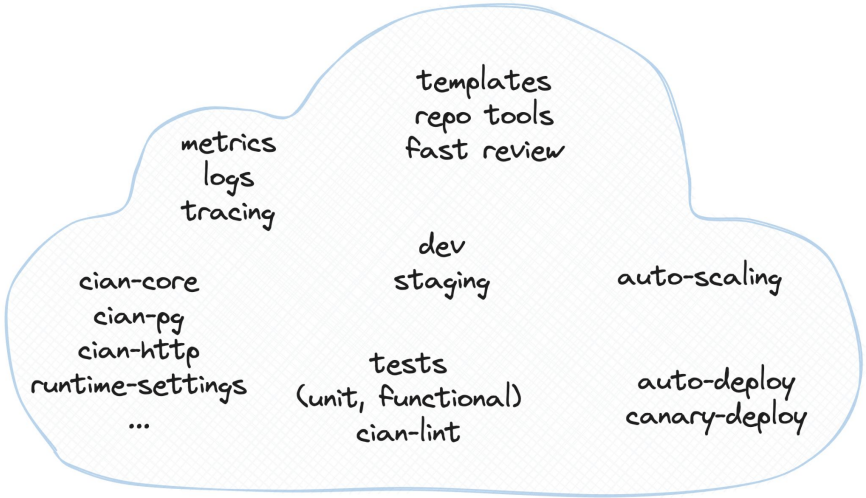
Эволюция MLOps в Циан

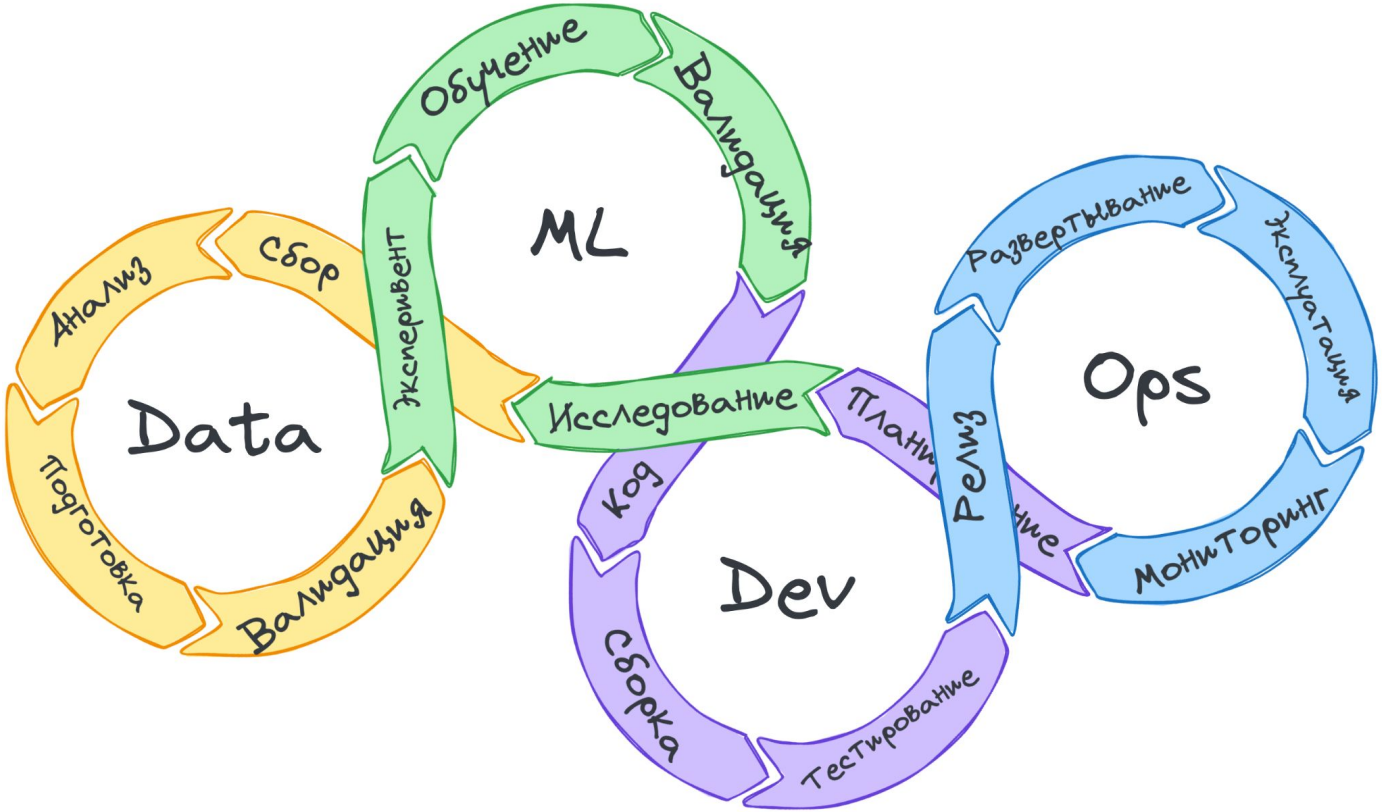
Роман Песков

старший инженер-разработчик
ML-платформа, Циан

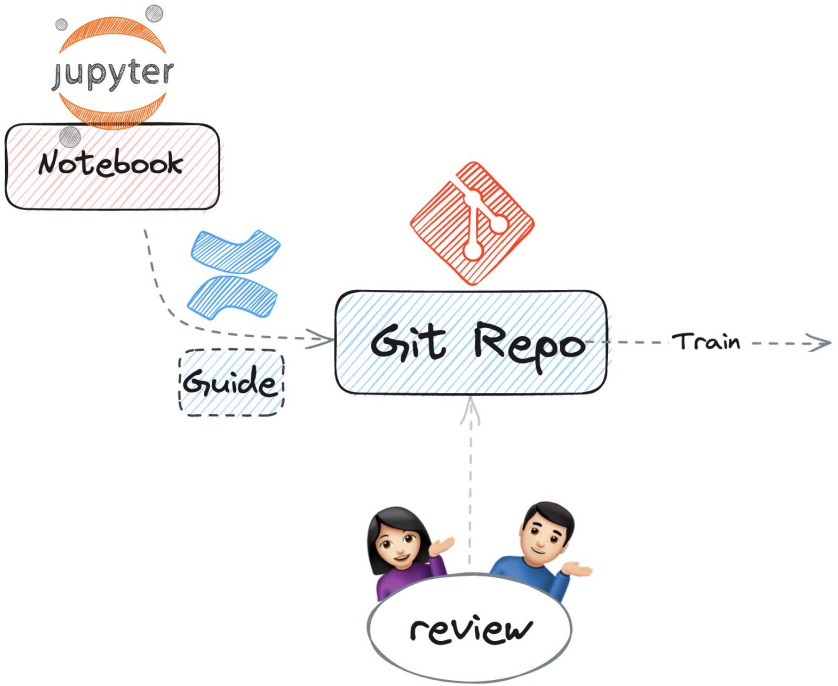
PiterPy / 2023



- шаблоны микросервиса, библиотеки;
- готовые либы для работы с БД, кешем, метриками, логами и т.п.;
- удобное написание функциональных тестов;
- линтеры, форматеры, swagger-check;
- codegen;
- dev/staging среды;
- автодеплой, canary-деплой;
- автоскейлинг;
- ...

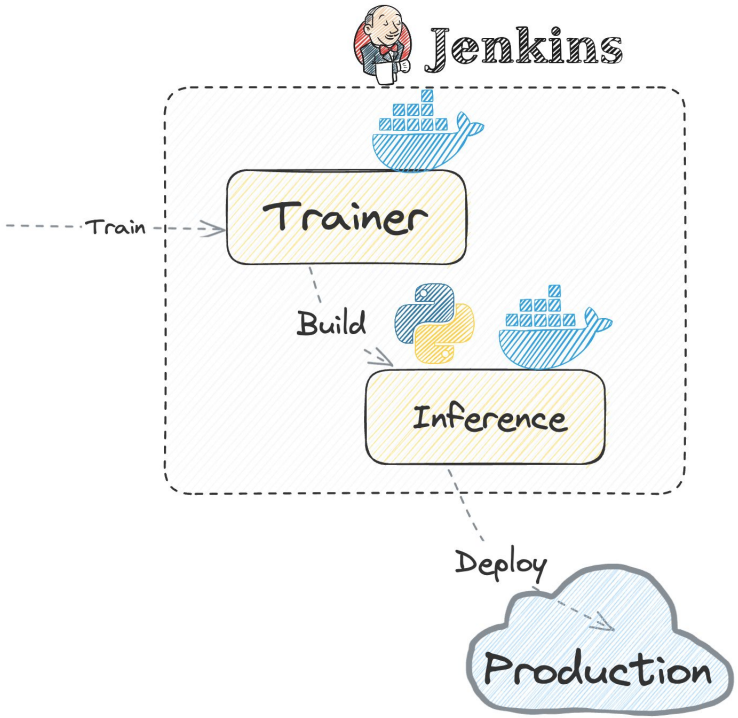

Data
Анализ
Сбор
Подготовка
Валидация
ML
Обучение
Валидация
Эксперимент
Исследование
Dev
Код
Сборка
Тестирование
Ops
Развертывание
Эксплуатация
Мониторинг

