



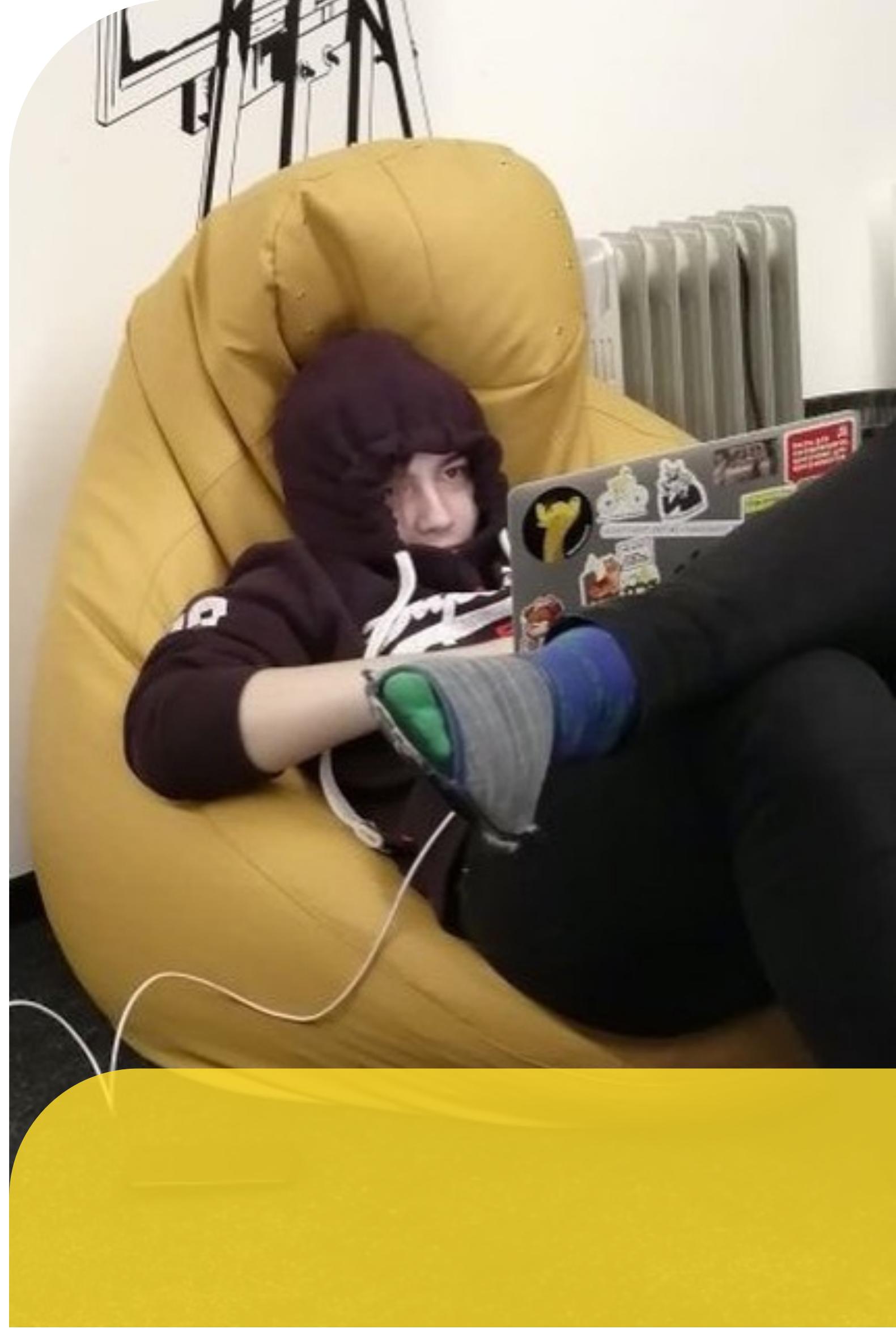
ТИНЬКОФФ

Кто такой ваш CI/CD

И как с ним дружить

Николай Тихонов

- Фронтенд-разнорабочий
- Не любитель делать руками то, что можно автоматизировать



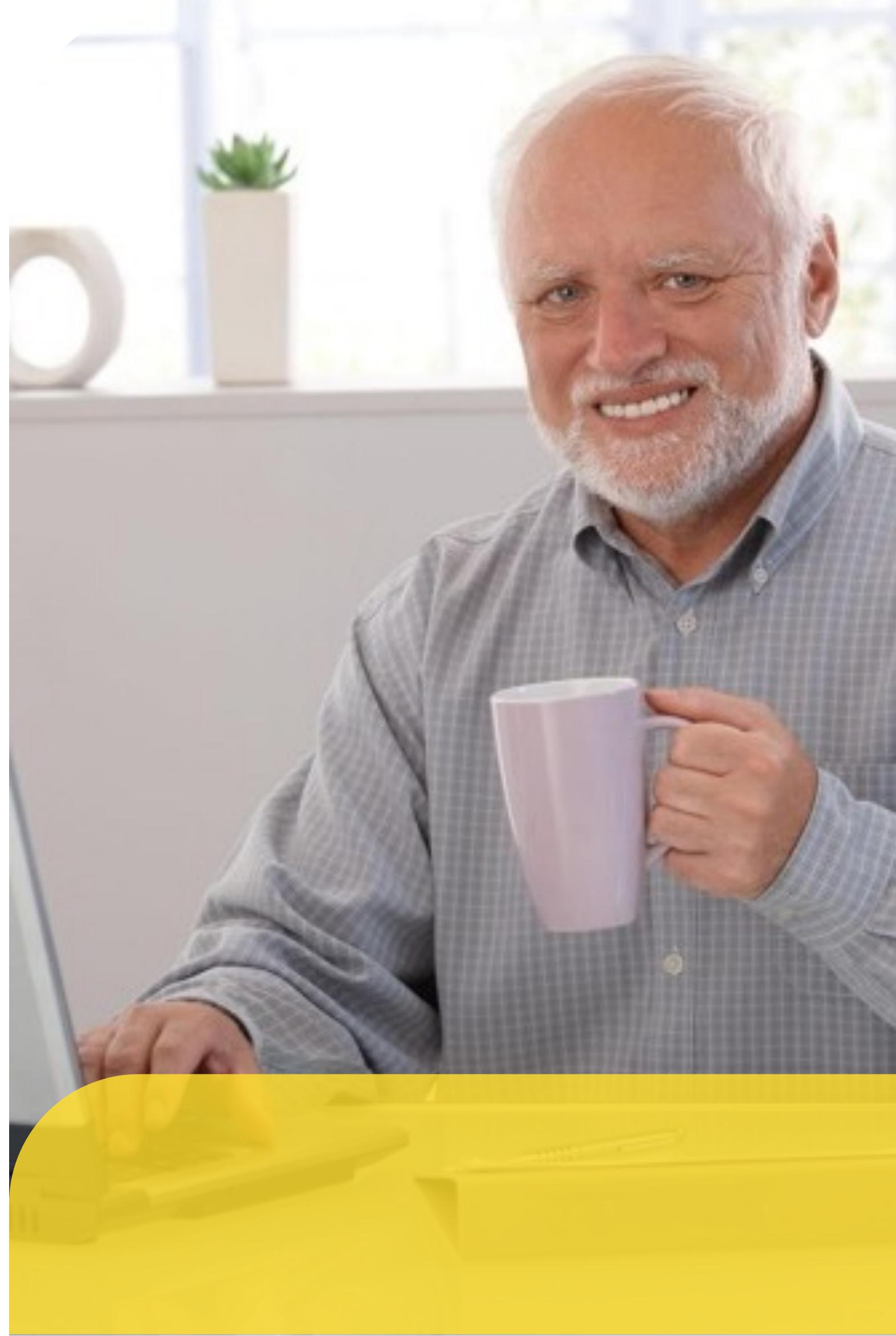
О чём сегодня поговорим?

- ➡ Что было до CI
- ➡ Важность автоматизации
- ➡ На чем и как можно делать CI
- ➡ Автоматизируем поставку сами

О чём сегодня не поговорим?



