cd .. dir

python3 -m venv venv

venv\Scripts\activate

python -m pip install django

django-admin.py startproject mysite .

vytvoří se potřebné soubory

python manage.py migrate

python manage.py runserver

http://127.0.0.1:8000/

ctrl c ukončíš

Vytvoření nové aplikace

python manage.py startapp blog

Řekni. že má blok používat – připiš do settings.py - blog

INSTALLED\_APPS = (

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

'blog',

)

Otevři blog/models.py, odstraň vše, co v něm je, a vlož následující kód:

from django.db import models

from django.utils import timezone

class Post(models.Model):

author = models.ForeignKey('auth.User', on\_delete=models.CASCADE)

title = models.CharField(max\_length=200)

text = models.TextField()

created\_date = models.DateTimeField(

default=timezone.now)

published\_date = models.DateTimeField(

blank=True, null=True)

def publish(self):

self.published\_date = timezone.now()

self.save()

def \_\_str\_\_(self):

return self.title

**Post –** jmeno tridy, klidne si dej vlastni  
**models.Model** -  znamená, že Post je Django Model, takže Django ví, že by to mělo být uloženo v databázi.

* models. CharField - takto definuješ text s omezeným počtem znaků.
* models. TextField - toto použiješ na dlouhý text bez omezení. To zní ideálně pro obsah blog post, ne?
* models. DateTimeField - definuje datum a čas.
* models. ForeignKey - definuje odkaz do jiného modelu.

Informace, že se zmenil kod

python manage.py makemigrations blog

Django pro nás připravil soubor migrace, který budeme muset aplikovat na naši databázi.

Python manage.py migrate blog

Otevřeme soubor blog/admin.py a nahradíme jeho obsah následujícím kódem:

from django.contrib import admin

from .models import Post

admin.site.register(Post)

Pythonu manage.py runserver

<http://127.0.0.1:8000/admin/>

python manage.py createsuperuser