Ohsiha harjoitustyö, vaihe 1

1. Harjoitustyön funktio

Harjoitustyön tarkoituksena on muodostaa yhdistysten käyttötarkoitukseen soveltuva web-pohjainen kulukorvausjärjestelmä. Tarkoitus on muodostaa interaktiivinen järjestelmä, johon pystytään syöttämään kulukorvaushakemuksia mahdollisimman helppokäyttöisesti.

Kulukorvausjärjestelmään liitetään käyttäjäautentikointi ja luodaan yhdistyksen hallitukselle keinot kirjautumalla seurata kulukorvauksia ja ottaa ne käsittelyyn. Kulukorvausjärjestelmä tallentaa tiedot tietokantaan. Ohjelmaan on tarkoitus lisätä myös analytiikkamahdollisuus, jonka avulla voidaan esimerkiksi seurata kulujen jakaantumista henkilöiden ja ajankohdan perusteella.

Lisäideoita on mahdollinen Telegram Chatbot, jonka kautta käyttäjät voi mahdollisimman matalalla kynnyksellä syöttää kulukorvaushakemuksia.

2. Valitut tekniikat

Harjoitustyö toteutetaan käyttäen web-viitekehyksenä Djangoa. Tämä on valittu suurimmaksi osaksi kattavan dokumentaationsa vuoksi, sekä aikaisemman Python -ohjelmointi kokemukseni takia. Pythonista käytetään versiota 3.6.3. ja Djangosta versiota 2.0.1. Harjoitustyössä olen päättänyt käyttää harjoituksen vuoksi PostgreSQL-tietokantaratkaisua. PostgreSQL:stä käytetään versiota 10.2.

Olen valinnut kurssilla esitellyn Atomin tekstieditoriksi, sillä se tuntuu erittäin monikäyttöiseltä alkutestauksen perusteella ja intuitiivisesti väittäisin tästä olevan apua harjoitustyön edetessä myöhempiin vaiheisiinsa. Versionhallintaan käytän GitHubia, sillä ko. alustaan tutustuminen tuntuu yhä relevantimmalta.

3. Asennusprosessi

Tässä kappaleessa käydään läpi valittujen tekniikoiden asennusprosessi, siltä osin, kun niitä on komentoriviltä käsitelty. Niin sanottuja ”tavallisia” setup-tyylisiä asennuksia ei käydä tässä harjoitustyössä läpi. Djangoon liittyvä asennusprosessi noudattaa hyvin pitkälti teknologiademo-luennon kaavaa yhdistettynä Djangon tutoriaaleihin.

3.1. Paketinhallinta & virtuaaliympäristö

Latasin paketinhallintaa varten pip-paketinhallintaohjelmiston Pythonia varten. PIP-laajennuksen sain päivittämällä Pythonini uusimpaan versioon. Virtuaaliympäristö ladattiin komentorivin kautta komennolla teknologiademon ohjeiden mukaisesti

```
pip install virtualenv
```

3.2. Django -web-kehys

Django asennettiin komennolla

```
pip install Django
```

Tämän jälkeen varmistuin Djangon asennuksen onnistumisesta kutsumalla python-tulkkia komentorivillä komennolla *py.* Tulkkiin syötin seuraavat komennot

```
import Django

print(Django.get_version())
```

3.3. PostgreSQL

PostgreSQL 10.2. asennettiin koneelle lataamalla EnterpriseDB:n setup. Tietokannan vaihto Djangon oletustietokannasta (sqlite) toteutettiin seuraamalla Django-girlsin ohjeistusta. Tietokanta vaihdettiin harjoitustyön hakemiston settings.py -tiedostoon seuraavaksi:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'kulukorvaukset',
        'USER': '*****',
        'PASSWORD': '*******',
        'HOST': 'localhost',
        'PORT': '5432',
    }
}
```

Tämän jälkeen asennettiin harjoitustyön hakemistoon paketti, joka mahdollistaa PostgreSQL:n ja Pythonin välisen kommunikaation. Tämä toteutettiin aktivoimalla virtuaaliympäristö ja ajamalla komennot:

```
easy_install C:\psycopg2.exe
python -c "import psycopg2"
```

Tämän jälkeen toteutetaan muutokset virtuaaliympäristöstä Djangoon komennolla.

```
py manage.py migrate
```

```
(OHSIHA~1) C:\Users\Teme\Google Drive\Kouluhommat\TITE\ohsiha_HT>py manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying sessions.0001_initial... OK
```

Muutokset onnistuivat, joka tarkoittaa, että harjoitustyön hakemisto saa yhteyden tietokantaan. Tietokannalle luotiin vielä lopuksi pääkäyttäjä komennolla:

```
py manage.py createsuperuser –username *****
```

4. Tähän mennessä toteutettu toiminnallisuus

Tietokantaan yhdistämisen jälkeen projektitiedostoon luotiin sovellus *login*, johon toteutetaan palveluun kirjautuminen ja myöhemmässä vaiheessa myös käyttäjäautentikointi. Tätä varten tiedostoon */login/models.py* luodaan alustava pohja tietokantarakenteelle. Tätä tullaan muokkaamaan reilusti rekisteröitymistä luotaessa.