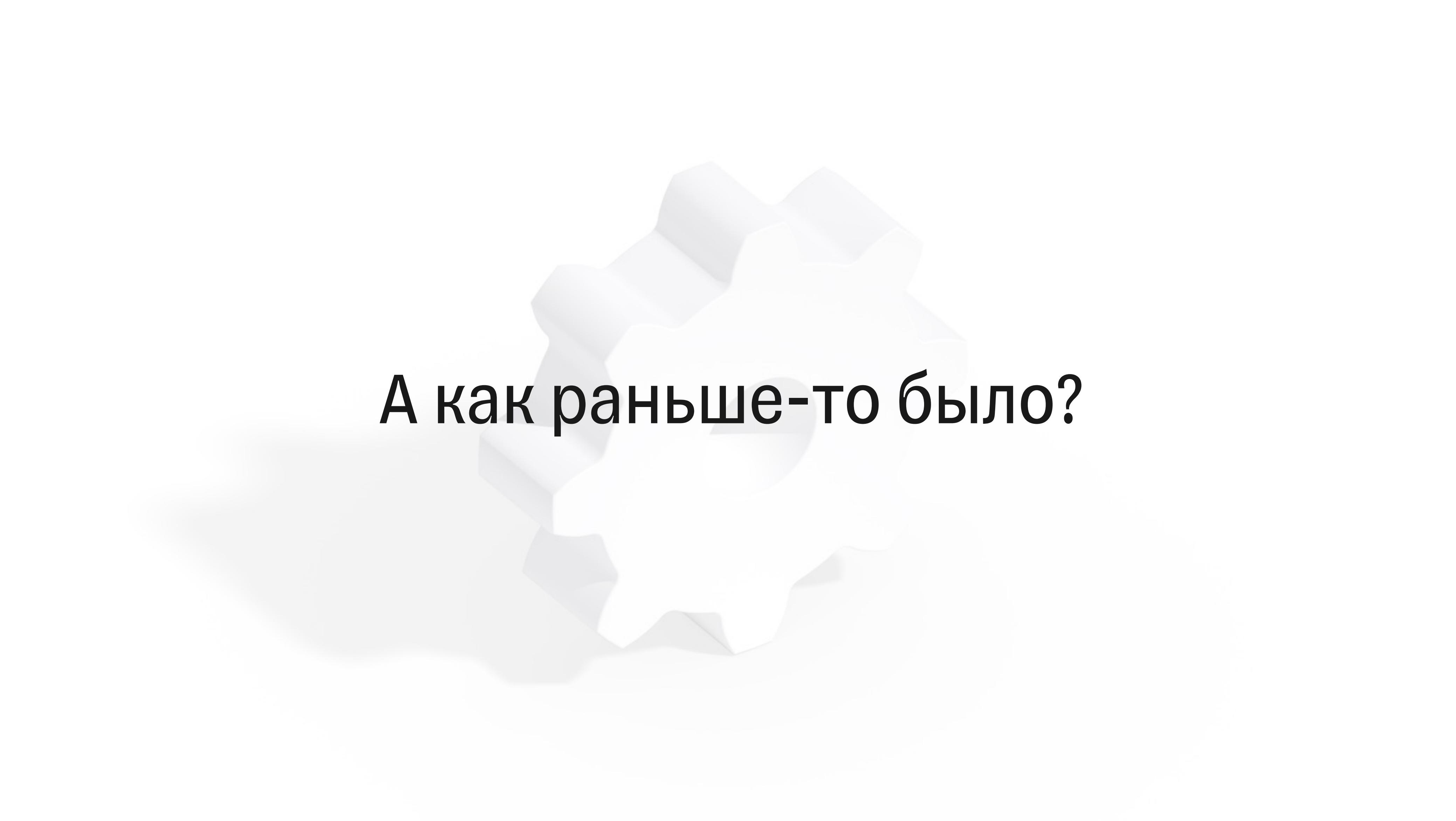
Документация к CI/CD системам



Геннадий Гарольдов

Старший Старый разработчик

Прошел путь от обычного работяги в 80-х до
руководителя



А как раньше-то было?

Темные доСI/CDшные времена...



Распространяем дискетами

Пилим продукт X лет, записываем его на
дискеты и отвозим в магазины



Пользователи пользуются

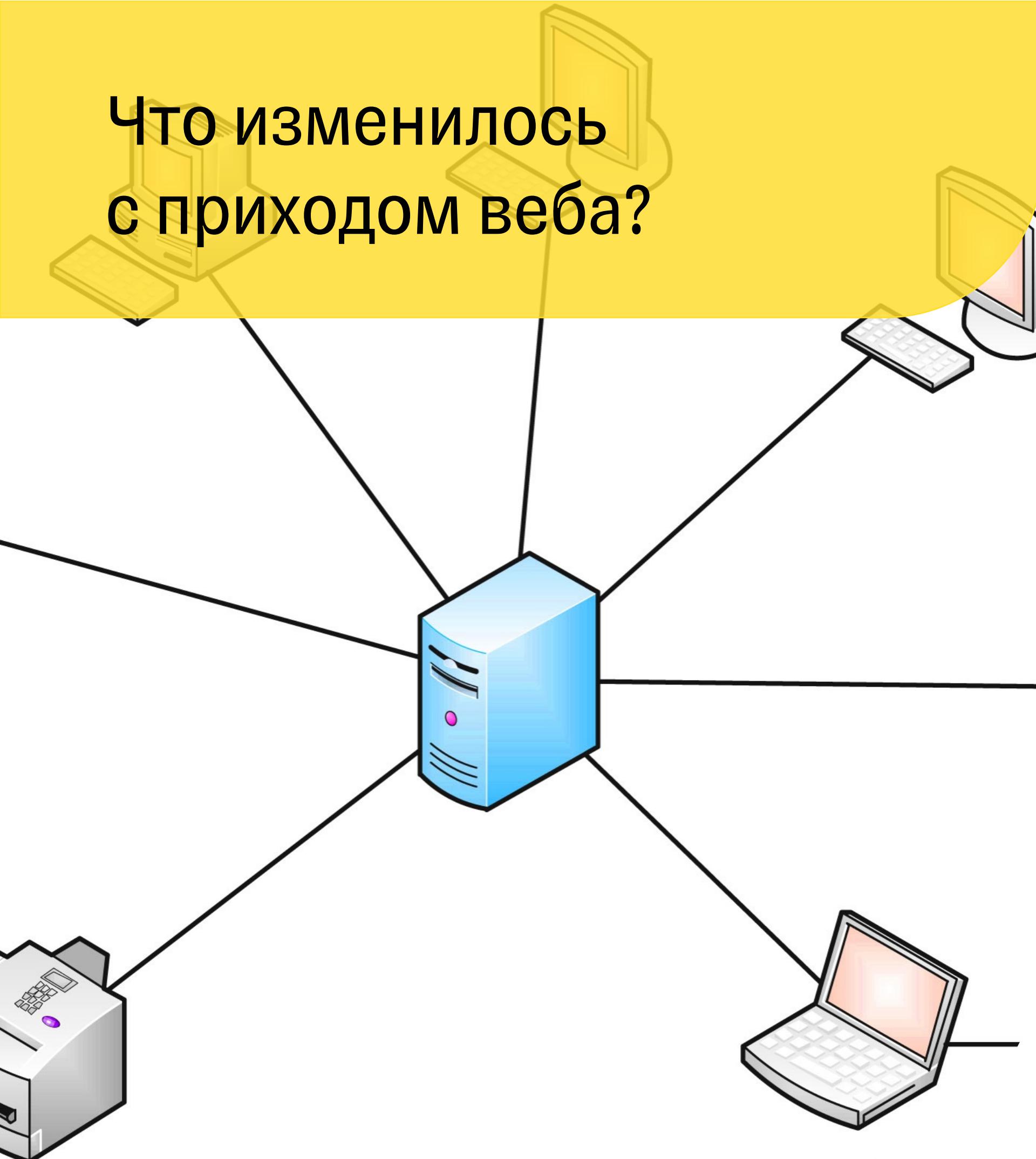
Понимаем, что вместе с продуктом
разрабатывали ещё и баги



Отзываем продукцию из магазинов

Чтобы пофиксить баги, нужно перезаписать
все дискеты мира

Что изменилось с приходом веба?



Теперь не надо продавать дискеты!

Вместо этого, чтобы обновить приложение, достаточно просто
зайти на сервер и залить туда нужные файлы

Обновление приложения на сервере



Супер примитивно

Настраиваем свой личный ПК как сервер и
просто меняем нужные файлики

Обновление приложения на сервере



Супер примитивно

Настраиваем свой личный ПК как сервер и просто меняем нужные файлики



Примитивно

Арендуем сервер в Heroku/DigitalOcean/Я.Облаке, заходим на него, меняем файлики и запускаем

Обновление приложения на сервере



Супер примитивно

Настраиваем свой личный ПК как сервер и просто меняем нужные файлики



Примитивно

Арендуем сервер в Heroku/DigitalOcean/Я.Облаке, заходим на него, меняем файлики и запускаем



Менее примитивно

Арендуем сервер там же, но используем Docker-образы

А если...



...у нас много сервисов и серверов?



...у нас много окружений? (тест/прод)



...нам надо автотестиовать приложение?



№ 18, 2021

ТАКИЕ ИНФОРМАЦИИ АДРЕСОВАНА ТЕМ, КТО ПЛАНИРУЕТ ДЕПЛОЙ В ПРОД В БЛИЖАЙШЕЕ ВРЕМЯ.
СПЕЦИАЛЬНО ДЛЯ НАШЕГО ЖУРНАЛА АСТРОЛОГ ЕЛЕНА ВАЛЕНТИНОВНА МАЗОВА ВЫСЧИТАЛА
В №20 ВАС ЖДЕТ АСТРОКАЛЕНДАРЬ НА НОЯБРЬ. НЕ ПРОПУСТИТЕ!

9

ТОЛЬКО У НАС!

Когда лучше деплоить в продакшн.

