Скачивание необходимых инструментов

```
parallels@debian-gnu-linux-12:~/uber$ curl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 | bash
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 11903  100 11903    0     0  14028      0 --:--:-- --:--:-- --:--:-- 14020
Downloading https://get.helm.sh/helm-v3.16.4-linux-arm64.tar.gz
Verifying checksum... Done.
Preparing to install helm into /usr/local/bin
[sudo] password for parallels:
helm installed into /usr/local/bin/helm
parallels@debian-gnu-linux-12:~/uber$ helm version
version.BuildInfo{Version:"v3.16.4", GitCommit:"7877b45b63f95635153b29a42c0c2f4273ec45ca", GitTreeState:"clean", GoVersion:"
go1.22.7"}
```

```
parallels@debian-gnu-linux-12:~$ curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-
arm64
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 99.0M  100 99.0M    0     0  4474k      0  0:00:22  0:00:22 --:--:-- 6713k
parallels@debian-gnu-linux-12:~$ sudo install minikube-linux-arm64 /usr/local/bin/minikube
parallels@debian-gnu-linux-12:~$ minikube version
minikube version: v1.34.0
commit: 210b148df93a80eb872ecbeb7e35281b3c582c61
parallels@debian-gnu-linux-12:~$ minikube start --driver=docker
😄  minikube v1.34.0 on Debian 12.7 (arm64)
✨  Using the docker driver based on user configuration
```

```
parallels@debian-gnu-linux-12:~$ minikube start --driver=docker
😄  minikube v1.34.0 on Debian 12.7 (arm64)
✨  Using the docker driver based on user configuration

🧯  The requested memory allocation of 1977MiB does not leave room for system overhead (total system memory: 1977MiB). You
ay face stability issues.
💡  Suggestion: Start minikube with less memory allocated: 'minikube start --memory=1977mb'

📌  Using Docker driver with root privileges
👍  Starting "minikube" primary control-plane node in "minikube" cluster
🚜  Pulling base image v0.0.45 ...
💾  Downloading Kubernetes v1.31.0 preload ...
    > preloaded-images-k8s-v18-v1...:  91.95 MiB / 307.61 MiB  29.89% 101.52 Ki
    > gcr.io/k8s-minikube/kicbase...:  100.22 MiB / 441.45 MiB  22.70% 101.61 K
    > index.docker.io/kicbase/sta...:  441.45 MiB / 441.45 MiB  100.00% 124.94
❗  minikube was unable to download gcr.io/k8s-minikube/kicbase:v0.0.45, but successfully downloaded docker.io/kicbase/stab
e:v0.0.45 as a fallback image
🔥  Creating docker container (CPUs=2, Memory=1977MB) ...
    > kubectl.sha256:  64 B / 64 B [-------------------------] 100.00% ? p/s 0s
    > kubelet.sha256:  64 B / 64 B [-------------------------] 100.00% ? p/s 0s
    > kubeadm.sha256:  64 B / 64 B [-------------------------] 100.00% ? p/s 0s
    > kubelet:  71.18 MiB / 71.18 MiB [--------------] 100.00% 1.64 MiB p/s 44s
    > kubectl:  52.44 MiB / 52.44 MiB [----------] 100.00% 143.24 KiB p/s 6m15s
    > kubeadm:  54.25 MiB / 54.25 MiB [-----------] 100.00% 92.80 KiB p/s 9m59s

    ▪ Generating certificates and keys ...
    ▪ Booting up control plane ...
    ▪ Configuring RBAC rules ...
🔗  Configuring bridge CNI (Container Networking Interface) ...
🔎  Verifying Kubernetes components...
    ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟  Enabled addons: storage-provisioner, default-storageclass
💡  kubectl not found. If you need it, try: 'minikube kubectl -- get pods -A'
🏄  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

Проверка

```
parallels@debian-gnu-linux-12:~$ minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured

parallels@debian-gnu-linux-12:~$ kubectl config use-context minikube
Switched to context "minikube".
parallels@debian-gnu-linux-12:~$ kubectl get nodes
NAME       STATUS   ROLES           AGE     VERSION
minikube   Ready    control-plane   6d12h   v1.31.0
parallels@debian-gnu-linux-12:~$
```

Создали Dockerfile в корне проекта. Далее собираем из него образ

(docker build -t uber:latest)

Open ▾ ⊞ Dockerfile ~/uber

```
FROM nginx:alpine
COPY src /usr/share/nginx/html
```

Там же создаем Deployment.yaml в папке k8s/staging k8s/production

Open ▾ ⊞ deployment.yaml ~/uber

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: uber
spec:
  replicas: 2
  selector:
    matchLabels:
      app: uber
  template:
    metadata:
      labels:
        app: uber
    spec:
      containers:
      - name: uber
        image: uber:latest
        imagePullPolicy: Never
        ports:
        - containerPort: 80
```

И Service.yaml в папке k8s/staging k8s/production

Open

service.yaml
~/uber

```yaml
apiVersion: v1
kind: Service
metadata:
  name: uber-service
spec:
  selector:
    app: uber
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
  type: NodePort
```

