



Устройство Gitlab CI. Построение процесса непрерывной поставки

План

- Непрерывная поставка ПО
- Gitlab CI. Функционал и концепции
- Непрерывная поставка с Gitlab CI

Непрерывная поставка ПО

Процесс разработки и эксплуатации ПО, при котором каждое изменение может быть выкачено в боевое окружение

Непрерывная поставка ПО

- Ускоряет и упрощает выпуск продукта
- Повышает качество продукта
- Сокращает время поиска и устранения дефектов
- Снижают стресс от релизов
- Повышают прозрачность процессов
- Обеспечивают командное взаимодействие

Непрерывная поставка ПО

Меняется взаимодействие в команде

- Изменения происходят часто и быстро — меняется процесс
- Окружения стандартные, разработка лучше понимает в каких условиях ПО работает в production — общий язык с эксплуатацией

Непрерывная поставка ПО

Повышается качество продукта

- Запуск в эталонном окружении
- Автотесты и анализаторы кода
- Одни и те же подходы для всех окружений
- Доступность команде мониторинга и логов ПО

Непрерывная поставка ПО

Уменьшение рисков и стресса

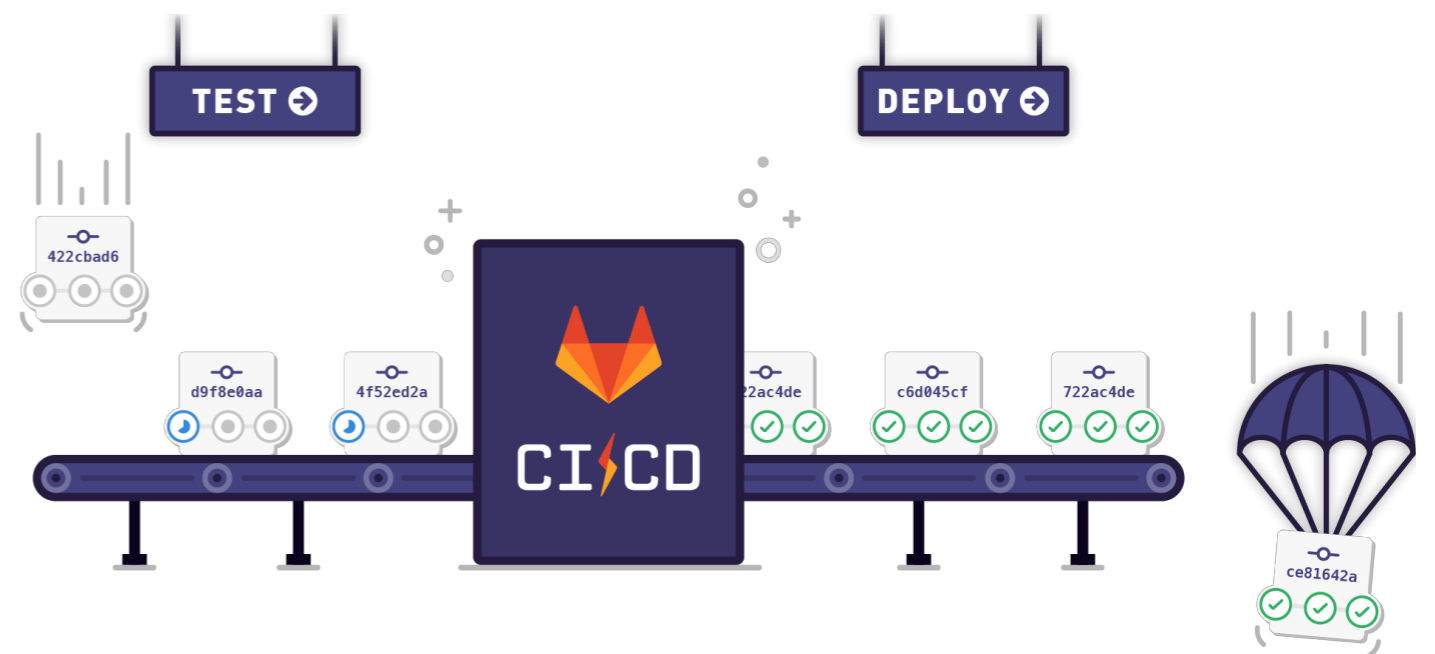
- Небольшие и частые изменения
- Меньше изменений за раз – проще откаты
- Проще отлаживать проблемы и разбирать инциденты

Из чего состоит CD и чем Gitlab может помочь

- **Version Control**
- **> Continuous Integration**
- **> Deployment Automation**
- **Monitoring**
- **Trunk-Based Development**
- **Test Automation**
- Test Data Management
- Shift Left on Security
- Loosely Coupled Architecture
- Empowered Teams
- Proactive Notification
- ...

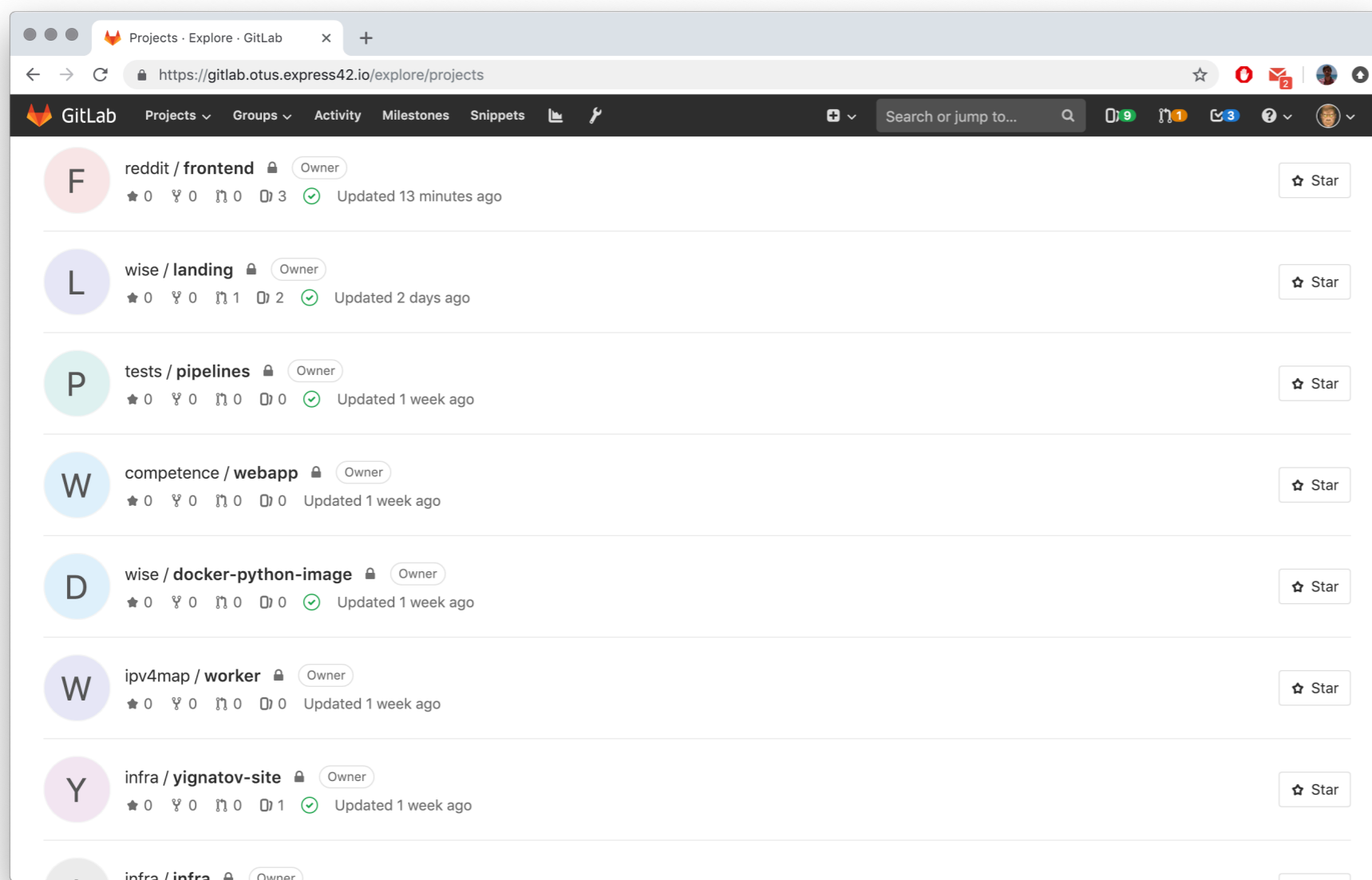
Gitlab CI

- Развивается с 2011 года
- SaaS и приватные инсталляции
- Community Edition
- Enterprise Edition
 - Multiple LDAP / AD server support
 - Remote repository mirroring
 - Support
 - HA
 - ...