T.ME/DEVOPSINA

Октябрь

ЗНАК ЗОДИАКА	БЛАГОПРИЯТНЫЕ ДНИ	НЕЙТРАЛЬНЫЕ ДНИ	НЕВЛАГОПРИЯТНЫЕ ДНИ
ОВЕН	12, 14, 18, 19, 23, 24, 25, 26, 27	1, 2, 7, 10, 11, 15, 16, 21, 22, 29, 30	3, 4, 5, 6, 8, 9, 13, 17, 20, 28, 31
ТЕЛЕЦ	1, 2, 14, 15, 21, 22, 26, 27, 29, 30	3, 4, 5, 8, 9, 12, 16, 17, 31	6, 7, 10, 11, 13, 18, 19, 20, 23, 24, 25, 28
БЛИЗНЕЦЫ	1, 2, 3, 4, 5, 16, 17, 23, 24, 25, 29, 30, 31	7, 10, 11, 14, 15, 18, 19	6, 8, 9, 12, 13, 20, 21, 22, 26, 27, 28
РАК	3, 4, 5, 7, 18, 19, 26, 27, 31	8, 9, 12, 16, 17, 21, 22	1, 2, 6, 10, 11, 13, 14, 15, 20, 23, 24, 25, 28, 29, 30
ЛЕВ	1, 2, 7, 8, 9, 21, 22, 29, 30	10, 11, 14, 15, 18, 19, 23, 24, 25	3, 4, 5, 6, 12, 13, 16, 17, 20, 26, 27, 28, 31
ДЕВА	3, 4, 5, 8, 9, 10, 11, 23, 24, 25, 31	12, 16, 17, 21, 22, 26, 27	1, 2, 6, 7, 13, 14, 15, 18, 19, 20, 28, 29, 30
ВЕСЫ	7, 10, 11, 12, 26, 27	1, 2, 14, 15, 18, 19, 23, 24, 25, 29, 30	3, 4, 5, 6, 8, 9, 13, 16, 17, 20, 21, 22, 28, 31
СКОРПИОН	1, 2, 8, 9, 12, 14, 15, 29, 30	3, 4, 5, 16, 17, 21, 22, 26, 27, 31	6, 7, 10, 11, 13, 18, 19, 20, 23, 24, 25, 28
СТРЕЛЕЦ	3, 4, 5, 10, 11, 14, 15, 16, 17, 31	1, 2, 7, 18, 19, 23, 24, 25, 29, 30	6, 8, 9, 12, 13, 20, 21, 22, 26, 27, 28
КОЗЕРОГ	7, 12, 16, 17, 18, 19, 31	3, 4, 5, 8, 9, 21, 22, 26, 27	1, 2, 6, 10, 11, 13, 14, 15, 20, 23, 24, 25, 28, 29, 30
ВОДОЛЕЙ	8, 9, 14, 15, 18, 19, 21, 22	1, 2, 7, 10, 11, 23, 24, 25, 29, 30	3, 4, 5, 6, 12, 13, 16, 17, 20, 26, 27, 28, 31
РЫБЫ	10, 11, 16, 17, 21, 22, 23, 24, 25	3, 4, 5, 8, 9, 12, 26, 27, 31	1, 2, 6, 7, 13, 14, 15, 18, 19, 20, 28, 29, 30



ТИНЬКОФФ

Что и как автоматизировать?

Старый процесс доставки на сервер

01

Устанавливаем
зависимости

Вряд ли мы будем
писать приложение,
не зависящее от
сторонних библиотек
или фреймворков

02

Тестируем
приложение

Перед тем, как думать
о поставке кода,
нужно его
протестировать.

03

Собираем
приложение

04

Разворачиваем
приложение

Доставляем код
до места назначения.
Как раз тут кладем
нужные файлы
на сервер

05

Убеждаемся,
что все работает

Опционально неплохо
бы убедиться в том,
что код
действительно доехал
в нужном нам виде.

Старый процесс доставки на сервер

01

Устанавливаем
зависимости

Вряд ли мы будем
писать приложение,
не зависящее от
сторонних библиотек
или фреймворков

02

Тестируем
приложение

Перед тем, как думать
о поставке кода,
нужно его
протестировать.

03

Собираем
приложение

04

Разворачиваем
приложение

Доставляем код
до места назначения.
Как раз тут кладем
нужные файлы
на сервер

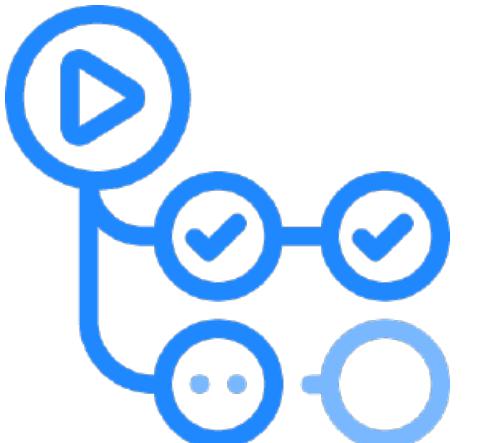
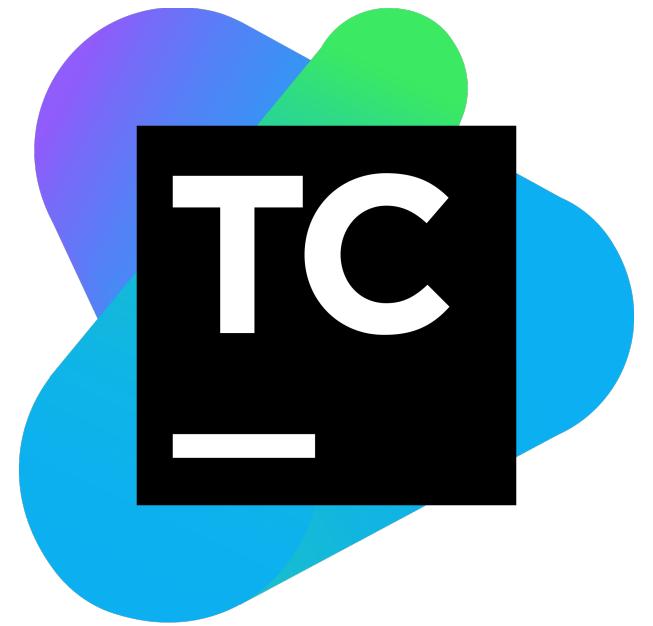
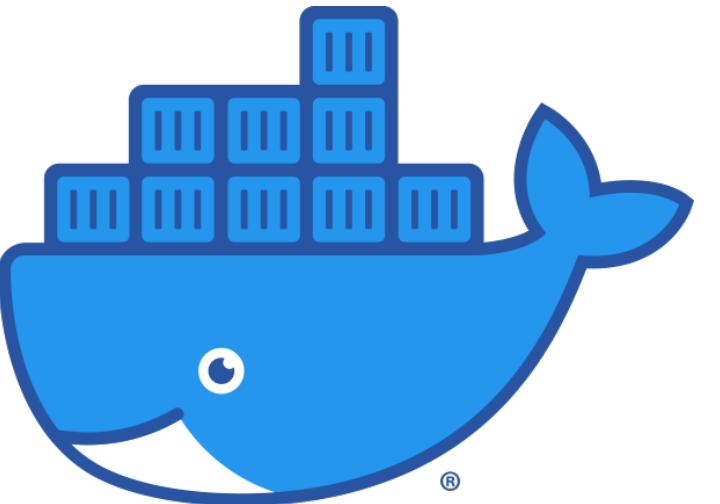
05

Убеждаемся,
что все работает

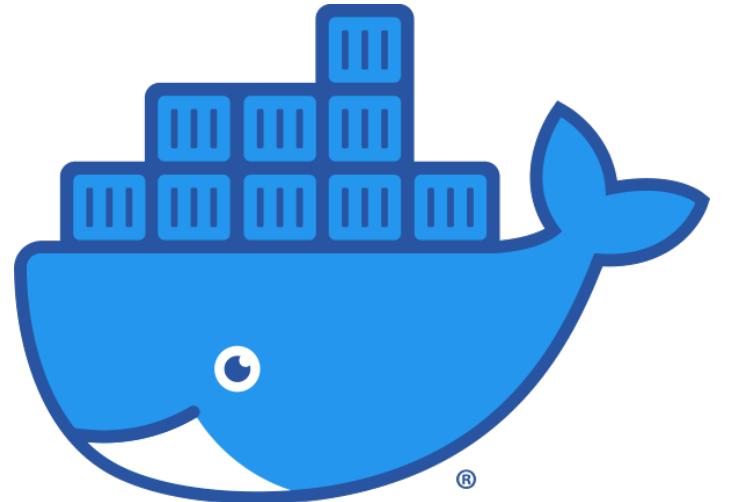
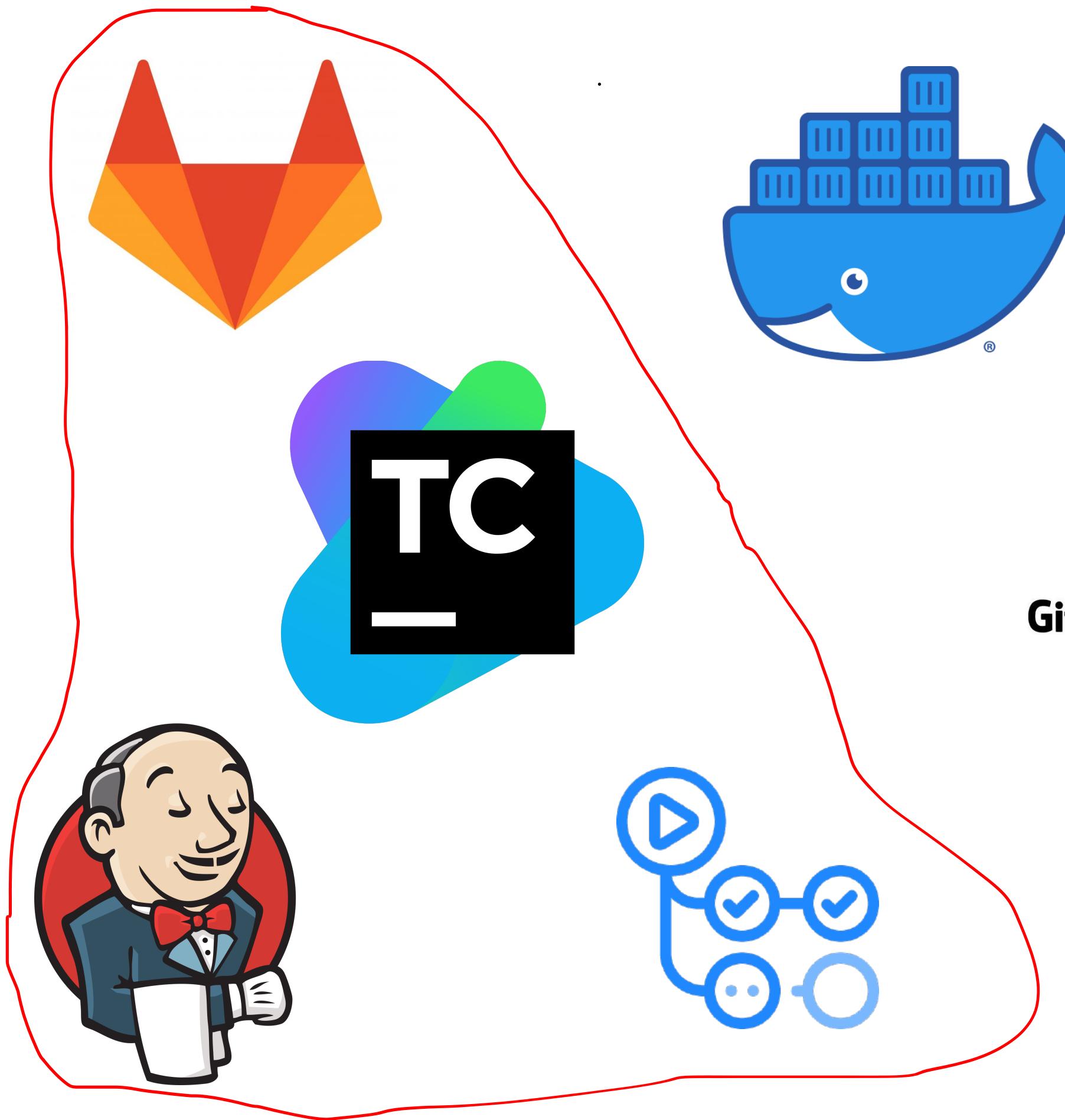
Опционально неплохо
бы убедиться в том,
что код
действительно доехал
в нужном нам виде.

