Deploying application on AWS

Yesin Oleksii

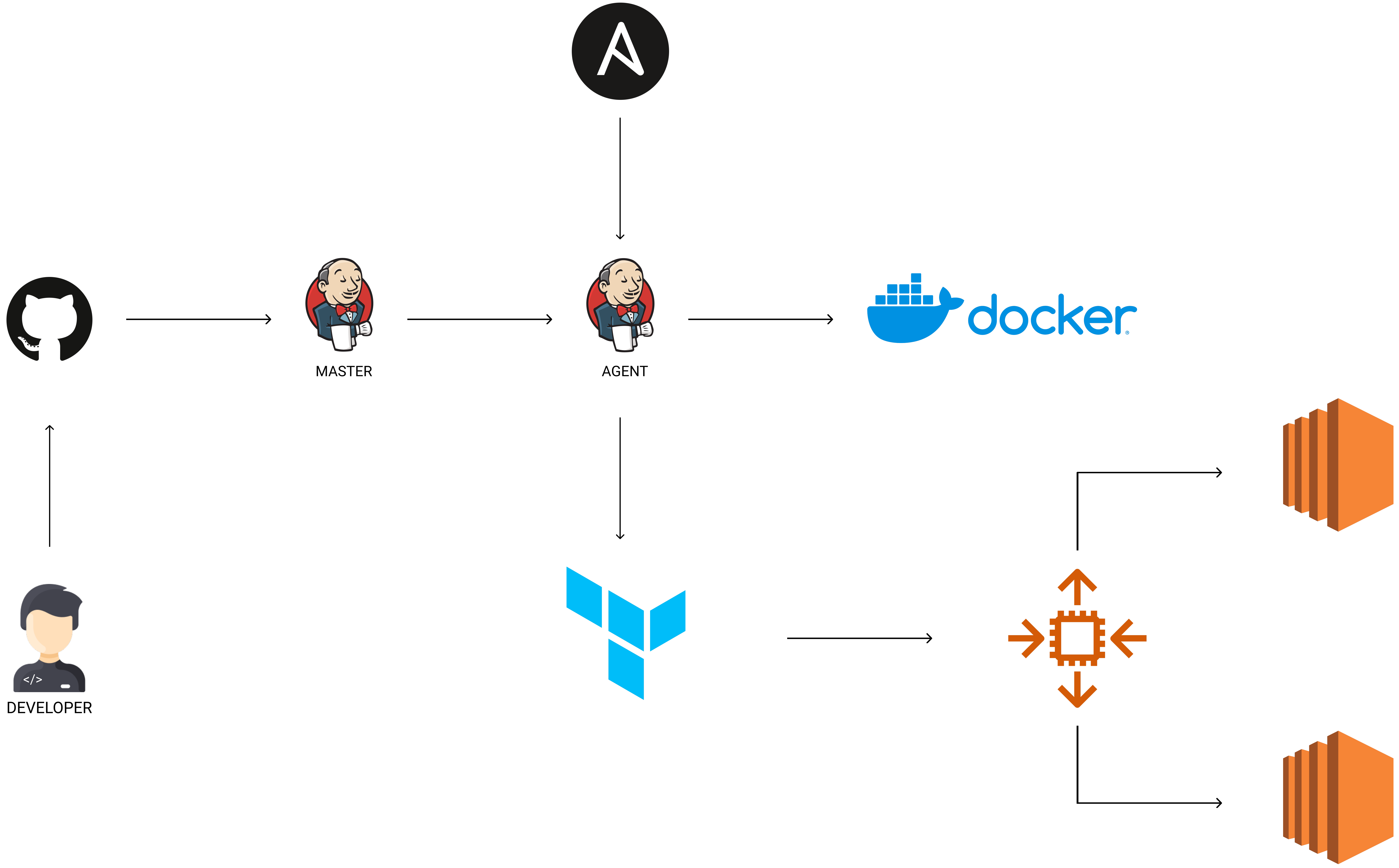
Goals of this project

- Eliminate downtime while deploying a new version of an application
- Storage of containers on DockerHub
- Deploying an application on AWS