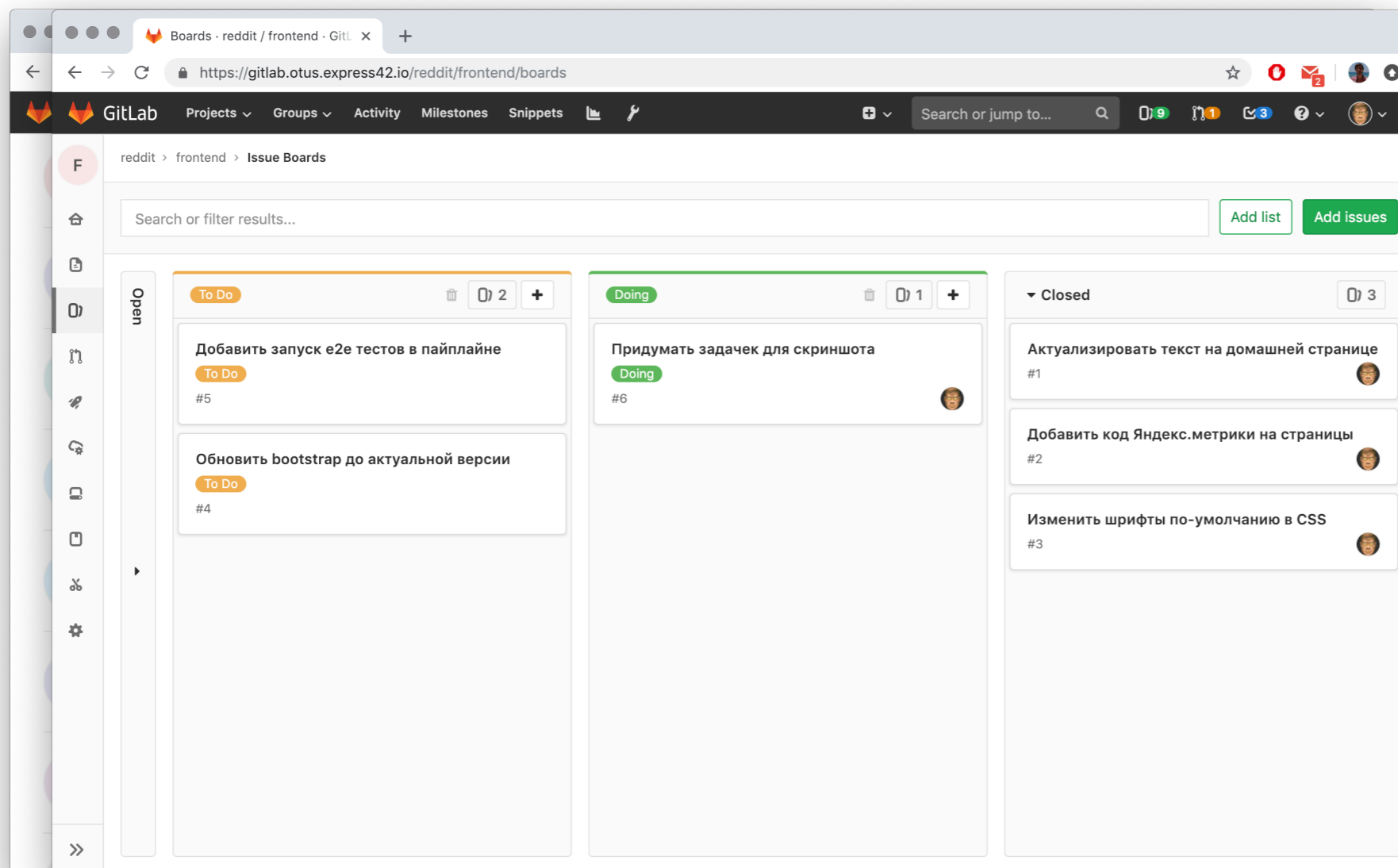
Функционал Gitlab CI

Проекты и группы проектов



Функционал Gitlab CI

Таск-треккер и доски



Функционал Gitlab CI

Репозитории

reddit > frontend > Repository

master frontend / +

History Find file Web IDE

changed text on homepage to look better
Yury Ignatov authored 3 hours ago 2985bf9e

Name	Last commit	Last update
conf/manifests	add pipeline definition and kube manifests	17 hours ago
e2e	initial commit	18 hours ago
src	changed text on homepage to look better	3 hours ago
.editorconfig	initial commit	18 hours ago
.gitignore	initial commit	18 hours ago
.gitlab-ci.yml	change url in pipeline	14 hours ago
Dockerfile	add pipeline definition and kube manifests	17 hours ago
README.md	initial commit	18 hours ago
angular.json	initial commit	18 hours ago
package-lock.json	initial commit	18 hours ago
package.json	initial commit	18 hours ago

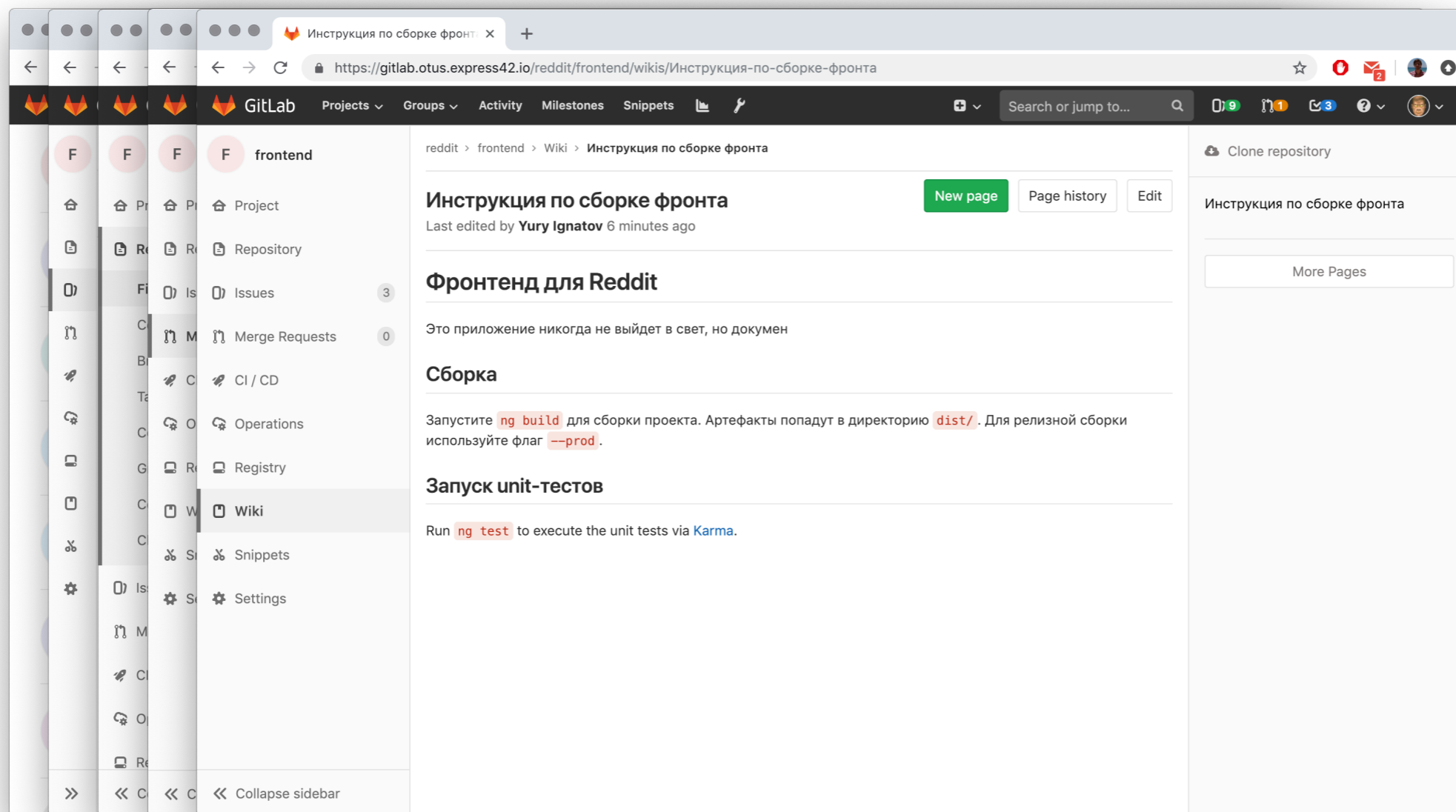
Функционал Gitlab CI

Merge requests

The screenshot displays a GitLab Merge Request (MR) interface. The browser address bar shows the URL: `https://gitlab.otus.express42.io/reddit/frontend/merge_requests/5`. The page title is "Resolve 'Добавить код Яндекс.метрики на страницы'". The MR is marked as "Merged" and was opened 13 hours ago by Yury Ignatov. It closes issue #2 (closed). The MR details include a request to merge branch `2-add-ya-metrika` into `master`. Two CI pipelines are shown: Pipeline #108 passed for commit `8a94676f` on branch `2-add-ya-metrika`, and Pipeline #109 passed for commit `046e83e4` on branch `master`. The MR was merged by Yury Ignatov 13 hours ago, with the changes merged into `master` with commit `046e83e4`. The interface also shows options to "Revert" or "Cherry-pick", and a "Remove Source Branch" button. The right sidebar contains settings for the MR, including "Assignee", "Milestone", "Labels", "Lock merge request", and "Notifications".