Коэффициент горения

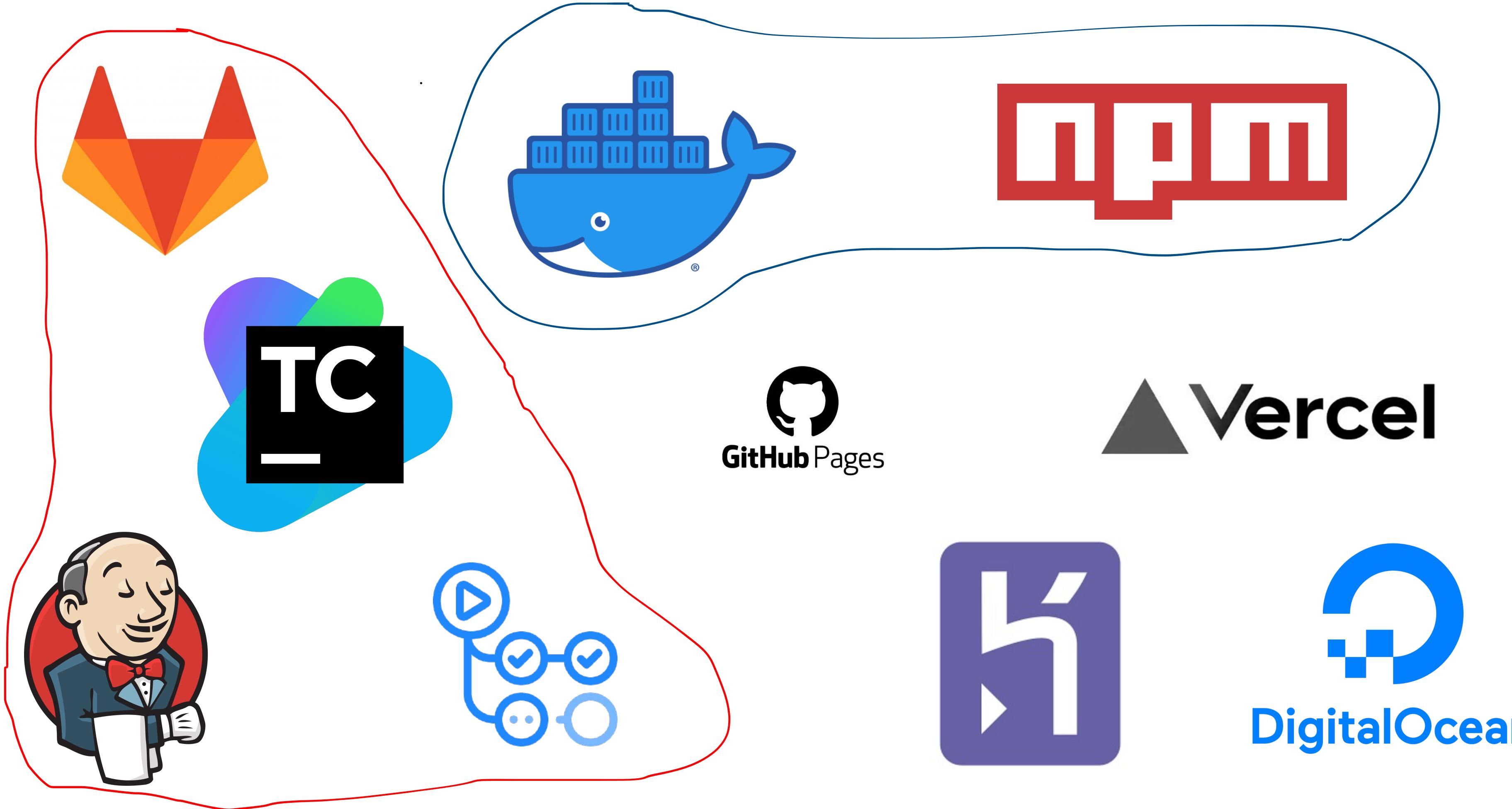
Что нам помогает автоматизировать?



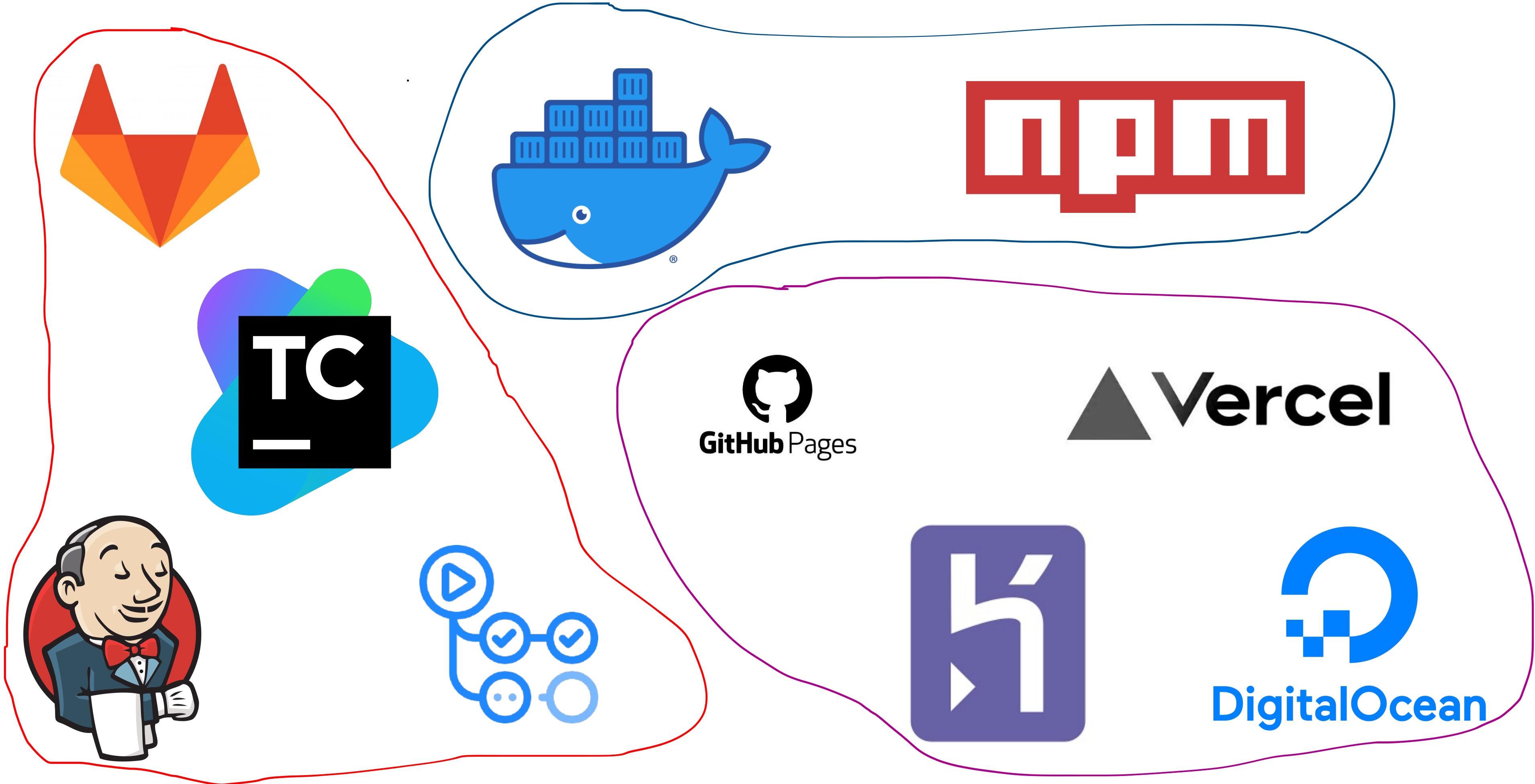
Что нам помогает автоматизировать?



