# HARJOITUSTYÖN 2. VAIHE: TUKI KIRJAUTUMISEEN JA REKISTERÖITYMISEEN

## YLEISTÄ

Harjoitustyön toisessa vaiheessa oli tarkoitus luoda esimerkkipalvelu valitsemallani teknologialla. Päätin luoda tuen kirjautumiselle sekä rekisteröinnille sekä CRUD-toiminnot. Päädyin luomaan yksinkertaisen muistiinpanotoiminnon, sillä netistä löytyi hyvät linkit toiminnon rakentamiseen sekä muiden kurssilaisten raportteja selailemalla toiminnot vaikutti kaikista helpoimmalta toteuttaa. Lähdin rakentamaan toista vaihetta jo ensimmäisessä harjoitustyön vaiheessa luotujen toimintojen päälle.

## TUKI KIRJAUTUMISELLE SEKÄ REISTERÖITYMISELLE

Loin tuen kirjautumiselle *LearnDjango* -nimisen sivuston ohjeiden avulla. Aluksi loin *templates* -kansion sisälle kansion nimeltä *registration*. Tämän jälkeen *registration* -kansion sisälle loin tiedoston *login.html*. Tiedostoon kirjattiin kuvassa 1 näkyvät rivit.

applikaatio > templates > registration > login.html > a

```
1   {% extends 'base.html' %}
2
3   {% block  title %}Login{% endblock %}
4
5   {% block content %}
6   <h2>Sisäänkirjautuminen</h2>
7   <form method="post">
8       {% csrf_token %}
9       {{form.as_p}}
10      <button type="submit">Kirjaudu</button>
11  </form>
12  <a href="{% url 'signup' %}">Rekisteröidy täällä!</a>
13  {% endblock %}
```

**Kuva 1**. *login.html* -tiedosto.

Tämän jälkeen tein seuraavan päivityksen *settings.py* -tiedostoon. Tiedostoon kirjoitettu komento *” ’DIRS’: [os.path.join(BASE_DIR, ’templates*)]* kertoo *Djangolle templates* -kansion sijainnin.

```
55  TEMPLATES = [
56      {
57          'BACKEND': 'django.template.backends.django.DjangoTemplates',
58          'DIRS': [os.path.join(BASE_DIR, 'templates')],
59          'APP_DIRS': True,
60          'OPTIONS': {
61              'context_processors': [
62                  'django.template.context_processors.debug',
63                  'django.template.context_processors.request',
64                  'django.contrib.auth.context_processors.auth',
65                  'django.contrib.messages.context_processors.messages',
66              ],
67          },
68      },
69  ]
70
71  WSGI_APPLICATION = 'Projekti.wsgi.application'
```

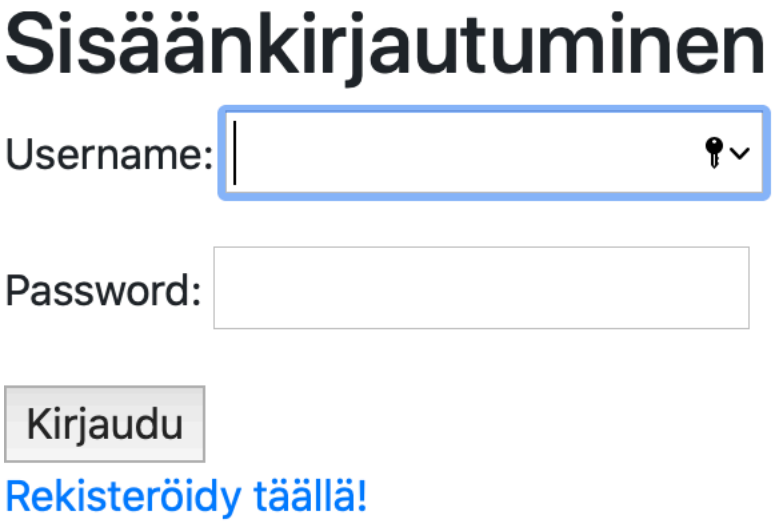
**Kuva 2.** *settings.py* -tiedosto.

