

# Etteplan MORE Backend-ennakkotehtävän käyttöohje

## 1. Projektin käynnistäminen

Projektin toimivuutta on helpointa tarkistaa avaamalla lähdekoodi Visual Studio 2017 tai 2019, ja ajamalla se sieltä käsin. Projekti suositellaan käynnistettävän Debug-tilassa (rakentamalla painamalla pelkästään F5), koska projektille ei ole tehty käyttöliittymää, ja ainut tapa miten sen voi sammuttaa olisi muuten tehtävienhallinnasta manuaalisesti. Debug-tilassa projektin voi sammuttaa suoraan Visual Studiassa.

Ennen kuin ajat projektin, kopioi muistiin projektikansiossa löytyvästä EtteplanMORE.ServiceManual.ApplicationCore -kansiossa olevan HuoltotehtDB.mdf-tiedoston tiedostosijainti.

Esimerkiksi omalla tietokoneellani se on

*"D:\Työnhakutesti\servicemanual\_csharp\_master\EtteplanMORE.ServiceManual.ApplicationCore\HuoltotehtDB.mdf"*

Kun tämä sijainti on tallessa, avaa lähdekoodista tiedosto FactoryDeviceService.cs Visual Studiolla. Tiedosto löytyy ApplicationCoren kansioista Services.

Tiedoston rivillä 16 (tai niillä seuduilla) lukee muuttujan "connectionString" määrittely. Tähän tekstiin pitää kopioida HuoltotehtDB.mdf-tiedoston polku. Polku kopioidaan heti " AttachDbFilename=" -kohdan ja ";Integrated Security=True" kohdan väliin.

Kopioinnin jälkeen muista lisätä tupla-\ merkit jokaiseen kohtaan, missä on yksittäinen \-merkki.

Esimerkki oikein kirjoitetusta connectionString-muuttujasta käyttäen ylläolevaa polkua:

```
readonly static string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=D:\\Työnhakutesti\\servicemanu
al_csharp_master\\EtteplanMORE.ServiceManual.ApplicationCore\\HuoltotehtDB.mdf;
Integrated Security=True";
```

Yritin tehdä connectionStringistä dynaamisen, jotta se osaisi etsiä tietokantatiedoston itse suoraan projektikansioista, mutta se kulutti niin paljon aikaa eikä yrittämäni ratkaisut toiminut, joten minun oli pakko tehdä tämä ratkaisu näin.

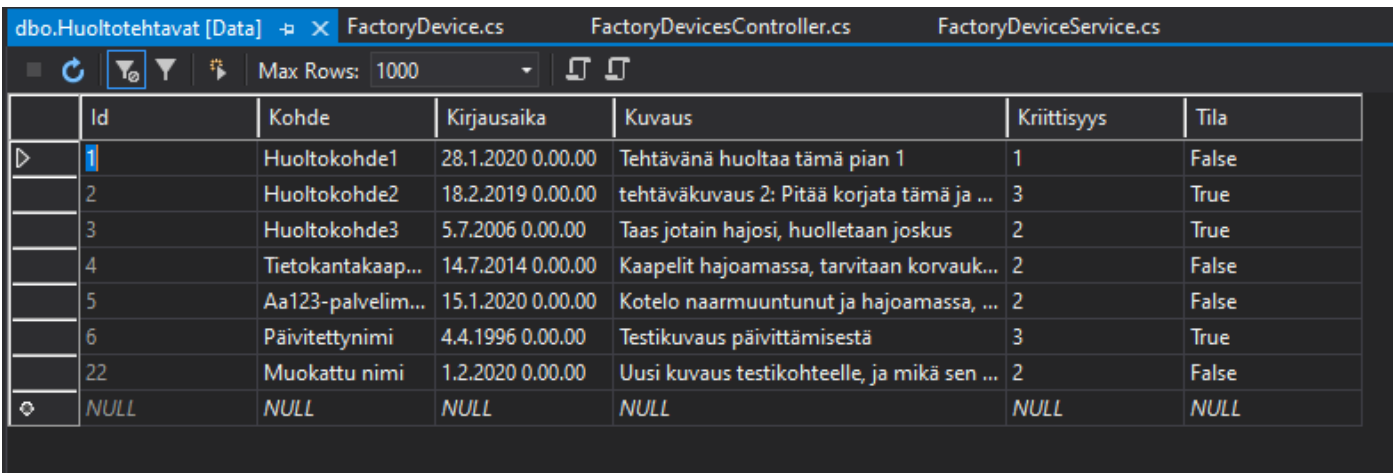
Kun käynnistät Debug-muodossa projektin, pitäisi selaimeen avautua ikkuna, joka ottaa yhteyttä localhostiin portin 50143 kautta (<http://localhost:50143/>). Siitä tiedät, että sovellus on käynnissä.

## 2. Selitykset tietokannasta

Projektin tiedot on tallennettu HuoltotehtDB.mdf-nimiseen SQL-tietokantatiedostoon. Tietokanta sisältää yhden taulun nimeltä Huoltotehtavat, ja taulu sisältää kuusi kolumnia:

- Id (int, määritelty Identity. automaattisesti kasvaa uusien syötteiden myötä)
- Kohde (nvarchar(50))
- Kirjausaika (datetime)
- Kuvaus (nvarchar(200))
- Kriittisyys (int, 1 = kriittinen, 2 = tärkeä, 3 = lievä)
- Tila (boolean, 0 = avoin, 1 = huollettu)

Tietokantaa voi tarkastella ja muokata Visual Studiossa. Tuplaklikkaamalla tietokannan tiedostoa Solution Explorerissa ottaa automaattisesti yhteyden siihen.



