**🧱 Что ты построишь:**

| **Компонент** | **Назначение** |
| --- | --- |
| ✅ PM2 Cluster | Распараллеливание Node.js |
| ✅ CI через GitHub Actions | Авто-сборка, авто-тесты |
| ✅ DockerHub Push | Публикация образа |
| ✅ Prod Dockerfile | Multistage, non-root |
| ✅ HTTPS + Nginx | Шифрование и защита |
| ✅ Loki / Grafana | Логирование и мониторинг |
| ✅ Kubernetes (опц.) | Деплой в кластер |

**✅ ЗАДАНИЯ:**

**🔹 1. Используй PM2 кластерный режим**

📄 ecosystem.config.js

js

КопироватьРедактировать

module.exports = {

apps: [{

name: "api",

script: "index.js",

instances: "max",

exec\_mode: "cluster",

env: {

NODE\_ENV: "production"

}

}]

};

Запуск:

bash

КопироватьРедактировать

pm2 start ecosystem.config.js

**🔹 2. Напиши Multistage Dockerfile**

Dockerfile

КопироватьРедактировать

# Stage 1 — build

FROM node:18 as build

WORKDIR /app

COPY package\*.json ./

RUN npm install

COPY . .

# Stage 2 — production

FROM node:18-alpine

WORKDIR /app

COPY --from=build /app .

RUN addgroup -S app && adduser -S app -G app

USER app

ENV NODE\_ENV=production

RUN npm prune --production

EXPOSE 3000

CMD ["node", "index.js"]

**🔹 3. CI: GitHub Actions workflow**

📄 .github/workflows/ci.yml

yaml

КопироватьРедактировать

name: CI/CD

on:

push:

branches: [main]

jobs:

build-and-test:

runs-on: ubuntu-latest

services:

mongo:

image: mongo

ports: ["27017:27017"]

steps:

- uses: actions/checkout@v3

- uses: actions/setup-node@v3

with:

node-version: 18

- run: npm install

- run: npm test

env:

MONGO\_URI: mongodb://localhost:27017/testdb

**🔹 4. Публикация в Docker Hub**

Добавь шаг после тестов:

yaml

КопироватьРедактировать

- name: Docker Login

uses: docker/login-action@v2

with:

username: ${{ secrets.DOCKER\_USERNAME }}

password: ${{ secrets.DOCKER\_PASSWORD }}

- name: Build and Push

run: |

docker build -t ${{ secrets.DOCKER\_USERNAME }}/api:latest .

docker push ${{ secrets.DOCKER\_USERNAME }}/api:latest

**🔹 5. Мониторинг через Grafana Loki**

📁 docker-compose.monitor.yml

yaml

КопироватьРедактировать

services:

loki:

image: grafana/loki:2.7.1

ports:

- "3100:3100"

promtail:

image: grafana/promtail

volumes:

- ./logs:/var/log

- ./promtail-config.yaml:/etc/promtail/promtail.yaml

📄 promtail-config.yaml

yaml

КопироватьРедактировать

server:

http\_listen\_port: 9080

positions:

filename: /tmp/positions.yaml

clients:

- url: http://loki:3100/loki/api/v1/push

scrape\_configs:

- job\_name: logfiles

static\_configs:

- targets:

- localhost

labels:

job: api

\_\_path\_\_: /var/log/\*.log

**🔹 6. Добавь логирование в JSON**

📄 logger.js

js

КопироватьРедактировать

const fs = require('fs');

const logFile = fs.createWriteStream('./logs/app.log', { flags: 'a' });

module.exports = (level, ...args) => {

const entry = JSON.stringify({

time: new Date().toISOString(),

level,

message: args.join(' ')

}) + '\n';

logFile.write(entry);

console.log(entry);

};

**🔹 7. Деплой на VPS через GitHub Actions**

Добавь в Secrets:

* SSH\_KEY
* HOST
* USER

📄 .github/workflows/deploy.yml

yaml

КопироватьРедактировать

- name: Deploy to VPS

uses: appleboy/ssh-action@v0.1.10

with:

host: ${{ secrets.HOST }}

username: ${{ secrets.USER }}

key: ${{ secrets.SSH\_KEY }}

script: |

cd /home/user/app

git pull

docker-compose down

docker-compose up -d --build

**🔹 8. (Опц.) Деплой в Kubernetes**

📄 deployment.yaml

yaml

КопироватьРедактировать

apiVersion: apps/v1

kind: Deployment

metadata:

name: api

spec:

replicas: 3

selector:

matchLabels:

app: api

template:

metadata:

labels:

app: api

spec:

containers:

- name: api

image: youruser/api:latest

ports:

- containerPort: 3000

**✅ РЕЗУЛЬТАТ УРОВНЯ 3**

| **Функция** | **Статус** |
| --- | --- |
| Multistage Dockerfile | ✅ |
| PM2 кластер | ✅ |
| CI: тесты + билд | ✅ |
| DockerHub Push | ✅ |
| Мониторинг с Loki | ✅ |
| GitHub Actions Deploy | ✅ |
| Kubernetes-ready | 🔄 (опционально) |