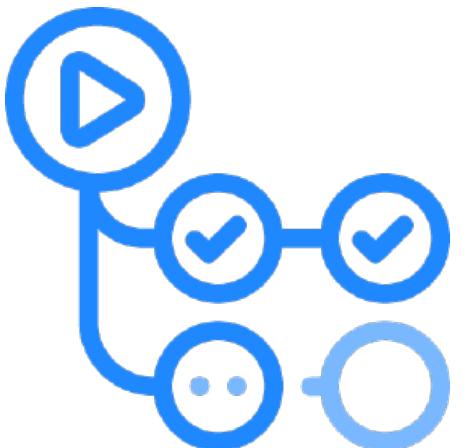
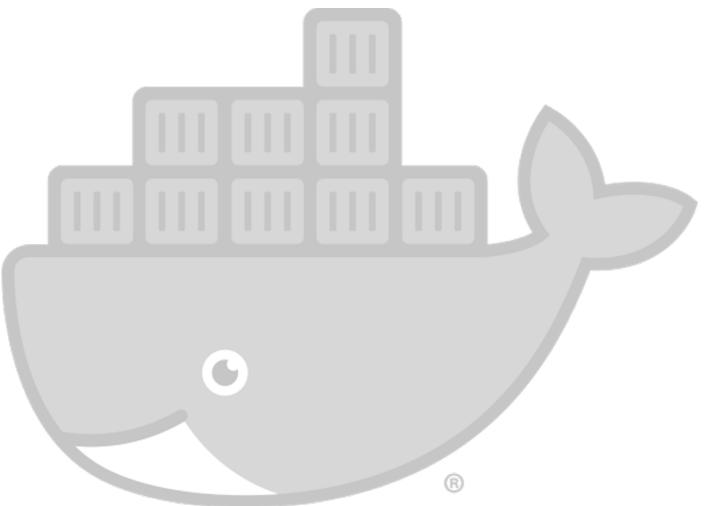
Что нам помогает автоматизировать?



Что нам помогает автоматизировать?



Что нам помогает автоматизировать?





ТИНЬКОФФ

Кто такой этот ваш докер?

Зачем нужен Docker?



IT WORKS
on my machine

Не все пользователи пользуются одной и той же сборкой компьютера, одной и той же операционной системой, одними и теми же комплектующими.

Docker делает запуск и работу приложения одинаковой везде!

Чтобы решить эту проблему, можно написать приложение, которое будет запускать другие приложения “внутри себя”. Тогда можно убедиться в том, что наше приложение ведет себя одинаково и на нашем ПК и на сервере!

Ну а ещё он изолирует наше приложение от машины, на которой оно крутится

Из чего он состоит?



Docker Image

Наше собранное приложение.
Можно сказать, что это просто
образ ОС со снимком
файловой системы со всеми
нужными файлами для запуска
приложения



Docker File

Описание нашего приложения:
какие файлы куда скопировать,
какие команды запускать



Docker Engine

Как раз то, что запускает наши
приложения

Из чего он состоит?



`FROM node`

`COPY . .`



Docker

`RUN npm install`

Наше со

Можно

образ О

файловой системы со всеми

нужными файлами для запуска

приложения

Docker Engine

раз то, что запускает наши

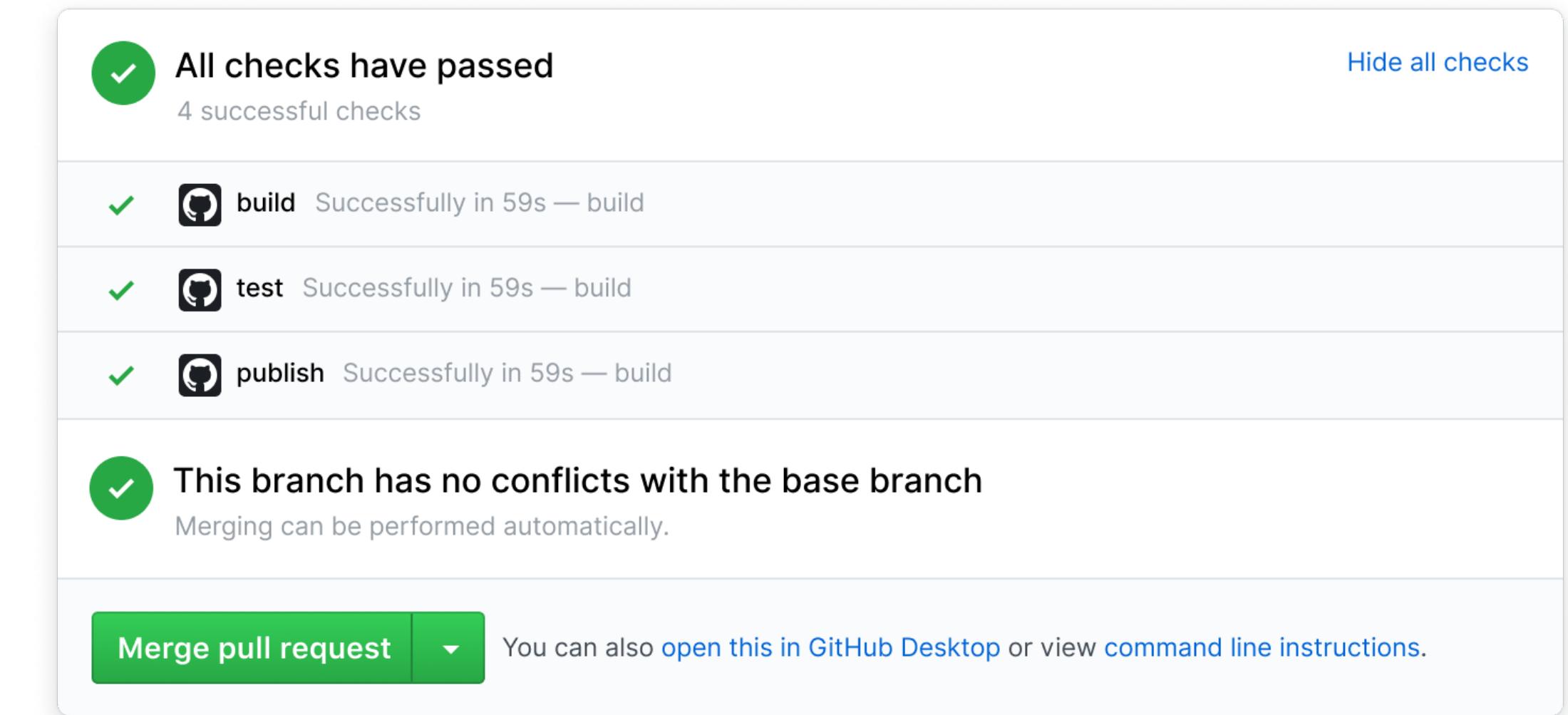
приложения



ТИНЬКОФФ

Кто такие эти ваши Github Actions?