Функционал Gitlab CI

wiki



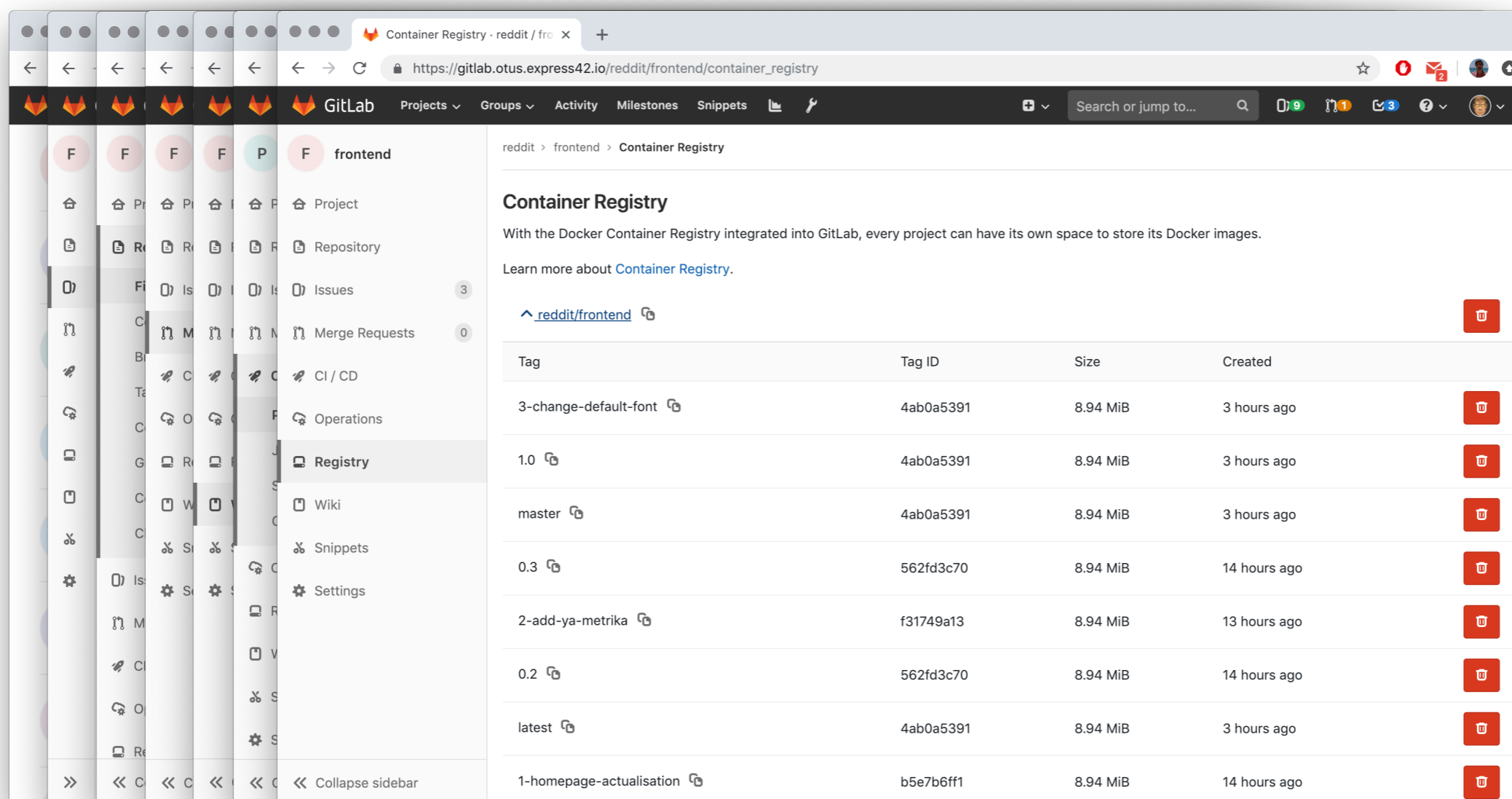
Функционал Gitlab CI

CI/CD Pipelines

The screenshot displays the GitLab web interface for a CI/CD pipeline. The browser address bar shows the URL `https://gitlab.otus.express42.io/tests/pipelines/pipelines/115`. The left sidebar contains navigation options: Pipelines, Jobs, Schedules, Charts, Operations, Registry, Wiki, Snippets, and Settings. The main content area is titled "add more tests" and shows a pipeline with 16 jobs. The pipeline is divided into four stages: Build, Qa, Stage, and Release. The Build stage includes jobs: docker-build, linter, static-analysis, and unittest. The Qa stage includes: auto-tests, database-migr..., deploy-test, integration, and manual-tests. The Stage stage includes: cab, database-migr..., deploy-stage, and loadtest. The Release stage includes: database-migr..., deploy-prod, and notifications. A "Play" button is visible over the integration job in the Qa stage. The top of the interface shows the GitLab logo and navigation tabs: Projects, Groups, Activity, Milestones, and Snippets.

Функционал Gitlab CI

Хранилище артефактов



The screenshot displays the GitLab Container Registry interface for the 'reddit/frontend' project. The left sidebar shows the project navigation menu with 'Registry' selected. The main content area shows the 'Container Registry' page with a table of Docker images.

Container Registry

With the Docker Container Registry integrated into GitLab, every project can have its own space to store its Docker images.

Learn more about [Container Registry](#).

[^ reddit/frontend](#)

Tag	Tag ID	Size	Created
3-change-default-font	4ab0a5391	8.94 MiB	3 hours ago
1.0	4ab0a5391	8.94 MiB	3 hours ago
master	4ab0a5391	8.94 MiB	3 hours ago
0.3	562fd3c70	8.94 MiB	14 hours ago
2-add-ya-metrika	f31749a13	8.94 MiB	13 hours ago
0.2	562fd3c70	8.94 MiB	14 hours ago
latest	4ab0a5391	8.94 MiB	3 hours ago
1-homepage-actualisation	b5e7b6ff1	8.94 MiB	14 hours ago

Функционал Gitlab CI

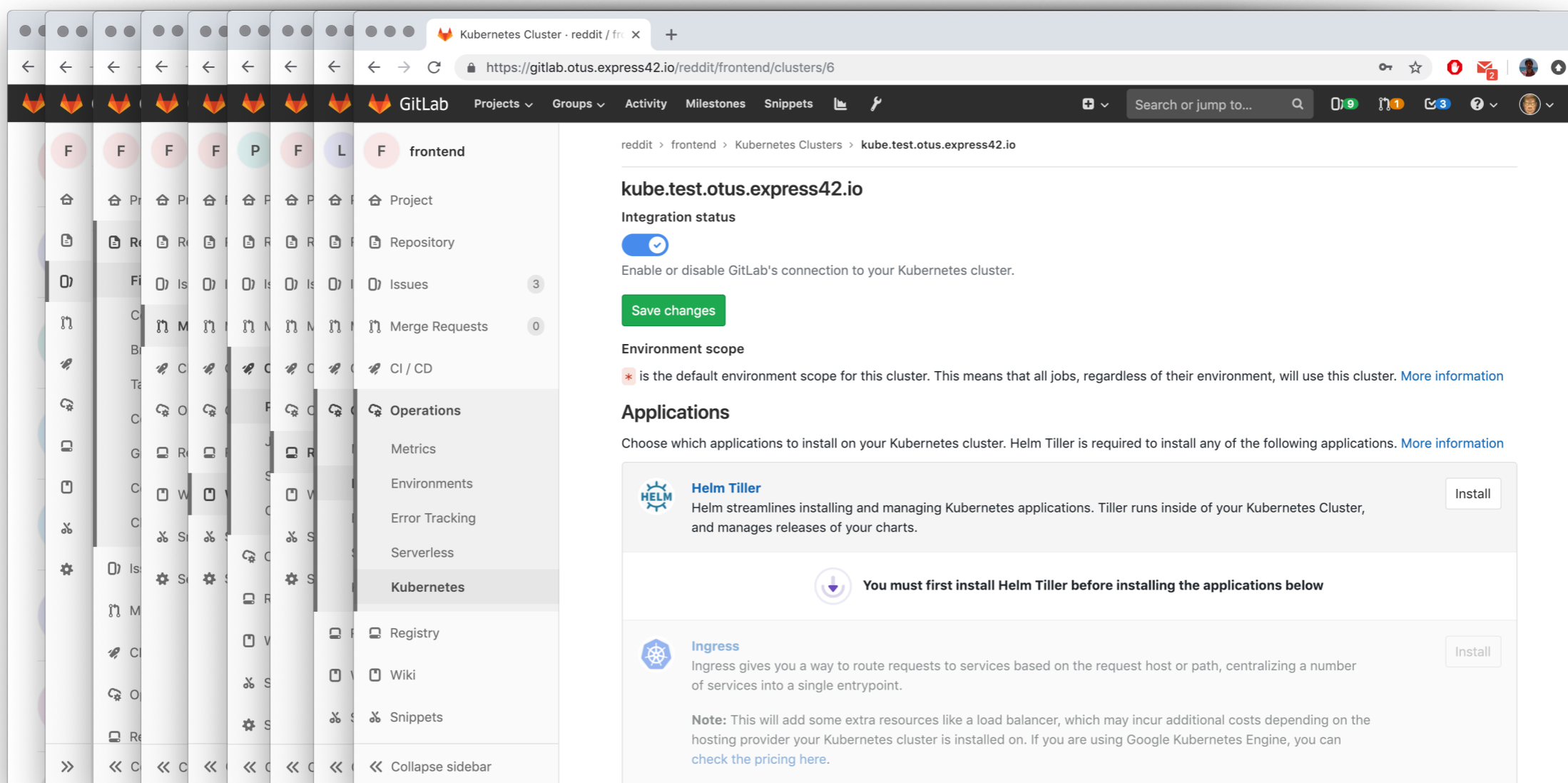
Статус окружений

The screenshot displays the GitLab web interface for the 'Environments' section of a project. The left sidebar shows the navigation menu with 'Environments' selected. The main content area shows a list of environments under the 'production' group. The table below summarizes the visible data:

Environment	Deployment	Job	Commit	Updated
production	#44 by [user]	deploy_live #307	9df1e203 just disable tests for no...	2 days ago
review 3				
staging	#34 by [user]	deploy_review #209	336ff015 generate domain in .git...	3 months ago
test-01	#37 by [user]	deploy_review #225	767ed2aa Update home.html	3 months ago
test-02	#45 by [user]	deploy_review #426	9df1e203 just disable tests for n...	3 minutes ago

Функционал Gitlab CI

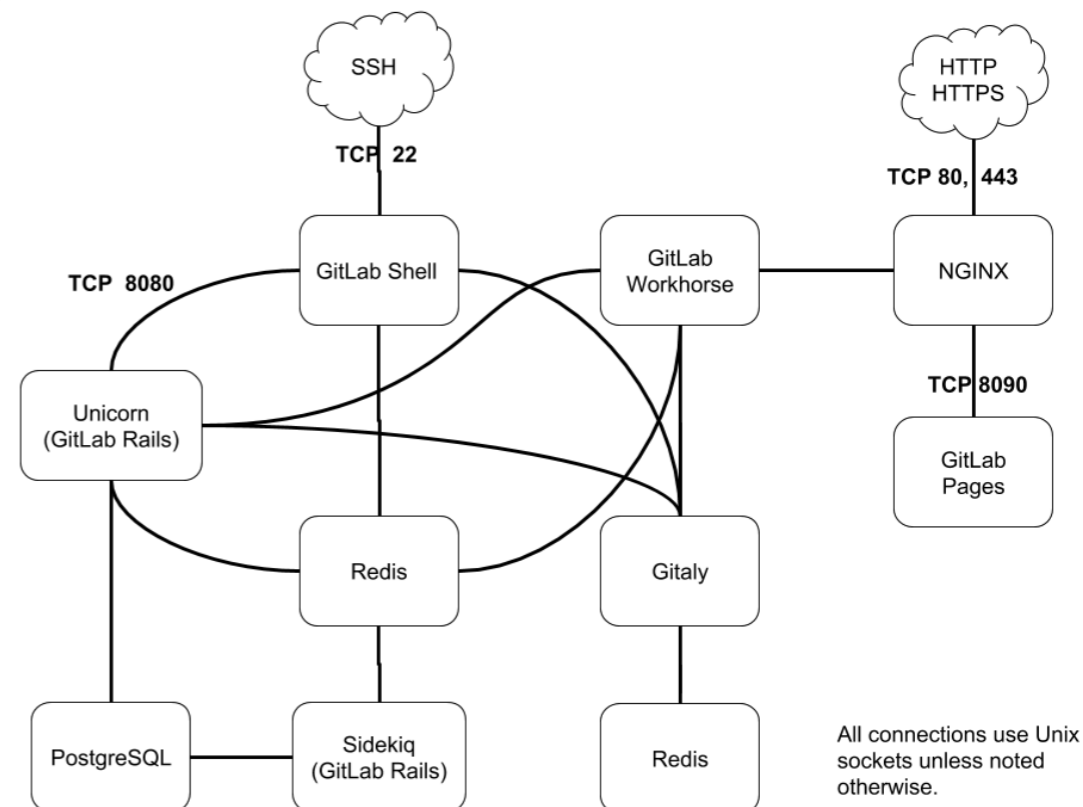
Интеграция с kubernetes, prometheus, sentry и многое другое



