BDUI КАК СТРАТЕГИЯ РАЗРАБОТКИ МОБИЛЬНОГО ПРИЛОЖЕНИЯ

Денис Александров

Разработчик Яндекс

О ЧЁМ ДОКЛАД

01 Почему мы выбрали BDUI

02 Как интегрировали

03 Планы на будущее

04 Выводы

ЧТО ИМЕЕМ

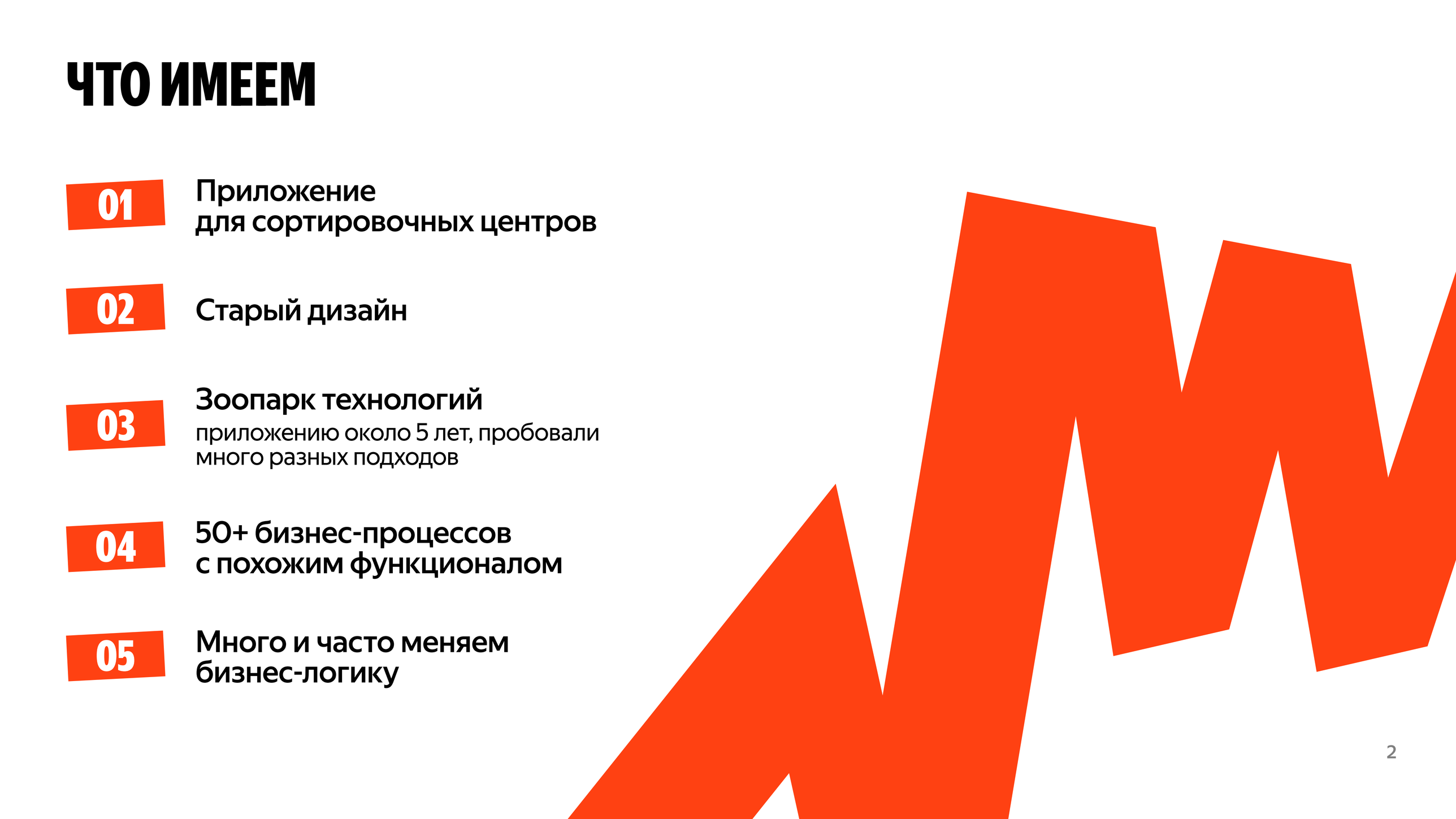
01 Приложение для сортировочных центров

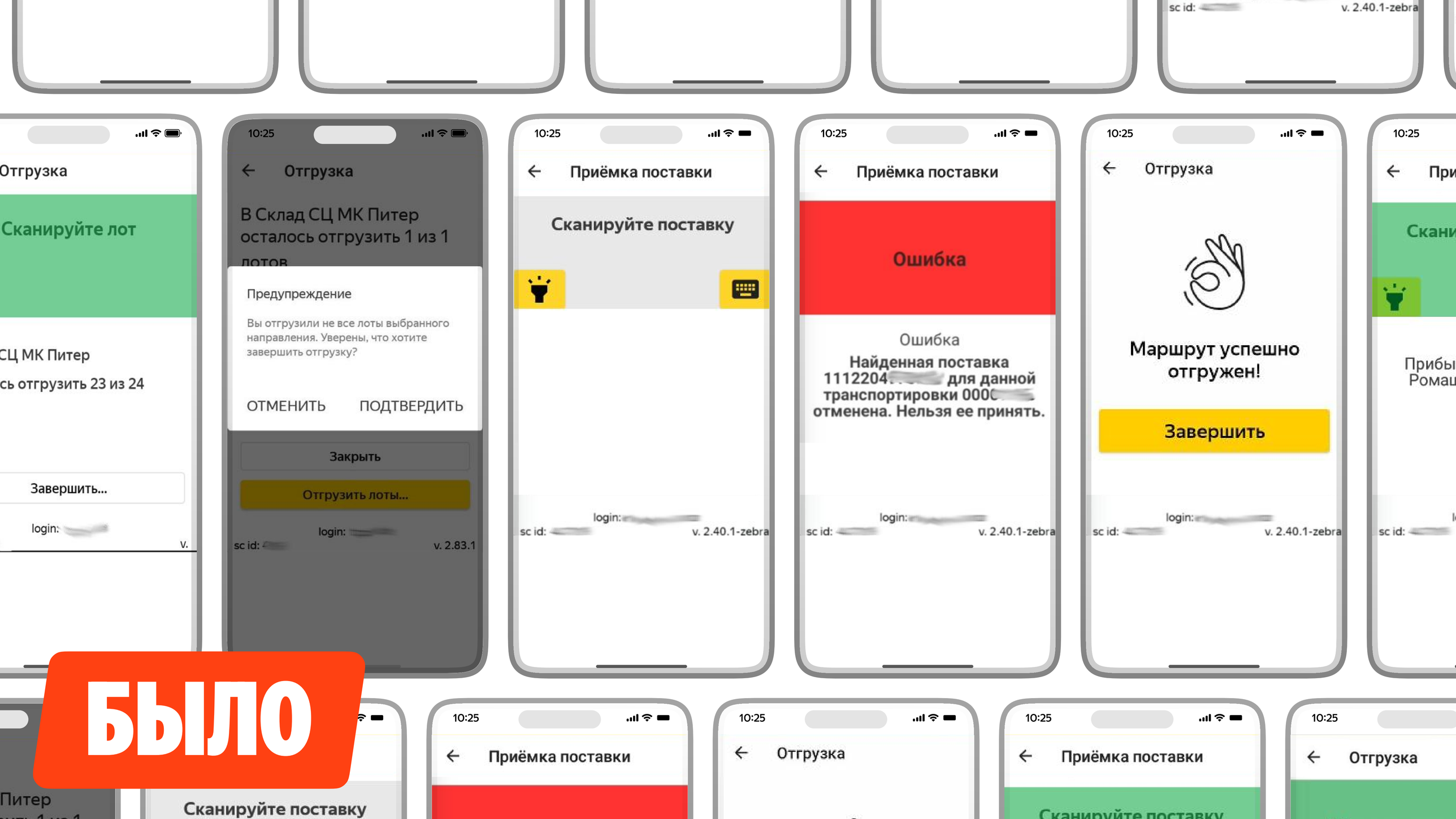
02 Старый дизайн

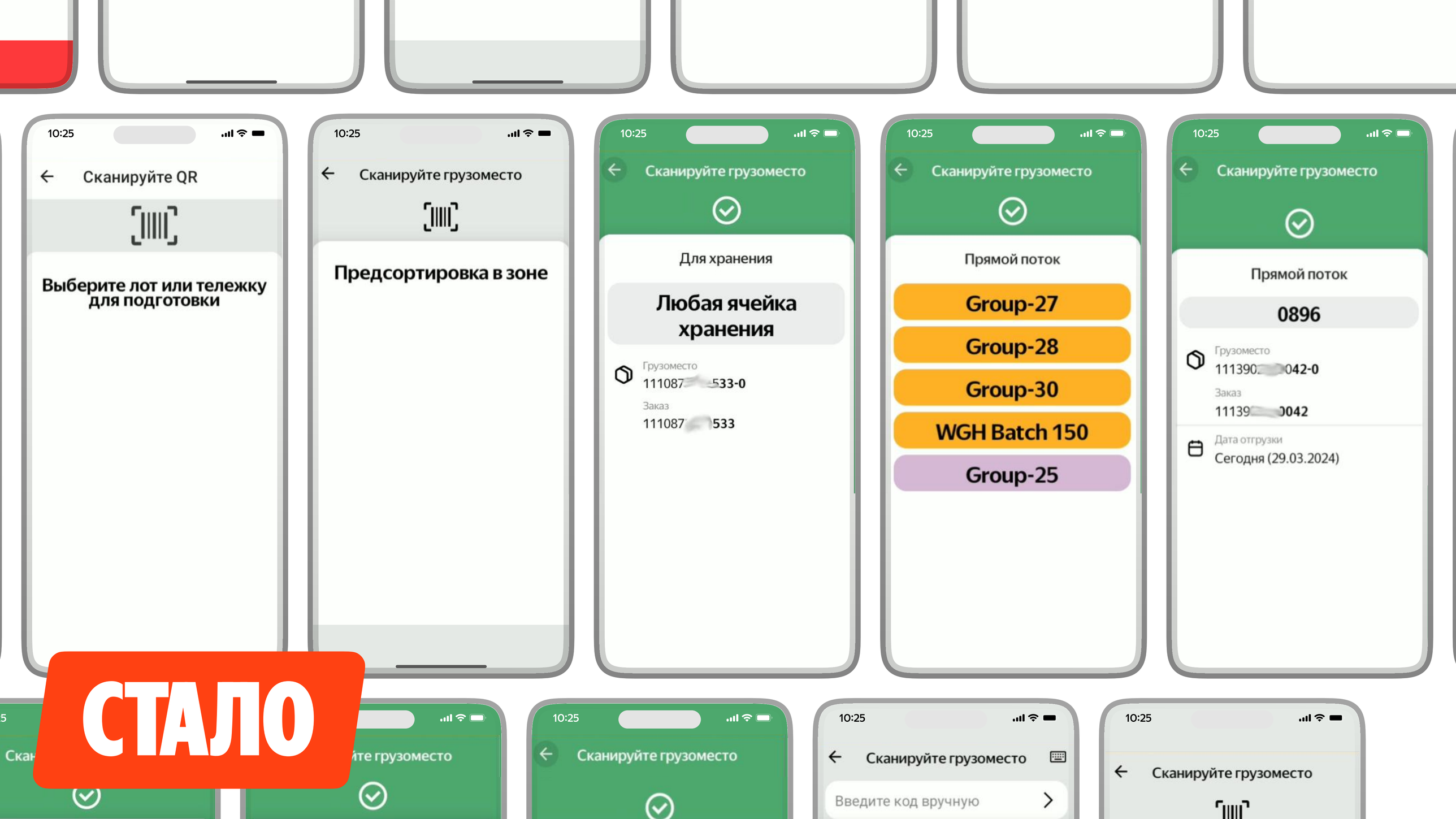
03 **Зоопарк технологий**
приложению около 5 лет, пробовали много разных подходов

04 50+ бизнес-процессов с похожим функционалом

05 Много и часто меняем бизнес-логику