Tasks of this project

- Keep containers on DockerHub
- Realize green/blue deployment
- Implement CI/CD

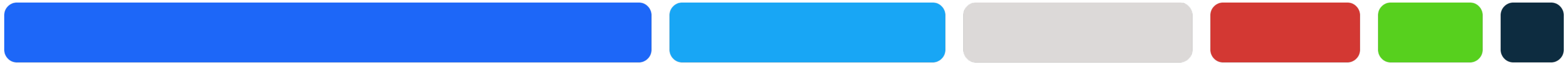
Project Scheme



Tools

- Ansible
- Docker
- Terraform
- Jenkins
- AWS
- GitHub

Languages

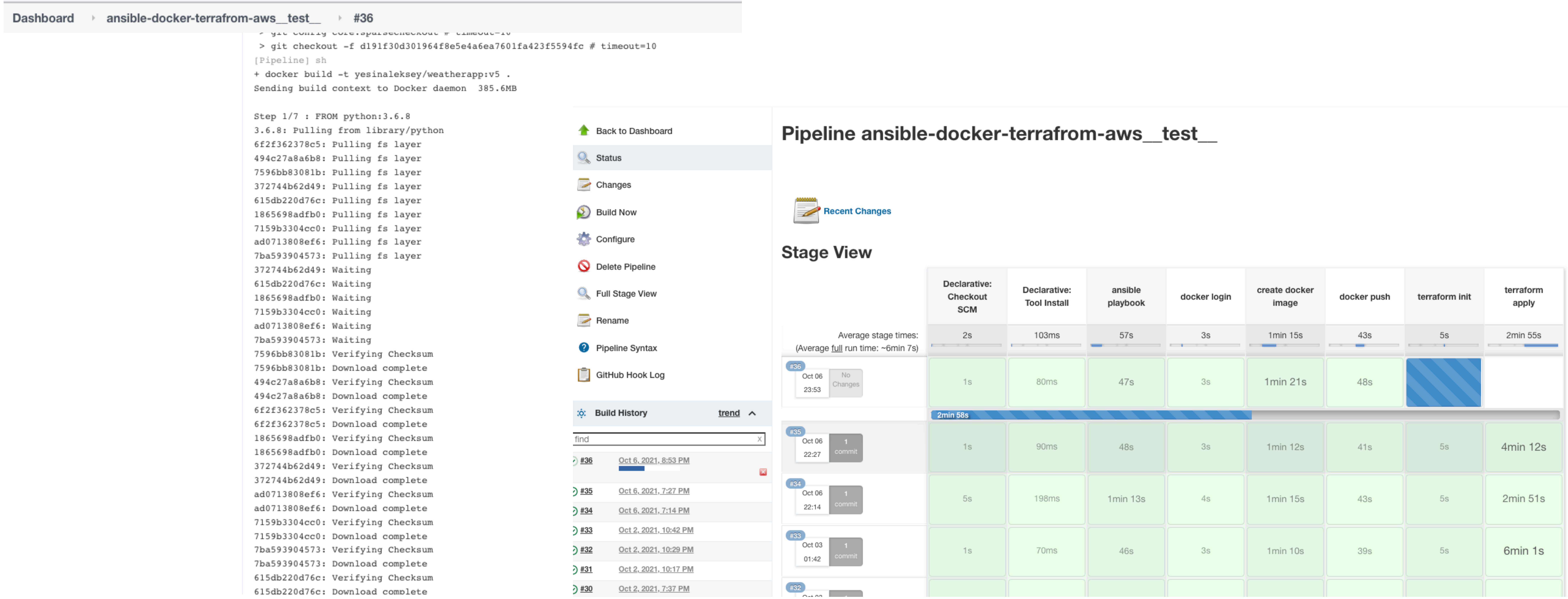


●	PYTHON	47.3%
●	GROOVY	17.9%
●	HCL	17.2%
●	HTML	12.1%
●	SHELL	4.0%
●	DOCKERFILE	1.5%

Purpose

Achieve high-quality deployment to production and smooth operation of our website

Implementation



```
 > git config core.sparsecheckout # timeout=10
 > git checkout -f d191f30d301964f8e5e4a6ea7601fa423f5594fc # timeout=10
[Pipeline] sh
+ docker build -t yesinaleksey/weatherapp:v5 .
Sending build context to Docker daemon  385.6MB

Step 1/7 : FROM python:3.6.8
3.6.8: Pulling from library/python
6f2f362378c5: Pulling fs layer
494c27a8a6b8: Pulling fs layer
7596bb83081b: Pulling fs layer
372744b62d49: Pulling fs layer
615db220d76c: Pulling fs layer
1865698adfb0: Pulling fs layer
7159b3304cc0: Pulling fs layer
ad0713808ef6: Pulling fs layer
7ba593904573: Pulling fs layer
372744b62d49: Waiting
615db220d76c: Waiting
1865698adfb0: Waiting
7159b3304cc0: Waiting
ad0713808ef6: Waiting
7ba593904573: Waiting
7596bb83081b: Verifying Checksum
7596bb83081b: Download complete
494c27a8a6b8: Verifying Checksum
494c27a8a6b8: Download complete
6f2f362378c5: Verifying Checksum
6f2f362378c5: Download complete
1865698adfb0: Verifying Checksum
1865698adfb0: Download complete
372744b62d49: Verifying Checksum
372744b62d49: Download complete
ad0713808ef6: Verifying Checksum
ad0713808ef6: Download complete
7159b3304cc0: Verifying Checksum
7159b3304cc0: Download complete
7ba593904573: Verifying Checksum
7ba593904573: Download complete
615db220d76c: Verifying Checksum
615db220d76c: Download complete
```