Устройство Gitlab CI

- Ruby on Rails, Go
- Postgres, Redis
- Sidekiq, Nginx, Prometheus

 GitLab Application Architecture

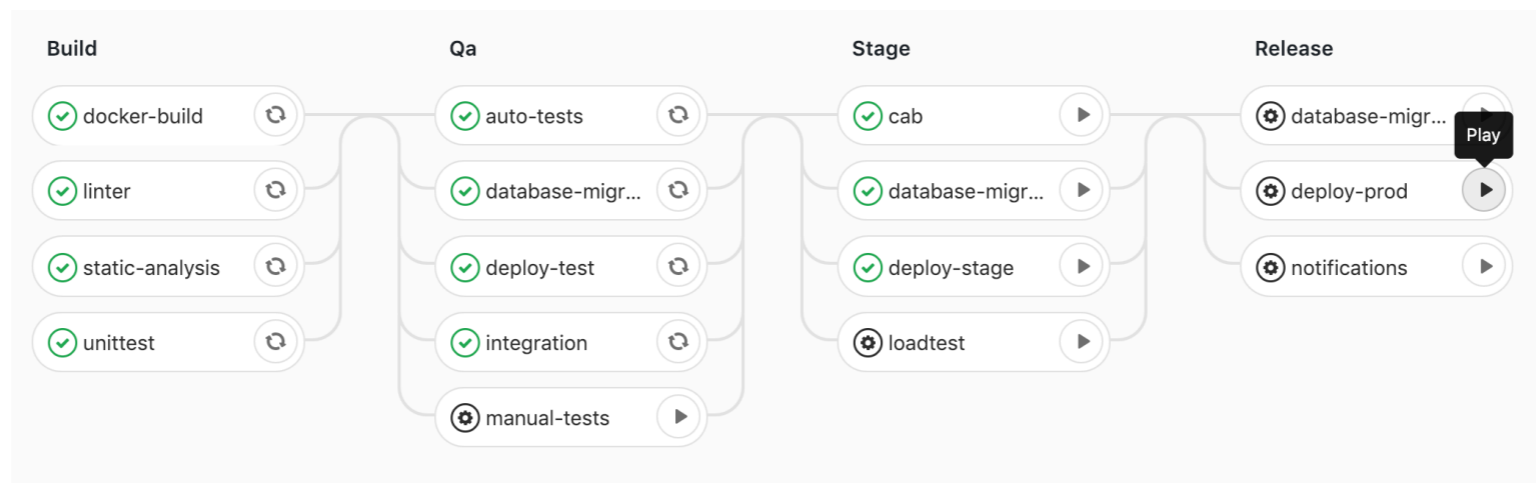


<https://docs.gitlab.com/ce/development/architecture.html>

<https://about.gitlab.com/handbook/engineering/infrastructure/production-architecture/>

CI/CD Pipelines

- Пайплайн определяется на репозиторий файлом `.gitlab-ci.yml`
- Состоит из задач (jobs)
- Задачи поделены на этапы (stages)
- Задачи выполняются на Gitlab Runners
- Запускается по коммиту, по кнопке, по расписанию, по триггеру



Gitlab Runner

- Процесс, выполняющий Jobs
- Состояния runner (shared, group, specific)
- Executors
- Теги

Admin Area > Runners

A 'Runner' is a process which runs a job. You can set up as many Runners as you need. Runners can be placed on separate users, servers, even on your local machine.

Each Runner can be in one of the following states:

- **shared** - Runner runs jobs from all unassigned projects
- **group** - Runner runs jobs from all unassigned projects in its group
- **specific** - Runner runs jobs from assigned projects
- **locked** - Runner cannot be assigned to other projects
- **paused** - Runner will not receive any new jobs

Set up a shared Runner manually

1. [Install GitLab Runner](#)
2. Specify the following URL during the Runner setup:
`https://gitlab/`
3. Use the following registration token during setup:
`sqQ3scTCP...`
4. Start the Runner!

Reset runners registration token

Runners currently online: 1

Type	Runner token	Description	Version	IP Address	Projects	Jobs	Tags	Last contact
shared	1e62610c	main	11.3.1	10.251.1.1	n/a	347		11 minutes ago

Job

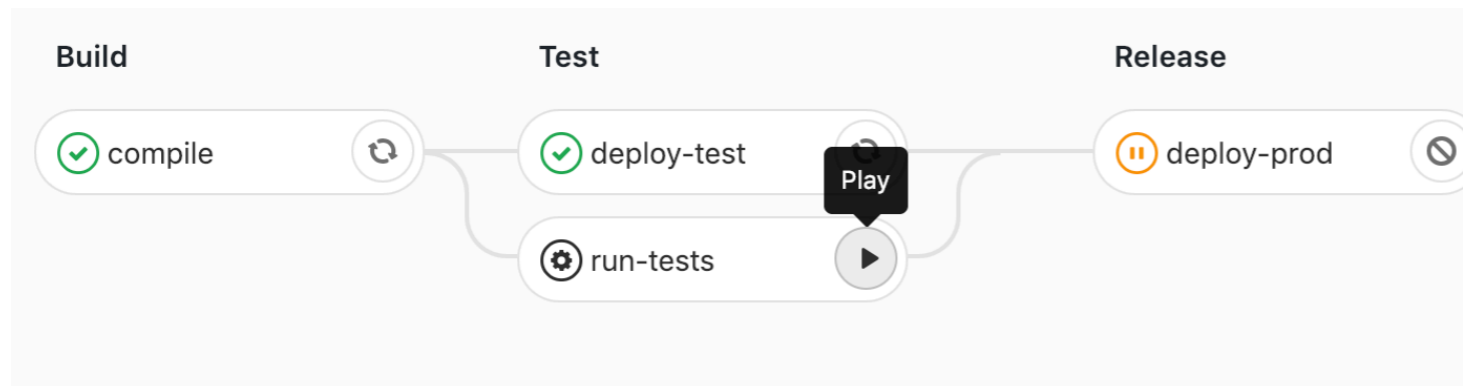
- Определяется кодом
- Относится к конкретному этапу (stage) пайплайна
- Каждый Job выполняется независимо
- Могут использовать переменные
- Директивы `only/except, when, allow_failure, retry`

Описание пайплайна .gitlab-ci.yml

Pipeline состоит из нескольких этапов

Jobs в одном этапе выполняются параллельно

Stages выполняются последовательно



```
1 stages:
2   - build
3   - test
4   - release
5
6 compile:
7   stage: build
8   script:
9     - ./compile-code.sh
10
11 deploy-test:
12   stage: test
13   script:
14     - ./deploy-test.sh
15
16 run-tests:
17   stage: test
18   when: manual
19   script:
20     - ./run-tests.sh
21
22 deploy-prod:
23   stage: release
24   script:
25     - ./deploy-prod.sh
26
27
```

Variables

- Встроенные
- Пользовательские
- Секретные

<https://docs.gitlab.com/ce/ci/variables/>

Variable Name	Version	Description
CI_PIPELINE_TRIGGERED	all	The flag to indicate that job was triggered by a pipeline.
CI_PIPELINE_URL	11.1	0.5 Pipeline details URL
CI_PROJECT_DIR	all	all The full path where the repository is checked out. If the <code>builds_dir</code> parameter is relative to the value of <code>builds_dir</code> , see Advanced Runner .
CI_PROJECT_ID	all	all The unique id of the current project.
CI_PROJECT_NAME	8.10	0.5 The project name that is currently being built (actually it is project folder name).
CI_PROJECT_NAMESPACE	8.10	0.5 The project namespace (user/group) that is currently being built.
CI_PROJECT_PATH	8.10	0.5 The namespace with project name.
CI_PROJECT_PATH_SLUG	9.3	all <code>CI_PROJECT_PATH</code> lowercased and <code>CI_PROJECT_PATH_SLUG</code> lowercase except <code>0-9</code> and <code>a-z</code> replaced by <code>-</code> and domain names.
CI_PROJECT_URL	8.10	0.5 The HTTP(S) address to access the project.
CI_PROJECT_VISIBILITY	10.3	all The project visibility (internal/public/private).
CI_REGISTRY	8.10	0.5 If the Container Registry is enabled, the address of GitLab's Container Registry.

Variables

- Встроенные
- Пользовательские
- Секретные

The screenshot displays the GitLab CI/CD Settings page for a project named 'frontend'. The page is titled 'Environment variables' and includes a 'Collapse' button. Below the title, there is a descriptive paragraph: 'Environment variables are applied to environments via the runner. They can be protected by only exposing them to protected branches or tags. You can use environment variables for passwords, secret keys, or whatever you want. You may also add variables that are made available to the running application by prepending the variable key with `K8S_SECRET_`. [More information](#)'.

The main content area contains a table of environment variables:

Variable Name	Value	Protected
AUTOTESTS_USERNAME_TEST	testuser	Protected (Off)
AUTOTESTS_PASSWORD_TEST	testtesttest	Protected (Off)
AUTOTESTS_USERNAME_PROD	*****	Protected (On)
AUTOTESTS_PASSWORD_PROD	*****	Protected (On)
Input variable key	Input variable value	Protected (Off)

At the bottom of the table, there are two buttons: 'Save variables' (green) and 'Hide values' (blue).

Below the main content area, there is a table with the following data:

Variable Name	Version	Visibility	Description
CI_PROJECT_URL	8.10	0.5	The HTTP(S) address to ac
CI_PROJECT_VISIBILITY	10.3	all	The project visibility (inter
CI_REGISTRY	8.10	0.5	If the Container Registry is address of GitLab's Contai