Исследование и эксперименты



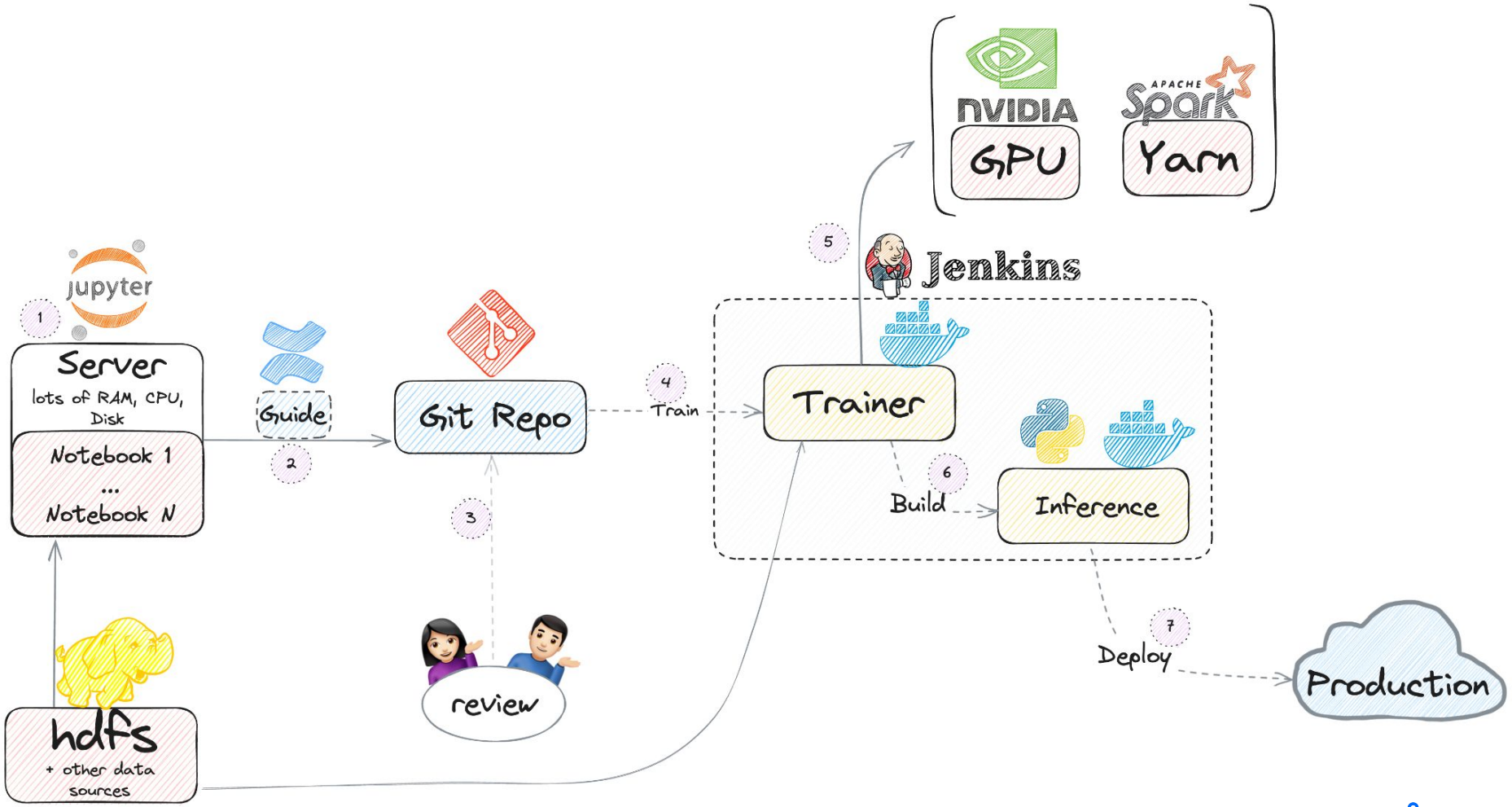
- Исследование и написание кода в личном Jupyter-ноутбуке на сервере с большим количеством ресурсов.
- Оформление Jupyter-ноутбука согласно документации, сохранение в git-репозиторий.
- Ревью «по запросу».

Обучение и Production-этап



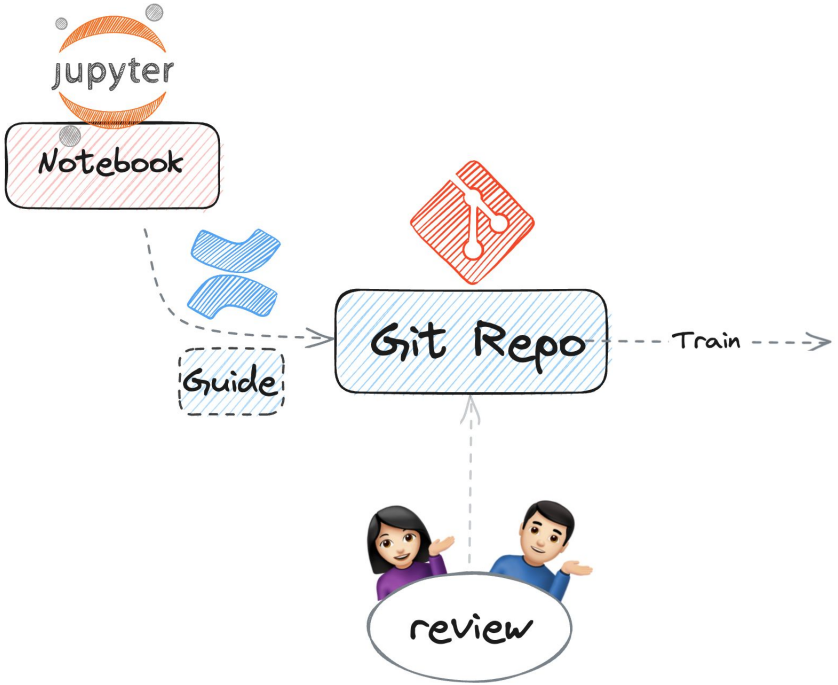
- Отдельные джобы обучения под каждую модель.
- Отдельные джобы для сборки образа и деплоя модели в production под каждую модель.
- Ручной запуск.

циан


NVIDIA
GPU
APACHE
Spark
Yarn
5
Jenkins
jupyter
1
Server
lots of RAM, CPU, Disk
Notebook 1
...
Notebook N
Guide
2
Git Repo
3
4
Train
Trainer
6
Build
Inference
7
Deploy
Production
review
hdfs
+ other data sources