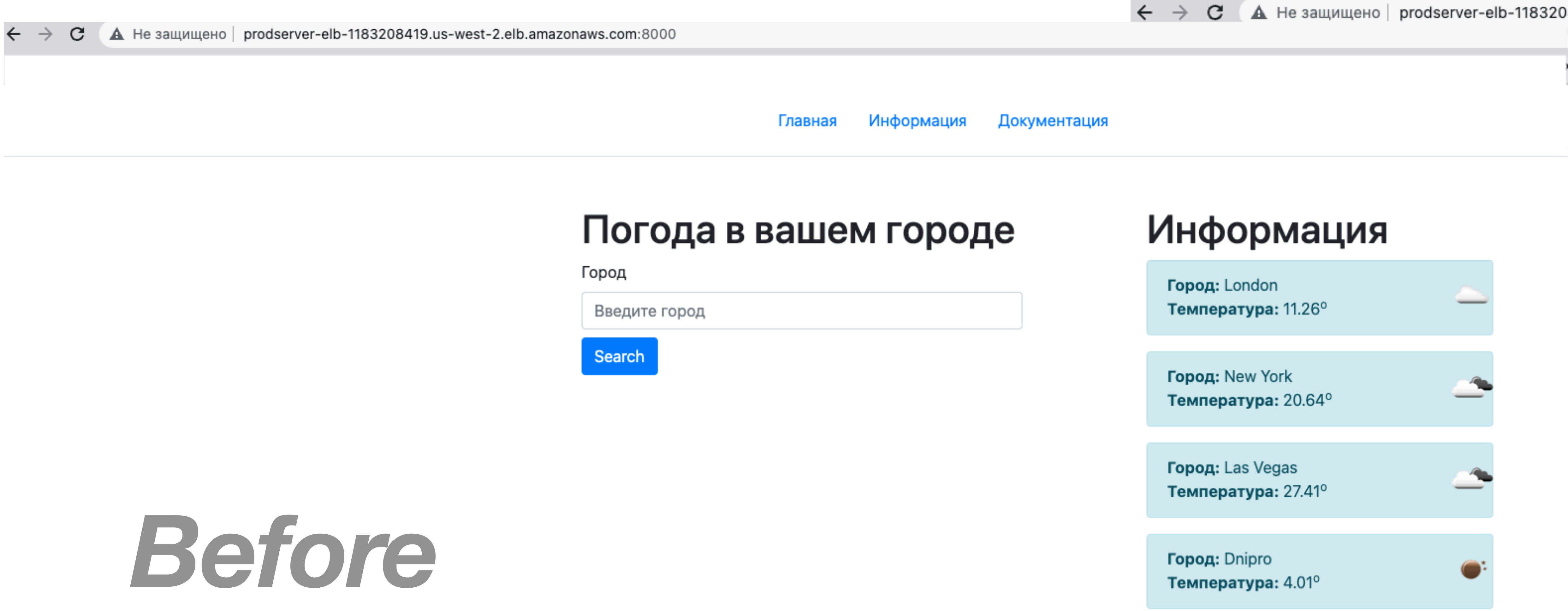
#36	Oct 6, 2021, 8:53 PM
#35	Oct 6, 2021, 7:27 PM
#34	Oct 6, 2021, 7:14 PM
#33	Oct 2, 2021, 10:42 PM
#32	Oct 2, 2021, 10:29 PM
#31	Oct 2, 2021, 10:17 PM
#30	Oct 2, 2021, 7:37 PM

Pipeline ansible-docker-terrafrom-aws__test__

Stage View

	Declarative: Checkout SCM	Declarative: Tool Install	ansible playbook	docker login	create docker image	docker push	terraform init	terraform apply
Average stage times: (Average full run time: ~6min 7s)	2s	103ms	57s	3s	1min 15s	43s	5s	2min 55s
#36 Oct 06 23:53 No Changes	1s	80ms	47s	3s	1min 21s	48s		
2min 58s								
#35 Oct 06 22:27 1 commit	1s	90ms	48s	3s	1min 12s	41s	5s	4min 12s
#34 Oct 06 22:14 1 commit	5s	198ms	1min 13s	4s	1min 15s	43s	5s	2min 51s
#33 Oct 03 01:42 1 commit	1s	70ms	46s	3s	1min 10s	39s	5s	6min 1s
#32								