Variables

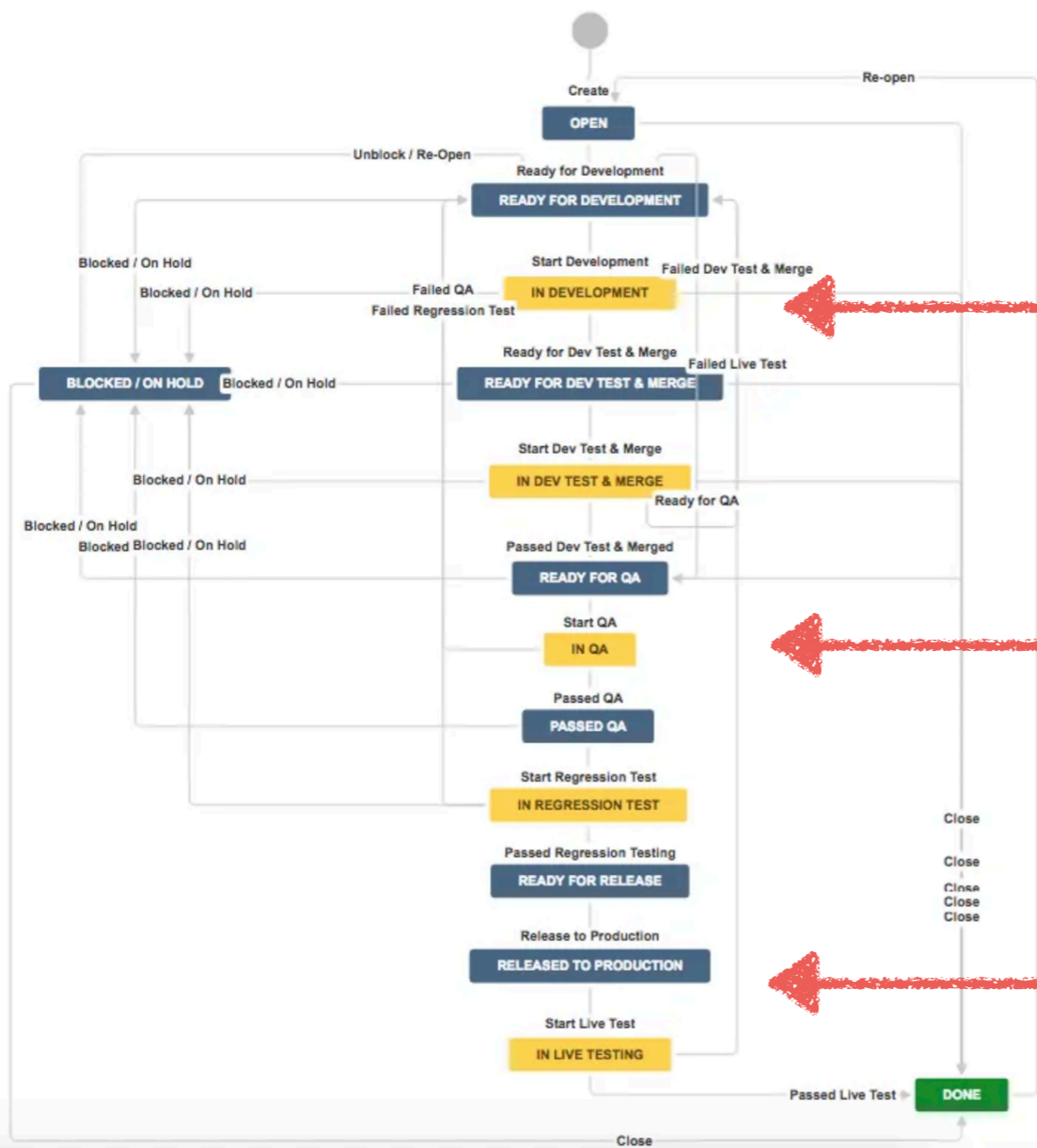
```
variables:  
  REGISTRY_URL: registry.otus.express42.io  
  CONTAINER_IMAGE: $REGISTRY_URL/$CI_PROJECT_NAMESPACE/$CI_PROJECT_NAME  
  
before_script:  
  - docker info  
  - docker login -u gitlab-ci-token -p $CI_JOB_TOKEN $REGISTRY_URL  
  
build:  
  stage: build  
  script:  
    - docker build --pull -t $CONTAINER_IMAGE .  
    - docker tag $CONTAINER_IMAGE:latest $CONTAINER_IMAGE:${CI_COMMIT_SHA}  
    - docker push $CONTAINER_IMAGE:${CI_COMMIT_SHA}  
  
test:  
  stage: test  
  script:  
    - docker pull $CONTAINER_IMAGE:${CI_COMMIT_SHA}  
    - docker run $CONTAINER_IMAGE:${CI_COMMIT_SHA} nginx -t  
    - docker run $CONTAINER_IMAGE:${CI_COMMIT_SHA} ./tests.sh
```

Непрерывная поставка ПО

Процесс разработки и эксплуатации ПО, при котором каждое изменение может быть выкачено в боевое окружение

Какой путь проходит изменение?

Путь изменения: воркфлоу

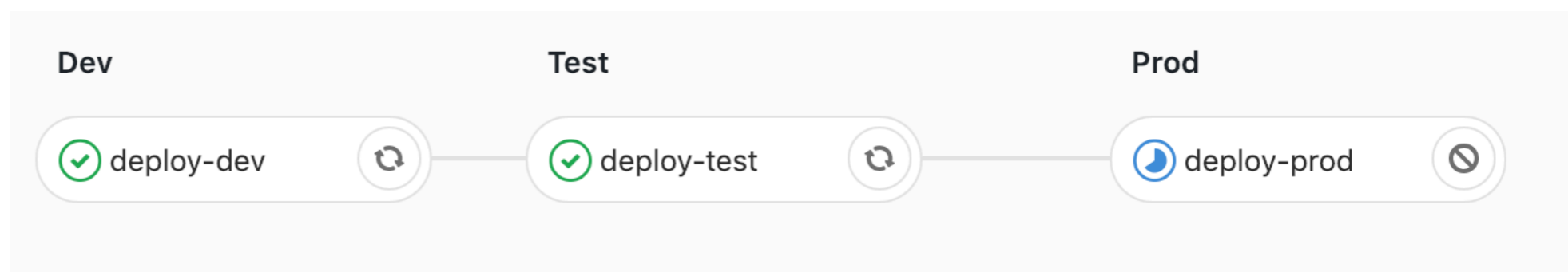


Окружение разработки

Окружение тестирования

Продакшн

Путь изменения: деплой



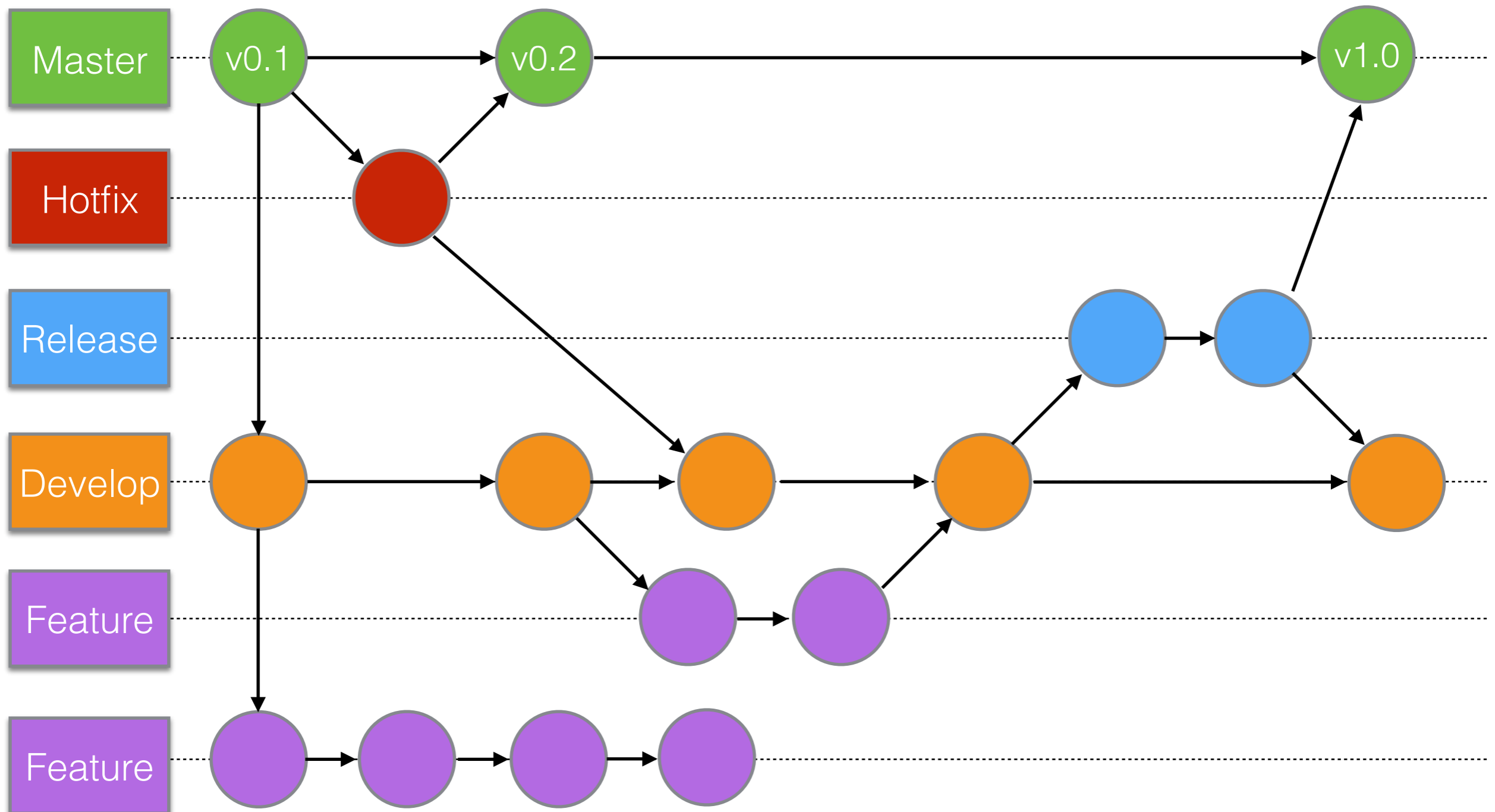
Изменение



```
landing/migrations/0003_auto_20181007_2026.py 0 → 100644 View file @ f41d8db3
```

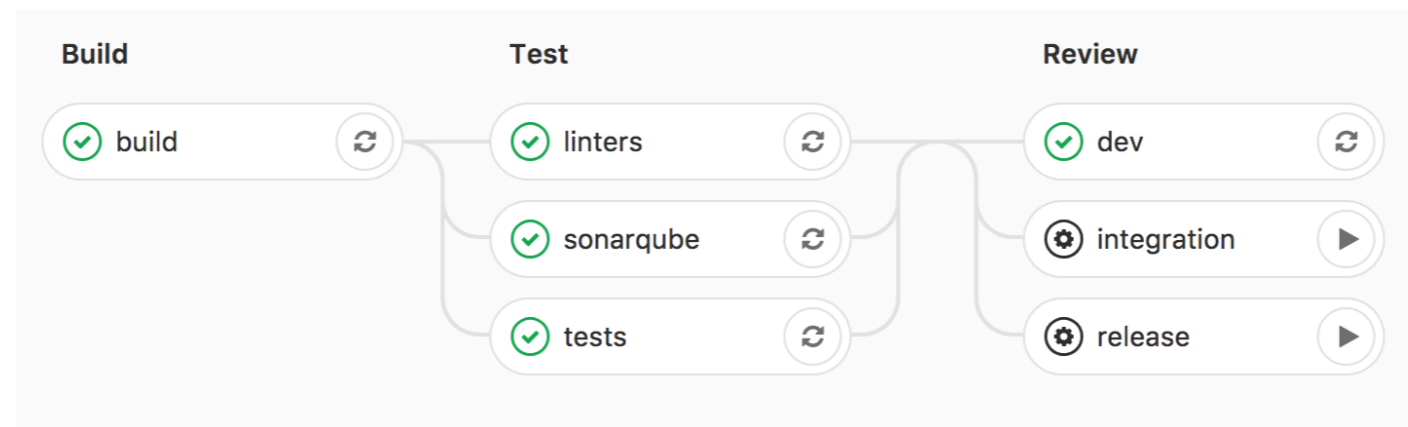
```
8 + class Migration(migrations.Migration):
9 +
10 +     dependencies = [
11 +         ('landing', '0002_auto_20181007_2013'),
12 +     ]
13 +
14 +     operations = [
15 +         migrations.CreateModel(
16 +             name='Concept',
17 +             fields=[
18 +                 ('id', models.AutoField(auto_created=True, primary_key=True, serialize=False,
19 + verbose_name='ID')),
20 +                 ('symbol', models.CharField(max_length=64)),
21 +                 ('display_name', models.CharField(max_length=64)),
22 +                 ('created_date', models.DateTimeField(default=django.utils.timezone.now)),
```

