

Contents

Uputstvo za Djangular web app (postavka foldera za backend-Django i frontend-Angular)	2
Podesavanja backend-a:	2
Podesavanje Django Secret Key exposed on GitHub (dotenv)	2
Podeasavanje JSON Web Token (JWT)	3
Podesavanje frontend-a:	4

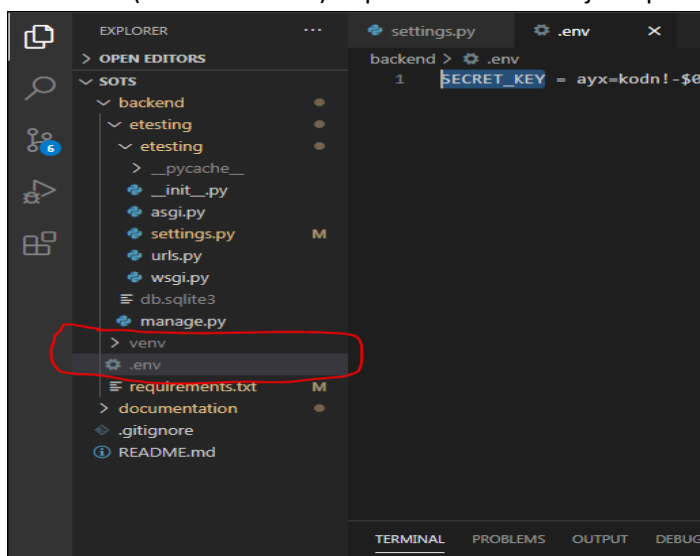
Uputstvo za Django web app (postavka foldera za backend-Django i frontend-Angular)

Podesavanja backend-a:

1. Napravi repo na github-u
2. Manuelno napravi folder backend
3. U backend folderu napravi virtuelno okruzenje komandom
python -m venv venv
4. Aktiviraj napravljeni okruzenje komandom
venv\Scripts\activate.bat
5. Eksportuju requirements.txt u backend folder
pip freeze > requirements.txt
6. Instalirati sledece zavisnosti komandom
pip install django
pip install djangoRESTframework
7. Kreiranje Django projekta komandom
django-admin startproject etesting
8. Pokreni napravljeni Django projekat
python manage.py runserver
9. Uradi prvu migraciju u projektu
python manage.py migrate
10. Fdasdf

Podesavanje Django Secret Key exposed on GitHub (dotenv)

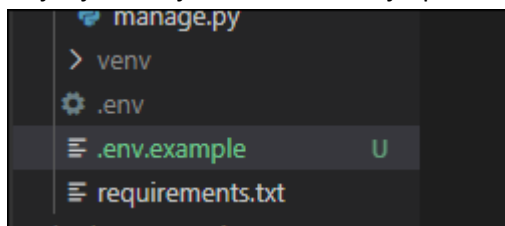
1. Instalirati zavisnost komandom
pip install python-dotenv
2. U settings.py na vrh fajla copy-past sledece komande:
from dotenv import load_dotenv
load_dotenv()
3. U root folderu (backend folder) napraviti novi .env fajl. Napraviti novi .env fajl na nivou venv folder



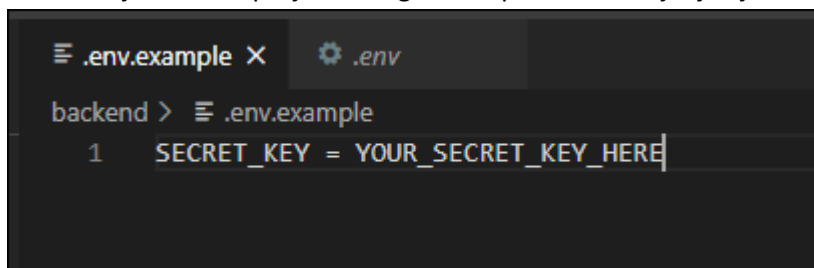
4. U .env fajl copy paste SECRET_KEY iz settings.py fajla. Kopirati po principu key = value samo vodi racuna da value bude bez navodnika
5. .env fajl je izuzet od push-ovanja sa .gitingore bas kao sto je i venv folder izuzet

```
# Environments
.env
.venv
env/
venv/
ENV/
env.bak/
venv.bak/
```

6. U settings.py zameniti SECRET_KEY sa SECRET_KEY = os.getenv('SECRET_KEY')
7. Dodati jos jedna fajl na nivou .env fajla pod imenom .env.example



8. Svrha ovog .env.example fajla je da svima koji kloniraju ili skidaju projekata da uputstvo kako treba podesiti sistemske varijable da bi projekat mogao da se pokrene. Ovaj fajl nije izuzet sa .gitignore fajlom.



