



CONTINUOUS DEPLOYMENT

na przykładzie aplikacji w Django



Wojciech Lichota - STX Next
Lipiec 2017



Wojciech Lichota



STX Next

wojciech@lichota.pl

[@wlichota](#)

<http://lichota.pl>

ZAWINĘLIŚMY
DO
GDAŃSKA



1. Wstęp

2. Workflow

3. Narzędzia

4. Wyzwania



1. Wstęp

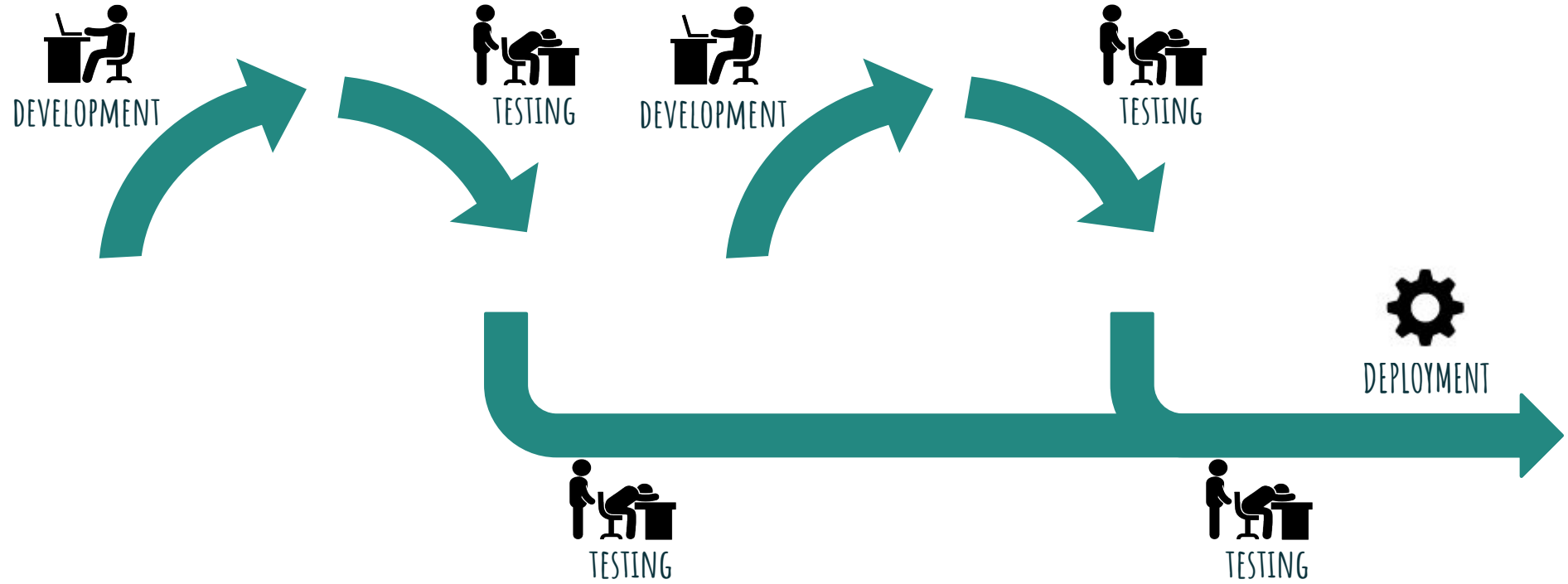
2. Workflow

3. Narzędzia

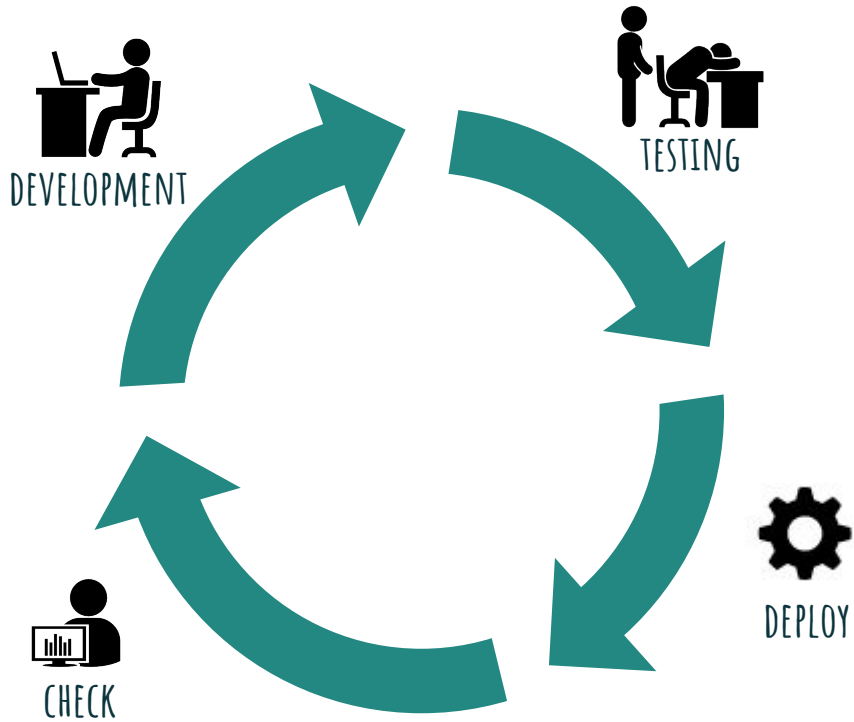
4. Wyzwania



Iterative Deployment



Continuous Deployment



Results: 81

All (507)

Application (81)

Confirmation (7)

Verification (103)

Interview (95)

Decision (109)

Employment (103)

#	Name	E-Mail / Phone Number	City	Position (Status)	Source	Age	Experience	Creation Date	Latest Activity Date	
1.	Van Hernandez Former Employee django css	test480@stxnext.com 123 880 789	Łódź	Senior Python Developer APPLICATION	Recommendation	39	7	2014-09-30	2014-09-30	2
2.	Nadia Shupe php Employee angular	test478@stxnext.com 123 862 789	Poznań	Front End Developer APPLICATION	Recommendation	25	0	2014-09-30	2014-09-30	1
3.	Marion Bunker php html	test476@stxnext.com 123 179 789	Poznań	QA APPLICATION	Direct search	36	2	2014-09-30	2014-09-30	5
4.	Penny Perze	test465@stxnext.com 123 242 789	Piła	Front End Developer APPLICATION	CV database	39	5	2014-09-30	2014-09-30	1
5.	Patricia Padilla python php html InBox django	test444@stxnext.com 123 433 789	Łódź	Front End Developer APPLICATION	Job fair	21	3	2014-09-30	2014-09-30	2

