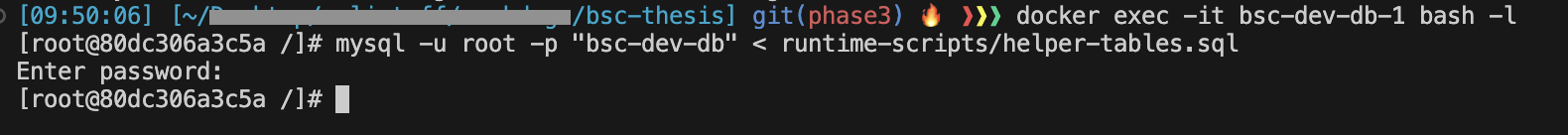
### Leírás:

A futtatás egy többlépéses parancssor eredménye lesz: TODO KÉPEK

1. Indítsuk el a docker-container-t
   1. Nyissunk meg egy terminált
   2. Győződjünk meg, hogy fut a docker daemon
   3. Navigáljunk a db mappába
   4. Adjuk ki a ’docker-compose up’ parancsot

. ábra: Sikeres docker-compose utolsó üzenetei

1. Generáljuk le a segéd táblákat
   1. Sikeres docker container indítás után nyissunk egy új terminált ( az előzőt hagyjuk megnyitva!)
   2. Adjuk ki a `docker exec -it bsc-dev-db-1 bash -l` parancsot
   3. Ezzel megnyitottunk a konténerünkben egy terminált, itt adjuk ki a `mysql -u root -p "bsc-dev-db" < runtime-scripts/helper-tables.sql` parancsot
   4. Adjuk meg a docker-compose.yml-ben definiált jelszót („pwd” alapesetben)
   5. Sikeres futás esetén zárjuk be ezt a terminált



. ábra: Sikeres runtime-scripts futtatás

1. Fordítsuk le és futtassuk a szerver kódot
   1. Navigáljunk a server mappába
   2. Győződjünk meg róla, hogy rendelkezünk node és npm parancsokkal
   3. Adjuk ki az `npm ci` parancsot
   4. Adjuk ki az `npm run start` parancsot , ez elvégzi mindkét lépést
   5. Siker esetén hagyjuk nyitva ezt a terminált is
2. Futtasuk a kliens kódot
   1. Navigáljuk a client/tool\_analytics\_frontend mappába
   2. Adjuk ki az `npm ci` parancsot
   3. Adjuk ki az `npm run dev` parancsot
   4. Siker esetén ezt a terminált is hagyjuk nyitva

### run-script.js

A fejlesztés elejétől terveztem , hogy ezt a bonyolult folyamatot a lehető legegyszerűbb formára szeretném hozni.

Ezért alternatívaként, készítettem egy extra modult. A szkrpit maga egy javascript-es szálkezelésre épülő szinkronizált parancsvégrehajtó modul , amivel a fenti parancsokat hajtuk végre egy terminálban. Szükséges a fent említett három dependencia ebben az esetben is.

Használatához csak lépjünk be a fő mappába egy terminálból és adjuk ki a `node runtime-script.js` nevű parancsot. A program tájékoztatni fog éppen melyik komponensen dolgozik és hogy ez milyen eredménnyel zárult, ha valami rész sikertelenül zárul, az egész leáll. Felszabadítja maga után a portokat, egy terminált foglal el és egyedül a docker container-t nem törli bezárás/ újra indításkor, azt manuálisan lehet meg tenni ha szükséges.

### Extra tudni valók:

A db mappa tartalmaz egy markdown file-t ahol a docker-specifikus extra információk találhatóak.

## Felhasználói felület:

A felület elsődleges célja a feldolgozott adatok interaktív és könnyen értelmezhető bemutatása-összehasoníltása. Minden oldal erre törekszik ,de más megközelítésből. Lehetőség van a teljes adathalmazból készült kimutatások megtekintésére, vagy Tool specifikusan megtekinti és összehasonlítani az adatokat. A felületen emellett pár kisegítő lehetőség is adott, mint például a nyelv válltás és sötét módra válltás és tooltip dobozok.