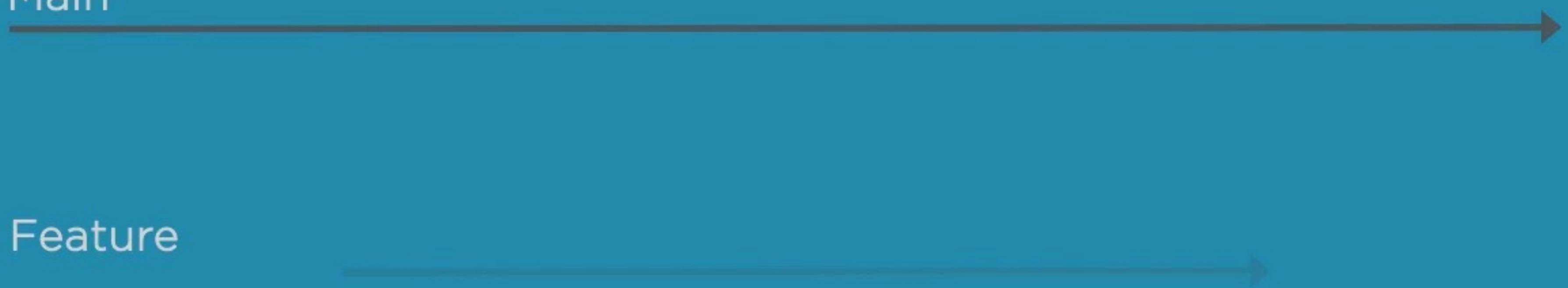
То самое,
что запускает
команды за нас



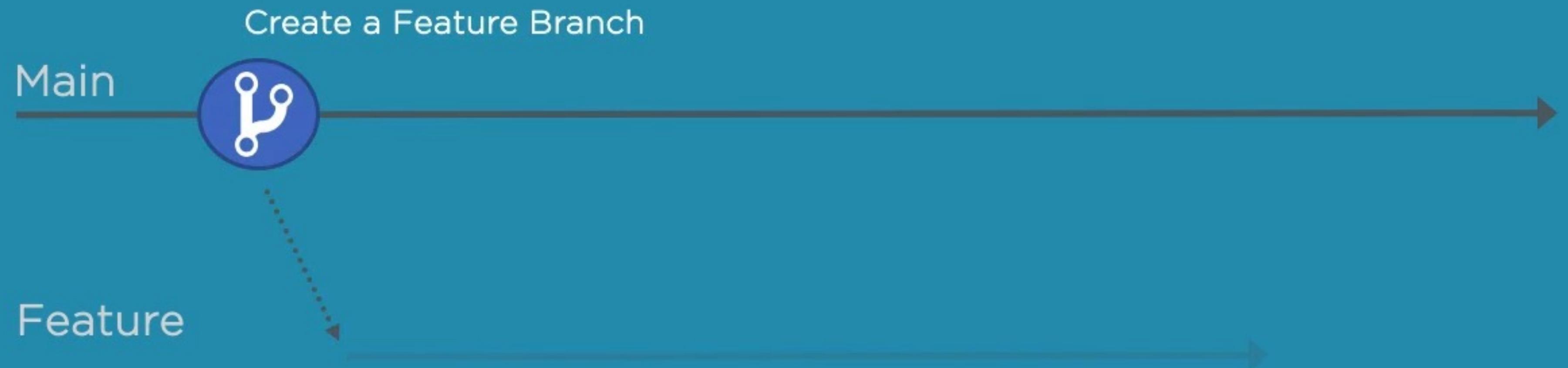
GitHub Flow

Main

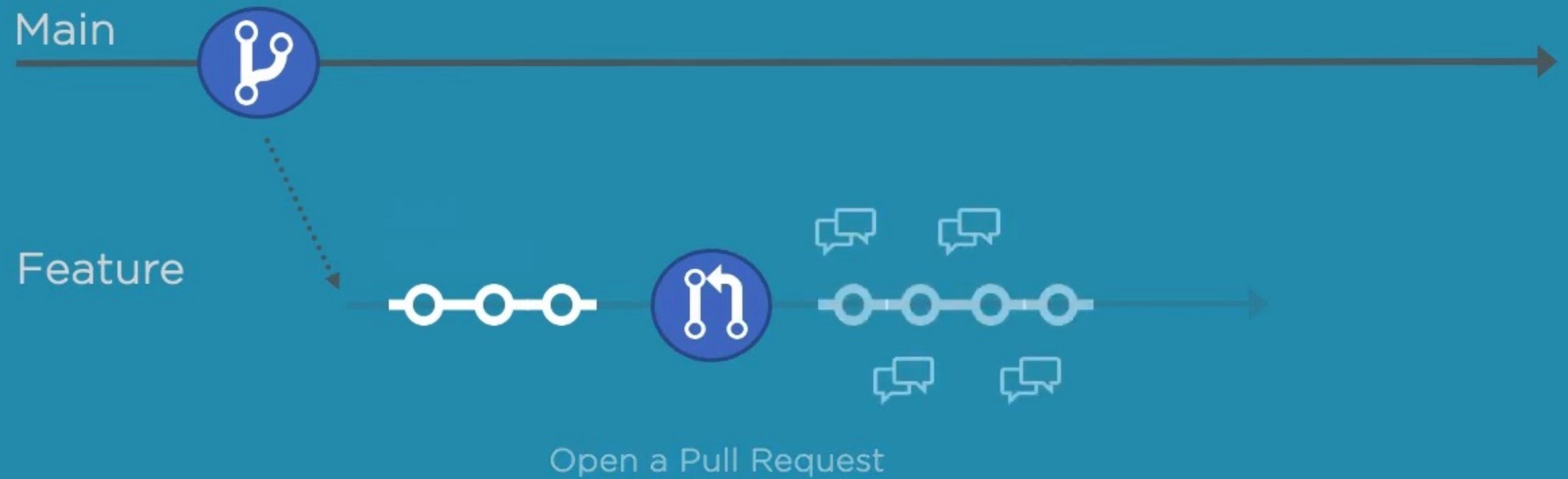
Feature



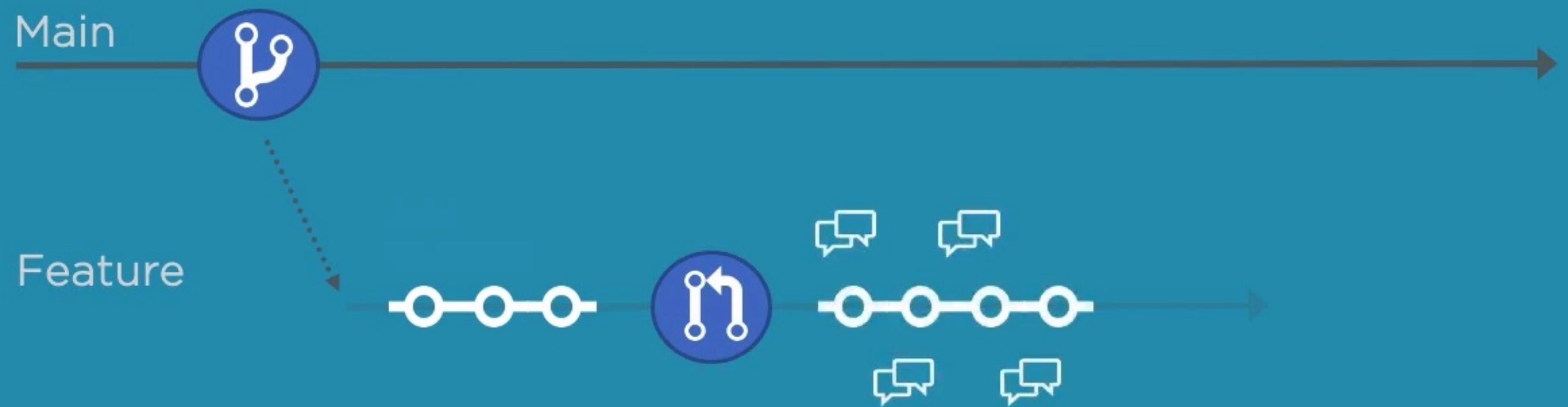
GitHub Flow



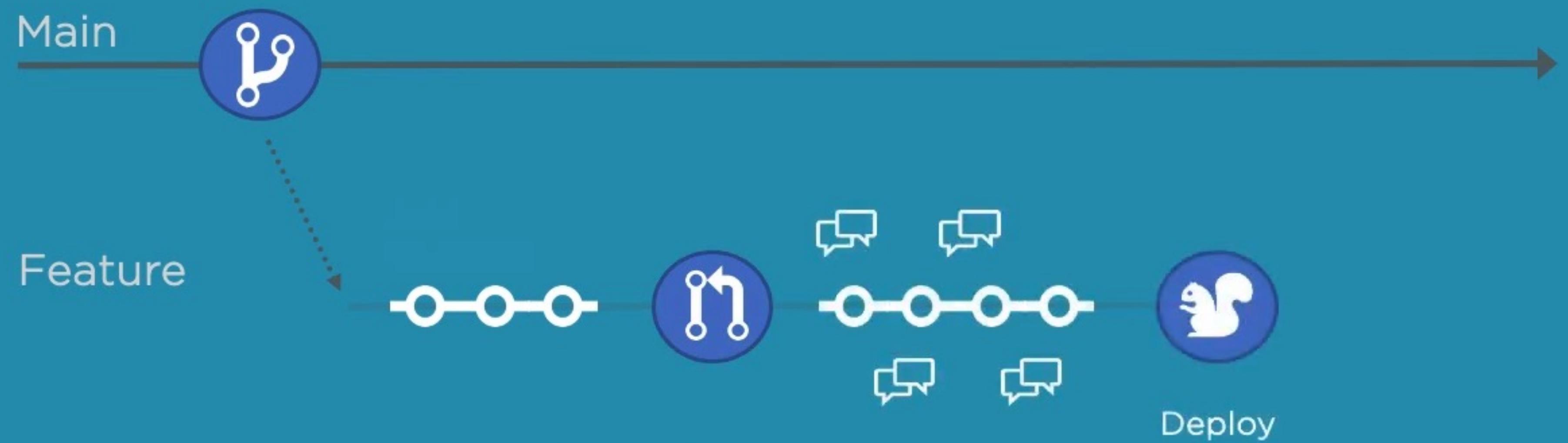
GitHub Flow



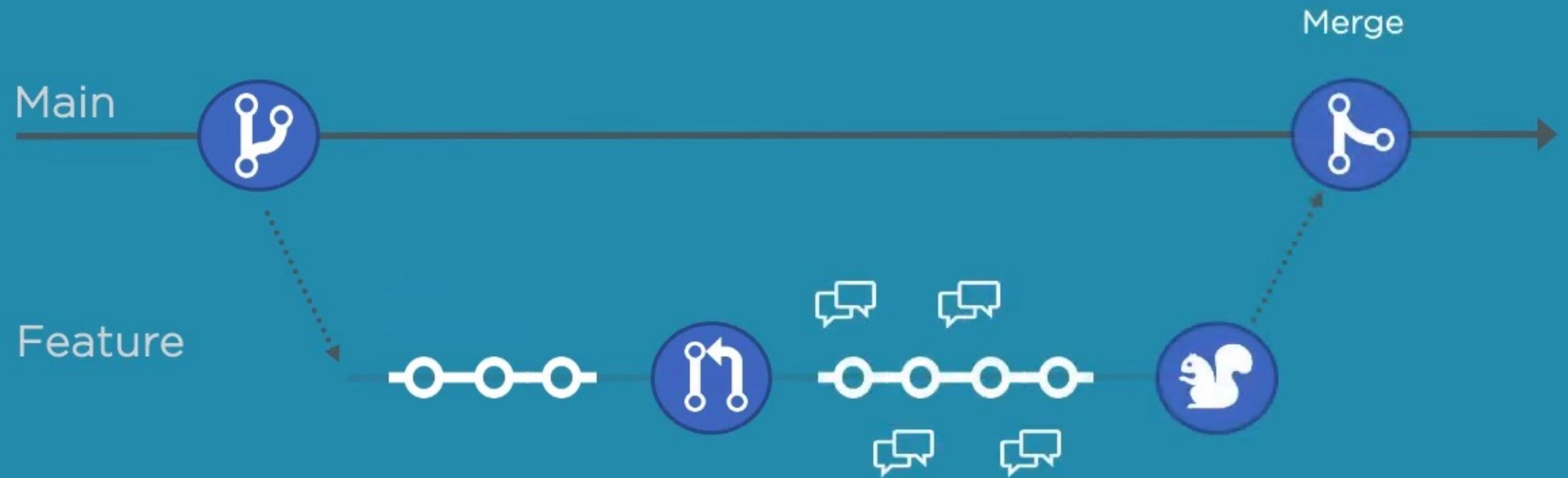
GitHub Flow



GitHub Flow

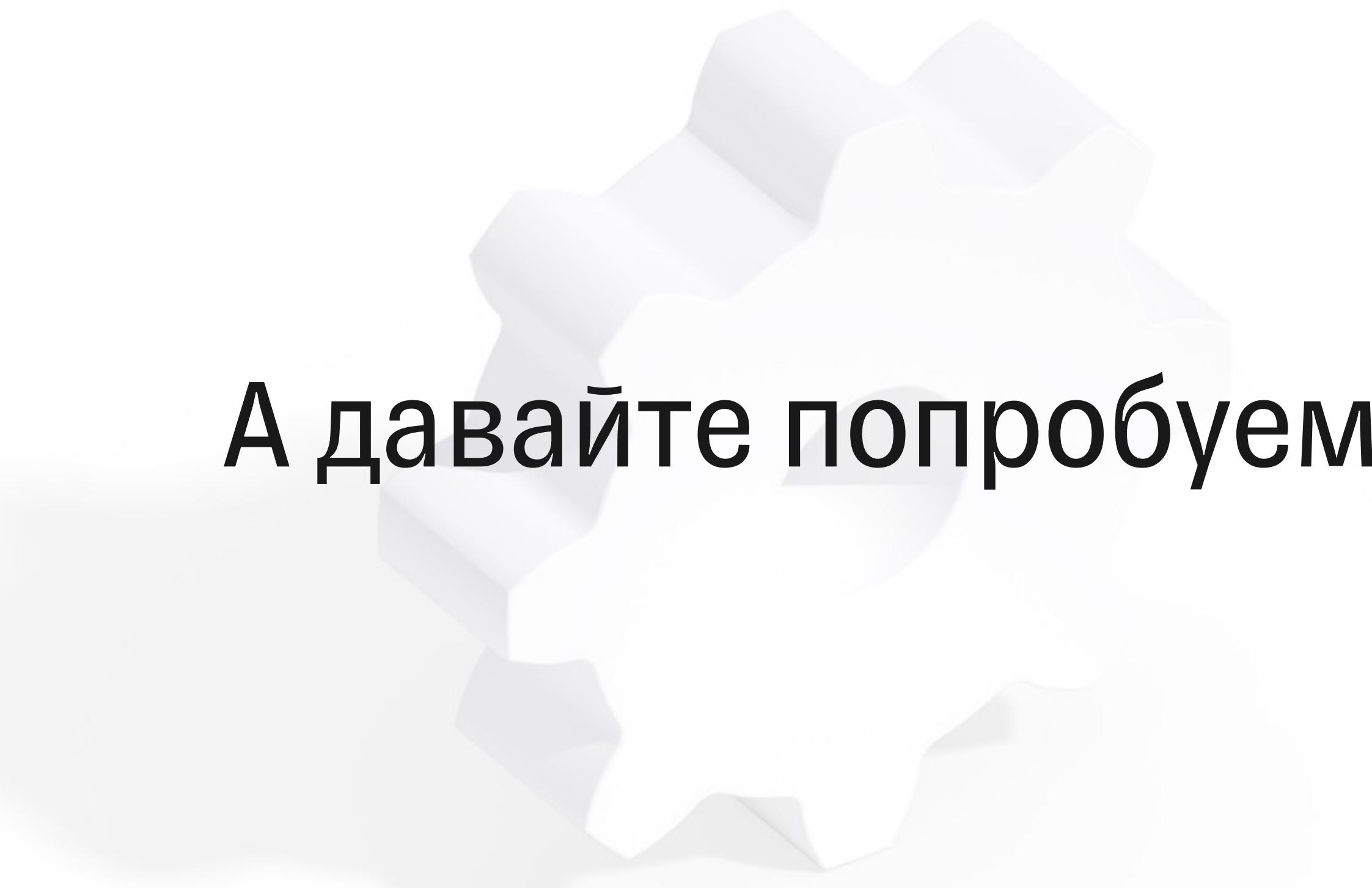


GitHub Flow



GitHub Flow with GitHub Actions





А давайте попробуем!

Новый процесс доставки кода

01

Делаем коммит в
репозиторий

Просто делаем свою
работу

02

CI/CD работает

А мы отдыхаем

03

Успокаиваем себя

Если все тесты
прошли, значит нам
нечего бояться

и можно отправлять
наш код в главную
ветку

04

Деплой деплоит

Автоматически