STX Next RMS

typeahead.js

Django

CKEditor factory-boy

South Django REST Framework

Bootstrap python-linkedin
bleach Selectize.js

python-social-auth PostgreSQL
Django-REST-Swagger

nose AngularJS

Underscore.js

RMS Team


 **jakubjanuzik** #1
470 commits / 18,397 ++ / 9,593 --

 **marek-leszczynski** #2
188 commits / 5,841 ++ / 2,327 --


 **mnowakowska** #3
157 commits / 6,905 ++ / 2,324 --


 **romanlevin** #4
117 commits / 4,480 ++ / 2,291 --


 **dpolcyn** #5
116 commits / 3,950 ++ / 3,667 --

 **KKrisu** #6
113 commits / 20,790 ++ / 20,382 --

 **alubeck** #7
107 commits / 21,785 ++ / 11,395 --

 **ArGroc** #8
95 commits / 6,961 ++ / 4,232 --

 **mrpear** #9
88 commits / 3,340 ++ / 2,390 --

 **ImEagle** #10
82 commits / 3,782 ++ / 1,175 --

 **pkucmus** #11
74 commits / 8,411 ++ / 21,515 --

 **add00** #12
70 commits / 2,593 ++ / 1,341 --

 **mjelonek92** #13
55 commits / 1,789 ++ / 1,068 --