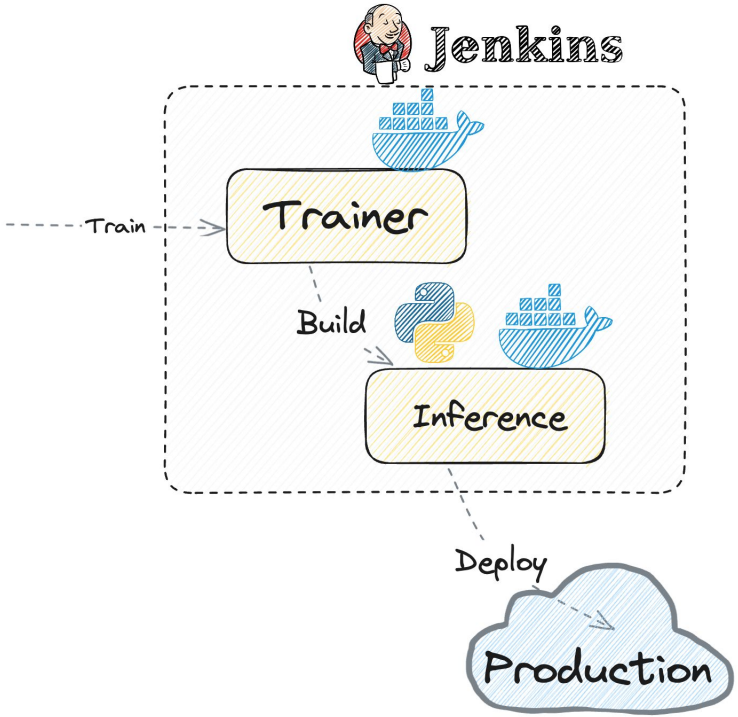
циан

Исследование и эксперименты

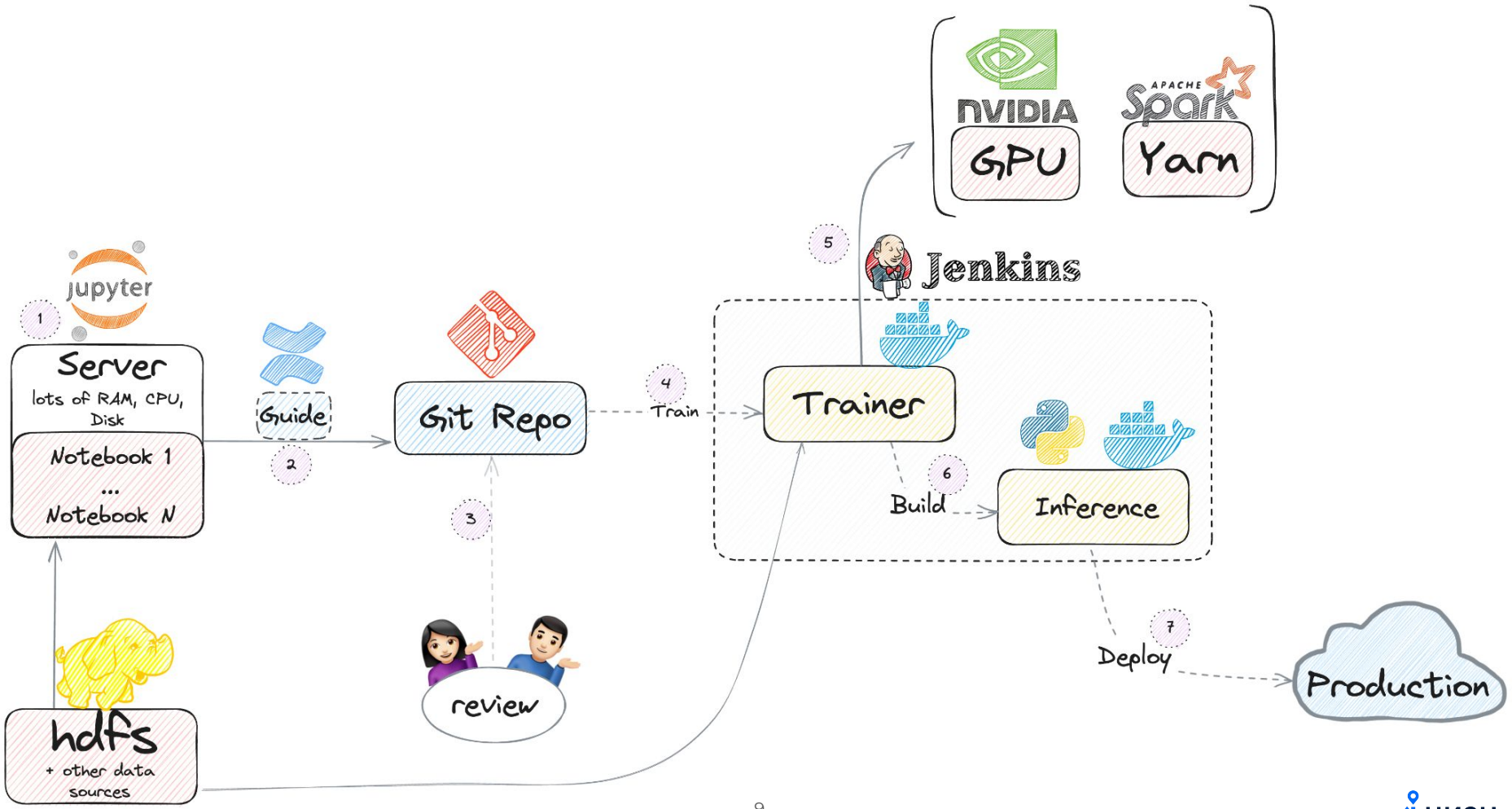


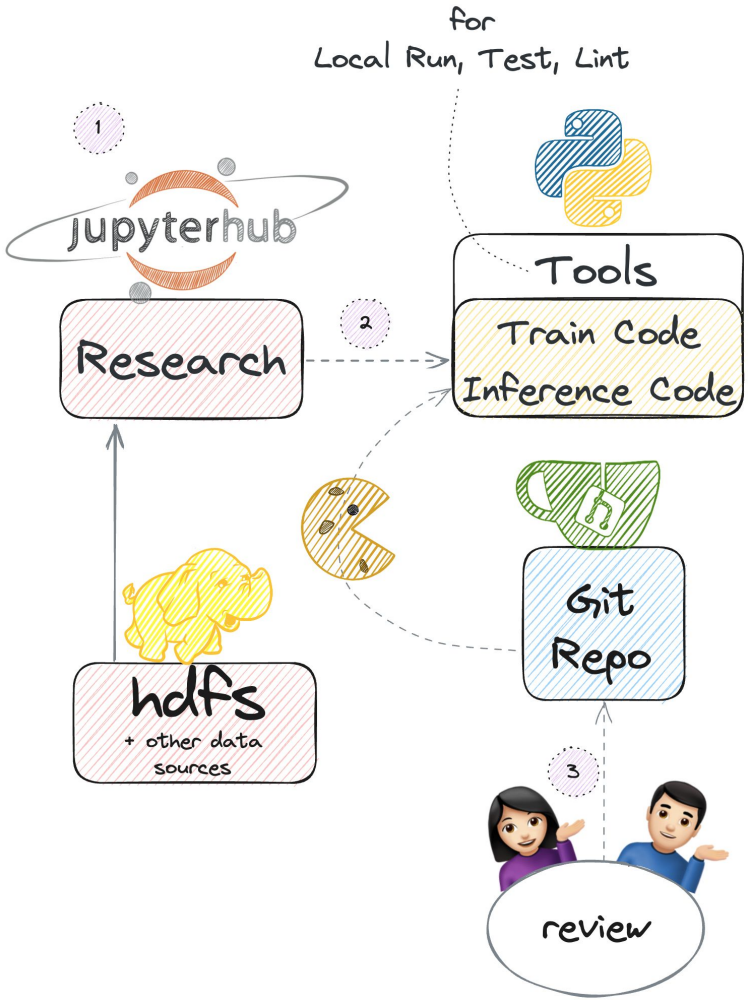
- Борьба за ресурсы.
- Постепенное «замусоривание».
- Слабая изоляция окружений.
- Соблазн, запустить модель для прода там же, где экспериментировали.

Обучение и Production-этап



- Отсутствие автоматизации.
- Нет model registry.
- Нет управления качеством.
- Ручной трейн и деплой.
- Сложности с мониторингом и логами.
- Привыкаешь к «плохому».


NVIDIA
GPU
APACHE
Spark
Yarn
5
Jenkins
jupyter
1
Server
lots of RAM, CPU, Disk
Notebook 1
...
Notebook N
Guide
2
Git Repo
3
4
Train
Trainer
6
Build
Inference
7
Deploy
Production
review
hdfs
+ other data sources



- JupyterHub для изоляции Jupyter-ноутбуков.
- Шаблон модели, cookiecutter.
- Инструменты для миграции из ipynb в структуру шаблона.

циан

Шаблон репозитория модели

```
# конфигурация модели
01    model.toml


# зависимости для обучения
02    requirements.txt
03    train_system_libs.txt

# зависимости для инференса
04    inference_requirements.txt
05    inference_system_libs.txt
```

```
# код модели
05    ml_model/
06        train/  # код обучения
07            train.py
08        inference/  # код инференса
09            inference.py
10        shared/  # общие модули

# функциональные тесты
11    tests_functional/
12        v1_predict/
13            test_request_01.json
```

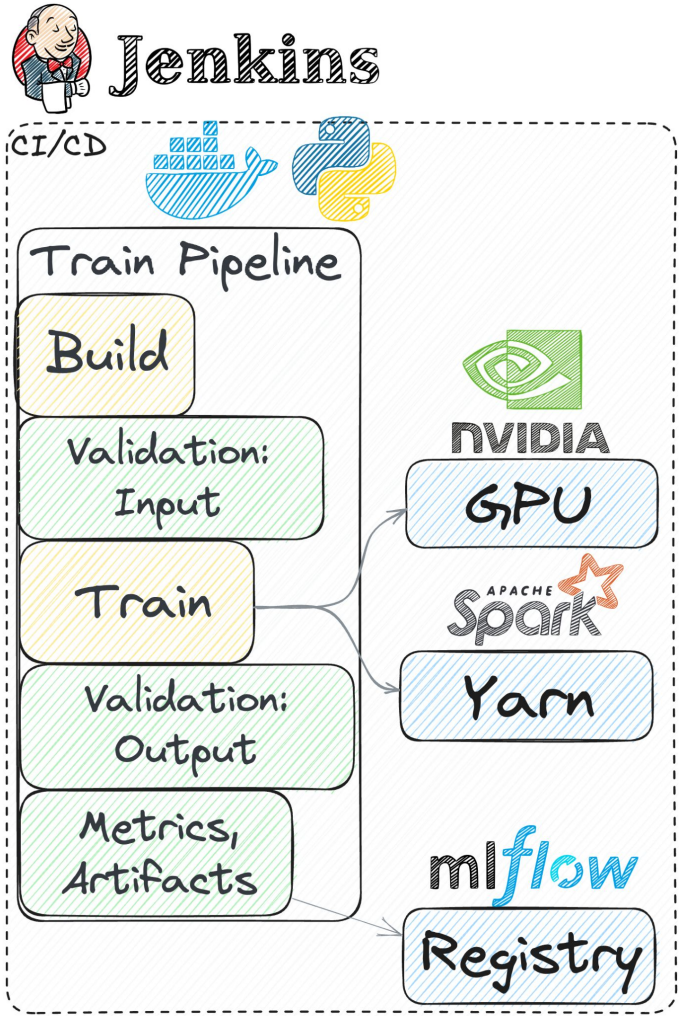
Конфигурация модели

```
[model]
model_name = "ml-model-test"
model_version = 1
model_type = "classification"
team = "ml-platform"

[runtime]
python_version = "3.10"

[inference_resources]
cpu_limit = 512
memory_limit = 1024
...

[train_resources]
memory_limit = 2000
cpus_number = 1
...
```

```
[train_metrics]
recall.min = 0.9

[dvc]
train_dataset_name = "some-dataset"

[stat_monitoring]
sampling_ratio = 0.5
```

train.py

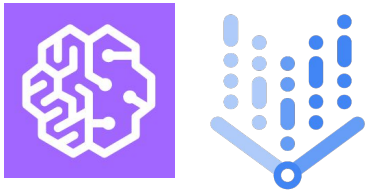
```
# обучаем модель
01    clf = svm.SVC(gamma='scale')
02    iris = datasets.load_iris()
03    X, y = iris.data, iris.target

# сохраняем модель
04    with open('sklearn_model.pkl', 'wb') as f:
05        pickle.dump(clf, f)
06    artifacts = {'sklearn_model': 'sklearn_model.pkl'}

# сохраняем информацию об артефактах в inference_artifacts.json
07    Path('inference_artifacts.json').write_text(json.dumps(artifacts))
```



- Подготовленные образы трейнера GPU/CPU.
- Валидация входных/выходных данных.
- Валидация метрик.
- Сохранение в MLflow.