### Beállítások

Az oldalak tetején találhat egy sáv a címmel. Ennek a fenti sávnak a jobb szélén található egy fogaskerék gomb mely kattintásra megjeleníti a beállítások ablakot, amiben válthatunk sötét/világos mód és angol/magyar nyelv között. A sötét mód lecseréli a háttér színeket egy sötét árnyalatúra és minden szöveget fehérre állít.

### Navigálás

A képen szöveg, képernyőkép, Betűtípus, embléma látható

Automatikusan generált leírás

. ábra: Navigációs menü

Az oldal tetején található egy sáv amiben a projekt címe látható. Kattintásra ez a cím hozza elő a navigációs ablakokat. Három oldal közül választhatunk, mindegyik más adatok megjelenítésére szolgál. Ezeket fogom most részletezni:

### Total page

A teljes eszköztárról készült kimutatásokat találhatjuk itt és ez szolgál jelenleg landoló oldalként TODO

### Per Tool page

A képen képernyőkép, szöveg, szoftver, diagram látható

Automatikusan generált leírás

. ábra: Per tool oldal

Ezen az oldalon kiválaszhatunk egy Tool-t és csak a rá specifikus adatokból készült diagrammokat láthatjuk. Az oldal tetején van egy lenyitható ablak amiben kiválatszhajtuk a Tool-t az elérhetőek listájából. Majd ezután láthatjuk ahogy dinnamikusan megjelennek az adatok a lentebb lévő eddig üres diagrammokon. A legalsó földrajzi adatokat bemutató diagramm itt is lenyitásra jelenik meg.

### Compare page

A képen szöveg, szoftver, diagram, Számítógépes ikon látható

Automatikusan generált leírás

. ábra: Compare oldal

Több kiválasztott eszköz összehasonlítására is lehetőség van. Az oldal tetején található buborékokból kiválasztva kis kártyákon jelennek meg a Tool-hoz köthető főbb jellemzők. (Teljes esemény szám, első használat éve , növekvési trend). Mellete pedig az összefelhasználásból kapott diagramm látható.

# Fejlesztői dokumentáció

## Leírás

Az Wireless Tools Analytics egy komplex (***háromrétegű kliens-szerver*** architektúrájú) rendszer, amely adatbázis-kezelést használva lehetővé teszi az adatok hatékony és strukturált kezelését. Az alkalmazás széles körű lehetőségeket biztosít az adatok lekérdezésére, feldolgozására és megjelenítésére különböző grafikonokon keresztül.

A tervezés során számos fontos kérdés merült fel a megvalósítást illetően. Ezek külön fel vannak tüntetve a rétegek részletes jellemzésénél. Fontos szempontnak találtam a könnyű és lokális kezelhetőséget ezért az alkalmazás kiegészült egy futtatást segítő szkrip-tel is. Ez segíti az egyébként aszinkronikus parancsok szinkronizált végrehajtását.

Az adatbázist egy lokális docker konténer fogja képezni, mivel így külsős service-k nélkül , offline is teljeskörű funkcionalitást érhetünk el. Emellett az adatbázis állapota is könnyen verziókezelhetővé vált inicializáló szkriptek segítségivel. Biztosítva a konzisztenciát különböző platformok és futtatási környezetek között.

Verziókezeléshez github-ot használtam.

### Projektstruktúra leírása:

## Futtatási lépések fejlesztőknek:

## Fejlesztési terv- és folyamatok:

A projekt elkészítését három részre osztottam , mindegyiknek külön git branch-e van , amiket befejezéskor vissza mergeltem a main branch-re:

1. fázis: Tervezés , technológiák kiválasztása , 3 modul alap szerkezetének elkészítése
2. fázis: Modulok összekötése és részletes kidolgozása
3. fázis: Utólagos módosítások , tesztelés , dokumentálás

### Technológiák

|  |  |  |
| --- | --- | --- |
| Adatbázis | Szerver | Kliens |
| * Docker * MySQL | * Express (Node.js) * Express MySQL plugin | * Quasar * Chart.js ( Vue ) * Pinia * Axios |

## Adatbázisrétegek

### Docker file

itt mutasd be pl docker kompose file-t , milyen kommandok vannak , milyen nehezség volt a custom init script, mi az a volume , miért azt használod , miért nem alap initben van a runtime script , stb...

