bcs Documentation

Release 0.1

Daniel Roy Greenfeld

Mar 30, 2018

Contents

1 Install **3**

2 Deploy **5**

3 Developing with Docker **7**
- 3.1 Setting up . . . 7
- 3.2 Deployment . . . 9
- 3.3 Building and running your app on EC2 . . . 10
- 3.4 Security advisory . . . 10

4 Django instalacija koplet na RPI s static IP 89.212.90.184 **11**
- 4.1 OSNOVNO . . . 11
- 4.2 Python3, Pip in Virtualenv . . . 11
- 4.3 COOKIECUTTER S PREDNASTAVLJENIM DJANGOM . . . 12
- 4.4 POSTGRES . . . 12
- 4.5 DJANGO . . . 12
- 4.6 READTHEDOCS . . . 13

5 BCS **15**
- 5.1 Standardizirani popisi del(Building Construction Specifications - Tehnične specifikacije del za gradnje) . . . 15

6 Indices and tables **21**

Contents:

CHAPTER 1

Install

This is where you write how to get a new laptop to run this project.

CHAPTER 2

Deploy

This is where you describe how the project is deployed in production.

CHAPTER 3

Developing with Docker

You can develop your application in a Docker container for simpler deployment onto bare Linux machines later. This instruction assumes an Amazon Web Services EC2 instance, but it should work on any machine with Docker > 1.3 and Docker compose installed.

3.1 Setting up

Docker encourages running one container for each process. This might mean one container for your web server, one for Django application and a third for your database. Once you're happy composing containers in this way you can easily add more, such as a Redis cache.

The Docker compose tool (previously known as fig) makes linking these containers easy. An example set up for your Cookiecutter Django project might look like this:

```
webapp/ # Your cookiecutter project would be in here
    Dockerfile
    ...
database/
    Dockerfile
    ...
webserver/
    Dockerfile
    ...
production.yml
```

Each component of your application would get its own Dockerfile. The rest of this example assumes you are using the base postgres image for your database. Your database settings in *config/base.py* might then look something like:

```
DATABASES = {
        'default': {
            'ENGINE': 'django.db.backends.postgresql_psycopg2',
            'NAME': 'postgres',
            'USER': 'postgres',
```

```
            'HOST': 'database',
            'PORT': 5432,
        }
    }
```

The Docker compose documentation explains in detail what you can accomplish in the *production.yml* file, but an example configuration might look like this:

```
database:
    build: database
webapp:
    build: webapp:
    command: /usr/bin/python3.4 manage.py runserver 0.0.0.0:8000 # dev setting
    # command: gunicorn -b 0.0.0.0:8000 wsgi:application # production setting
    volumes:
        - webapp/your_project_name:/path/to/container/workdir/
    links:
        - database
webserver:
    build: webserver
    ports:
        - "80:80"
        - "443:443"
    links:
        - webapp
```

We'll ignore the webserver for now (you'll want to comment that part out while we do). A working Dockerfile to run your cookiecutter application might look like this:

```
FROM ubuntu:14.04
ENV REFRESHED_AT 2015-01-13

# update packages and prepare to build software
RUN ["apt-get", "update"]
RUN ["apt-get", "-y", "install", "build-essential", "vim", "git", "curl"]
RUN ["locale-gen", "en_GB.UTF-8"]

# install latest python
RUN ["apt-get", "-y", "build-dep", "python3-dev", "python3-imaging"]
RUN ["apt-get", "-y", "install", "python3-dev", "python3-imaging", "python3-pip"]

# prepare postgreSQL support
RUN ["apt-get", "-y", "build-dep", "python3-psycopg2"]

# move into our working directory
# ADD must be after chown see http://stackoverflow.com/a/26145444/1281947
RUN ["groupadd", "python"]
RUN ["useradd", "python", "-s", "/bin/bash", "-m", "-g", "python", "-G", "python"]
ENV HOME /home/python
WORKDIR /home/python
RUN ["chown", "-R", "python:python", "/home/python"]
ADD ./ /home/python

# manage requirements
ENV REQUIREMENTS_REFRESHED_AT 2015-02-25
RUN ["pip3", "install", "-r", "requirements.txt"]

# uncomment the line below to use container as a non-root user
```

```
USER python:python
```

Running *sudo docker-compose -f production.yml build* will follow the instructions in your *production.yml* file and build the database container, then your webapp, before mounting your cookiecutter project files as a volume in the webapp container and linking to the database. Our example yaml file runs in development mode but changing it to production mode is as simple as commenting out the line using *runserver* and uncommenting the line using *gunicorn*.