9. Izvori:
 - 9.1. <https://dev.to/vladyslavnya/how-to-protect-your-django-secret-and-oauth-keys-53fl>
 - 9.2. <https://pypi.org/project/python-dotenv/>
 - 9.3. <https://www.youtube.com/watch?v=17UVejOw3zA&t=446s>

Podasavanje JSON Web Token (JWT)

1. Instaliraj zavisnost:
pip install django-rest-framework-jwt
2. Fasdfa

Formiranje i punjenje tabela u bazi podataka

1. Formiranje praznih tabla (kostur): Sa ove dve komande formiraju se prazne tabele u bazi podataka koje nisu popunjene. Pravi se migracija od modela do praznih tabela zato se i zovu migracije.
python manage.py makemigrations
python manage.py migrate
2. Punjenje praznih tabela (populate db): Za punjenje baze podataka koriste se dummy podaci koji se zovu fixtures. Postoje globalni i lokalni fixtures. Lokalni fixtures su podaci koji se vezuju za svaku aplikaciju u projektu ponaosob i nalaze se u fixtures folderu svake aplikacije. Svaka aplikacija ima folder fixtures koji se rucno pravi. U tim folderim se nalaze lokalni fixtures sa podacima za konkretnu app u projektu. Lokalni fixtures se koriste za testiranje aplikacije. Globalni fixtures prave se na nivou projekta i obuhvataju dummy podatke na nivou citavog projekta (projekat = suma svih app(komponenti)). Globalni fixtures koristi se za punjenje baze podataka inicijalnim podacima. Globani fixture nalazi se na nivou projekta. Globani fixture se nalazi na nivou db.sqlite3 fajla.

- 2.1. db.sqlite3 fajl predstavlja binarni fajl koji nije citljiv za ljude. U ovom fajlu se nalaze podaci iz lokalne baze podataka. Od push-ovanja je izuzet sa .gitignore fajlom. Ne moze da se koristi kao .sql skripta u Springu zato sto se vezuje za lokalni racunar. Sqlite3 baza podataka predstavlja embedded (ugradjenu) bazu podataka bas kao sto je H2 za Spring. db.sqlite3 fajl ne moze da se koristi na nivou github tima (ideja da ceo tim koristi iste inicijalne podatke) zato sto su svi podaci koji se nalaze u ovoj skripti vezani za lokani racunar i skripta je izuzeta od push-ovanja razlika u odnosu na Spring i .sql skriptu.
- 2.2. Eksportovanje db.sqlite3 fajla u globalni fixtures: Pozicioniraj se na nivo db.sqlite3 skripte u folder projekta. Dodaj sve sto mas u db.sqlite3 lokalnu skriptu (dodavanje ide preko admin panela, fronta, terminala). Proveriti na sajtu <https://sqliteonline.com> da li su svi podaci dodati koji treba da budu dodati. Ici na opciju File open DB. Kada mas sve u lokalnu exportuju db.sqlite3 u db.json fajl sledecom komandom. Na ovaj nacin lokani podaci su prebaceni u fajl koji nije izuzet od push-ovanja i ceo tim moze da koristi iste podatke. Push-uj projekat na git.
python manage.py dumpdata --indent 4 > db.json
- 2.3. Ucitavanje globalnog fixture. Povuces sa gita najnoviju verziju projekta. Odradis dve gore navedene komande za migraciju. Ako izbiju greske izbrisati lokalni db.sqlite3 fajl koji je konfliktan. Ponovo pokrenuti komande za migraciju. Biti na nivou db.sqlite3 fajla i kucati sledecu komandu. Sa ovim je baza svih pojeadinacnih clanova tima popunjena isitim inicijalnim podacima
python manage.py loaddata db.json
- 2.4. Proveriti na sajtu <https://sqliteonline.com> da li su svi podaci ucitani koji treba da budu ucitani. Ici na opciju File open DB
3. YAML db.yaml sve isto kao i sa db.json samo mora da se instalira sledeca zavisnost
pip install PyYAML
Nakon sto je instalirana zavisnost koriste se sledece komande za dump-ovanje i load-ovanje :
python manage.py dumpdata --indent 4 > db.yaml
python manage.py loaddata db.yaml
4. Izvori:
 - 4.1. <https://docs.djangoproject.com/en/3.1/howto/initial-data/>

Podesavanje frontend-a:

dafdsafsd