Komennon jälkeen saman *settings.py* -tiedoston loppuun lisäsin komennot *”LOGIN_REDIRECT_URL = ’home’”* ja *”LOGOUT_REDIRECT_URL = ’home’”* ohjaamaan käyttäjä kotisivulle kirjautumisen jälkeen.

```
123  LOGIN_REDIRECT_URL = 'home'
124  LOGOUT_REDIRECT_URL = 'home'
125
```

**Kuva 3.** *settings.py* -tiedoston loppuosa.

Tämän jälkeen selaimessa avautui sisäänkirjautumisnäkymä.




**Kuva 4.** Selainnäkymä.

Kirjautumisnäkymä ei vielä tässä kohtaa ole toiminnallinen ja rekisteröitymisen tuki puuttuu.

Tämän jälkeen tein *base.html* sekä *home.html* -tiedostot. *base.html* -tiedosto sisältää lähinnä sivujen perusrakenteen, kuten tyylitiedoston hakemisen, ja itse *home.html* -tiedosto puolestaan sisältää selainnäkymässä näkyvät komponentit. Kuvissa 5 ja 6 näkyy tiedostoihin tehdyt komennot.

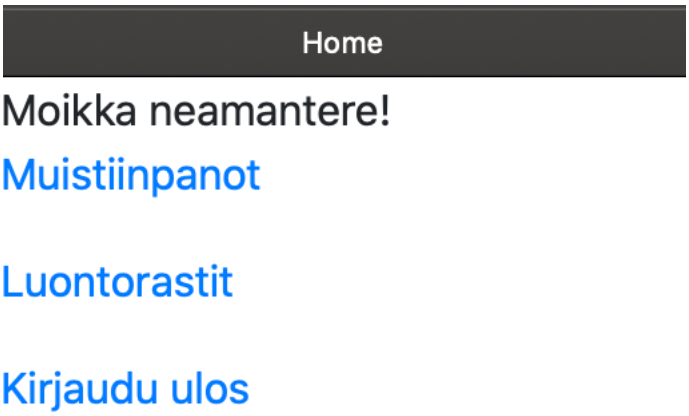

**Kuva 5.** *base.html* -tiedosto.

applikaatio > templates > <> home.html > ...