Both are set to run on port *0.0.0.0:8000*, which is where the Docker daemon will discover it. You can now run *sudo docker-compose -f production.yml up* and browse to *localhost:8000* to see your application running.

3.2 Deployment

You'll need a webserver container for deployment. An example setup for Nginx might look like this:

```
FROM ubuntu:14.04
ENV REFRESHED_AT 2015-02-11

# get the nginx package and set it up
RUN ["apt-get", "update"]
RUN ["apt-get", "-y", "install", "nginx"]

# forward request and error logs to docker log collector
RUN ln -sf /dev/stdout /var/log/nginx/access.log
RUN ln -sf /dev/stderr /var/log/nginx/error.log
VOLUME ["/var/cache/nginx"]
EXPOSE 80 443

# load nginx conf
ADD ./site.conf /etc/nginx/sites-available/your_cookiecutter_project
RUN ["ln", "-s", "/etc/nginx/sites-available/your_cookiecutter_project", "/etc/nginx/
↪sites-enabled/your_cookiecutter_project"]
RUN ["rm", "-rf", "/etc/nginx/sites-available/default"]

#start the server
CMD ["nginx", "-g", "daemon off;"]
```

That Dockerfile assumes you have an Nginx conf file named *site.conf* in the same directory as the webserver Dockerfile. A very basic example, which forwards traffic onto the development server or gunicorn for processing, would look like this:

```
# see http://serverfault.com/questions/577370/how-can-i-use-environment-variables-in-
↪nginx-conf#comment730384_577370
upstream localhost {
    server webapp_1:8000;
}
server {
    location / {
        proxy_pass http://localhost;
    }
}
```

Running *sudo docker-compose -f production.yml build webserver* will build your server container. Running *sudo docker-compose -f production.yml up* will now expose your application directly on *localhost* (no need to specify the port number).

3.3 Building and running your app on EC2

All you now need to do to run your app in production is:

- Create an empty EC2 Linux instance (any Linux machine should do).
- Install your preferred source control solution, Docker and Docker compose on the news instance.
- Pull in your code from source control. The root directory should be the one with your *production.yml* file in it.
- Run *sudo docker-compose -f production.yml build* and *sudo docker-compose -f production.yml up*.
- Assign an Elastic IP address to your new machine.
- Point your domain name to the elastic IP.

Be careful with Elastic IPs because, on the AWS free tier, if you assign one and then stop the machine you will incur charges while the machine is down (presumably because you're preventing them allocating the IP to someone else).

3.4 Security advisory

The setup described in this instruction will get you up-and-running but it hasn't been audited for security. If you are running your own setup like this it is always advisable to, at a minimum, examine your application with a tool like OWASP ZAP to see what security holes you might be leaving open.

CHAPTER 4

Django instalacija koplet na RPI s static IP 89.212.90.184

4.1 OSNOVNO

za štart

```
Sl.tipkovnica ....sudo dpkg-reconfigure keyboard-configuration
SSH server .......sudo apt install openssh-server
Miška ............sudo apt-get install gp
Vim...............sudo apt-get install vim
Git ..............sudo apt-get install git
```

4.2 Python3, Pip in Virtualenv

Prvi korak je instalacija Pythona3 in Pip in virtualnega okolja V ubuntu16.04 je python3 že instaliran sicer pa download iz https://www.python.org/downloads/python3.6.3 .Iz download direktorija Python-3.6.3 izvršimo ukaze

```
./configure
sudo make
sudo make install
```

Instalacija Pip

```
sudo apt-get install python3-pip
sudo python3.5 -m pip install -upgrade pip setuptools wheel
```

Virtualno okolje z mojim "local" za development

```
sudo pip3 install virtualenv
sudo pip3 install https://github.com/pypa/virtualenv/tarball/master
virtualenv -p python3 local
```

4.3 COOKIECUTTER S PREDNASTAVLJENIM DJANGOM

Prvi korak za prvo instalacijo : Na osnovnem direktoriju /home/pavlovicr naložimo cookiecutter, odpremo virtualno okolje in inštaliramo nov projekt z django aplikacijo

```
source local/bin/activate
pip install --upgrade cookiecutter
cookiecutter https://github.com/pydanny/cookiecutter-django
```

