

## Harjoitustyö vaihe 2: Tuki kirjautumiselle ja rekisteröitymiselle ja CRUD-toiminnot

Kuvaan tässä harjoitustyön osan 2 raportissa, miten toteutan kirjautumis- ja rekisteröitymistoiminnot ja CRUD-toiminnot eli lisää, näytä, päivistä ja poista -toiminnot. En ole vielä täysin päättänyt harjoitustyöni aiheita. Toteutan tuen kirjautumiselle ja rekisteröitymiselle ja CRUD-toiminnot siitä näkökulmasta, että voisin hyödyntää niitä lopullisessa harjoitustyössäni, vaikka en ole aiheesta vielä täysin varma.

### Kirjautuminen sivulle

Olin edellisessä vaiheessa lisännyt mysite-projektin alle applikaation testapp, jota hyödynnän tässä vaiheessa, kun luon tuen kirjautumiselle. Ensin projektitason *urls.py* tiedostoa muokattiin alla olevan kuvan mukaan. Lisäsin auth appin *urls.py* tiedostoon.

```
16 from django.contrib import admin
17 from django.urls import path, include
18 from django.views.generic.base import TemplateView
19
20 urlpatterns = [
21     path('admin/', admin.site.urls),
22     path('testapp/', include('django.contrib.auth.urls')),
23     path('', TemplateView.as_view(template_name='home.html'), name='home'),
24 ]
25 |
```

Loin *templates.py*-kansioon projektitasolla. Templates.py-kansioon loin kansion *registration* ja sen alle *login.html*.

```
templates > registration > <> login.html > form
1 <h2>Login</h2>
2 <form method="post">
3     {% csrf_token %}
4     {{ form.as_p }}
5     <button type="submit">Login</button>
6 </form>
```

Päivitin *settings.py* etsimään projektitasolta *templates*-kansiota. Templates-kohtaan lisättiin DIRS riville

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

Lisäsin *templates*-kansioon *base.html* ja *home.html*. Muokkasin myös *registration* kansion alla olevaa *login.html* dokumentaation ohjeiden mukaisesti.

```
templates > <> base.html > html
1  <html>
2  <head>
3      <meta charset="utf-8">
4      <title>{% block title %}Django Auth Tutorial{% endblock %}</title>
5  </head>
6  <body>
7      <main>
8          {% block content %}
9          {% endblock %}
10     </main>
11 </body>
12 </html>
```

```
templates > <> home.html > ...
1  {% extends 'base.html' %}
2
3  {% block title %}Home{% endblock %}
4
5  {% block content %}
6  {% if user.is_authenticated %}
7      Hi {{ user.username }}!
8  {% else %}
9      <p>Et ole kirjautunut sisään, kirjautuisitko sisään?</p>
10     <a href="{% url 'login' %}">login</a>
11 {% endif %}
12 {% endblock %}
```

```
templates > registration > <> login.html > ...
1  {% extends 'base.html' %}
2
3  {% block title %}Login{% endblock %}
4
5  {% block content %}
6  <h2>Login</h2>
7  <form method="post">
8      {% csrf_token %}
9      {{ form.as_p }}
10     <button type="submit">Login</button>
11 </form>
12 {% endblock %}
13
```

Lisäsin myös *settings.py* tiedoston loppuun seuraavat kohdan, jotka ohjaavat käyttäjän kotisivulle.

```
LOGIN_REDIRECT_URL = 'home'
```

```
LOGOUT_REDIRECT_URL = 'home'
```

Testasin, toimiiko sovellus.

```
python3 manage.py runserver
```

## Login

Et ole kirjautunut sisään, kirjautuisitko sisään?

[login](#)

Username:

Password:

Login

Seuraavaksi lisäsin uloskirjautumislinkin muokkaamalla *home.html*-tiedostoa.

```
templates > <> home.html > ...
1  {% extends 'base.html' %}
2
3  {% block title %}Home{% endblock %}
4
5  {% block content %}
6  {% if user.is_authenticated %}
7      Hi {{ user.username }}!
8      <p><a href="{% url 'logout' %}">logout</a></p>
9  {% else %}
10     <p>Et ole kirjautunut sisään, kirjautuisitko sisään?</p>
11     <a href="{% url 'login' %}">login</a>
12  {% endif %}
13  {% endblock %}
```

Muokkasin testapp-kansion alla olevaa *urls.py*-tiedostoa.

```
from django.contrib import admin
from django.urls import path, include
from django.views.generic.base import TemplateView

urlpatterns = [
    path('admin/', admin.site.urls),
    path('testapp/', include('django.contrib.auth.urls')),
    path('', TemplateView.as_view(template_name='home.html'), name='home'),
]
```

Seuraavaksi loin superuserin komentorivillä, jotta kirjautuminen sisään on mahdollista.

```
Username (leave blank to use 'anutamminen'):
Email address: anu.tamminen@tuni.fi
Password:
Password (again):
This password is too short. It must contain at least 8 characters.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
```

Asettamani salasana oli liian lyhyt, mutta se ei ollut työn toteutuksen kannalta merkityksellistä, joten ohitin sen. Testasin toimintaa.