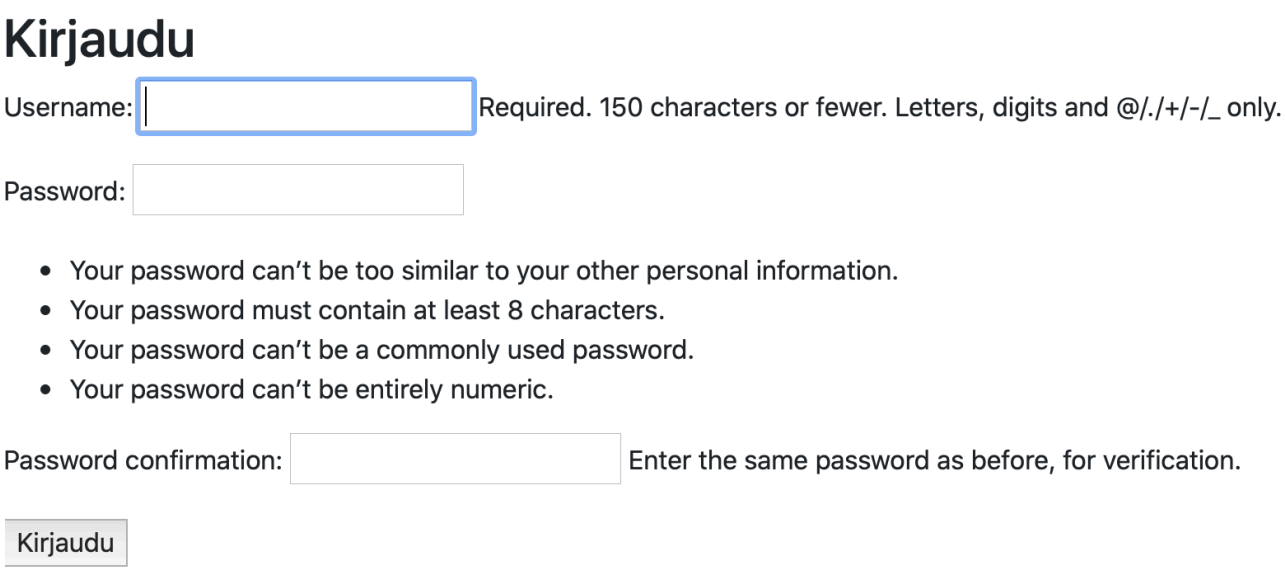
```
1   {% extends 'base.html' %}
2
3   {% block title %}Home{% endblock %}
4
5   {% block content %}
6   {% if user.is_authenticated %}
7       Moikka {{ user.username }}!
8       <p><a href="{% url 'book_list' %}">Muistiinpanot</a></p>
9       <p><a href="{% url 'show_data' %}">Luontorastit</a></p>
10      <p><a href="{% url 'logout' %}">Kirjaudu ulos</a></p>
11  {% else %}
12      <p>Et ole kirjautunut sisään</p>
13      <a href="{% url 'login' %}">Kirjaudu sisään</a>
14  {% endif %}
15  {% endblock %}
```

**Kuva 6.** *home.html* -tiedosto.

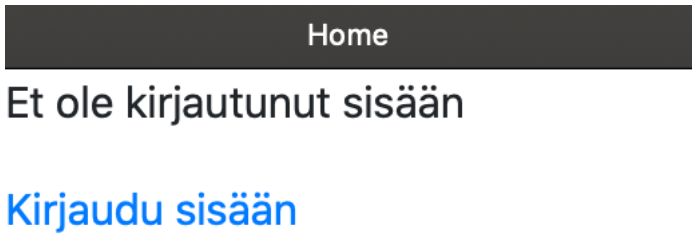
Tiedostojen muutosten jälkeen selaimessa pystyi suorittamaan rekisteröinnin, sisäänkirjautumisen sekä uloskirjautumisen. Kotisivun näkymä on nähtävissä kuvassa 7, rekisteröinnin selainnäkymä kuvassa 8 ja uloskirjautumisen selainnäkymä kuvassa 9.




**Kuva 7.** Kotisivun näkymä rekisteröinnin jälkeen.




**Kuva 8.** Rekisteröinnin selainnäkymä.




**Kuva 9.** Uloskirjautumisen selainnäkymä.

## CRUD-TOIMINNOT

Seuraavaksi tein CRUD-toiminnot eli lisää – näytä – päivitä – poista -toiminnot. Tein toiminnot *Django CRUD* -nimisen netistä löytyvän ohjeen avulla.

Aluksi loin uuden apin nimeltä *books*. Apin luonnin jälkeen *settings.py* -tiedostoon lisättiin tieto uudesta *books* -nimisestä apista, jotta uutta appia voitaisiin käyttää.

```
33  INSTALLED_APPS = [
34      'django.contrib.admin',
35      'django.contrib.auth',
36      'django.contrib.contenttypes',
37      'django.contrib.sessions',
38      'django.contrib.messages',
39      'django.contrib.staticfiles',
40      'applikaatio.apps.ApplikaatioConfig',
41      'books',
```