	Id	Kohde	Kirjausaika	Kuvaus	Kriittisyys	Tila
▶	1	Huoltokohde1	28.1.2020 0.00.00	Tehtävänä huoltaa tämä pian 1	1	False
	2	Huoltokohde2	18.2.2019 0.00.00	tehtäväkuvaus 2: Pitää korjata tämä ja ...	3	True
	3	Huoltokohde3	5.7.2006 0.00.00	Taas jotain hajosi, huolletaan joskus	2	True
	4	Tietokantakaap...	14.7.2014 0.00.00	Kaapelit hajoamassa, tarvitaan korvauk...	2	False
	5	Aa123-palvelim...	15.1.2020 0.00.00	Kotelo naarmuuntunut ja hajoamassa, ...	2	False
	6	Päivitetynimi	4.4.1996 0.00.00	Testikuvaus päivittämisestä	3	True
	22	Muokattu nimi	1.2.2020 0.00.00	Uusi kuvaus testikohteelle, ja mikä sen ...	2	False
⚙	NULL	NULL	NULL	NULL	NULL	NULL

Kuva 1 Tietokannan sisältö testivaiheessa

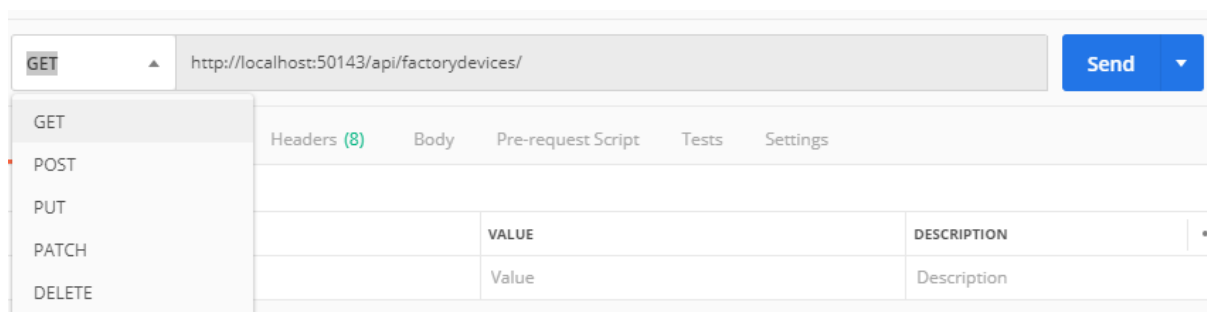
### 3. Tietoa projektista

- Projekti on rakennettu tehtävänannossa tarjotun ServiceManual-projektin päälle
- Projektissa on tehtävänannon mukaisesti jätetty pois autentikaatiot, autorisaatiot, ja muut turvallisuustoimet (kuten kunnollinen SQL-injektioilta esto), ja keskitytty toiminnallisuuden rakentamiseen
- Projektipohjan mukana tulleita yksikkötestejä ei ole muokattu, joten ne ei todennäköisesti toimi uuden projektin kanssa.
- Koodi on kommentoitu, mutta jos on kysyttävää, voi sähköpostitse kysyä lisää.

### 4. Projektin testaaminen

Kun projekti on Visual Studiossa laitettu käyntiin, voi sitä testata Postmanilla (<https://www.getpostman.com/>).

Seuraavaksi listataan kaikki kutsut mitä projektille voi tehdä. Ne listataan HTTP-metodien mukaan. Tämä metodi valitaan Postmanissa URL-syötekentän vasemmalta puolelta.



pohja-URL, johon kutsut laitetaan, on <http://localhost:50143/api/factorydevices/>

GET:

- Hae kaikki tietokannan rivit: <http://localhost:50143/api/factorydevices/>
- Hae yksi rivi tietokannasta: <http://localhost:50143/api/factorydevices/><huoltotehtävän id>
- Hae kohteen nimen perusteella tietokannasta: <http://localhost:50143/api/factorydevices/><kohteen nimi>
  - o Kohteen perusteella hakiessa tietokannasta haetaan kaikki kohteet, jotka vastaavat <kohteen nimi>-parametria.
- Kaikki haut järjestetään automaattisesti ensin kriittisyyden mukaan vakavimmasta lievimpään, ja sitten päivämäärän mukaan uusimmasta vanhimpaan.

#### POST:

- <http://localhost:50143/api/factorydevices/>
  - Syötettävät tiedot laitetaan kutsun kehoon, katso kansiossa oleva kuva POSTluonti.png
  - Avataan Body-välilehti, asetetaan asetukseksi raw, ja laitetaan sisällön tyyppi JSON
  - Kirjoitetaan Bodyyn syötettävät tiedot, kuten kuvassa. Kirjausajan ja Id:n asettaa sovellus

#### PUT:

- <http://localhost:50143/api/factorydevices/> <muokattavan huoltotehtävän id>
  - Muokattavan tehtävän Id menee URL:ään
  - Kutsun kehoon laitetaan muokattavat tiedot, kaikkia tietoja voi muokata paitsi Id:tä ja Kirjausaikaa. Lisää kuvassa PUTpäivitys.png

#### DELETE:

- <http://localhost:50143/api/factorydevices/> <poistettavan huoltotehtävän id>
  - Poiston tulos kuvassa DELETEpoisto.png