```
class Palautus(models.Model):
    palauttaja = models.CharField(max_length=200)
    palautuspvm = models.DateTimeField('korvauksen pvm')
```

Rakenne aktivoidaan *settings.py* -tiedostossa lisäämällä *INSTALLED_APPS* -listaan rivi:

```
'login.apps.LoginConfig'
```

Muutokset tallennettiin projektihakemistoon ja tietokantaskeemaan komennoilla:

```
py manage.py makemigrations login
py manage.py sqlmigrate login 0001
```

Aikaisemmin luodulla pääkäyttäjällä voidaan kirjautua palveluun djangon sisäänrakennetun admin-toiminnallisuuden avulla. Palvelu käynnistetään ensin paikallisella palvelimella alla olevalla komennolla. Tämän jälkeen avaamalla osoite http://127.0.0.1:8000/admin/ selaimessa, saadaan järjestelmävalvojan näkymä käyttöön.

```
py manage.py runserver
```

4.1. Käyttäjän kirjautuminen

Käyttäjän kirjautumisen luominen toteutettiin mukaillen Django Login/Logout -tutoriaalia. Projektihakemiston *urls.py* -tiedoston *urlpatterns* -listaan lisättiin seuraavat rivit ja haettiin kirjastosta vaadittavat komennot.

```
from django.contrib import admin
from django.urls import include, path
urlpatterns = [
    path('admin/', admin.site.urls),
    path('login/', include('django.contrib.auth.urls')),
]
```

Tämän jälkeen login-sovelluksen hakemistoon perustettiin uusi kansio rekisteröitymistä varten. Tänne luotiin yksinkertaisten HTML-tiedostojen avulla hyvin

raaka kirjautumissivusto, joka kirjautuessa vie käyttäjän sivustolle, joka tervehtii kirjautujaa. Tätä varten tuli *settings.py* -tiedoston *TEMPLATES* -listaan lisätä rivi:

```
'DIRS': [os.path.join(BASE_DIR, 'templates')],
```

Urls.py -tiedoston *urlpatterns* -listaan lisättiin vielä seuraava rivi, joka ohjaa käyttäjän aloitussivulla koti.html -kirjautumissivulle.

```
path('', TemplateView.as_view(template_name='koti.html'), name='koti'),
```

Tästä on huomattava, että mikäli sivun oletustunnisteen perään ei laiteta mitään, ohjaa sivusto silloin kirjautumissivulle. Mikäli kirjoitetaan tunnisteen perään esimerkiksi */admin,* ohjautuu käyttäjä järjestelmänvalvojan kirjautumiseen. Kirjautumissivujen HTML-tiedostot ovat toistaiseksi perusrakenteeltaan kehitetty William Vincentin tutoriaalin perusteella, mutta toiminnallisuuden lisääntyessä niitä tullaan muokkaamaan.

5. Kolme helppoa/hankalaa tekijää

1. Djangon valinta helpotti merkittävästi projektissa alkuun pääsyä, sekä kattavan dokumentaationsa ja tutoriaaliensa avulla sai aikaan ”näkyviä” tuloksia hyvin pian, kuten esimerkiksi kirjautumissivun. Tämä osaltaan motivoi, sekä auttoi ymmärtämään syy-seuraussuhteita.
2. PostgreSQL voi osoittautua harjoitustyön myöhemmässä vaiheessa helpottavaksi tekijäksi, mutta tietokantaan yhdistämisessä tuli hyvin paljon erilaisia ongelmia (esim. polut virtuaaliympäristöön, Django piti asentaa uudestaan kohdehakemistoon jne.). Näistä yli pääseminen kuitenkin kasvatti osaamista merkittävästi.
3. Atom tekstieditorina on helposti ymmärrettävä ja lähestyttävä, sekä monien liitännäistensä avulla siitä on muodostunut tehokas ja kokonaisvaltainen työkalu.

6. Lähteet

Tärkeimpinä lähteinä toimivat seuraavat:

Ohjelmallinen sisällönhallinta – luentodiat + teknologiademot

Djangon dokumentaatio

Djangon Polls-sovelluksen tutoriaali

William Vincentin Django-tutoriaali käyttäjän kirjautumiseen ja autentikaatioon

Django Girls -tutoriaali: PostgreSQL installation

Näiden lisäksi ongelmissa on turvauduttu etenkin Stackoverflown ohjeisiin.