Файл пайплайна CI/CD, использующий GitHub Actions, описывает автоматизированный процесс интеграции и доставки программного обеспечения. Он состоит из нескольких ключевых компонентов, которые выполняются в ответ на определенные события, такие как коммиты в ветку `master`. Давайте разберем этот файл по частям.

Основные компоненты файла

1. Определение пайплайна

```
name: CI/CD

on:
    push:
      branches: [ "master" ]
```

Этот блок определяет имя пайплайна (`CI/CD`) и событие, при котором он будет запускаться — в данном случае это `push` в ветку `master`.

2. Права доступа

```
permissions:
    contents: read
    id-token: write
```

Здесь устанавливаются права доступа для выполнения действий в рамках пайплайна. В данном случае разрешается чтение содержимого репозитория и запись токена идентификации.

PROF

3. Задание работ (jobs)

Пайплайн состоит из трех основных работ: `run_tests`, `publish_image` и `deploy_application`.

run_tests

```
run_tests:
    runs-on: ubuntu-latest
```

Эта работа выполняется на последней версии Ubuntu. Она включает следующие шаги:

- **Проверка кода**:

  ```
  - uses: actions/checkout@v4
  ```

Этот шаг загружает код из репозитория.

- **Настройка Python**:

```
- name: Set up python 3.12
  uses: actions/setup-python@v3
  with:
      python-version: "3.12"
```

Устанавливается версия Python для выполнения тестов.

- **Установка Make**:

```
- name: Set up make
  run: |
      sudo apt update
      sudo apt install -y make
```

Устанавливается утилита Make, необходимая для запуска тестов.

- **Запуск тестов**:

```
- name: Run tests
  run: |
      make test
```

Выполняются тесты, определенные в Makefile.

PROF

publish_image

```
publish_image:
    name: Push Docker image to Docker Hub
    runs-on: ubuntu-latest
    needs: run_tests
```

Эта работа зависит от успешного завершения работы `run_tests`. Она отвечает за публикацию Docker-образа:

- **Логин в Docker Hub**:

```
- name: Log in to Docker Hub
  uses: docker/login-
action@f4ef78c080cd8ba55a85445d5b36e214a81df20a
  with:
      username: ${{ secrets.DOCKER_LOGIN }}
      password: ${{ secrets.DOCKER_PASS }}
```

Здесь происходит аутентификация на Docker Hub с использованием секретов для безопасного хранения логина и пароля.

- **Извлечение метаданных**:

```
- name: Extract metadata (tags, labels) for Docker
  id: meta
  uses: docker/metadata-
action@9ec57ed1fcdbf14dcef7dfbe97b2010124a938b7
  with:
      images: arsenier/cicd-task
```

Этот шаг извлекает метаданные для образа Docker, такие как теги и метки.

- **Сборка и публикация Docker-образа**:

```
- name: Build and push Docker image
  id: push
  uses: docker/build-push-
action@3b5e8027fcad23fda98b2e3ac259d8d67585f671
  with:
      context: .
      file: ./build/Dockerfile
      push: true
      tags: ${{ steps.meta.outputs.tags }}
      labels: ${{ steps.meta.outputs.labels }}
```

Здесь происходит сборка образа по указанному Dockerfile и его публикация в Docker Hub с использованием ранее извлеченных тегов и меток.

PROF

deploy_application

```
deploy_application:
    name: Deploy application on the remote server
    runs-on: ubuntu-latest
    needs: publish_image
```

Эта работа выполняется после успешной публикации образа. Она отвечает за развертывание приложения на удаленном сервере:

- **Деплой через SSH**:

```
- name: Log into the server via SSH and deploy image
  uses: appleboy/ssh-action@v1.2.0
  with:
    host: ${{ secrets.SELECTEL_SERVER_IP }}
    username: ${{ secrets.SELECTEL_SERVER_USER }}
    key: ${{ secrets.SELECTEL_PRIVATE_SSH_KEY }}
    script: |
        docker ps -aq | xargs docker stop | xargs docker rm
        docker run --pull=always -d -p 5000:5000 arsenier/cicd-
task:master
```

  Этот шаг выполняет вход на удаленный сервер по SSH и разворачивает контейнер с приложением, останавливая и удаляя старые контейнеры перед запуском нового.