Inference

Облачные решения



Yandex Cloud

Amazon SageMaker, Google Vertex AI, Azure ML, etc.

+ Удобно, если уже в облаке.
+ Запуск в пару кликов из ноутбука.
+ Масштабируется.

- Управление доставкой изменений, качеством, безопасностью — «отменяют» удобство.
- Ограниченная кастомизация.
- Привязка к конкретному облаку.
- Не всегда очевидное ценообразование.

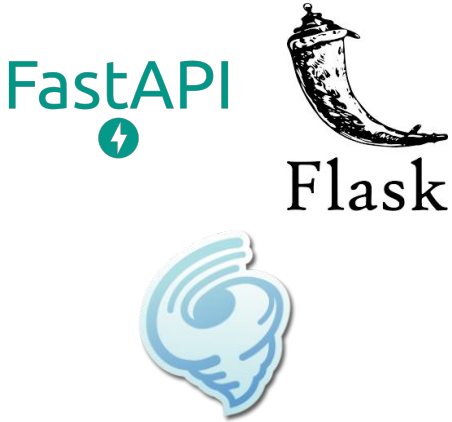
Готовые решения

Nvidia Inference Server, TorcheServe, Seldon Core, etc.

+ Много возможностей из коробки.
+ Производительность, бенчмарки.

- Может не хватать возможностей.
- Зависимость от решения.
- Может быть избыточным.

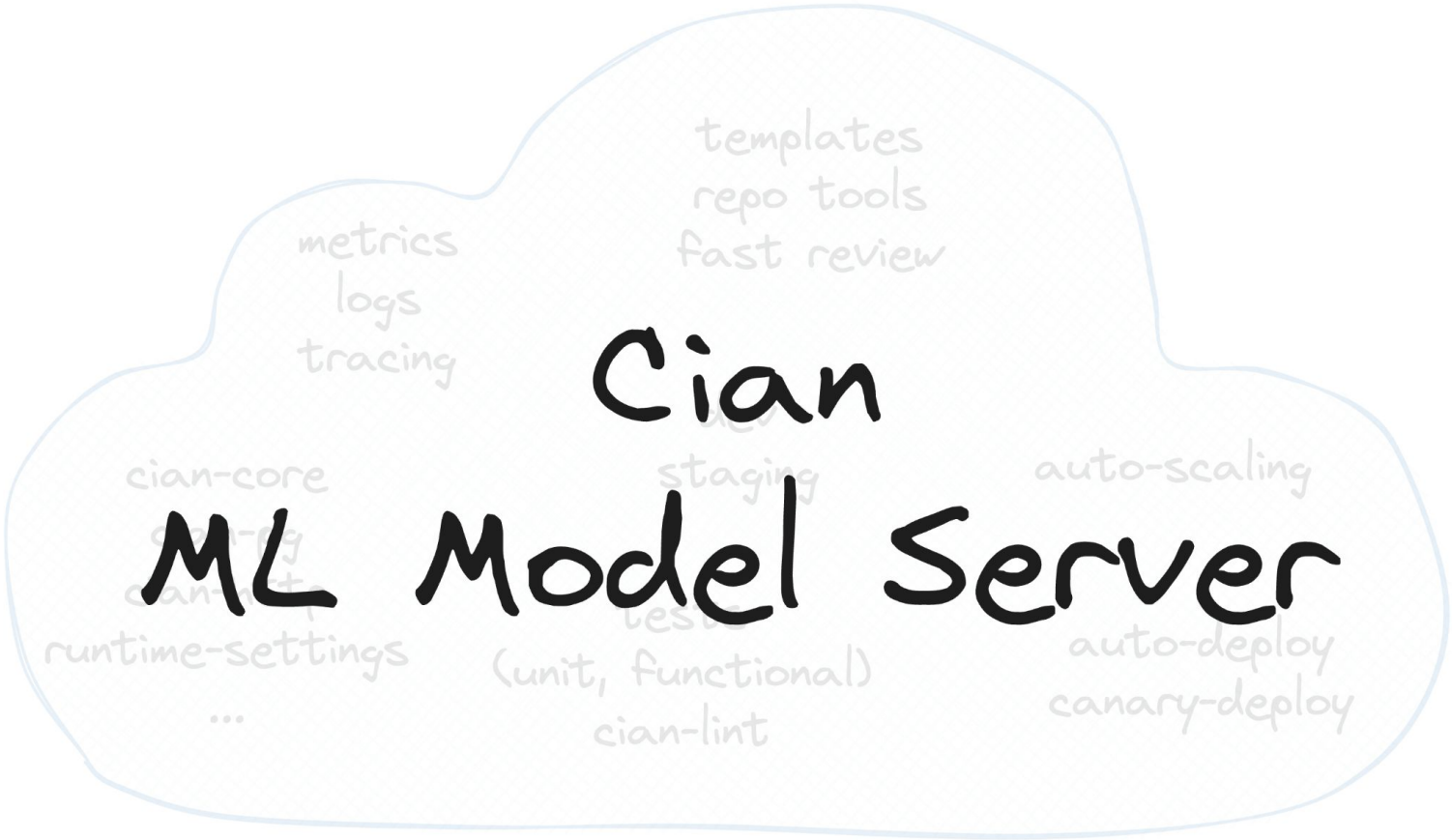
Свое решение



На основе FastAPI, Flask, Tornado, etc.

+ Гибкость.
+ Практически неограниченные возможности.
+ Совместимость с существующей инфраструктурой.

- Практически неограниченные возможности.


templates
repo tools
fast review
metrics
logs
tracing
Cian
cian-core
staging
auto-scaling
ML Model Server
runtime-settings
(unit, functional)
auto-deploy
canary-deploy
...
cian-lint

циан

inference.py

```
01    class TestModel(mlflow.pyfunc.PythonModel):
02        def load_context(self, context):
              # На этапе обучения мы сохранили информацию об артефактах
              # Артефакты из inference_artifacts.json будут доступны в context.artifacts
03            with open(context.artifacts['sklearn_model'], 'rb') as f:
04                self.sklearn_model = pickle.load(f)

          # делаем что-то с data, если нужно
05        def transform_input(self, context, data):
              ...

06        def predict(self, context, model_input):
06            return self.sklearn_model.predict(model_input)[:, np.newaxis]
```

schemas

```python
@dataclass
class Image
    filename: str
    raw_body: bytes
    pillow_image: JpegImageFile
```

```python
@dataclass
class UploadedFile:
    filename: str
    body: bytes
    content_type: str
```

```python
@dataclass
class ModelResponseSchema
# произвольный формат, например:
    classes: List[str]
```

model_mcs

```
01    def model_setup():
02        loaded_model = mlflow.pyfunc.load_model(model_uri=get_model_path())
03        model.set(loaded_model)

04    def setup() -> None:
05        cian_core.setup(..., additional_setup=model_setup, ...)

...

06    def execute_model_predict(model_input: Any) -> Any::
07        ml_model = model.get()
08        with statsd.timer('prediction.model-predict'):
09            predictions = ml_model.predict(model_input)
10        return predictions
```

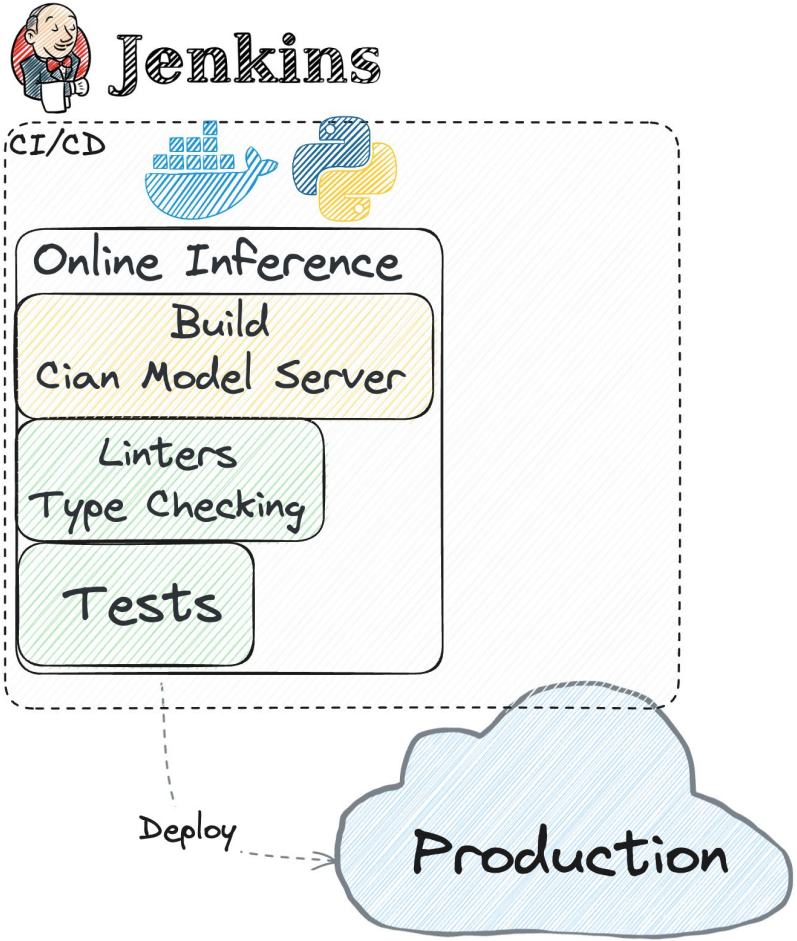
Тесты

test_request01.json

```
{
  "floatData": [[1.0, 2.0]],
  "strData": [["foo", "bar"]]
}
```