Prvi korak, za kopijo aplikacije, ki jo imamo na gitHubu. To uporabimo vedno ko je projekt živ sicer bomo imeli težave z različnimi inštaliranimi aplikacijami

```
git clone https://github.com/pavlovicr/bcs
```

Drugi korak: Inštaliramo še vse potrebne programe za naš operacijski sistem in python, ki smo jih naložili z instalacijo cookiecutter-django in so v direktoriju "utility"

```
cd utility
sudo ./install_os_dependencies.sh install
sudo ./install_python_dependencies.sh install
```

Pri zadnji komandi nas opozori naj si le te pridobimo iz naslova

```
wget https://bootstrap.pypa.io/get-pip.py --output-document=get-pip.py; chmod +x get-
↪pip.py; sudo -H python3 get-pip.py
```

Sedaj , ko imamo inštalirano vso potrebno programsko opremo za os in python zaženemo še potrebno programsko opremo za django za local development okolje

```
cd bcs
pip install -r requirements/local.txt
```

4.4 POSTGRES

v serverju postgres ustvarimo bazo

```
sudo su -l postgres
createdb bcs
```

in nastavimo novega uporabnika "ubuntu"

```
CREATE USER ubuntu WITH PASSWORD 'rolu9255';
```

4.5 DJANGO

```
python manage.py runserver 89.212.90.184:8000
python manage.py migrate
```

4.6 READTHEDOCS

:: za lepo html obliko navodil ali tudi modelsov in ostalega v index.rst vpišemo ime fajla “bcs_instalacija.rst”

CHAPTER 5

BCS

5.1 Standardizirani popisi del(Building Construction Specifications - Tehnične specifikacije del za gradnje)

5.1.1 Uvod

28. člen pravilnika o projektni dokumentaciji pravi :

. . . . se za celovit opis objekta v projektu za izvedbo izdela zbirno tehnično poročilo, ki vsebuje tudi skupen popis materiala in opreme z rekapitulacijo stroškov izgradnje, pri čemer je za njegovo izdelavo zadolžen koordinator . . . .

Vsebina Popisa materiala in opreme po zahtevah iz navedenega pravilnika se le deloma prekriva z vsebino POPISOV DEL, ki v praksi pomenijo nepogrešljiv dokument pri gradnjah. V POPISIH DEL ni samo popisa materiala in opreme. V popisu del so opisane sestave gradbenih konstrukcij in njihovih delov , tehnologije izvedbe, uporabljeni materiali in na njih vezane zahteve iz zakona o gradbenih proizvodih, pravila za zagotavljanje kvaliete, pravila pri obračunu in merjenju količin, varnostne zahteve, navedbe standardov itd.. Popisi del so pri gradnjah po sistemu ” na enoto mere ” ključni pri definiranju predmeta pogodbe in izračunu končne cene pogodbenih del.

Kljub temu, da so popisi del pomemben del gradbene dokumentacije, Slovenija ne premore enotne metodologije za njihovo izdelavo prav tako ni javno dostopnih zbirk popisov del, ki bi nadomestili zastarele standardizirane popise del (GIPPOS, Fabrizzio, ..). Ostali so le redki in to samo za določene zvrsti del.(Skupnost za ceste). Zanimivo, da se je zakonodajalec zelo potrudil pri oblikovanju knjige obračunskih izmer v “pravilniku o vsebini in načinu vodenja dnevnika o izvajanju del ter o načinu označitve gradbišča “, o vsebini pa ne duha ne sluha.

Pred več kot štiridesetimi leti smo bili slovenski gradbinci v svetovnem merilu pionirji tudi na področju standardiziranih popisov del, prirejenih celo za računalniško obdelavo. Danes, v obdobju globalizacije, standardizacije in digitalizacije imamo še vedno iste.

5.1.2 Cilj projekta

Izdelati sistem priprave, urejanja in vzdrževanja knjižnice standardiziranih popisov del , ki bo sodoben, digitaliziran, dostopen prek spleta in javen po principih “open source” ???