← → ↺ ⓘ 127.0.0.1:8000

Hi anutamminen!

[logout](#)

Nyt kirjautumisen jälkeen avautui yllä olevan kuvan mukainen ikkuna. Kun painaa logout, pääsee takaisin kirjautumisvaiheeseen.

## Rekisteröityminen sivulle

Seuraavaksi loin rekisteröitymissivun. Loin uuden applikaation nimeltä *accounts*.

```
python3 manage.py startapp accounts
```

Lisäsin projektitason *settings.py*-tiedostoon seuraavat muutokset.

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'testapp.apps.TestappConfig',
    'accounts.apps.AccountsConfig'
```

Eli lisäsin uuden applikaation *INSTALLED\_APPS* kohtaan.

Lisäsin *accounts*-kansioon *urls.py* tiedoston ja täydensin *views.py*-tiedoston.

```
accounts > urls.py > ...
1  from django.urls import path
2
3  from . import views
4
5
6  urlpatterns = [
7      path('signup/', views.SignUp.as_view(), name='signup'),
8  ]
```

```
accounts > views.py > ...
1  from django.shortcuts import render
2  from django.contrib.auth.forms import UserCreationForm
3  from django.urls import reverse_lazy
4  from django.views import generic
5
6
7  class SignUp(generic.CreateView):
8      form_class = UserCreationForm
9      success_url = reverse_lazy('login')
10     template_name = 'signup.html'
11
```

Täydensin myös projektitason *urls.py*-tiedostoa ja loin *signup.html*-tiedoston *templates*-kansioon.

```
from django.contrib import admin
from django.urls import path, include
from django.views.generic.base import TemplateView

urlpatterns = [
    path('admin/', admin.site.urls),
    path('testapp/', include('django.contrib.auth.urls')),
    path('accounts/', include('accounts.urls')),
    path('accounts/', include('django.contrib.auth.urls')),
    path('', TemplateView.as_view(template_name='home.html'), name='home'),
]
```

```
templates > <> signup.html > ...
1  {% extends 'base.html' %}
2
3  {% block title %}Sign Up{% endblock %}
4
5  {% block content %}
6      <h2>Sign up</h2>
7      <form method="post">
8          {% csrf_token %}
9          {{ form.as_p }}
10         <button type="submit">Sign up</button>
11     </form>
12 {% endblock %}
```

Ajoin palvelimen uudestaan ja kokeilin toimintaa menemällä osoitteeseen <http://127.0.0.1:8000/accounts/signup/>. Tein testikäyttäjän ja alla on esitetty testaus.

## Sign up

Username:  Required. 150 characters or fewer. Letters, digits and @/./+/-/\_ only.

Password:

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation:  Enter the same password as before, for verification.

Hi test!

[logout](#)

## CRUD-toiminnot

Loin sivustolle CRUD-toiminnot. En ole vielä päättänyt, mistä teemasta haluan hyödyntää dataa harjoitustyön seuraavissa vaiheissa. Luon sivustolle kuitenkin sellaisen ominaisuuden, josta on

hyötyä millä tahansa sivustolla. Seurauksena loin kommenttien lisäämisen mahdollisuuden. Käytin apuna YouTube-tutoriaalia.

Loin projektiin uuden applikaation nimelät *comments*.

```
python3 manage.py startapp comments
```

Lisäsin *comments*-applikaation projektitason *settings.py*-tiedostoon kohtaan `INSTALLED_APPS`. Seuraavaksi lisäsin *models.py*-tiedostoon mallin nimeltä *Comment*, jossa kommenttikohde ja arvosanakenttä.

```
comments > models.py > ...
1  from django.db import models
2
3  # Create your models here.
4
5  class Comment(models.Model):
6      kommentti = models.CharField(max_length = 200)
7      arvosana = models.IntegerField()
8
9      def __str__(self):
10         return self.kommentti
```

Tein migraatiot *comment*-applikaatiolle

```
python3 manage.py makemigrations
```

```
python3 manage.py migrate
```

Uusi applikaatio lisättiin projektitason *urls.py*-tiedostoon, jotta projekti voi käyttää uutta applikaatiota.

```
urlpatterns = [
    path('admin/', admin.site.urls),
    path('testapp/', include('django.contrib.auth.urls')),
    path('accounts/', include('accounts.urls')),
    path('accounts/', include('django.contrib.auth.urls')),
    path('', include('comments.urls')),
    path('', TemplateView.as_view(template_name='home.html'), name='home'),
]
```

Lisäsin myös applikaation *admin.py* tiedostoon luomani mallin. Loin *forms.py*-tiedoston ja sinne kuvan mukaisen koodin.

