**Harjoitustyön vaiheen 1 palautus**

**Johdanto**

En ole aiemmin paljoa koodaillut tai tuottanut ohjelmia. Pyrin luomaan avointa dataa rajapintojen avulla hyödyntävän koostepalvelun, koska tällaiseen tarkoitukseen löytyy paljon valmiita rajapintoja. Netistä myös löytynee Googlaamalla vinkkejä rajapintojen käyttään Djangolla ja Visual Studiolla.

**Käyttämäni teknologia ja ympäristö**

Kehitysympäristö: Django

Ohjelmointikieli: Python 3.8.1

Editori: MS Visual Studio Code

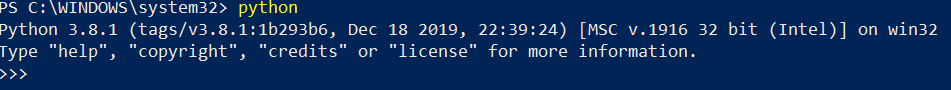
Tietokanta: SQLite

Käyttöjärjestelmä: Windows 10

Valitsin Djangon, koska sitä suositeltiin aloittelijoille. Ohjelmointikielenä käytän Pythonia, koska olen sitä käyttänyt aiemmin toisella kurssilla ja sitä käytetään myös tämän kurssin koodiklinikoilla. Jälkimmäisestä syystä valitsin myös Visual Studion editoriksi. SQLite on oletuksena Djangossa tietokannaksi, enkä näe syytä vaihtaa sitä.

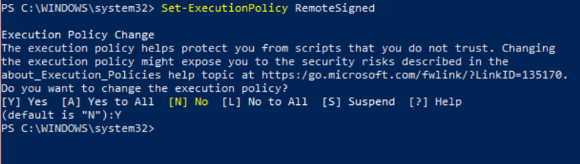
**Pythonin ja Djangon asennus sekä kehitysympäristön perustaminen**

Aluksi latasin Pythonin koodiklinikan ohjeiden mukaisesti. Tämä onnistui, koska tietokoneeltani löytyi asennuksen jälkeen Pythonin uusin versio ja Powershell tunnisti kielen ja version kuvan 1 mukaisesti.



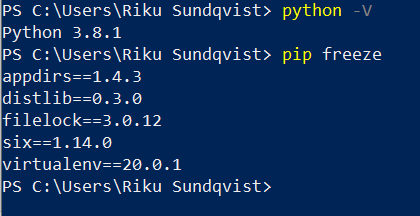
Kuva 1: Pythonin tunnistaminen Powershellissä

Seuraavaksi loin Powershellillä harjoitustyölle virtuaaliympäristön haluamaani tiedostosijaintiin. Aluksi sallittiin tietokoneelle scriptien suorittaminen antamalla kuvan 2 mukainen komento ja antamalla lupa (”Y”) Powershellin varmistaessa olenko varma.



Kuva 2: Scriptien salliminen

Pythonin toimivuus testattiin jo alussa, mutta testataan vielä pipin toimivuus *pip freeze* -komennolla.



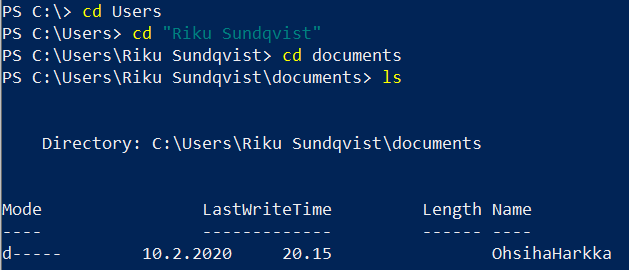
Kuva 3: Pip toimii

Seuraavaksi asennettiin virtuaaliympäristö komennolla:

*pip install virtual env*

Seuraavaksi loin harkkatyötä varten kansion haluamaani tiedostosijaintiin. Valitsin koodiklinikan esimerkin mukaisesti kansiolle sijainnin C:\Users\Riku Sundqvist\documents. Powershellissä kansioden välillä navigoitiin komennoilla *cd [kansion nimi*] haluamaansa kansioon ja *cd ..* vei takaisin päin. Halutessaan saatavilla olevat kansiot ja tiedostot sai näkyviin komennolla *cd ls*, joka tulosti listan sen hetkisen sijainnin alavalikosta.