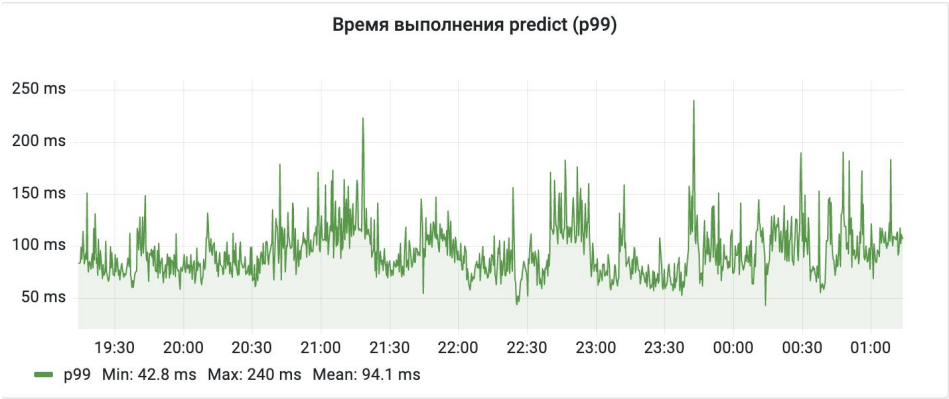
test_request01_response.json

```
{
  "predictions": [[3.0, ">60,⩽100"]]
}
```

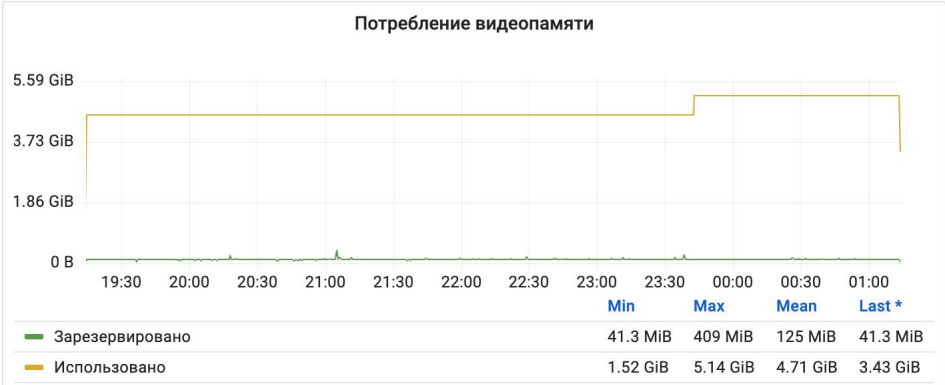


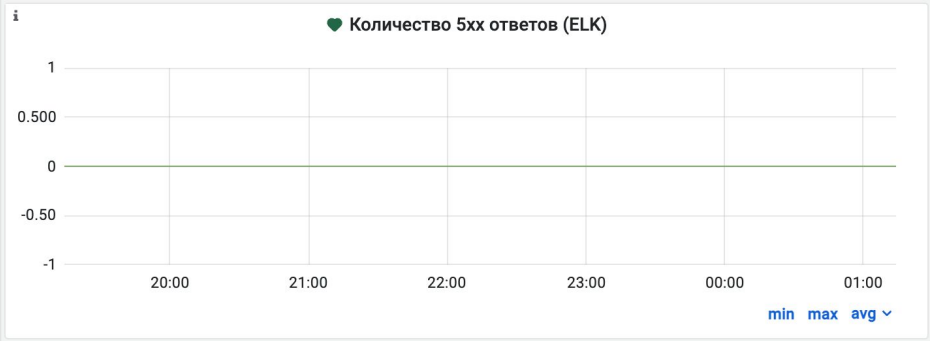
- Сборка микросервиса с моделью.
- Проверка форматирования, типов.
- Тесты.
- Деплой (+canary).

Production



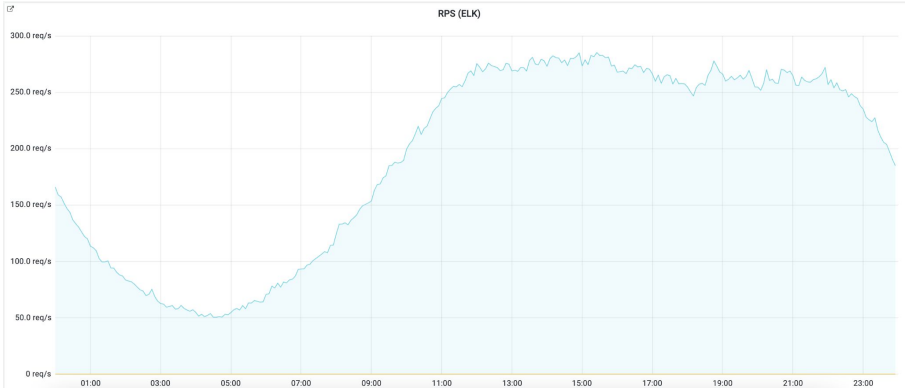
- Мониторинг, алерты в grafana.
- Логи в ELK.
- Автоскейлинг.
- Изменение ресурсов на лету.