Путь изменения: в git flow

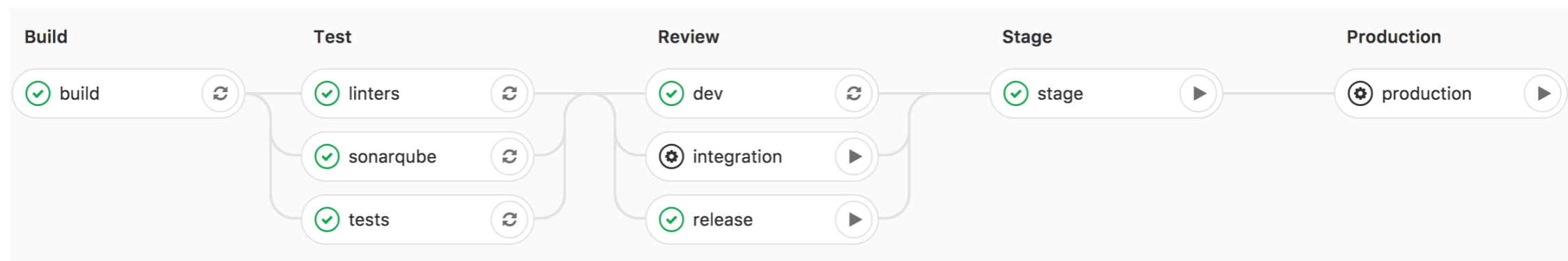


Не каждое изменение нужно выкатывать на все окружения

```
git commit -a -m 'Add more comments for code from make it more readable'  
git push
```

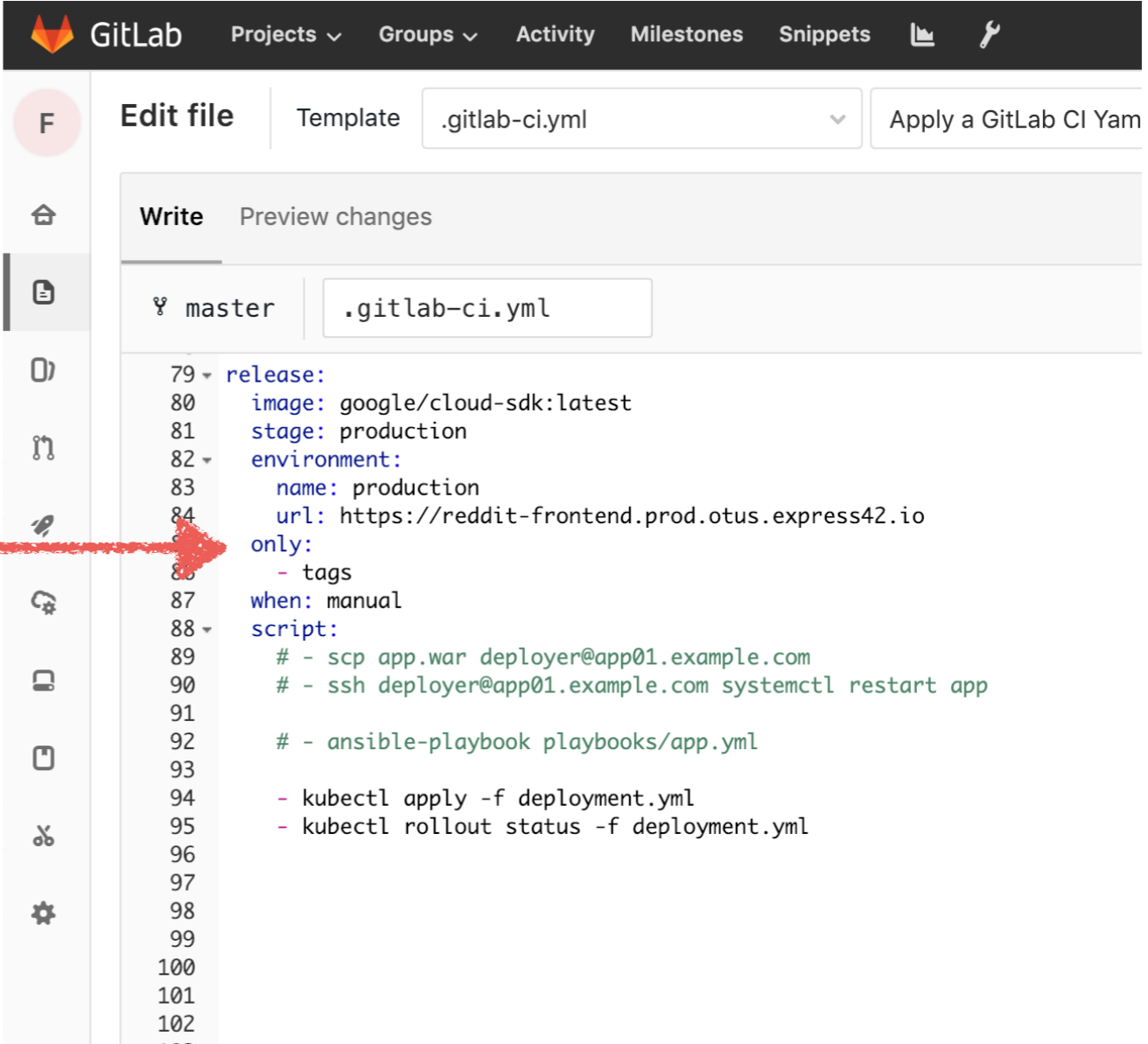


```
git commit -a -m 'New login page design'  
git tag 2.1.0  
git push --tags
```



Не каждое изменение нужно выкатывать на все окружения

Например, только изменения, помеченные тегом



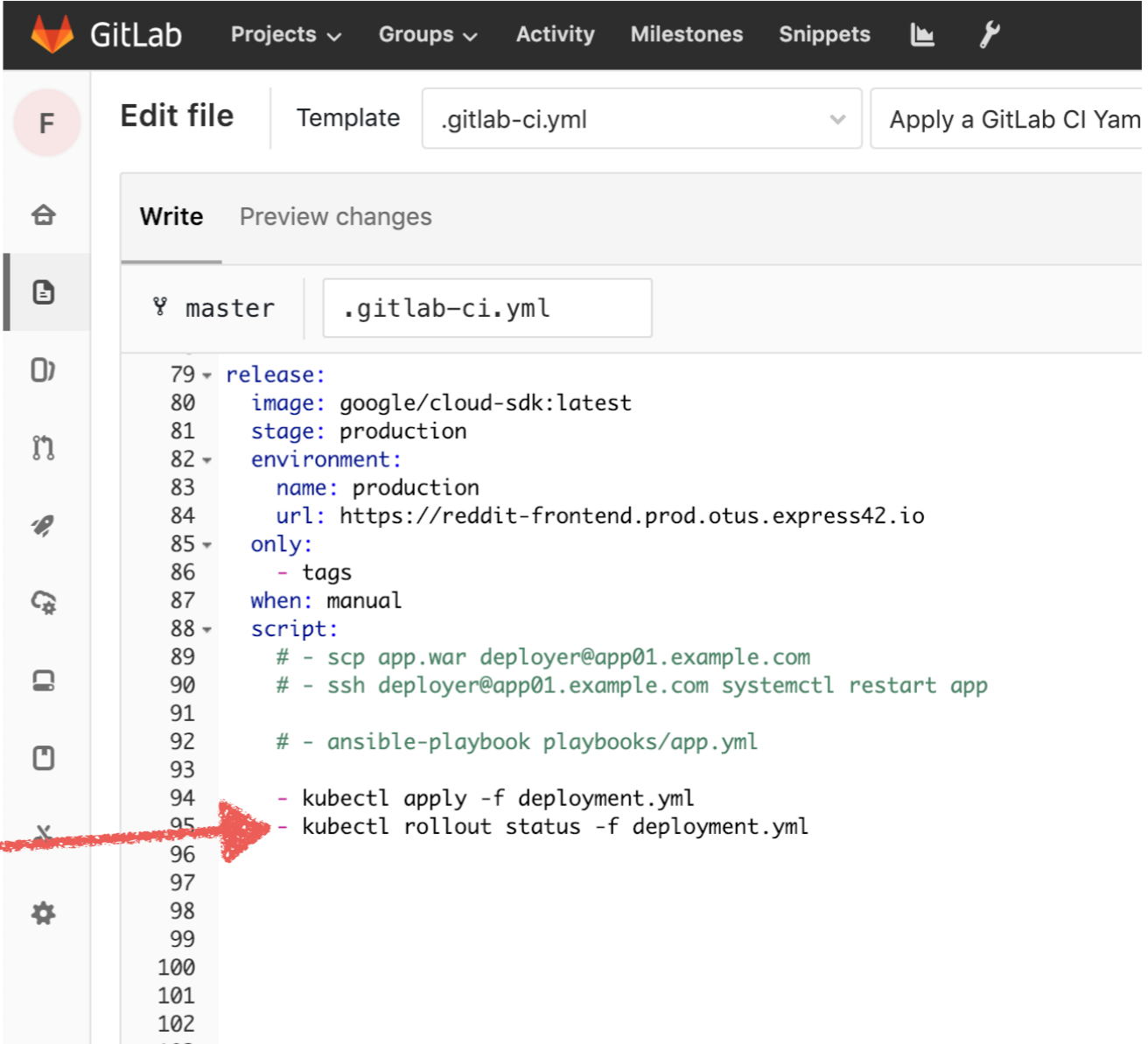
The screenshot shows the GitLab web interface for editing a file named `.gitlab-ci.yml`. The file is being edited on the `master` branch. The configuration defines a `release` job with the following settings:

```
79 release:
80   image: google/cloud-sdk:latest
81   stage: production
82   environment:
83     name: production
84     url: https://reddit-frontend.prod.otus.express42.io
85   only:
86     - tags
87   when: manual
88   script:
89     # - scp app.war deployer@app01.example.com
90     # - ssh deployer@app01.example.com systemctl restart app
91
92     # - ansible-playbook playbooks/app.yml
93
94     - kubectl apply -f deployment.yml
95     - kubectl rollout status -f deployment.yml
96
97
98
99
100
101
102
103
```

A red arrow points to the `only:` section, specifically to the `- tags` condition, which indicates that this job only runs when a new tag is pushed to the repository.