Создаем файл .gitlab-ci.yml в корне проект

```yaml
stages:
  - build
  - deploy_staging
  - deploy_production

variables:
  DOCKER_IMAGE: registry.gitlab.com/lera8lee/uber:$CI_COMMIT_SHORT_SHA

build:
  stage: build
  script:
    - docker build -t $DOCKER_IMAGE .
    - echo $CI_REGISTRY_PASSWORD | docker login -u $CI_REGISTRY_USER --password-stdin $CI_REGISTRY
    - docker push $DOCKER_IMAGE

deploy_staging:
  stage: deploy_staging
  script:
    - kubectl apply -f k8s/staging/deployment.yaml
    - kubectl apply -f k8s/staging/service.yaml
  environment:
    name: staging
    url: http://192.168.49.2:32277
  only:
    - main

deploy_production:
  stage: deploy_production
  script:
    - kubectl apply -f k8s/production/deployment.yaml
    - kubectl apply -f k8s/production/service.yaml
  environment:
    name: production
    url: http://192.168.49.2:32277
  only:
    - main
```

Регистрируем runner gitlab предварительно установив gitlab-ce и установив логин и пароль

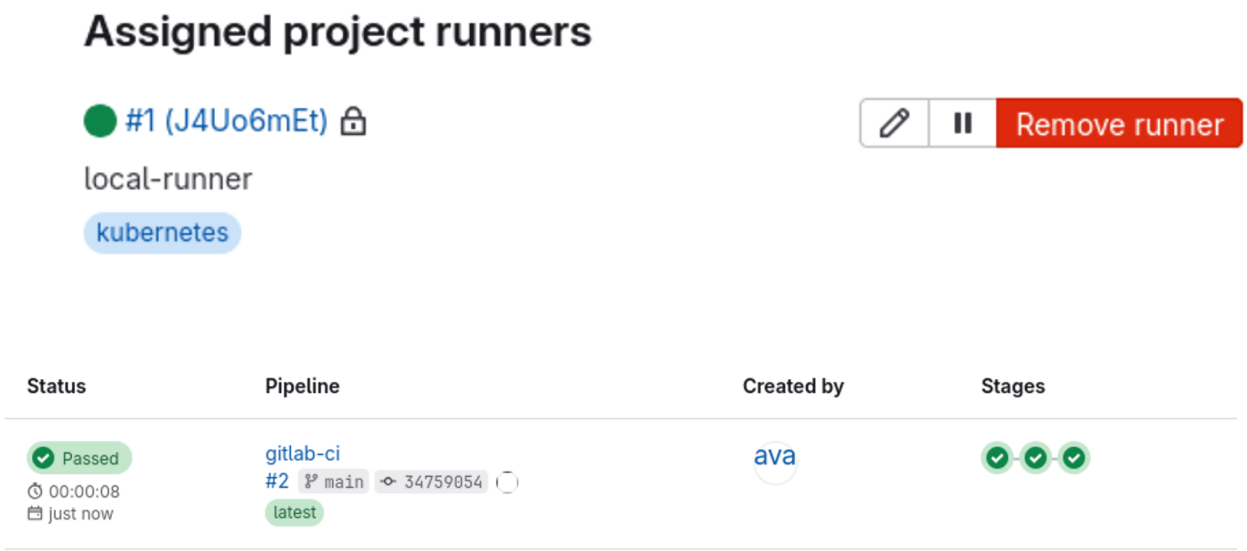
```
parallels@debian-gnu-linux-12:~$ sudo gitlab-runner register
Runtime platform                                    arch=arm64 os=linux pid=10085 revision=3153ccc6
ersion=17.7.0
Running in system-mode.

Enter the GitLab instance URL (for example, https://gitlab.com/):
http://localhost
Enter the registration token:
GR13489419prLYPzjM_oB91fH8hJV
Enter a description for the runner:
[debian-gnu-linux-12]: local-runner
Enter tags for the runner (comma-separated):
kubernetes
Enter optional maintenance note for the runner:
```

Проверка что раннер работает

```
parallels@debian-gnu-linux-12:~$ sudo gitlab-runner status
Runtime platform                                    arch=arm64 os=linux pid=12872 revision=3153ccc6
 version=17.7.0
gitlab-runner: Service is running
parallels@debian-gnu-linux-12:~$
```

Переходим в Settings -> CI/CD -> Runners. Раннер отображается в разделе “Project runners” и находится в статусе **Active**.



Administrator / kubernetes-project / Pipelines / #2

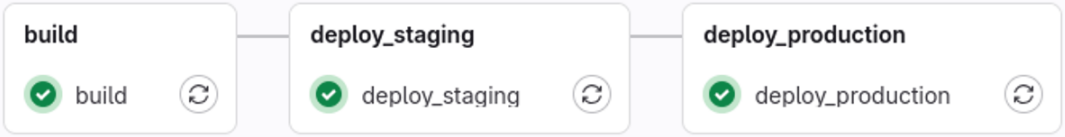
gitlab-ci

Passed **Administrator** created pipeline for commit `34759054` 4 hours ago, finished 5 minutes ago

For `main`

latest 11 jobs 8 seconds, queued for 5 seconds

Pipeline | Jobs 11 | Tests 0




```
parallels@debian-gnu-linux-12:~/uber$ minikube service uber-service --url
http://192.168.49.2:32277
```