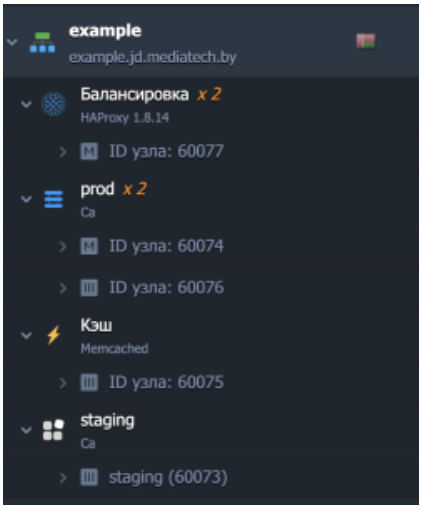
CI/CD Docker deployment

- docker , Gitlab.

Jelastic HTTP API ()

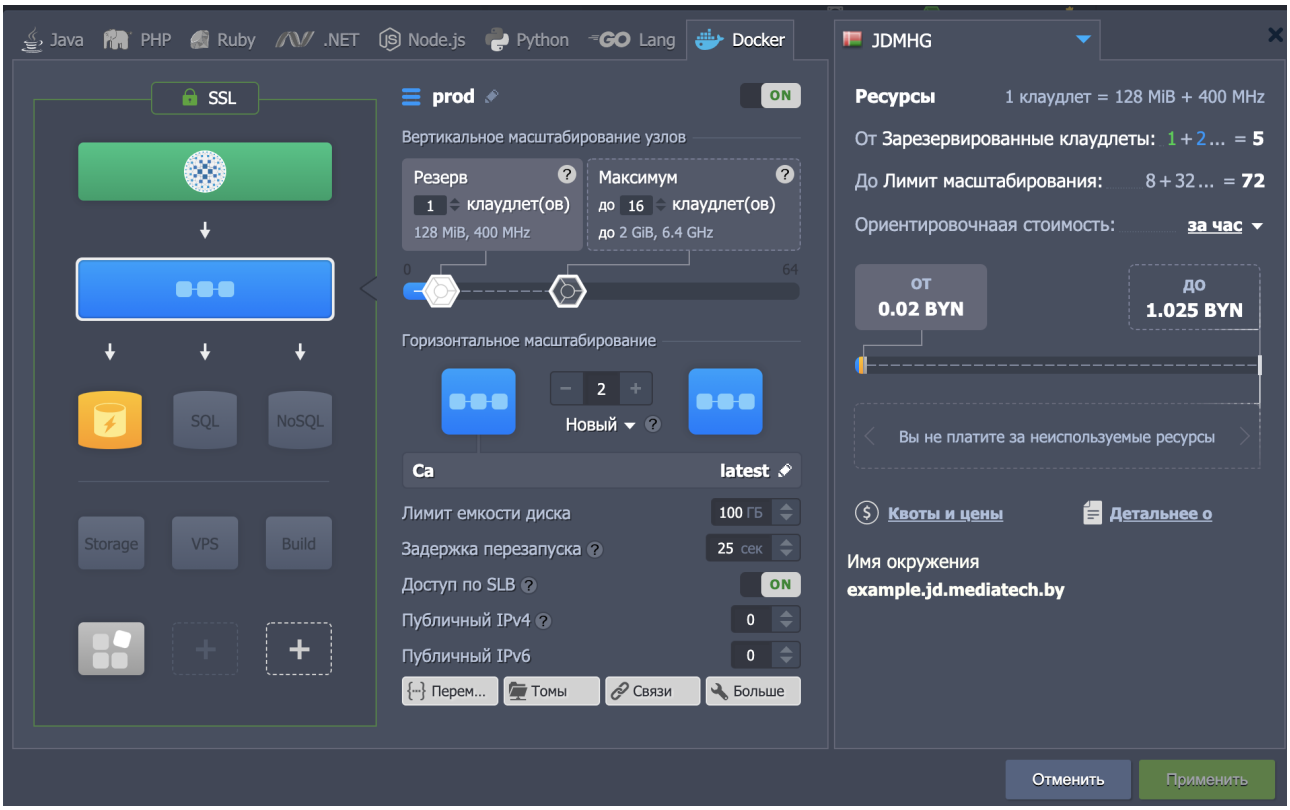
:



, docker web-, memcached, docker .