**Kuva 10.** *settings.py* -tiedosto.

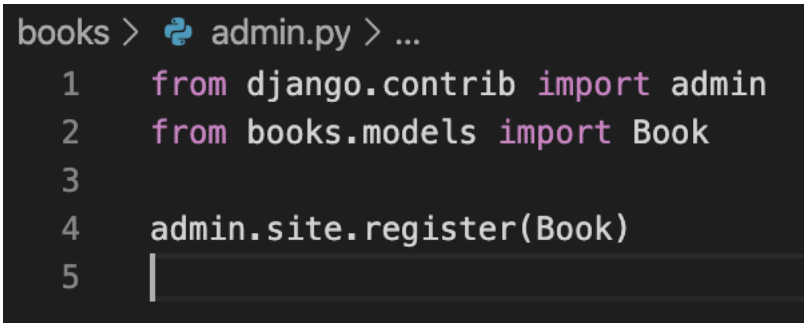
Tämän jälkeen *models.py* -tiedostoon luotiin muistiinpanotoiminnon pohja. Kuvassa 11 näkyy tiedostoon tehdyt muutokset, eli *Book* -luokan luominen.

books > models.py > ...

```
1   from django.db import models
2   from django.urls import reverse
3
4   class Book(models.Model):
5       nimi = models.CharField(max_length=200)
6       tiedot = models.TextField(max_length=200)
7
8       def __str__(self):
9           return self.nimi
10
11      def get_absolute_url(self):
12          return reverse('book_edit', kwargs={'pk': self.pk})
13
```

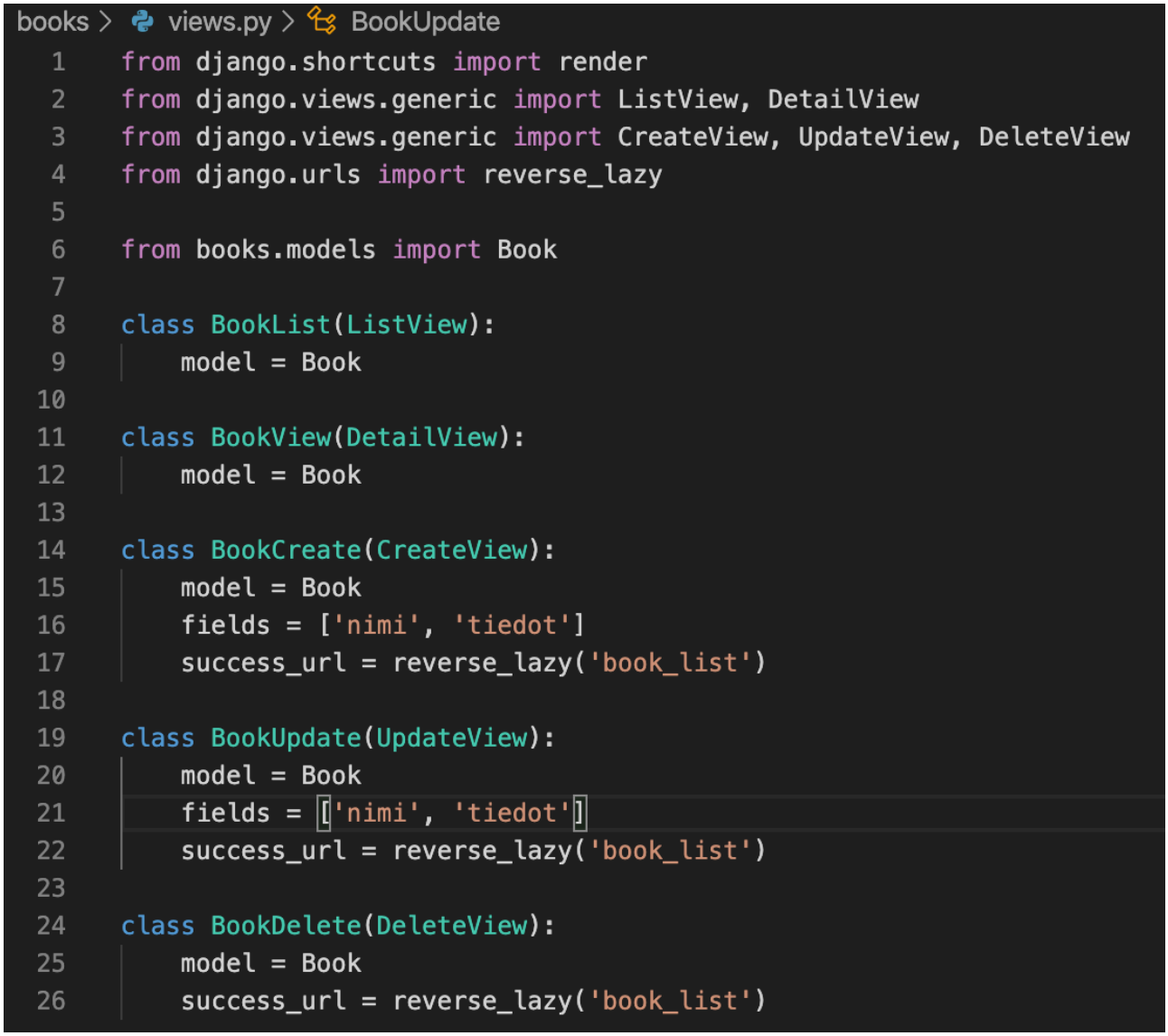
**Kuva 11.** *models.py* -tiedosto.