Az adatbázis réteg két fő részre bontható:

* **Első réteg:**A fő relációs adatbázisunk, ezek tartalmaznak minden beérkezett eseményt aminek a naplózásából a kimutatást végezzük, plusz a hozzájuk tartozó részletes információkat és azok kapcsolótábláit.  
  Ezek inicializálása létrehozása a docker konténer létrehozásakor automatikusan megtörténik.Ehhez a ***db/mock/data*** folder tartalmazza a szükséges sql parancsokat.  
    
  A réteg további 3-ra bontható:
  + EventLog tábla: Ez a fő táblánk, itt történik a naplózás minden esemény kiváltásakor.Megtudhatjuk belőle az esemény idejét, milyen parancsot és melyik eszközből kapott. Emelett refernciákat is tárol a felhasználó, lokáció és szoftver adatokról.
  + Információs táblák (User,Location,Software): Ezekben tároljuk a számontartott felhasználókat, lokációkat és szoftver információkat.
  + Kapcsoló táblák: Összekapcsolják a hozzájuk tartozó információs tábla egy elemét egy use session-el(StudioInstance). Ezen párok kerülnek referenciaként egy eseményhez.
* **Második réteg:**   
  A segéd táblák , ezek szolgálnak a kimutatásokhoz tartozó csoportosított lekérdezések ereményeivel. Erre azért volt szükség mivel az éles adatbázis nagyon nagymennyiségű adatot tárol, amik lekérdezése időbe telik és ha ez direkt a kliens kérésére történne az rontaná a reszponzivitását. Ehhez mountoljuk a db/mock/runtime-scripts mappa tartalmát, amit a konténer konzoljából futtathatunk a létrehozását követően.

## Szerver

Az alkalmazás lehetőséget biztosít az adatok dinamikus lekérdezésére a segéd táblákból és kisebb direkt lekérdezésekből. Főbb feladatai:

* **Kommunikáció az adatbázissal: a kapcsolat létrehozása , fenttartása**
* **Adatok feldolgozása: a tábla lekérdezések eredményeinek feldolgozása a megjelenítésnek alkalmas módon**
* **API biztosítása a kliens számára: ahol a feldolgozott adatokat lekérdezhetjük a megjelenítés oldalán**

Emelett session szinten saját magának is tárol segéd táblákat amik a feldolgozást és a címkézést segítik. Pl.: Évek , éppen használt eszközök , stb…

## Kliens

Az alkalmazás egyszerű és felhasználóbarát felhasználói felülettel rendelkezik, amely lehetővé teszi a felhasználók számára az adatok intuitív és könnyű kezelését, valamint a kívánt grafikonok kiválasztását és testreszabását az adatok megfelelő megjelenítéséhez.

## Adatbiztonság és Teljesítmény

Az alkalmazás maximális prioritást ad az adatbiztonságnak és az adatintegritásnak, miközben magas teljesítményt és gyors adatelérést biztosít a háromrétegű adatbázis struktúrájának hatékony kihasználásával.

itt mond h pl bevan egetve a credential a szerverbe

## Összegzés

Az X alkalmazás lehetővé teszi a felhasználók számára, hogy könnyen és hatékonyan navigáljanak a háromrétegű adatbázisban tárolt információk között. A grafikonok változatos megjelenési lehetőségei révén az adatok vizualizációja segíti a felhasználókat a döntéshozatalban és az adatelemzés során.

# Összefoglalás és további fejlesztési lehetőségek

* runtime script logolás debugolás részletesebben
* runtime script paraméterezés
* komponens futtatási paraméterezés : pl portok , mysql jelszavak stb
* mysql adatok beégetve vannak
* több lekérdezés , diagramm
* több összehasonlítás

# Irodalomjegyzék

# Melléklet