5.1.3 Aktivnosti in faznost

1. **faza**
    - **Izdelava dokumentacije**
        - vsebinski del
        - razvoj aplikacije
2. **faza**
    - Produkcija
    - Urejanje specifikacij na vzorčnem primeru stanovanjske gradnje do III.FAZE
3. **faza**
    - Izdelava dokumentacije za upravljanje sistema

1. Dokumentacija - vsebinski del

1.1 Splošno

Kaj je knjižnica popisov del ?

Knjižnica popisov del so sistematično urejeni seznami tehničnih specifikacij del (opisov del) , ki nastopajo pri gradnjah in z njin

- Tehnične specifikacije del opredeljujejo načine in postopke izvajanja del , izbiro uporabljenih materialov , strojev , opreme in orodja ter pogoje izvedbe del, ipd.
- Določila opredeljujejo pravila vezana na merjenje in obračun specificiranih del, zagotavljanje kakovosti, izvajanje varnostnih ukrepov ter spoštovanje zahtev zakonodaje in standardov za specificirana dela, ipd.

Kaj so standardizirani popisi del ?

Popisi del sprejeti kot standardizirani popisi del, s konsenzom investitorjev gradenj, projektantov objektov in izvajalcev del.?

1.2 Izhodišča

- **Standardizirane opise ločiti glede na vrste gradenj :**
    - visoke gradnej
    - nizke gradnje
    - hidrotehnični objekti
- Struktura in vsebine po zgledu TSC gov
- Strukturo razčlenjena modularno ??

1.3 Pomen izrazov

tehnične specifikacije del tehnični opisi del, ki nastopajo pri gradnjah

popisna postavke tehnična specifikacija posameznega dela

postavka osnovna specifikacija posameznega dela

specifikacija dodatna specifikacija posameznega dela

kriterij specifikacije merilo za opredelitev specifikacije

področje kriterija področje, ki ji kriterij pripada

skupina področja kriterija skupina, ki ji področje pripada

dela skupine sorodnih postavk

vrsta del skupine sorodnih del

skupina del

splošna določila pravila vezana na izvajanje skupine del

posebna določila pravila vezana na izvajanje posameznih del

vrsta določila kriterij vsebine določila

1.4 Vsebina

- tehnične specifikacije del
- splošna in posebna določila
- popisi del

1.4.1 Tehnične specifikacije del

Tehnične specifikacije del opredeljujejo vrsto del , načine in postopke izvajanja , izbiro uporabljenih materialov , strojev , opren

- Popisi del so sistematično urejene tehnične specifikacije posameznih del oziroma popisnih postavk.

Sistematično pomeni, da so posamezna dela zbrana po delih , dela po vrstah del in vrste del po skupinah del.

- Posamezna popisna postavka je sestavljena iz postavke, ki je osnovna tehnična specifikacija posameznega dela in specifikacij postavke, ki popisno postavko -podrobneje definirajo predmet posameznega dela.
- Kriteriji specifikacij razvrščajo specifikacije v skupine, po namenu kriterija.
- Postavkam pripadajo dela, delom vrste del, vrstam del skupine del.
- Kriterij

Primer popisne postavke :

A. GRADBENA DELA A/1 Betonska dela A/1.1 vgrajevanje betona A/1.1.1 Dobava in vgrajevanje betona C30/37

specifikacije					določilo	vrsta določila	skupina določila
skupina del	gradbena dela	kriterj specifikacije	področje specifikacije	skupina področja specifikacije	splošna	splošne zahteve	
vrsta del	betonska dela				posebna 1	obračun	
dela	vgrajevanje betona				posebna 2		
postavka	vgrajevanje betona						
enota mere	m3						
specifikacija 1	preseka 0-12 m3/m2,m1	presek konstrukcije			posebna3		
specifikacija 2	z dobavo betona C30/37	trdnostni razred					
specifikacija 3	XC4	odpornost na karbonatizacijo	razredi izpostavljenosti		SIST EN		
specifikacija 4	XF3	odpornost na zmrzovanje					
specifikacija 5	PVII	vodonepreputstpkt	splošno		posebna 8		
specifikacija 6	0-16 mm	max. zrno					
specifikacija 7	S4	konsistenca betona					
specifikacija 8	VB3	viden beton	razred vidne površine		SIST EN 13670		
specifikacija 9	P3	ravnost					
specifikacija 10	T3	tekstura					
specifikacija 11	C30	barvno odstopanje					

struktura in medsebojne zveze

Postavka je osnovna specifikacija posameznega dela in ima enoto mere.Sama zase nam pove samo za kakšno delo gre in nič več. Na primer “izkop jarkov”.