Отгрузка
Сканируйте лот
СЦ МК Питер
сь отгрузить 23 из 24
Завершить...
login:
v.
10:25
Отгрузка
В Склад СЦ МК Питер
осталось отгрузить 1 из 1
лотов
Предупреждение
Вы отгрузили не все лоты выбранного направления. Уверены, что хотите завершить отгрузку?
ОТМЕНИТЬ
ПОДТВЕРДИТЬ
Закрыть
Отгрузить лоты...
login:
sc id:
v. 2.83.1
Приёмка поставки
Сканируйте поставку
login:
sc id:
v. 2.40.1-zebra
Ошибка
Ошибка
Найденная поставка 1112204 для данной транспортировки 000 отменена. Нельзя ее принять.
Маршрут успешно отгружен!
Завершить
Скани
Прибы
Рома
БЫЛО
Питер
Сканируйте поставку


10:25
Сканируйте QR
Выберите лот или тележку для подготовки
10:25
Сканируйте грузоместо
Предсортировка в зоне
10:25
Сканируйте грузоместо
Для хранения
Любая ячейка хранения
Грузоместо
111087 533-0
Заказ
111087 533
10:25
Сканируйте грузоместо
Прямой поток
Group-27
Group-28
Group-30
WGH Batch 150
Group-25
10:25
Сканируйте грузоместо
Прямой поток
0896
Грузоместо
111390 042-0
Заказ
11139 0042
Дата отгрузки
Сегодня (29.03.2024)
СТАЛО
10:25
Сканируйте грузоместо
10:25
Сканируйте грузоместо
Введите код вручную
10:25
Сканируйте грузоместо