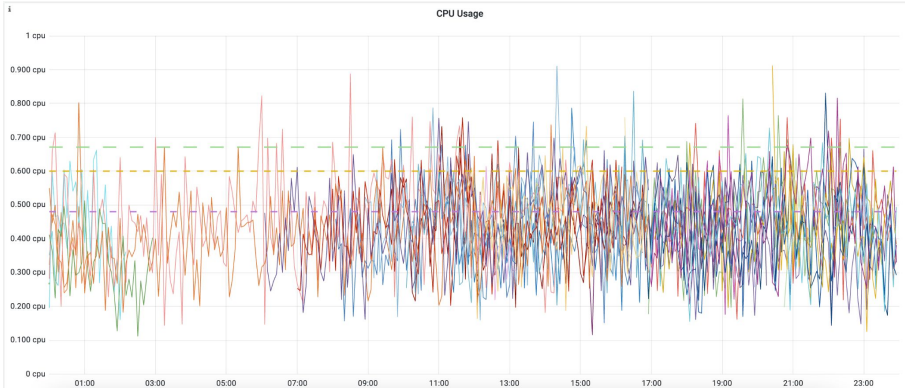


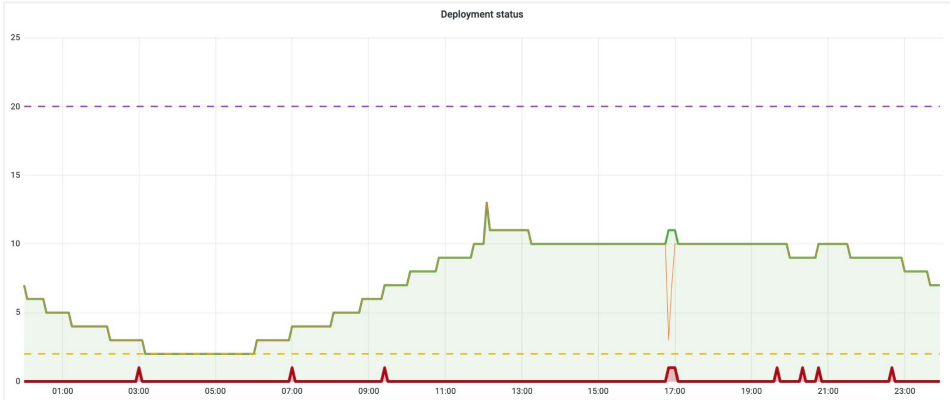


циан

Автоскейлинг



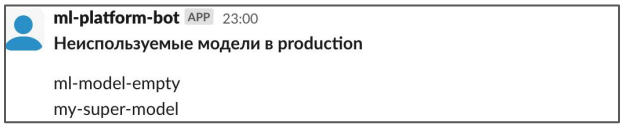




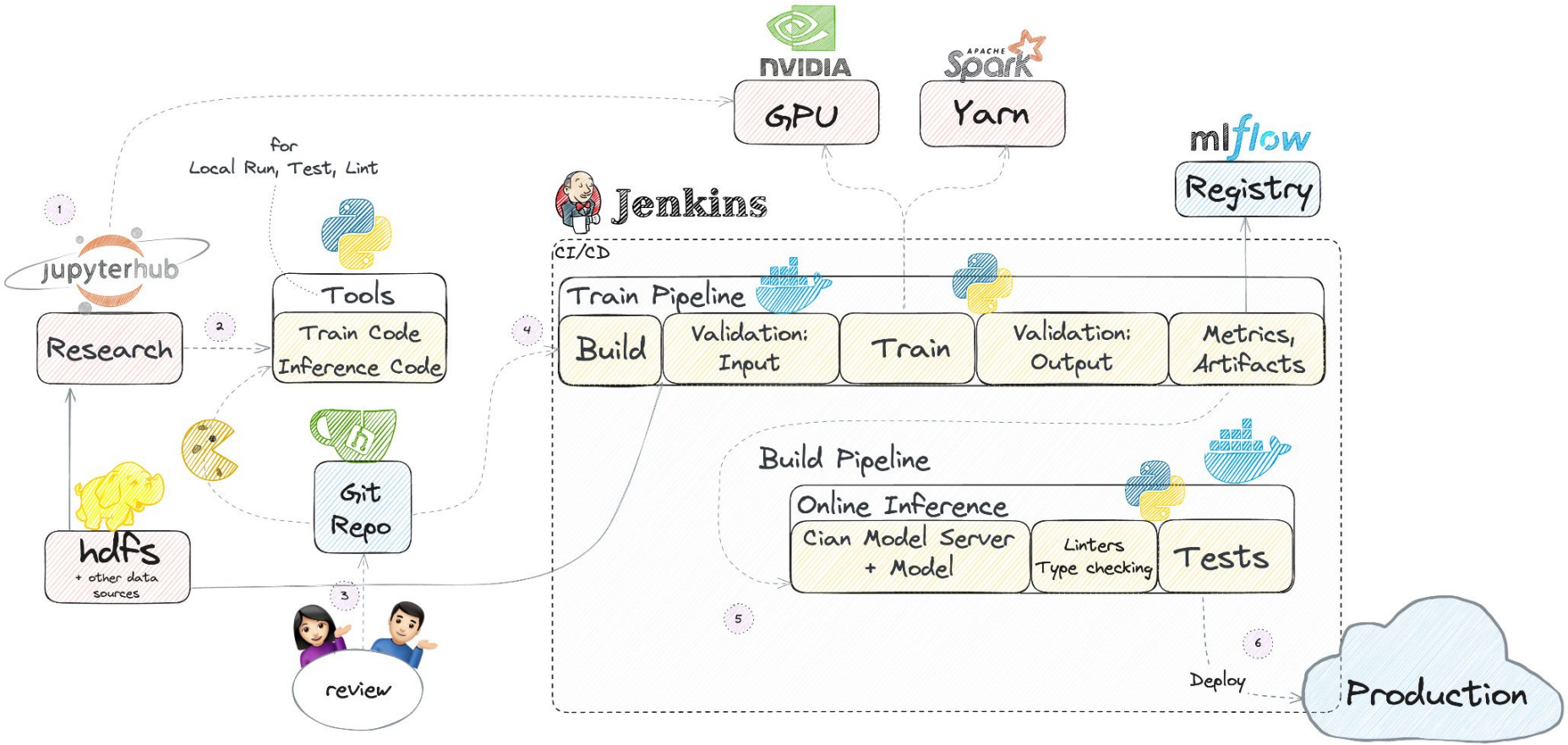
циан

Управление ресурсами

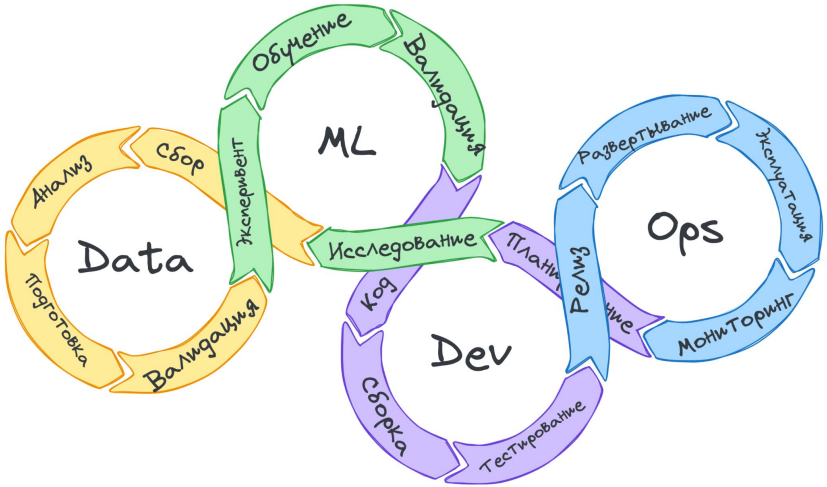
- запрошено
- рекомендовано
- порог для автоскейлинга



циан


NVIDIA
GPU
APACHE Spark
Yarn
mlflow
Registry
for
Local Run, Test, Lint
Jenkins
1
jupyterhub
Research
2
Tools
Train Code
Inference Code
4
CI/CD
Train Pipeline
Build
Validation: Input
Train
Validation: Output
Metrics, Artifacts
Git Repo
Build Pipeline
Online Inference
Cian Model Server + Model
Linters Type checking
Tests
hdfs
+ other data sources
3
review
5
6
Deploy
Production

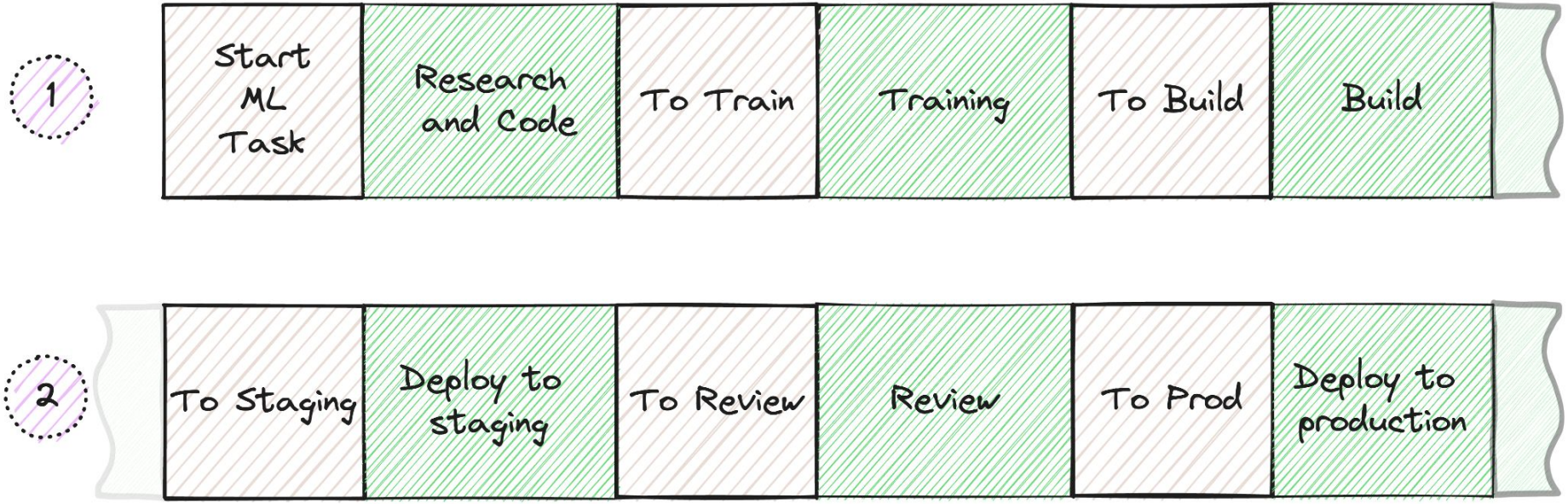
циан



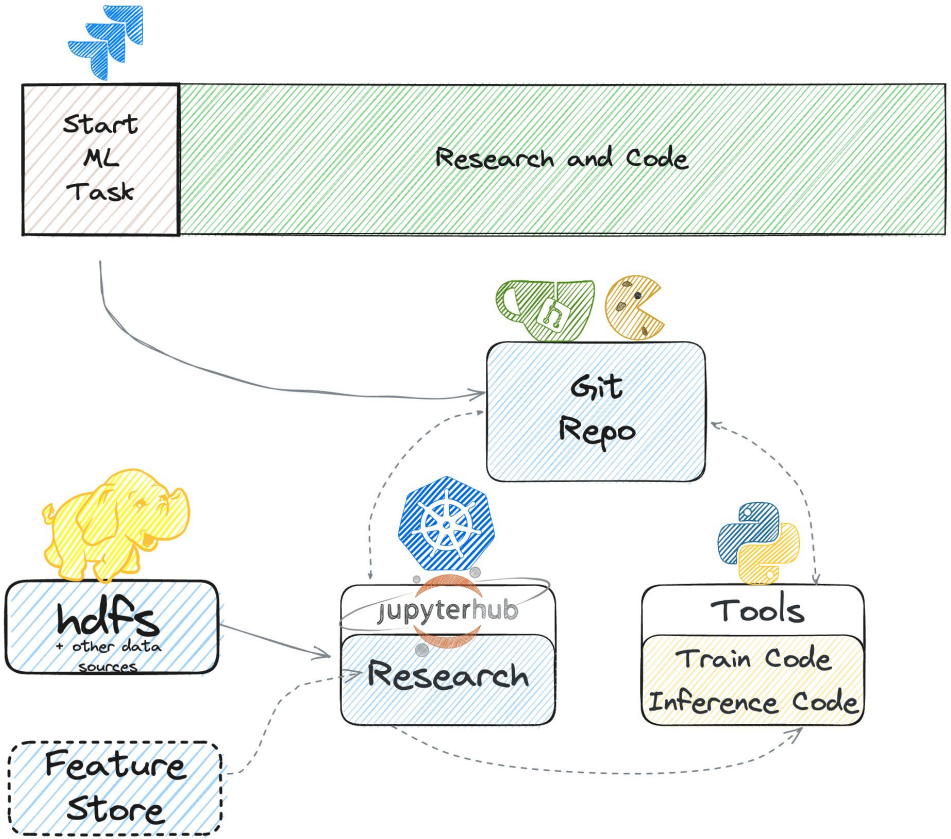
Что еще хотелось:

- удобное управление «одной кнопкой»
- интерфейс с результатами каждого этапа
- сбор и анализ метрик времени работы джоб
- количество успешных, неуспешных запусков, классификация причин
- TTM моделей