Tässä kohtaa merkittäviä vaikeuksia seurasi siitä, että nimelläni oleva kansio (Riku Sundqvist) käyttäjissä sisältää välilyönnin, jota ei käsitelty koodiklinikalla. Komento *cd Riku Sundqvist* tuotti siis virheviestin, koska *Riku* nimistä kansiota ei ole olemassa. Tätä pähkäilin hyvin kauan, kunnes Googlesta lopulta selvisi, että minun täytyy käyttää heittomerkkejä välilyönnin sisältävää kansiota käsitellessä. Oikea komento oli siis *cd ”Riku Sundqvist”* . Seuraavaksi loin ”OhsihaHarkka” -nimisen kansion komennolla *mkdir OhsihaHarkka* .

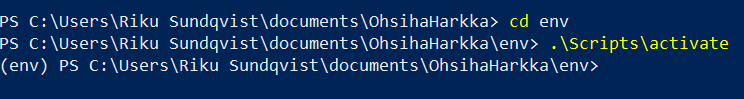


Kuva 4: Harkkatyökansioni sijainti ja sinne navigointi. Erityishuomio heittomerkkien käytölle välilyönnin sisältävää kansion nimeä käsiteltäessä.

Viimeinen komento virtuaaliympäristön luomiseksi oli *virtualenv env*, jossa *env* oli ympäristölle antamani nimi. Virtuaalympäristö käynnistettiin komennolla

*.\Scripts\activate*

Onnistuneen käyttöönoton merkiksi Powershellin komentoriviin ilmestyi *\env* -pääte.



Kuva 5: Virtuaaliympäristön käynnistys

Seuraavaksi asennettiin Django komennolla *pip install django*. Lisäksi Visual Studion käyttäjänä asensin PyLintin, joka on Python -koodauksessa apuväline virheiden tarkastamiseen. Asennus tapahtui komennolla *install pylint*.

Seuraavaksi loin Harkkatyö -nimisen Djangoprojetkin komennolla *django-admin startproject Harkkatyö*. Ennen tätä menin kansiossa yhden askeleen taaksepäin komennolla *cd ..* .

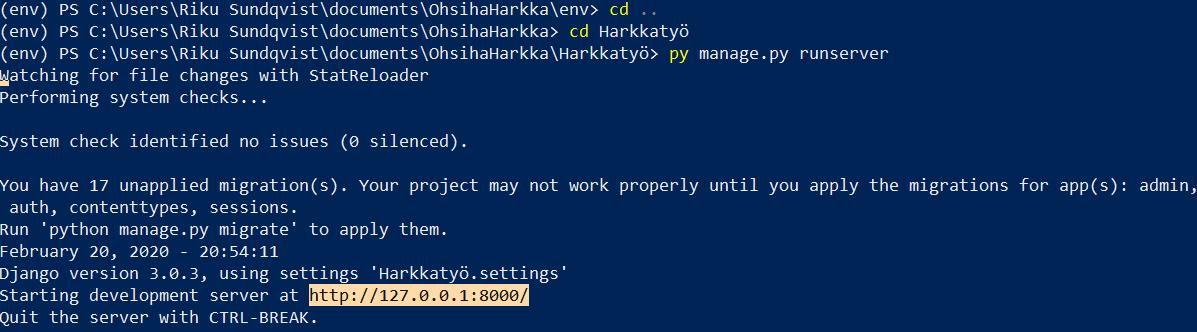
Seuraavaksi asensin Python lisäosan Visual Studioon. Tämän jälkeen avasin projektin Visual Studiossa komennolla

*code .* (Huom! välilyönti ennen pistettä)

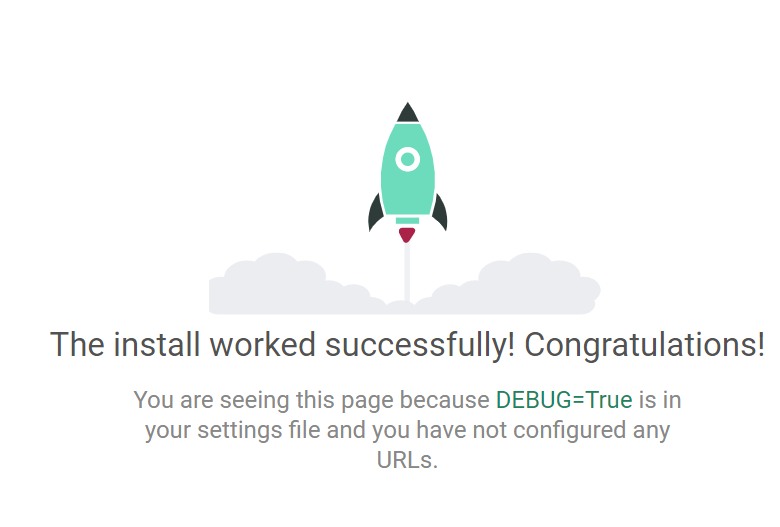
Ennen testausta valittiin vielä oikea tulkki:

ctrl+shift+p -> Python select interpreter

Testatakseni ympäristön toimivuuden navigoin takaisin projektiini ja ajoin Powershellissä komennon *py manage.py runserver*. Seuraavaksi syötettiin saatu URI (korostettuna kuvassa 6) selaimeen ja saatiin kuvassa 7 näkyvä ilmoitus ympäristön virheettömästä toiminnasta.



6: Komennot ympäristön testaamiseksi

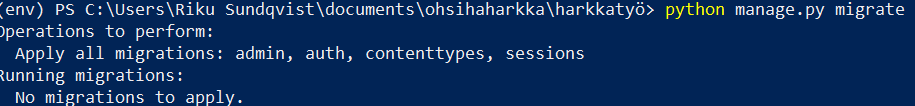


7: Selaimeen syötetyllä URIlla näkyvä viesti toimivasta virtuaaliympäristöstä

Seuraavaksi luotiin django applikaatio komennolla *py manage.py startapp applikaatio* ja tehtiin koodiklinikan mukaiset muutokset tarvittaviin kansioihin.

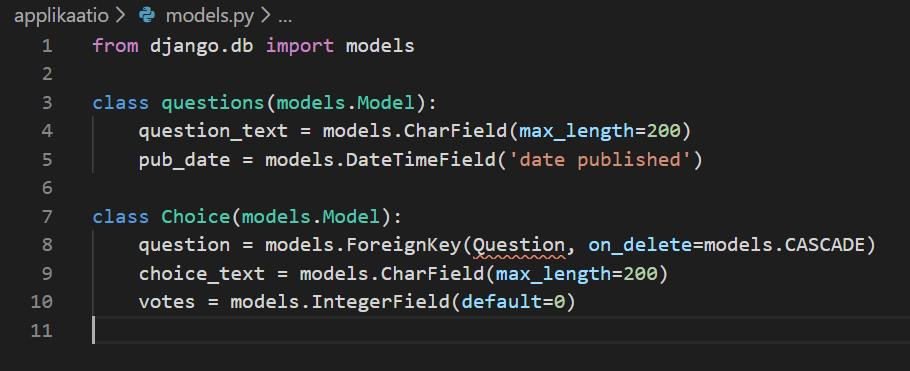
**Tietokannan käyttöönotto**

Otettiin SQLite oletustietokannaksi käyttöön komennolla *python manage.py migrate* kuvan 8 mukaisesti.



Kuva 8: Asetetaan oletustietokannaksi SQLite

Seuraavaksi tehtiin tarvittavat muutokset applikaation *models.py* tiedostoon, joka käsittelee tietokantoihin liittyvät seikat:



Kuva 9: Tarvittavat muutokset models.py tiedostoon

Muutokset otettiin käyttöön Powershellillä komennolla *py manage.py makemigrations applikaatio*. Viimeisenä vielä testattiin toimivuus komennolla *python manage.py shell*. Ainoa virhe oli kuvan 10 mukainen NameError, jota en saanut ratkaistua.



Kuva 10: virhe tietokantaa testatessa

**Helppoa:**

* Ohjeistus koodiklinikalla selkeää, mikä helpotti mukana pysymistä ympäristöä luodessa
* Powershellin komennot jäivät nopeasti mieleen
* Visual Studion käyttö helppoa selitteineen ja automaattisine täyttöehdotuksineen komentoja kirjoittaessa.

**Vaikeaa:**

* Pitää olla huolellinen: jokainen kirjoitusvirhe on kriittinen lopputuloksen kannalta Powershelliä ja Visual Studioa käyttäesä.
* Pitää tarkastaa, että tekee oikeat komennot oikeassa kansiossa.
* Jos putoaa kärryiltä koodiklinikkaa seuratessa, joutuu kelaamaan paljon taakse, että pääsee taas kiinni