ПОЧЕМУ BDUI

01 В приложении простая и предсказуемая верстка/переходы

02 Хочется ускорить разработку, проводить эксперименты
меньше TTM, билдов, релизов

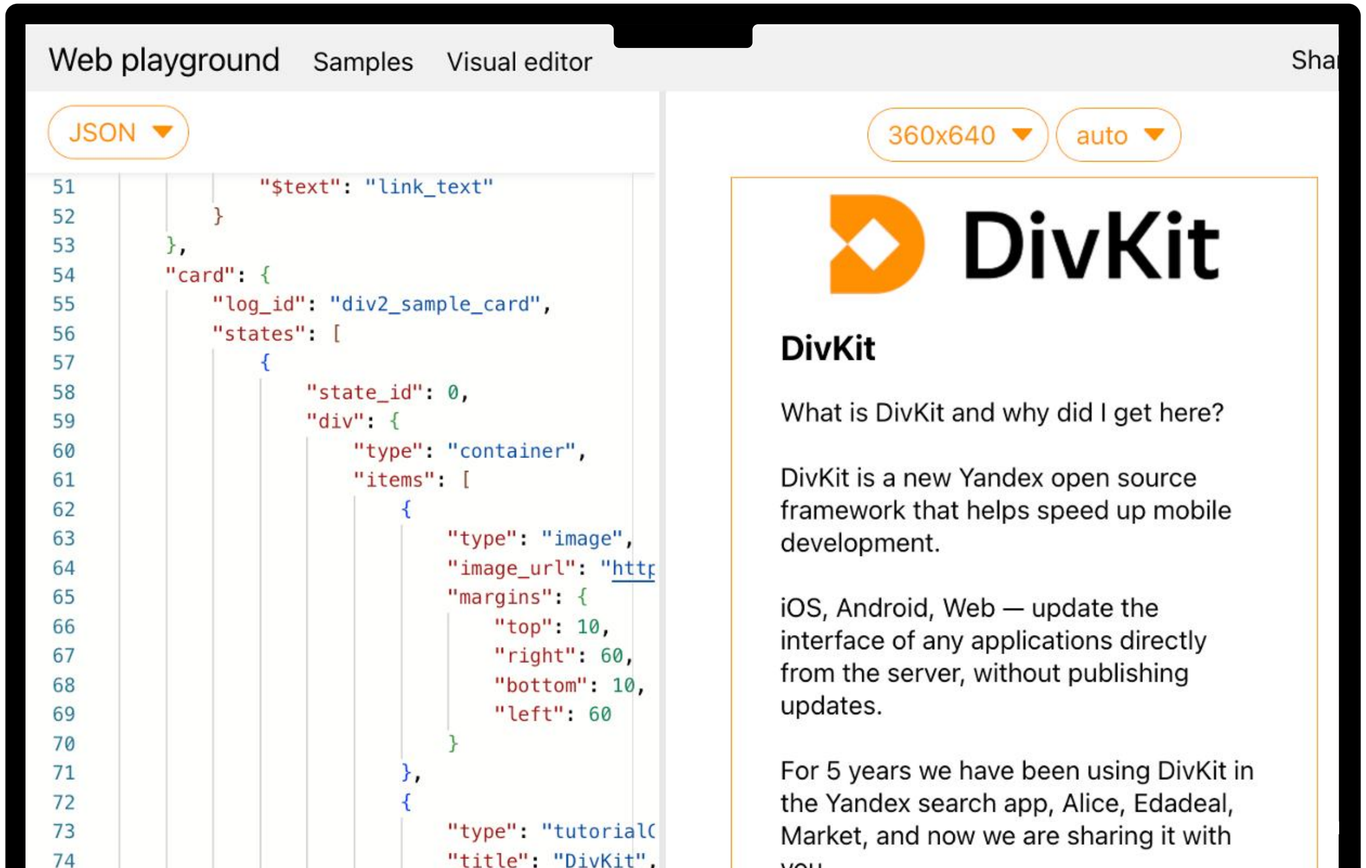
03 Потенциальная кроссплатформенная направленность
BDUI можно переиспользовать на других платформах

DIVKIT

01 Готовый UI фреймворк

02 Клиенты Android, iOS, Web

03 Космолёт по возможностям



ПОЧЕМУ МЫ НЕ ВЫБРАЛИ DIVKIT

01 **(Мы думали что) не хватало интерактивности**
мульти селект, счетчик, прогресс бар и т.п.