```
comments > forms.py > ...
1  from django import forms
2  from .models import Comment
3
4  class CommentForm(forms.ModelForm):
5      class Meta:
6          model = Comment
7          fields = ('kommentti', 'arvosana')
```

Lisäsin *comment*-applikaation *views.py*-tiedostoon ja tarvittavat funktiot CRUD-toimintoja varten. Loin kansion templates *comments*-applikaation alle

```
mkdir comments/templates
```

Loin templates-kansion alle html-tiedostot *comments.html*, *comments-form.html* ja *comment-delete-confirm.html* terminaalisssa.

```
touch comments/templates/comments.html
```

```
touch comments/templates/comments-form.html
```

```
touch comments/templates/comment-delete-confirm.html
```

Täydensin *templates*-kansion alle luotujen tiedostoihin alla olevissa kuvissa näkyvät html-koodit.

```
comments > templates > comments.html > html
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7  </head>
8  <body>
9      <ul>
10         {% for comment in comment %}
11         <a href="{% url 'update_comment' comment.id %}" >
12             <li> {{ comment.kommentti }} </li>
13         </a>
14         {% endfor %}
15     </ul>
16
17     <a href = "{% url 'create_comment' %}"> New comment</a>
18 </body>
19 </html>
```

```
comments > templates > comments-form.html > html > body
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7  </head>
8  <body>
9      <h1>New/Update comment</h1>
10
11      <form method = "POST">
12          {% csrf_token %}
13          {{ form }}
14          <button type = "submit">Save</button>
15      </form>
16
17      {% if comment %}
18          <a href = "{% url 'delete_comment' comment.id %}">Delete</a>
19      {% endif %}
20
21 </body>
22 </html>
```

```
comments > templates > comment-delete-confirm.html > html > body > form
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7  </head>
8  <body>
9      <form method = "POST">
10          {% csrf_token %}
11          <h2> Are you sure you want to remove {{ comment.kommentti }} ?</h2>
12          <button type = "submit">Confirm</button>
13      </form>
14  </body>
15  </html>
```

Täydensin *views.py*-tiedostoon seuraavan koodin.

```
comments > views.py > ...
1  from django.shortcuts import render, redirect, get_object_or_404
2  from .models import Comment
3  from .forms import CommentForm
4
5  # Create your views here.
6
7  def list_comments(request):
8      comments = Comment.objects.all()
9      return render(request, 'comments.html', {'comments': comments})
10
11 def create_comment(request):
12     form = CommentForm(request.POST or None)
13
14     if form.is_valid():
15         form.save()
16         return redirect('list_comments')
17
18     return render(request, 'comments-form.html', {'form': form})
19
20 def update_comment(request, id):
21     comment = Comment.objects.get(id=id)
22     form = CommentForm(request.POST or None, instance=comment)
23
24     if form.is_valid():
25         form.save()
26         return redirect('list_comments')
27
28     return render(request, 'comments-form.html', {'form': form, 'comment': comment})
29
30 def delete_comment(request, id):
31     comment = Comment.objects.get(id=id)
32
33     if request.method == 'POST':
34         comment.delete()
35         return redirect('list_comments')
36
37     return render(request, 'comments-delete-confirm.html', {'comment': comment})
38
```

Loin comment-applikaation alle *urls.py*-tiedoston, johon lisäsin seuraavan koodin.

```
comments > urls.py > ...
1  from django.urls import path
2  from .views import list_comments, create_comment, update_comment, delete_comment
3
4  urlpatterns = [
5      path('', list_comments, name='list_comments'),
6      path('new', create_comment, name='create_comment'),
7      path('update/<int:id>/', update_comment, name='update_comment'),
8      path('delete/<int:id>/', delete_comment, name='delete_comment'),
9  ]
```

Luotuani tarpeelliset koodit testasin sovelluksen toimintaa.

```
python3 manage.py runserver
```



Add comment

Kommentti:

Hieno!

Arvosana:

5

Are you sure?

Are you sure you want to delete the selected comment?

Summary

▪ Comments: 1

Objects

▪ Comment: [Hieno!](#)

Yes, I'm sure

No, take me back

## Helppoa/vaikeata

- Tuki kirjautumiseen ja rekisteröitymiseen oli helppo tehdä, kun vain noudatti tarkasti tutoriaalien ohjeita.
- Haasteellista oli, että en ole vielä pohtinut, mistä aiheesta teen sovellukseni lopullisesti. En ole varma onko CRUD-toiminnoista hyötyä lopullisessa työssä.
- Haasteellista oli ajoittain kokonaisuuden ymmärtäminen eli esimerkiksi miksi osa tehdään projektitasolla ja osa applikaatiotasolla.
- Huolimattomuus. Pienet errorit sen takia, että jäänyt pilkku pois tai sana koodissa kirjoitettu väärin 😊

## Lähteet

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

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

[https://www.youtube.com/watch?v=Kf9KB\\_TZY5U](https://www.youtube.com/watch?v=Kf9KB_TZY5U)

Aiemmat harjoitustyöt