Postavko natančno opišejo dodatne specifikacije, ki posameznim postavkam pripadajo. Specifikacije dodatno opisujejo postavke glede na sestavo konstrukcij in njihovih delov, način, pogoje in postopke izvajanja del, materiale ipd . Izkop jarkov “globine do 2m , v terenu III. ktg ” .

> Dodatne specifikacije so oblikovane glede na razne kriterije. Kriterij “globine izkopa” , “kriterij kategorije terena ” ipd. Kriteriji specifikacije so lahko zbrani po področjih.”pogoji dela”, “material ” ipd.

Posamezni postavki pridapa več specifikacij, posamezna specifikacija pa lahko pripada večim različnim posameznim postavkam.

Posamezni postavki pripadajo tudi “dela” iz katerih izhaja. Eni postavki ena “dela”, enim “delom” pa več postavk.

vrsta del pripada družini skupina del dela pripadajo družini vrsta del

Popisne postavke sestavljajo postavke s specifikacijami, ki jim pripadajo in podrobneje opisujejo postavko. Popisne postavke so organizirane v okviru del in vrste del , ki jim pripadajo.

Postavka je jedro popisne postavke in sama po sebi opredeljuje osnovni predmet dela in enoto mere. Specifikacije podrobneje definirajo postavko (prednmet dela)in pogoje izvedbe.

Postavke s specifikacijami tvorijo popisne postavke, ki jih sestavljamo modularno. Postavke izbiramo, sortiramo,zbiramo

Relacije :

vrsta del	skupina del	n : 1
dela	vrsta del	n : 1
postavka	dela	n : 1
specifikacija	postavka	n : n
kriterij specifikacije	specifikacije	1 : n
področje specifikacije	kriterij specifikacije	1 : n
splošna določila	skupina del	n : 1
splošna določila	dela	n : 1
splošna določila	postavka	n : 1
splošna določila	specifikacija	n : 1
splošna določila	vrsta določila	n : 1
vrsta določila	skupina določila	n : 1

1.4.2 Splošna in posebna določila

** Določila opredeljujejo pravila vezana na merjenje in obračun specificiranih del, zagotavljanje kakovosti, izvajanje varnostnih ukrepov ter spoštovanje zahtev zakonodaje in standardov za specificirana dela, ipd.**

Določila niso nič drugega kot specifikacije specifikacij, postavk, del in vrst del ter določila, ki veljajo za gradnje nasplošno.Za razliko od tehnično tehnoloških specifikacij ta določajo pravila glede uporabe zakonodaje, obračunov, varnosti, kakovosti ipd.

1.5 Shema

1.6 Vzorčni primer

1.7 Izhodišča za spletno aplikacijo

izpis :

POSTAVKA : strojni izkop temeljev globine do 2m v terenu III. ktg
koda : str,izk,tem,gl0-2,IIIktg | SE GENERIRA
stevilka : 1256783452 | SE GENERIRA

izbirna polja

NAČIN DELA			KATEGORIJA ZEMLJIŠČA	
ročno			1. ktg	
strojno			2. ktg	
brez			3. ktg	
ročni vnos			IV . ktg	
			22. ktg	

List view:

dela postavke

Detail view

Določitev URL

specifikacije/ - home/index specifikacije/postavke - list specifikacije/postavke/<id> - detail

> catalog/ — The home/index page. catalog/books/ — The list of all books. catalog/authors/ — The list of all authors. catalog/book/<id> — The detail view for the specific book with a field primary key of <id> (the default). So for example, /catalog/book/3, for the third book added. catalog/author/<id> — The detail view for the specific author with a primary key field named <id>. So for example, /catalog/author/11, for the 11th author added.

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Index

D

dela, **17**

K

kriterij specifikacije, **17**

P

področje kriterija, **17**
popisna postavke, **16**
posebna določila, **17**
postavka, **16**

S

skupina del, **17**
skupina področja kriterija, **17**
specifikacija, **17**
splošna določila, **17**

T

tehnične specifikacije del, **16**

V

vrsta del, **17**
vrsta določila, **17**