, , . , (zero-downtime deployment, blue/green deployment).

:



, **2**.

, - , HTTP API Jelastic. python:

```
#!/usr/bin/python3
import sys
import json
import requests

STAGING_TAG = 'master'
PRODUCTION_TAG = 'latest'

TOKEN = 'xxx' # https://docs.jelastic.com/personal-access-tokens/
ENV = 'example'
STAGING_ID = 60073


JDOMAIN = 'mycloud.by'
APPID = '58bdf83fea6af021e0c94ba13730fd6b' #  mycloud.by

if len(sys.argv) < 2 or len(sys.argv) > 3:
    print('Usage: ' + sys.argv[0] + ' [staging|production] (tag)')
    sys.exit(1)

mode = sys.argv[1]

try:
    tag = sys.argv[2]
except IndexError:
    tag = None

if mode == 'production':
    if not tag:
        tag = PRODUCTION_TAG
    URL = 'https://app.' + JDOMAIN + '/1.0/environment/control/rest/redeploycontainersbygroup'
    DATA = {
        'token': TOKEN,
        'envName': ENV,
        'nodeGroup': 'cp',
        'tag': tag,
        'isSequential': True,
        'useExistingVolumes': True }
elif mode == 'staging':
    if not tag:
        tag = STAGING_TAG
    URL = 'https://app.' + JDOMAIN + '/1.0/environment/control/rest/redeploycontainerbyid'
    DATA = {
        'token': TOKEN,
        'envName': ENV,
        'nodeId': STAGING_ID,
        'tag': tag,
        'useExistingVolumes': True
        }
else:
    raise ValueError("Invalid mode! Mode should be 'staging' or 'production'")

HEADERS = {
    'User-Agent': 'Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) '
                  'Chrome/56.0.2924.87 Safari/537.36',
    'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8',
    'Accept-Encoding': 'gzip, deflate, sdch',
    'Accept-Language': 'en,en-US;q=0.8,ru;q=0.6,uk;q=0.4',
    'Connection': 'keep-alive'
}

response = requests.post(URL, headers=HEADERS, data=item)
data = json.loads(json.dumps(response.json()))
print(data)
assert data.get("result", 1) == 0
```

, - production, staging. , docker docker registry .

, .

git-

Gitlab CI/CD

Gitlab CI/CD *.gitlab-ci.yml*

:

```
stages:
- build
- release

variables:
  CONTAINER_TEST_IMAGE: $CI_REGISTRY_IMAGE:$CI_COMMIT_REF_NAME
  CONTAINER_RELEASE_IMAGE: $CI_REGISTRY_IMAGE:latest

build:
  image: docker:latest
  stage: build
  before_script:
    - docker login -u gitlab-ci-token -p $CI_BUILD_TOKEN $CI_REGISTRY
  script:
    - docker build -f Dockerfile --pull -t $CONTAINER_TEST_IMAGE .
    - docker push $CONTAINER_TEST_IMAGE

deploy_staging:
  image: python:3
  stage: release
  script:
    - python deploy.py staging $CI_COMMIT_REF_NAME
  when: manual

deploy_production:
  image: python:3
  stage: release
  before_script:
    - docker login -u gitlab-ci-token -p $CI_BUILD_TOKEN $CI_REGISTRY
  script:
    - docker pull $CONTAINER_TEST_IMAGE
    - docker tag $CONTAINER_TEST_IMAGE $CONTAINER_RELEASE_IMAGE
    - docker push $CONTAINER_RELEASE_IMAGE
  after_script:
    - python deploy.py production
  only: master
  when: manual
```

Gitlab CI/CD, pipeline, , dev production.