 **lukaszjagodziniski** #14
55 commits / 43,288 ++ / 22,084 --

 **bitrut** #15
42 commits / 1,810 ++ / 947 --

 **avalanchy** #16
38 commits / 11,246 ++ / 1,941 --


 **nihilifer** #17
30 commits / 538 ++ / 136 --

 **Exorades** #18
22 commits / 718 ++ / 835 --

 **romankierzkowski** #19
19 commits / 2,086 ++ / 1,215 --


 **sargo** #20
14 commits / 217 ++ / 129 --

 **Alkemic** #21
10 commits / 339 ++ / 106 --

 **AcidWeb** #22
7 commits / 162 ++ / 407 --

 **Gruszks** #23
7 commits / 348 ++ / 125 --

 **msopocko** #24
1 commit / 16 ++ / 5 --

 **dasm** #25
1 commit / 88 ++ / 84 --



1. Wstęp

2. Workflow

3. Narzędzia

4. Wyzwania

GIT Workflow

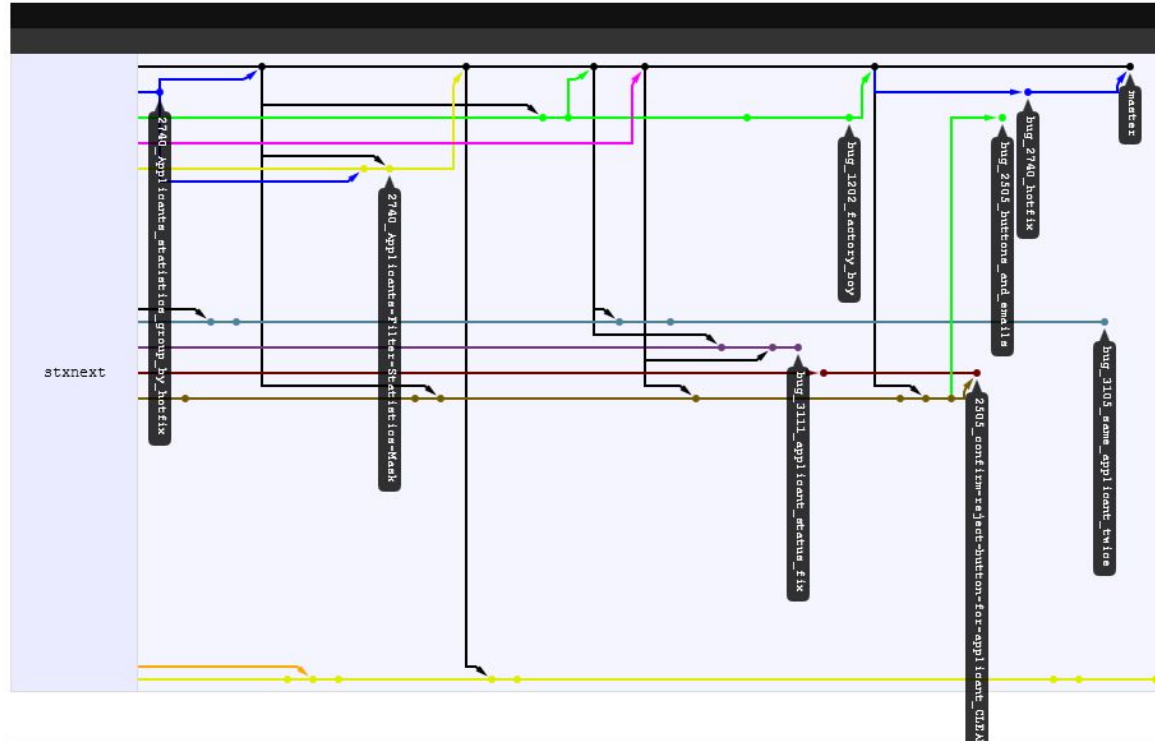
Graph Members

The rms network graph

Keyboard shortcuts available

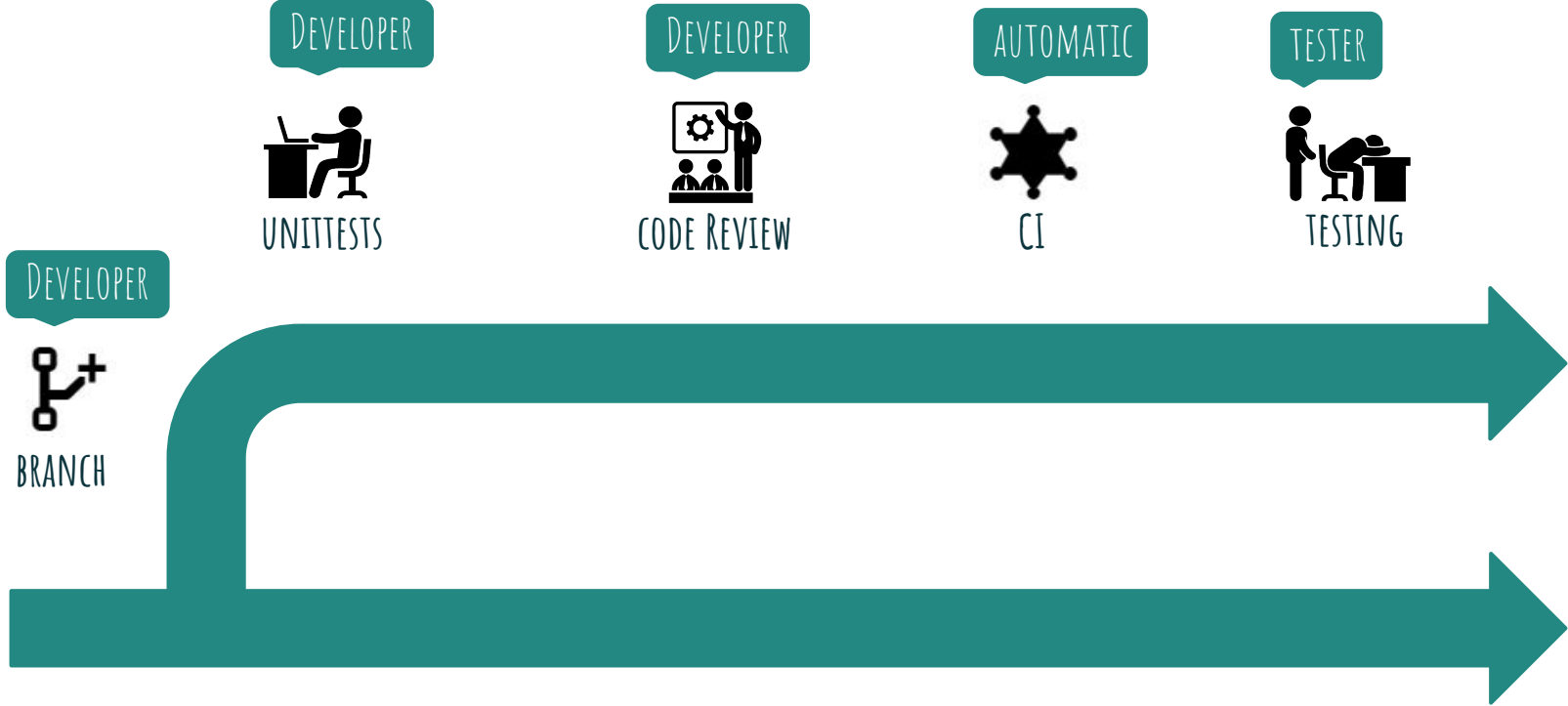
All branches in the network using stxnext/rms as the reference point. Read our blog post about how it works.