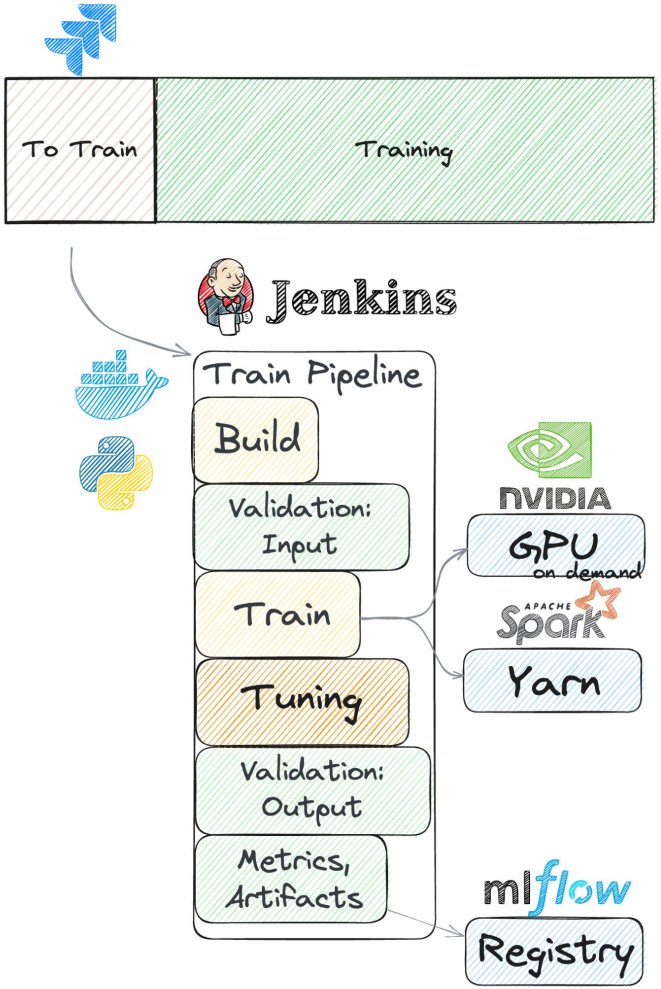
Jira + Comunda



циан



- Автоматическое создание репозитория.
- Далее ресерч в JupyterHub k8s.



Robot added a comment -

Обучение модели завершено

[Success] TRAIN_RESULT

- Трейн модели успешно завершен. Артефакты доступны по ссылке в MLflow.

Артефакты:

- MLflow train url.
- yarn_serviceam.log
- yarn_stderr.txt
- yarn_stdout.txt

ml-platform-bot APP 07:12

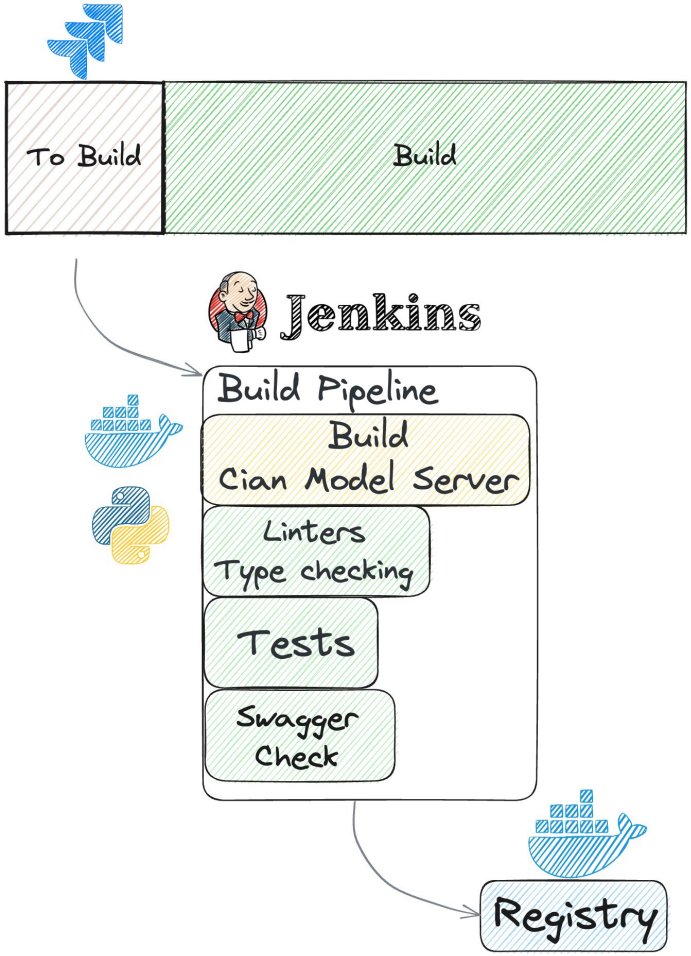
Train succeeded!

Model: **ml-model-test-gpu v1**
Branch: **master**

Train metrics:
n/a

Open job in Jenkins
Open run in MLFlow

циан



Robot added a comment - 31.10.2023, 11:27

[ERROR] Произошла ошибка
[ERROR] **BUILD_RESULT**

Артефакты:

- JOBEXTRAOUT_LINK_MCS_HEALTH_WARNINGS

[Failed] **TESTING_RESULT**

- Перезапускались 2 функциональных теста:
  tests_functional.test_built_model.test_built_model_v1_predict.test_built_model_v1_predict[test_input0],
  tests_functional.test_built_model.test_built_model_v1_predict.test_built_model_v1_predict[test_input1]
- Проверка Swagger схем сервиса успешно пройдена. Отчет доступен по ссылке.

Артефакты:

- mypy_report.txt (1 errors)
- fmt_report.txt (0 errors)
- swagger report (Errors: 0, Warnings: 0)

Ошибки:

- **<<CODE>>** Упали тесты, подробности по приложенной ссылке.

Robot added a comment - 31.10.2023, 18:00

Сборка и тесты прошли успешно

[Success] **BUILD_RESULT**

Артефакты:

- JOBEXTRAOUT_LINK_MCS_HEALTH_WARNINGS

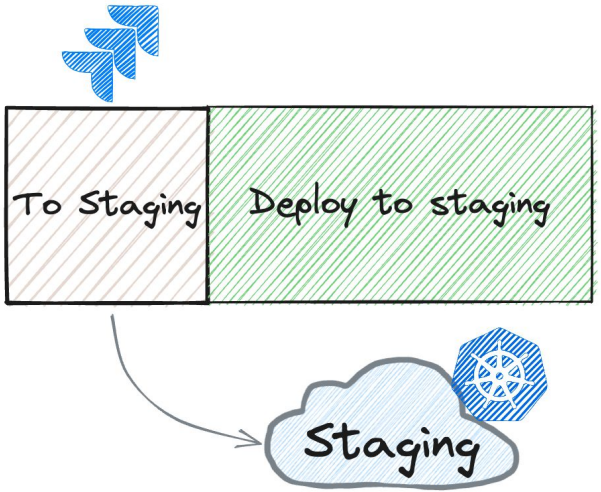
[Success] **TESTING_RESULT**

- Проверка Swagger схем сервиса успешно пройдена. Отчет доступен по ссылке.

Артефакты:

- diff-cover.html (100%)
- mypy_report.txt (1 errors)
- Overall coverage (68%)
- fmt_report.txt (0 errors)
- swagger report (Errors: 0, Warnings: 0)

циан



Robot added a comment -

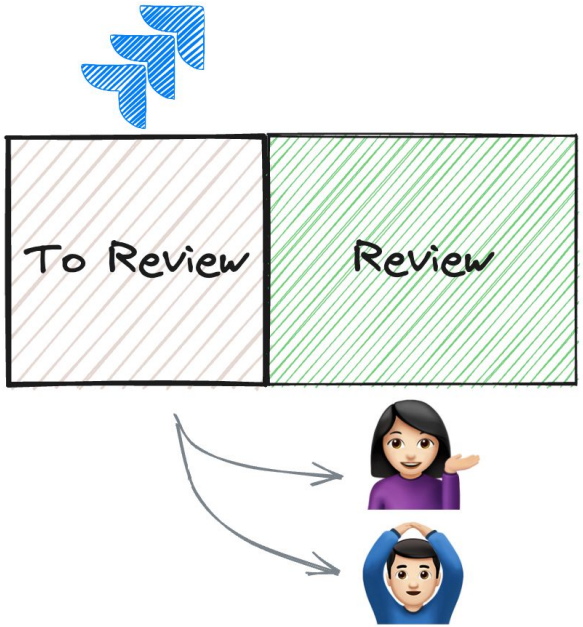
Выкладка завершена

[**Success**] **STAGING_RESULT**

- Деплой в staging-окружение завершен. Убедиться, что сервис модели успешно запустился и находится в Healthy/Synced статусе, можно перейдя по ссылке.

Артефакты:

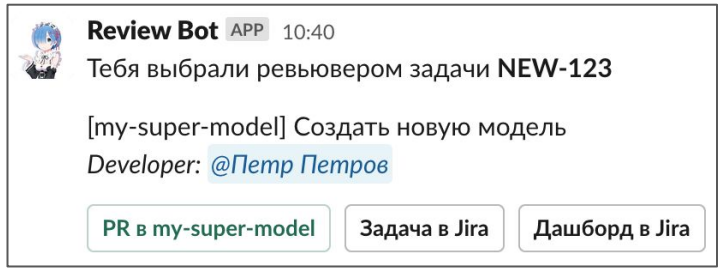
- Service Url.