Не каждое изменение нужно выкатывать на все окружения

Для частых релизов нужно обновление без простоев (Zero-downtime deploy)



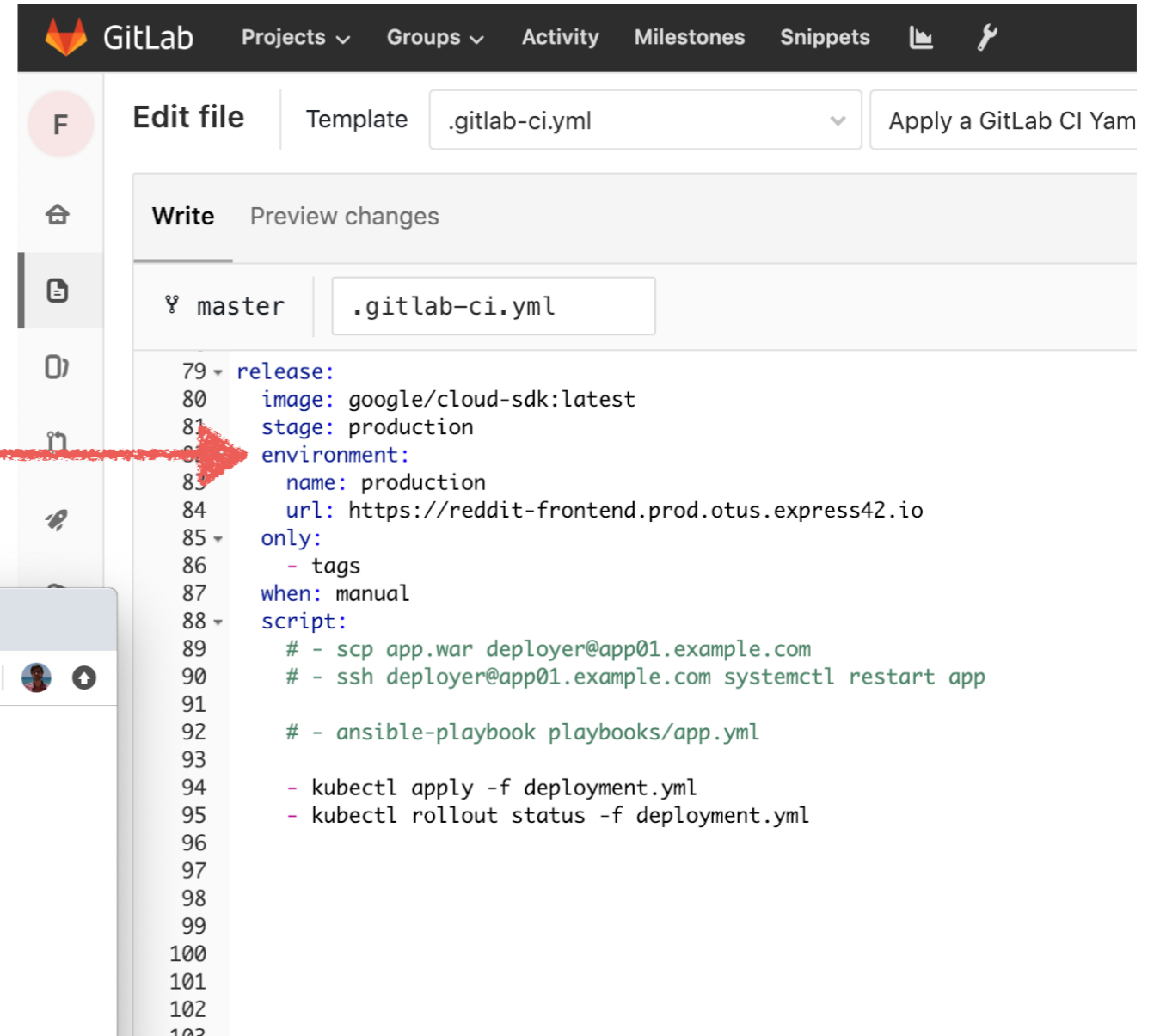
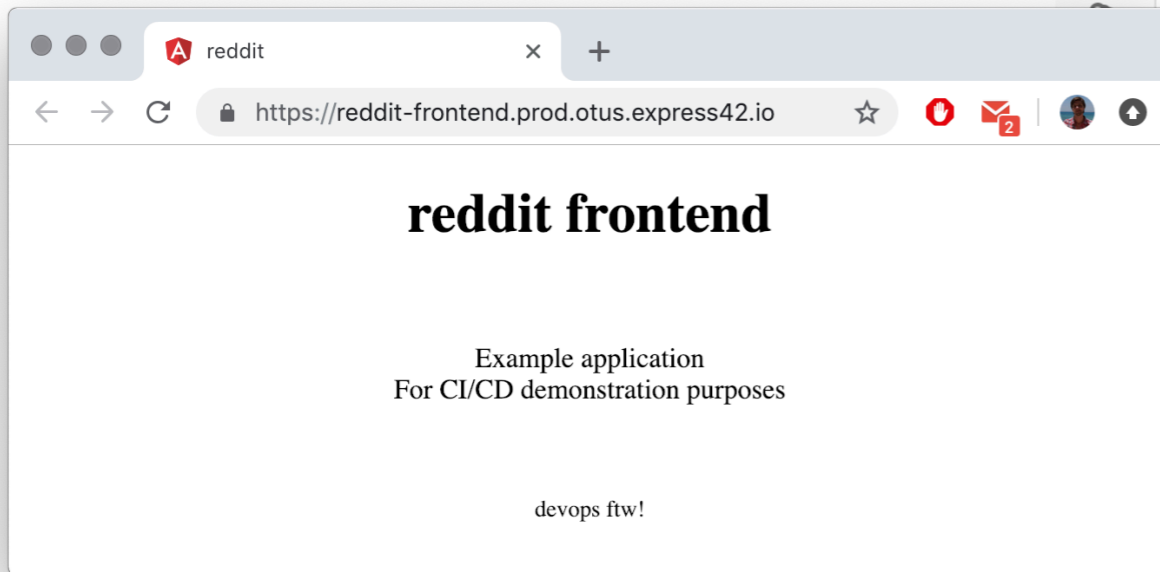
The screenshot shows the GitLab CI configuration editor for a file named `.gitlab-ci.yml`. The editor is in the 'Write' mode. The configuration includes a job named `release` with the following settings:

```
79 release:
80   image: google/cloud-sdk:latest
81   stage: production
82   environment:
83     name: production
84     url: https://reddit-frontend.prod.otus.express42.io
85   only:
86     - tags
87   when: manual
88   script:
89     # - scp app.war deployer@app01.example.com
90     # - ssh deployer@app01.example.com systemctl restart app
91
92     # - ansible-playbook playbooks/app.yml
93
94     - kubectl apply -f deployment.yml
95     - kubectl rollout status -f deployment.yml
96
97
98
99
100
101
102
103
```

A red arrow points to the `script` section, highlighting the deployment commands.

Не каждое изменение нужно выкатывать на все окружения

Кстати, про окружения



```
79 release:
80   image: google/cloud-sdk:latest
81   stage: production
82   environment:
83     name: production
84     url: https://reddit-frontend.prod.otus.express42.io
85   only:
86     - tags
87   when: manual
88   script:
89     # - scp app.war deployer@app01.example.com
90     # - ssh deployer@app01.example.com systemctl restart app
91
92     # - ansible-playbook playbooks/app.yml
93
94     - kubectl apply -f deployment.yml
95     - kubectl rollout status -f deployment.yml
96
97
98
99
100
101
102
103
```

Окружения в Gitlab CI

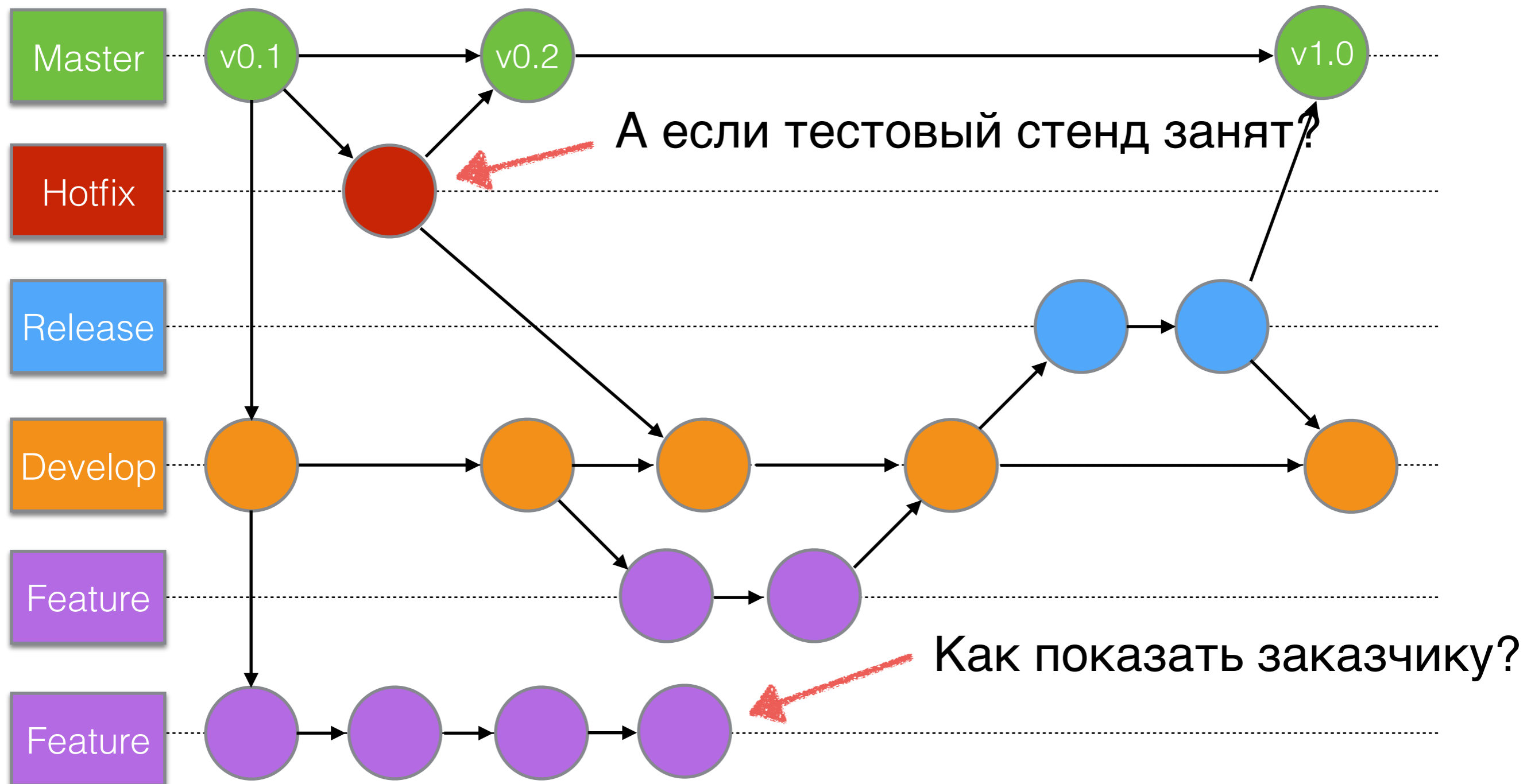
production

ID	Commit	Job	Created
#26	1.1 -> f321376f change title in base html	release (#469) by [user]	5 minutes ago
#24	1.0 -> 5c3d00b1 Merge branch '3-change-default-font'...	release (#390) by [user]	3 hours ago
#15	0.3 -> 031c5c11 change url in pipeline	release (#357) by [user]	16 hours ago
#12	0.1 -> 880042a2 Update app.component.ts	release (#348) by [user]	17 hours ago