каждое изменение
кода

05

Profit

Приложение на проде!

The possibilities are endless



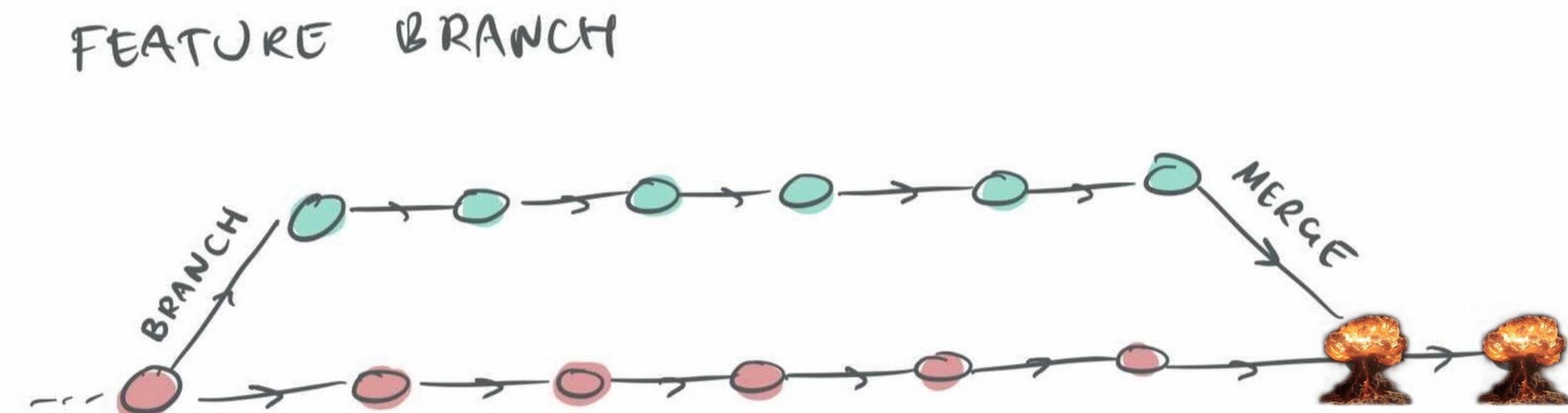
ТИНЬКОФФ

СІ в отрыве от процессов в
команде бесполезен

Почему CI — не панацея?

CI никак не избавляет нас от возможности
задеплоить “недеплоемое”

Никто не ручается за то, что разрабатывая огромные фичи по сто лет, при их мерже
мы получаем рабочий продукт, готовый ехать в прод



Trunk Based Development

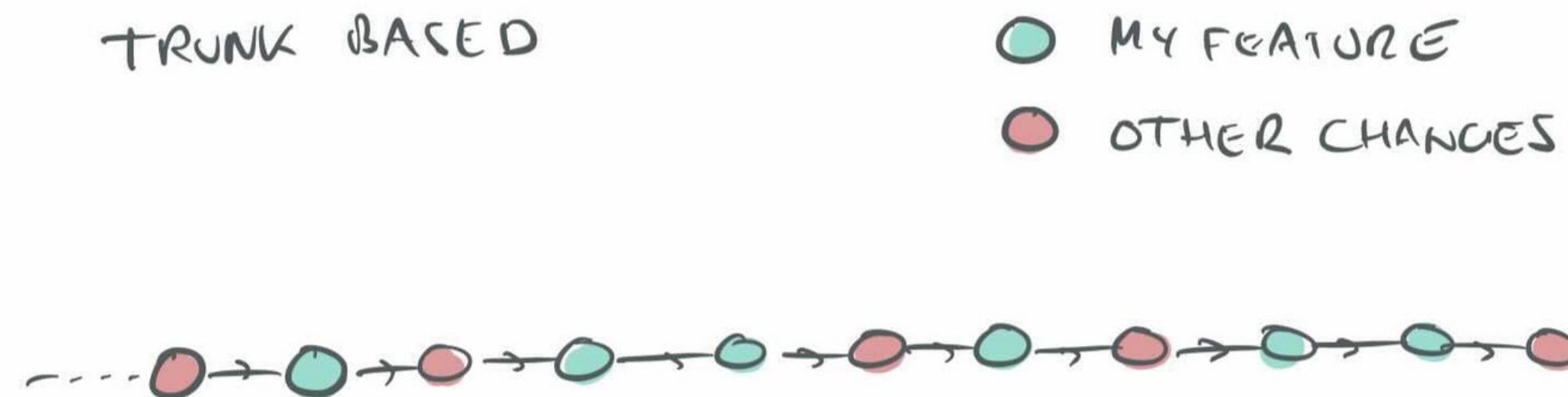
TRUNK BASED

- MY FEATURE
- OTHER CHANGES



В чем плюсы?

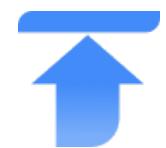
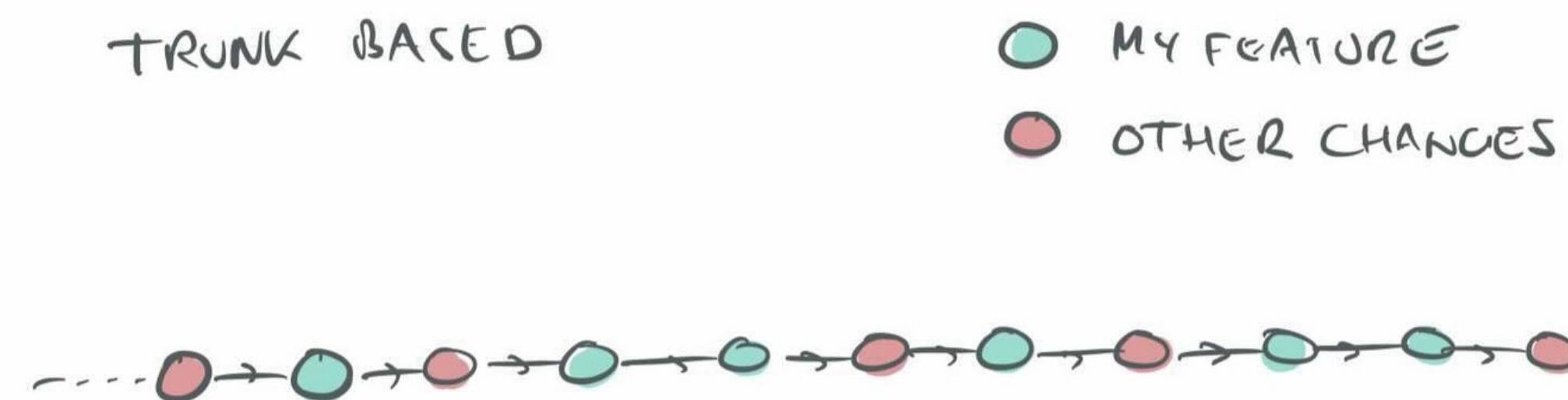
TRUNK BASED



Не копим конфликты

Ветки помимо trunk живут коммит-два, поэтому их просто смержить обратно очень быстро

В чем плюсы?