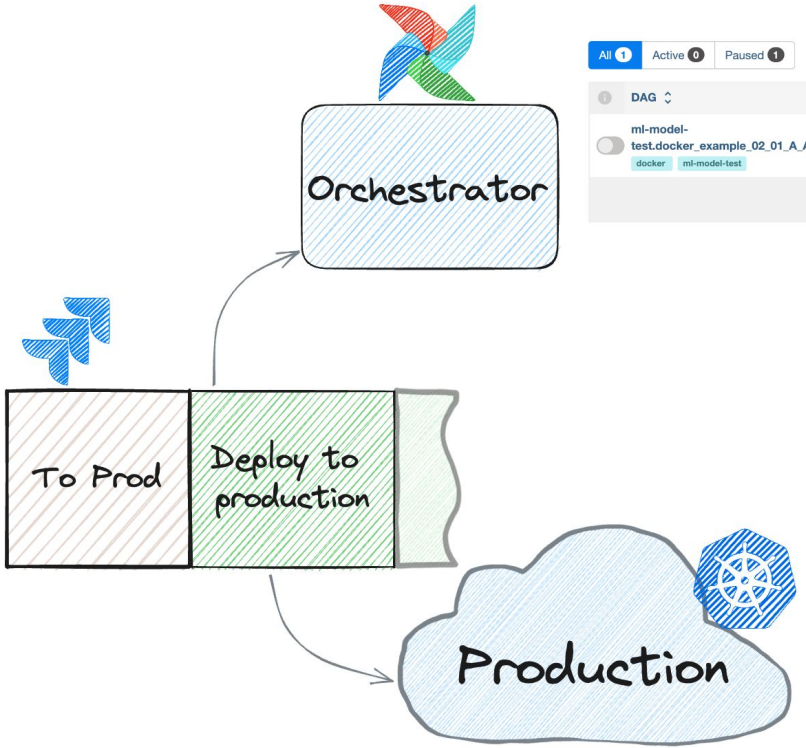


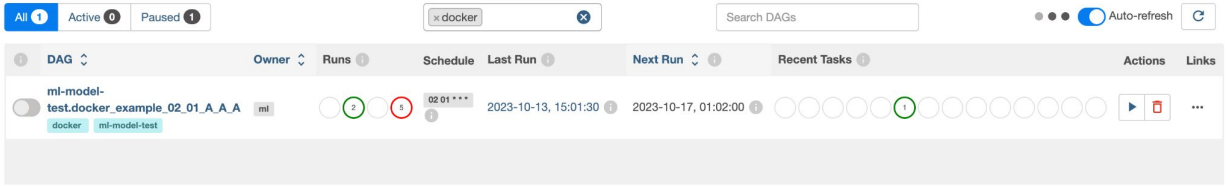
- Автоматическое назначение ревьюверов: один из мейнтейнеров модели/репозитория второй из всего пула ML

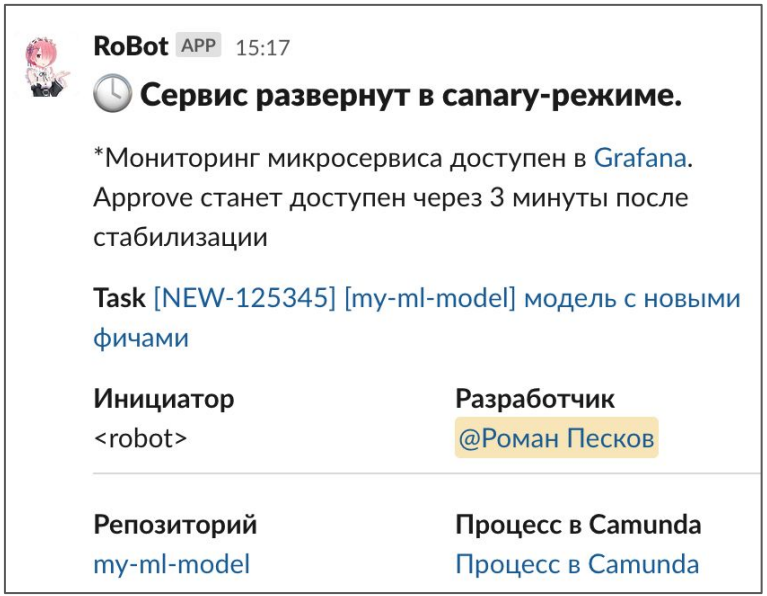
Developer:	Иван Иванов
Reviewers:	Андрей Андреев, Мария Машина, Роман Песков ...
Required Reviewers:	Андрей Андреев, Мария Машина

- Уведомления в slack о ревью

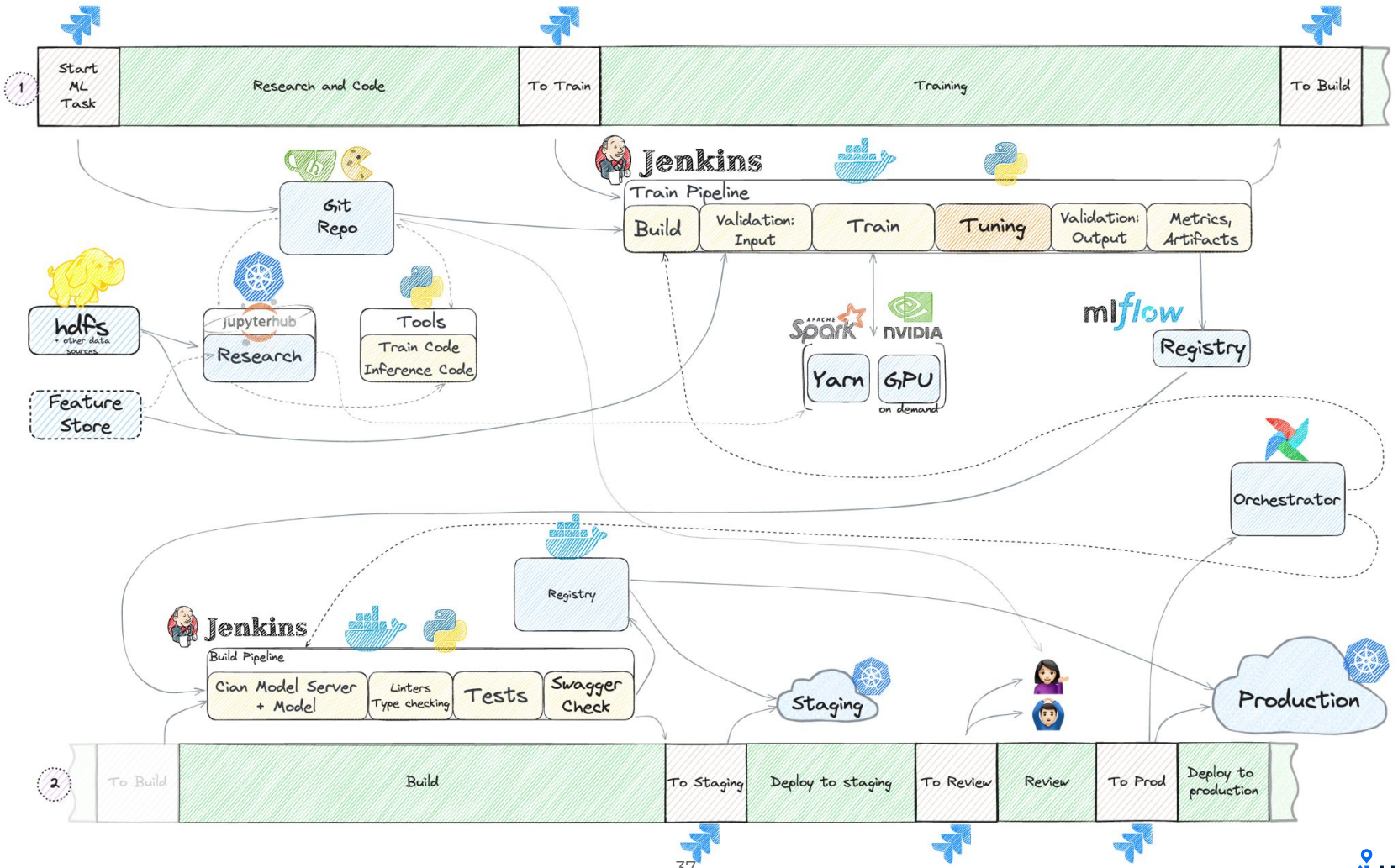



Orchestrator
To Prod
Deploy to production
Production


All 1
Active 0
Paused 1
docker
Search DAGs
Auto-refresh
DAG
Owner
Runs
Schedule
Last Run
Next Run
Recent Tasks
Actions
Links
ml-model-test.docker_example_02_01_A_A_A
ml
docker
ml-model-test
2023-10-13, 15:01:30
2023-10-17, 01:02:00


RoBot APP 15:17
Сервис развернут в canary-режиме.
*Мониторинг микросервиса доступен в Grafana. Approve станет доступен через 3 минуты после стабилизации
Task [NEW-125345] [my-ml-model] модель с новыми фичами
Инициатор
<robot>
Разработчик
@Роман Песков
Репозиторий
my-ml-model
Процесс в Camunda
Процесс в Camunda

циан


1
Start ML Task
Research and Code
To Train
Training
To Build
Git Repo
Jenkins
Train Pipeline
Build
Validation: Input
Train
Tuning
Validation: Output
Metrics, Artifacts
hdfs
+ other data sources
jupyterhub
Research
Tools
Train Code
Inference Code
Feature Store
Spark
NVIDIA
Yarn
GPU
on demand
mlflow
Registry
Orchestrator
Registry
Jenkins
Build Pipeline
Cian Model Server + Model
Linters Type checking
Tests
Swagger Check
Staging
Production
2
To Build
Build
To Staging
Deploy to staging
To Review
Review
To Prod
Deploy to production

➔ ручные запуски ⏬ ~0
➔ количество задач 📈 x3
➔ время в ревью 📉-50%
➔ время сборки 📉-15%

➔ развитие и адаптация 

не останавливаем процессы

легаси, с которым нужно бороться

➔ для ML-команд 

через обратную связь

борьба с привычками

➔ экспертиза DI, SRE, DE 

сильно упрощает

нужно договариваться

@peskovrn