Динамические окружения

- Изолированные окружения под задачу / фичу
- Создаются и удаляются “по кнопке”
- Должны быть короткоживущими, как и фича-ветки
- Детали реализации

Путь изменения: в git flow



Динамические окружения

deploy-feature:

stage: review

environment:

name: review/\$CI_BUILD_REF_NAME

url: **https://**reddit-frontend-\$CI_BUILD_REF_SLUG.test.otus.express42.io

on_stop: shutdown

script:

- ./create-environment-if-not-exists.sh
- ./deploy-app.sh

shutdown:

stage: review

variables:

GIT_STRATEGY: none

when: manual

environment:

name: review/\$CI_BUILD_REF_NAME

action: stop

script:

- ./destroy-environment.sh

Динамические окружения

Resolve "Поменять фон страницы на темный"

Closes #7

Edited 6 minutes ago by Yury Ignatov

The screenshot displays a GitHub Actions workflow for a pull request. The workflow includes a 'Request to merge' step, a 'Pipeline #126 passed' step, and a 'Merge' step. A red arrow points from the 'Merge' step to a browser window. The browser window shows a dark-themed application with the text 'reddit frontend', 'Example application For CI/CD demonstration purposes', and 'devops ftw!'.

Request to merge 7-change-background... into master

Open in Web IDE Check out branch

Pipeline #126 passed for 6c9a2074 on 7-change-background...

Deployed to review/7-change-backgr... 6 minutes ago

shutdown

deploy

Merge Remove source branch Modify commit message

Closes #7

https://reddit-frontend-7-change-background-to-dark.test.otus.express42.io

reddit frontend

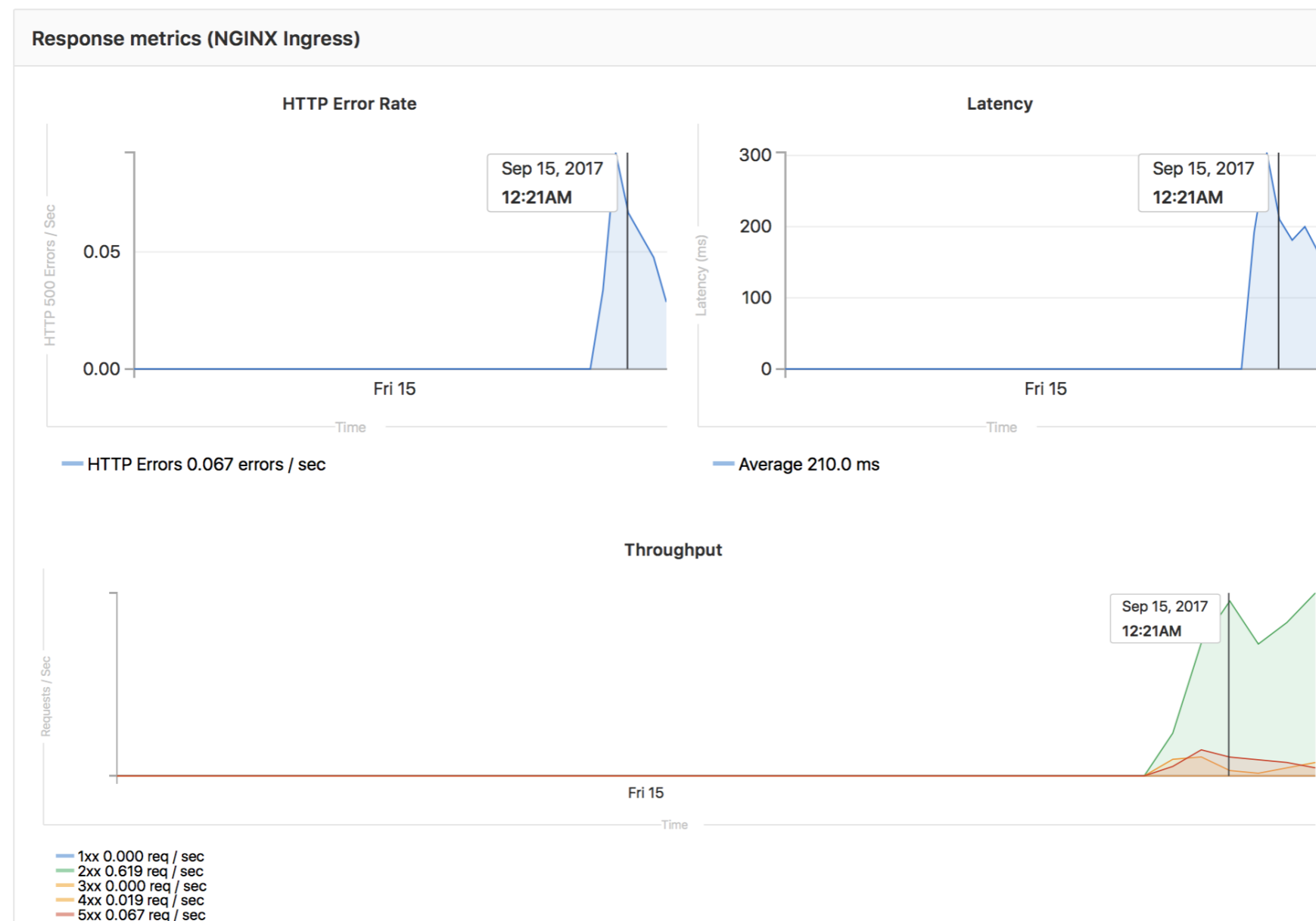
Example application
For CI/CD demonstration purposes

devops ftw!

Выглядит как магия

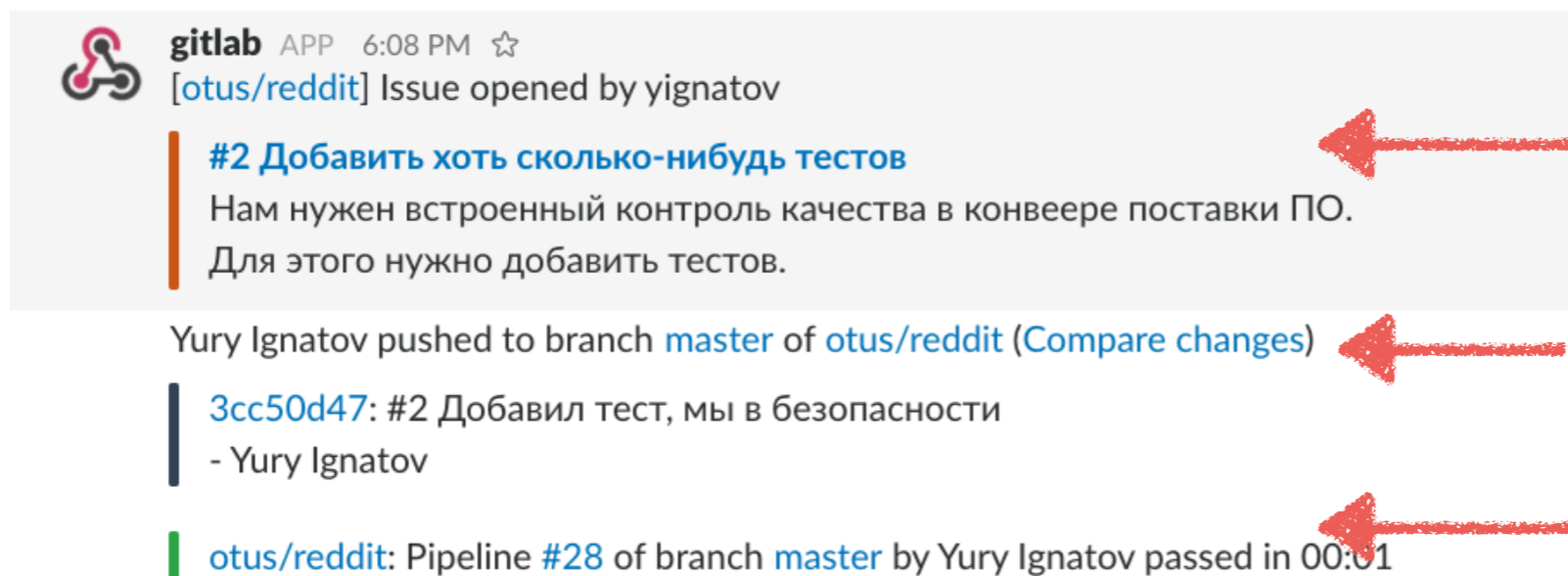
Мониторинг окружений

Environment: **production**



Интеграции

- Хорошие интеграции повышают прозрачность
- Много всего из коробки
- Легко интегрируется по HTTP API
- Подходит для ChatOps



The screenshot shows a chat window from the GitLab application. At the top, it says 'gitlab APP 6:08 PM ☆ [otus/reddit] Issue opened by yignatov'. Below this, there are three distinct messages, each with a colored vertical bar on the left and a red arrow pointing to the right:

- #2 Добавить хоть сколько-нибудь тестов** (blue bar): Нам нужен встроенный контроль качества в конвейере поставки ПО. Для этого нужно добавить тестов.
- 3cc50d47: #2 Добавил тест, мы в безопасности** (purple bar): - Yury Ignatov
- otus/reddit: Pipeline #28 of branch master** (green bar): by Yury Ignatov passed in 00:01

создали задачу

выполнили работу

получили фидбек

Спасибо!

- Ждем ваших вопросов
- Дело