Komentojen aktivoimiseksi terminaaliin kirjoitettiin seuraavat käskyt: *python3 manage.py makemigrations* ja *python3 manage.py migrate*. Tämän jälkeen *admin.py* -tiedostoon lisättiin vielä kuvassa 12 näkyvät komennot.




**Kuva 12.** *admin.py* -tiedosto.

Seuraavaksi CRUD -toiminnot aktivoitiin. *views.py* -tiedostoon tehtiin jokaiselle CRUD-toiminnolle oma luokka. Kuvassa 13 näkyy kaikki luodut luokat ja niiden sisäiset toiminnot.




**Kuva 13.** *views.py* -tiedosto.

Oleellisimmat luokat tiedostossa käyttäjän näkökulmasta ovat *BookCreate* , *BookUpdate* ja *BookDelete* -nimiset luokat, sillä nimenomaan näiden toimintojen avulla käyttäjä voi vapaasti luoda uusia muistiinpanoja sekä päivittää ja poistaa niitä. Jotta toiminnot saatiin aktivoitua, tein vielä seuraavat komennot uuteen *urls.py* -tiedostoon sekä loin tiedostot *book_confirm_delete.html, book_detail.html, book_form.html* ja *book_list.html*, joissa vielä konkreettisemmin määritellään CRUD-toimintojen toiminnallisuudet.

books > urls.py > ...

```python
from django.urls import path

from . import views

urlpatterns = [
    path('', views.BookList.as_view(), name='book_list'),
    path('view/<int:pk>', views.BookView.as_view(), name='book_view'),
    path('new', views.BookCreate.as_view(), name='book_new'),
    path('view/<int:pk>', views.BookView.as_view(), name='book_view'),
    path('edit/<int:pk>', views.BookUpdate.as_view(), name='book_edit'),
    path('delete/<int:pk>', views.BookDelete.as_view(), name='book_delete'),
]
```

**Kuva 14.** Uusi *urls.py* -tiedosto.

```
    <h1>Poista muistiinpano</h1>
    <form method="post">{% csrf_token %}
        Oletko varma että haluat poistaa "{{ object }}" ?
        <input type="submit" value="Ok" />
        <a href="{% url 'book_list' %}">Takaisin</a>
    </form>
```

**Kuva 15**. *book_confirm_delete.html* -tiedosto.

```
    <h1>Muistiinpanon tiedot</h1>
    <h2>Nimi: {{ object.nimi }}</h2>
    <p>Tiedot: {{ object.tiedot }}</p>
    <p><a href="{% url 'book_list' %}">Takaisin</a></p>
```