Show Help



- ★ jeden branch główny (master)
- ★ feature branch (oddzielny branch na każdą historyjkę)
- ★ pull requesty

QA Workflow




Deploy Workflow

TESTER



MERGE

AUTOMATIC



BACKUP

TESTER

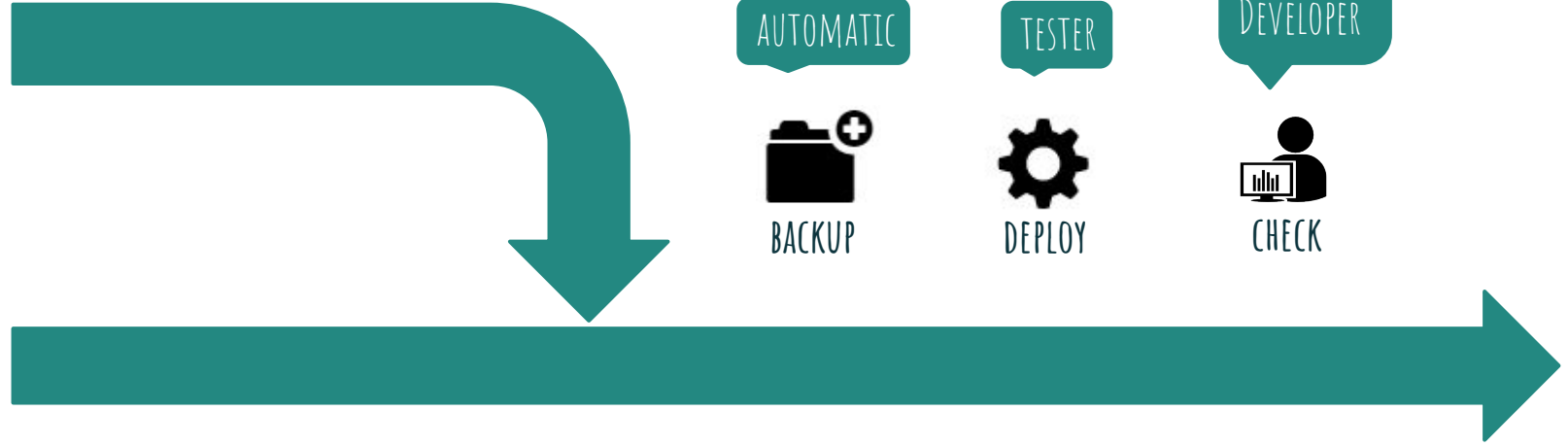


DEPLOY

AUTOMATIC
TESTER
DEVELOPER



CHECK





1. Wstęp

2. Workflow

3. Narzędzia

4. Wyzwania

Konkurs



Is Python the best programming language?

◆ **A:** Yes

◆ **B:** No



JENKINS

continuous
integration

S	W	Name ↓	Ostatni Sukces	Ostatni Błąd	Ostatni Czas Trwania
		RMS-live	1 day 18 hr - #136	—	3 min 43 sec
		RMS-PR-testing	23 hr - #65	23 hr - #64	1 min 24 sec
		RMS-staging	1 day 18 hr - #725	—	2 min 21 sec
		RMS-testing	1 day 18 hr - #10829	—	1 min 58 sec

Instancje:

- ★ testing - dla każdego z testerów, na feature branch
- ★ staging - z danymi testowymi
- ★ PR-testing - automatyczne budowanie każdego pull requesta
- ★ live - z danymi produkcyjnymi



JENKINS

★
continuous
integration

Przydatne wtyczki Jenkinsa

- ★ Violations Plugin - pep8, pylint
- ★ Post build task Plugin - kroki po prawidłowym buildzie (np. uruchomienie Fabrica)
- ★ Matrix Authorization Strategy Plugin - uprawnienia (np. kto może wrzucać na produkcję)
- ★ Github Plugin - automatyczne tagowanie
- ★ GitHub Pull Request Builder - automatyczne budowanie PR