Не копим конфликтъ

Ветки помимо trunk живут коммит-два, поэтому их просто смержить обратно очень быстро



Изменения становятся супер частыми

Пропадает выпадание разработчиков на фичи на несколько месяцев, становится легче вылавливать проблемы на ранней стадии фичи

В чем плюсы?

TRUNK BASED

- MY FEATURE
- OTHER CHANGES



Не копим конфликты

Ветки помимо trunk живут коммит-два, поэтому их просто смержить обратно очень быстро



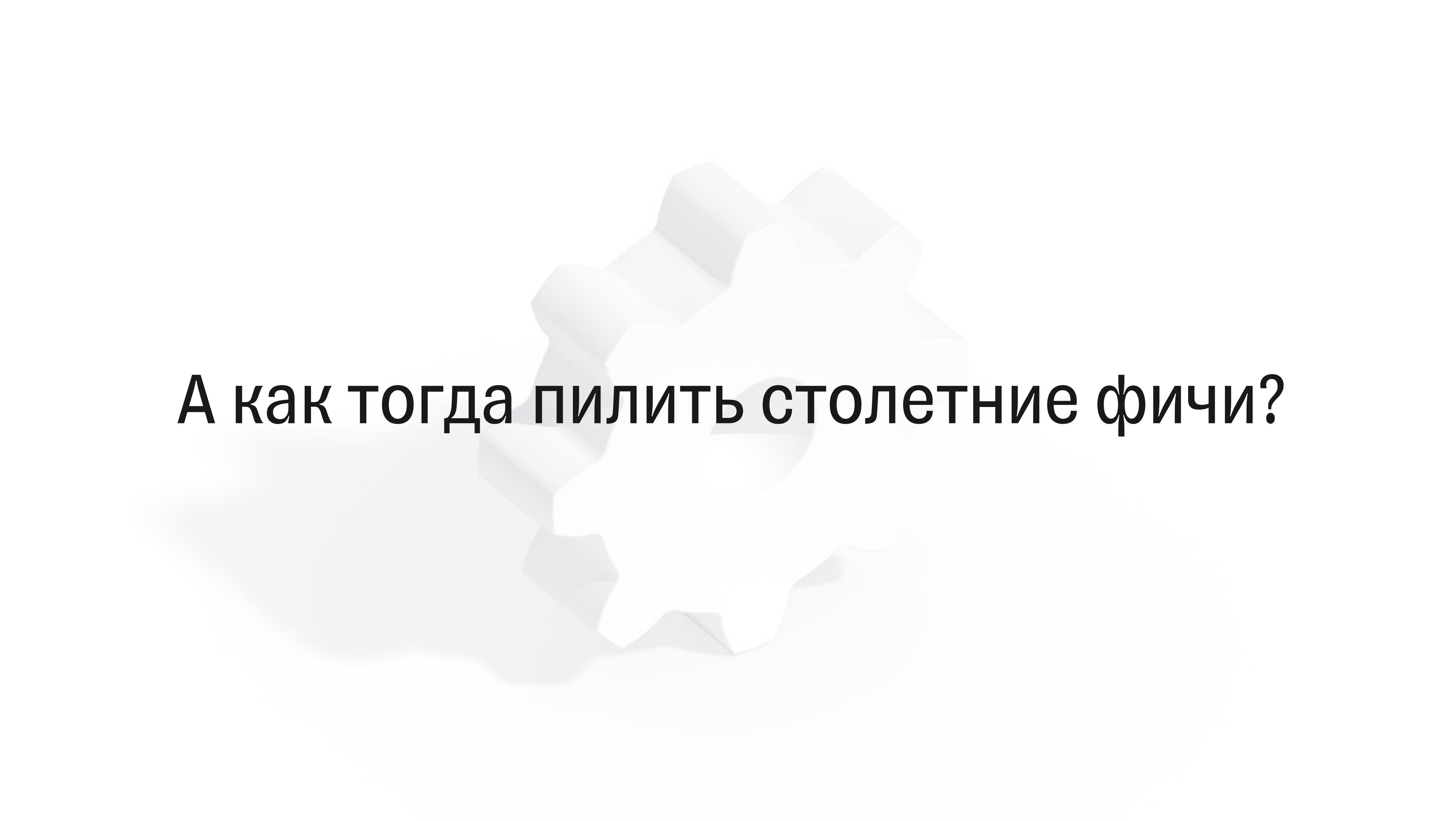
Изменения становятся супер частыми

Пропадает выпадение разработчиков на фичи на несколько месяцев, становится легче вылавливать проблемы на ранней стадии фичи



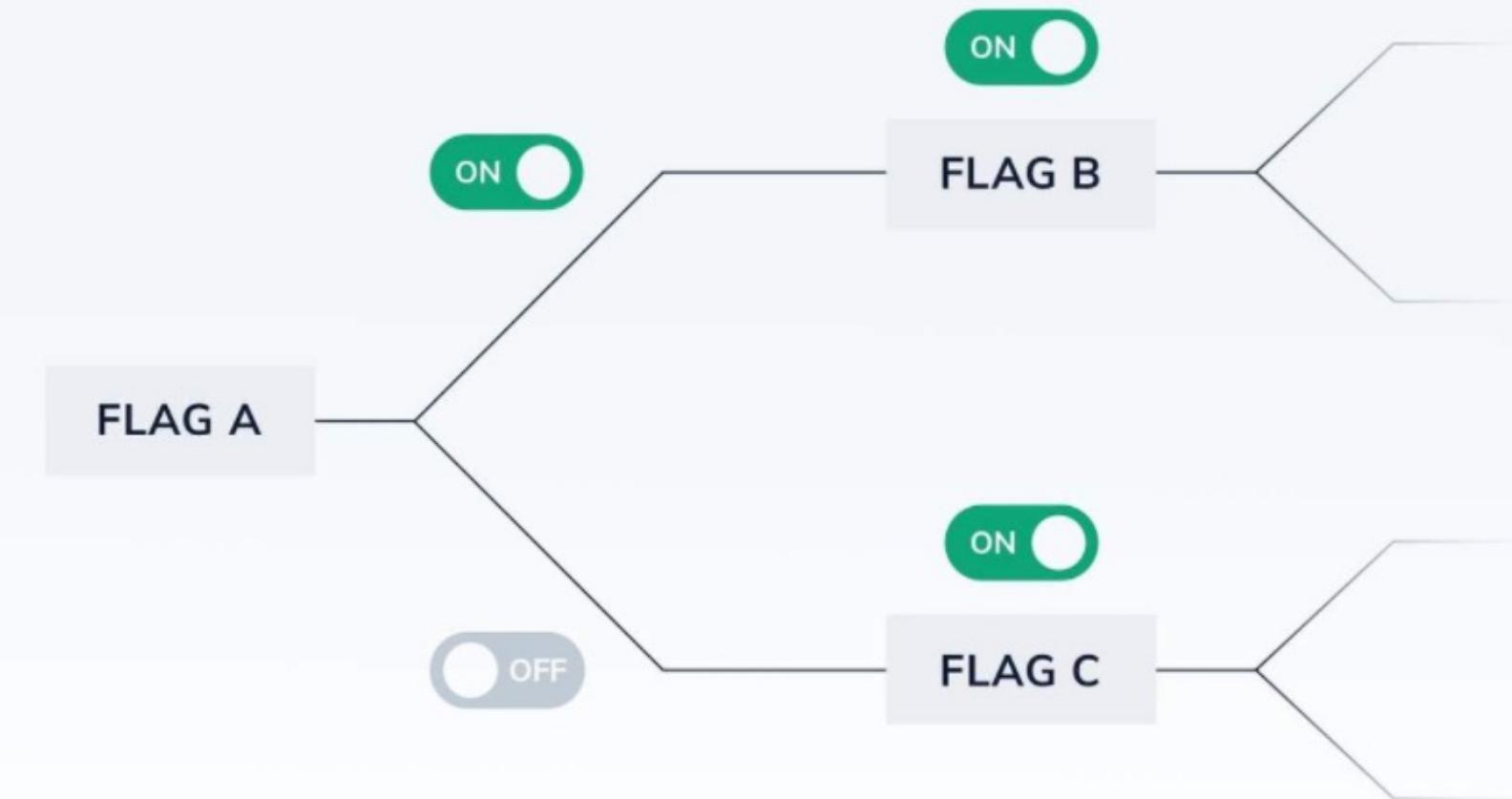
Быстрая обратная связь

Изменения быстро ломают trunk, а значит и чинят его настолько же быстро



А как тогда пилить столетние фичи?

Feature flags



Обычными if/else скрываем от пользователей

Сами же можем видеть новый функционал
по определенным условиям

Что получаем на выходе при совмещении?



Довольные пользователи

У которых приложение обновляется постоянно и работает стабильно



Довольные разработчики

Которые работают в разы быстрее и не делают операционную работу руками





ТИНЬКОФФ

Спасибо!

Вопросики?