Implementation



Не защищено | prodserver-elb-1183208419.us-west-2.elb.amazonaws.com:8000

Погода в вашем городе

Город

Введите город

Search

Информация

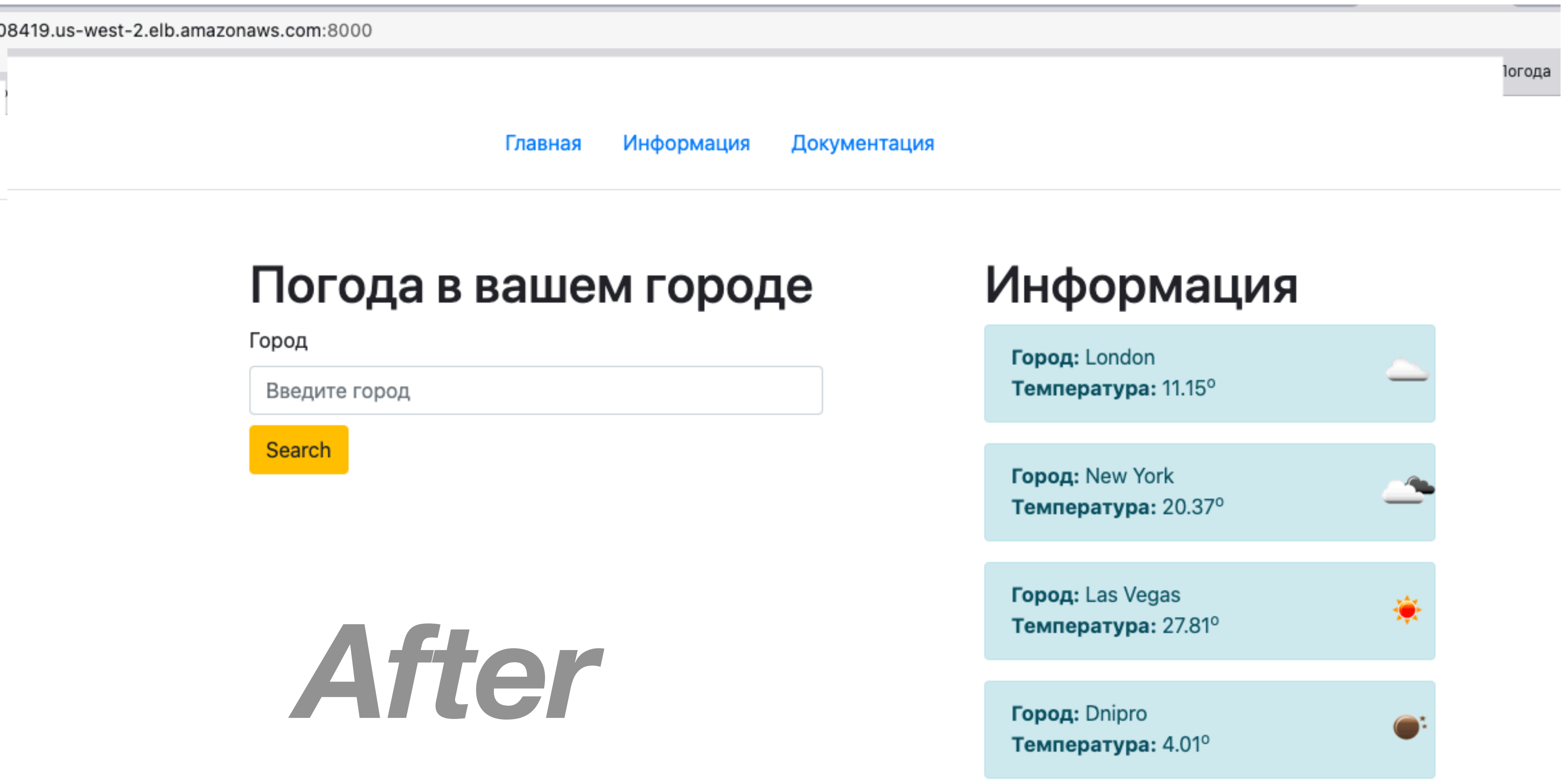
Город: London
Температура: 11.26°

Город: New York
Температура: 20.64°

Город: Las Vegas
Температура: 27.41°

Город: Dnipro
Температура: 4.01°

Before



Не защищено | prodserver-elb-1183208419.us-west-2.elb.amazonaws.com:8000

Погода в вашем городе

Город

Введите город

Search

Информация

Город: London
Температура: 11.15°

Город: New York
Температура: 20.37°

Город: Las Vegas
Температура: 27.81°

Город: Dnipro
Температура: 4.01°

After

Conclusions

Green/Blue Deployment implementation was successful

There are different approaches for achieving my goals:

- Automatical introduction of agent-server raising, using EC2plugin in Jenkins
- Instead of default database use PostgreSQL(for example)