Dodatki do Django wspomagające integrację z Jenkinsem:

- ★ django-nose - raporty odpalenia unittestów (opcja --with-xunit)
- ★ nosecover - raport pokrycia testami (opcja --with-xcover)



DOCKER



BUILDOUT

- ★ Tworzy reprodukowalne środowisko
- ★ Odseparowane sandboxy
- ★ Uproszczenie komend
(opcje uruchomienia w konfiguracji buildouta)

Przydatne recepty:

- ★ `djangorecipe`
- ★ `collective.recipe.template`
- ★ `buildout.recipe.uwsgi`
- ★ `mr.developer`



WEBPACK

★
module
bundler

- ★ Przygotowuje wszystkie zasoby frontendowe
-

Przykładowe zadania:

- ★ łączenie plików JS/CSS
- ★ kopiowanie plików (np. obrazków z bibliotek do projektu)
- ★ kompresja JS/CSS
- ★ obfuskacja JS (tylko live)
- ★ uruchomienie jshint, compass



FABRIC



remote
execution

- ★ Spięcie kilku komend (np. odpalenie buildout, webpacka, restarty serwerów) w jedno polecenie
- ★ Uruchomienie wybranych komend na zdalnych serwerach (poprzez ssh)



SENTRY



error
aggregation

- ★ Zbieranie wszystkich błędów występujących w aplikacji
- ★ Rozwiązanie On-Premise (self hosted)
- ★ Łatwa konfiguracja poprzez bibliotekę *Raven*
- ★ Wygodne powiadomienia

1. Wstęp

2. Workflow

3. Narzędzia

4. Wyzwania



Restarty aplikacji

Problem

Kilka instancji na wspólnym Apache i mod_wsgi.
Restart jednej instancji wpływa na chwilową niedostępność innych.

Rozwiązanie

Osobny uWSGI dla każdej instancji.

PROTIP

Użyj **Master FIFO** do zarządzania procesem uWSGI.

```
while lsof uwsgi_ctl; do
    echo Q > uwsgi_ctl
done
```

Baza danych

Problem

Dwa otwarte pull requesty i w obu migracja zmieniająca ten sam model.

Utrzymywanie wstecznych migracji, brak konfliktów.

Rozwiązanie

Instancja testing zawsze na świeżej kopii bazy danych. Plik migrations/version.txt z ostatnim numerem migracji.

PROTIP

```
# drop all tables
psql rms_testing -t -c "select 'drop table \" | tablename | '\" cascade;'
from pg_tables where schemaname = 'public'" | psql rms_testing
# copy tables
pg_dump rms_staging | psql rms_testing
```

Revert na live

Problem

Jeżeli wrzutka na **live** spowoduje błędy, potrzebny jest szybki sposób przywrócenia i uruchomienia poprzedniej wersji.

Rozwiązanie

- ★ Tagowanie kodu
- ★ Backup bazy danych
- ★ wrzutki tylko w godzinach gdy admin jest “pod ręką”
- ★ Automatyzacja rewertu

PROTIP

```
# backup db  
pg_dump -C rms > ~/rms_backup_$(date +"%Y%m%d%H%M").psql
```

Używaj narzędzia do monitoringu - np. **Sentry**, **Nagios**, **NewRelic**.

Podsumowanie

- ★ Przedstawione rozwiązanie wystarczające dla niekrytycznych aplikacji
- ★ Szybciej nowa rzecz na produkcji
- ★ Większa ilość pracy dla testera, mniej dla dewelopera
- ★ Zawsze zakładaj problemy (jeżeli coś się może zepsuć - to na pewno się zepsuje)



ZAWINĘLIŚMY
DO
GDAŃSKA