**Kuva 16.** *book_detail.html* -tiedosto.

```
4    <h1>Uusi rasti</h1>
5    <p>Lisää uusi rasti, jolla olet käynyt.</p>
6    <form method="post">{% csrf_token %}
7        {{ form.as_p }}
8        <input type="submit" value="Valmis" />
9        <a href="{% url 'book_list'  %}">Peruuta</a>
10   </form>
```

**Kuva 17.** *book_form.html* -tiedosto.

```
4   <h1>Muistiinpanot</h1>
5   <p><a href="{% url 'home'  %}">Kotisivulle</a></p>
6   <p>Tälle sivulle voit merkitä vapaita muistiinpanoja luontorasteista, joilla olet käynyt.</p>
7   <table border="1">
8   <thead>
9       <tr>
10      <th>Rastin nimi</th>
11      <th>Muistiinpanot</th>
12      <th>Näytä</th>
13      <th>Muokkaa</th>
14      <th>Poista</th>
15      </tr>
16  </thead>
17  <tbody>
18      {% for book in object_list %}
19      <tr>
20      <td>{{ book.nimi }}</td>
21      <td>{{ book.tiedot }}</td>
22      <td><a href="{% url "book_view" book.id %}">Näytä</a></td>
23      <td><a href="{% url "book_edit" book.id %}">Muokkaa</a></td>
24      <td><a href="{% url "book_delete" book.id %}">Poista</a></td>
25      </tr>
26      {% endfor %}
27  </tbody>
28  </table>
29
30  <a href="{% url "book_new" %}">Uusi muistiinpano</a>
```

**Kuva 18**. *book_list.html* -tiedosto.

Edellä esitettyjen komentojen seurauksena sain luotua toimivan muistiinpanotoiminnon selaimeen. Kuvassa 19 näkyy muistiinpanotoiminnon selainnäkymä, johon on onnistuneesti luotu yksi muistiinpano.


**Kuva 19.** Selainnäkymä muistiinpanosovelluksesta.

Halusin selainnäkymän visuaalisesta ilmeestä raikkaamman, joten jokainen applikaation sisällä oleva *.html* -tiedosto ”extendaa” *base.html* -tiedostoa, johon on lisätty tyylitiedosto alla olevan linkin kautta. En kuitenkaan pystynyt ”extendaamaan” muissa, kuten booksissa, applikaation *base.html* -tiedostoa, joten lisäsin linkin niihin erikseen.

Tyylitiedoston linkki:

*<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO" crossorigin="anonymous">*
*<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js" integrity="sha384-ChfqqxuZUCnJSK3+MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy" crossorigin="anonymous"></script>*

HELPPOA

- Netistä löytyneiden ohjeiden avulla oli suhteellisen helppoa toteuttaa kirjautumisen ja rekisteröinnin tuet sekä CRUD-toiminnot.
- Tyylitiedoston hakeminen netistä ja liittäminen *.html* -tiedostoihin onnistui näppärästi.
- Helppoa oli myös eri sisäisten linkkien lisääminen *.html* -tiedostoihin.

VAIKEAA/HAASTAVAA

- Liian yksinkertaisen tehtävänannon perusteella oli aluksi vaikea ymmärtää, mitä harjoitustyön toiselta vaiheelta edes haluttiin. Jonkinlaisen kuvan sai yksinkertaisesti selailemalla aikaisempiin toteutuskertoihin osallistuneiden kurssilaisten raportteja.
- Komentojen kirjottamisessa ja tiedostojen luonnissa tuli usein kirjoitusvirheitä. Tämän seurauksena välillä kesti todella kauan päästä etenemään työssä, sillä virheiden etsimiseen kului paljon aikaa.

## LÄHTEET

https://learndjango.com/tutorials/django-login-and-logout-tutorial

https://learndjango.com/tutorials/django-signup-tutorial

https://rayed.com/posts/2018/05/django-crud-create-retrieve-update-delete/

https://getbootstrap.com/docs/4.1/getting-started/download/