02 **Не регламентирует общение с бэкендом**
много допиливать

03 **Нет Kotlin + Compose**

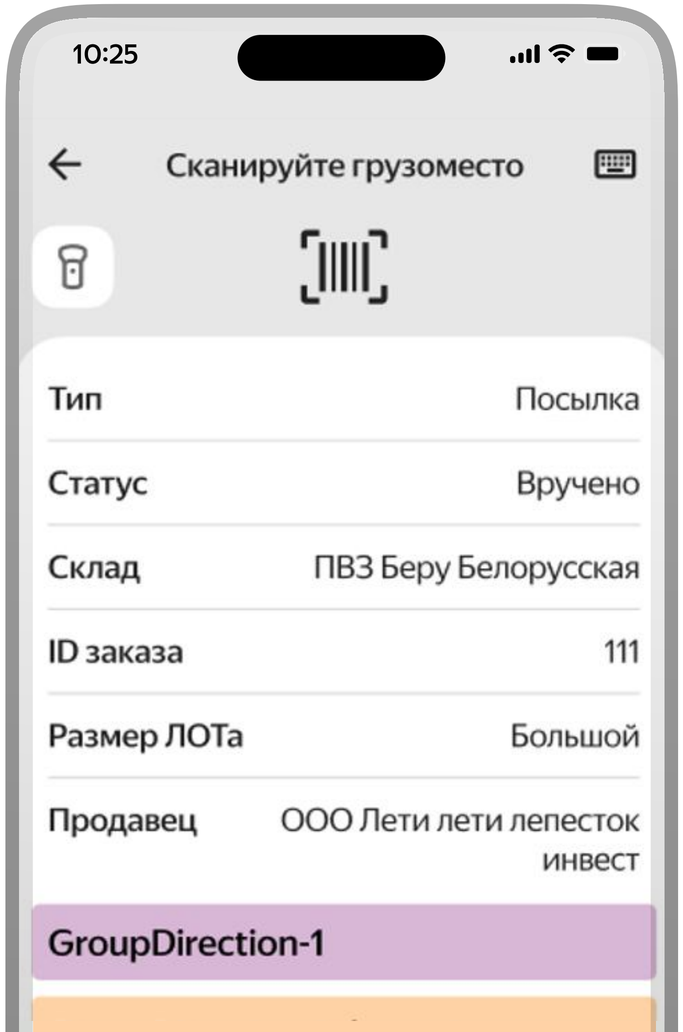
04 **Не было уверенности, что сможем быстро допилить DivKit до своих потребностей**

05 **Да че мы, не YetAnother-оиды что ли!**

ПРОРАБАТЫВАЕМ ТЕХНИЧЕСКОЕ РЕШЕНИЕ

Шаг 1

ДИЗАЙН-СИСТЕМА



Typography ТСД

Yandex Sans Display

Title 1 **Самый лучший сортировочный центр**

Title 2 **Самый лучший сортировочный центр**

Title 3 **Самый лучший сортировочный центр**

Body 1 Самый лучший сортировочный центр **Medium** **Bold**

Body 2 Самый лучший сортировочный центр **Medium** **Bold**

Caption 1 Самый лучший сортировочный центр **Medium** **Bold**

Caption 2 Самый лучший сортировочный центр **Medium** **Bold**

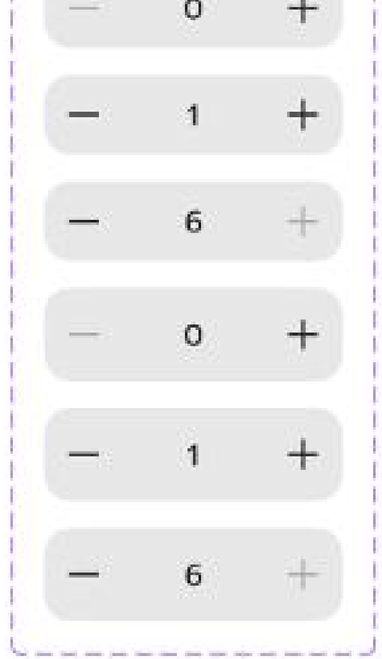
Palette ТСД

Предопределяем основные
и дополнительные цвета для ТСД и Android App

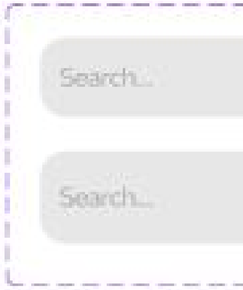
Accent colors: Yellow Main, Yellow, Red, Green, Blue

Grayscale
Палитра ахроматических цветов: Black, Coal Black, Graphite Gray, Ash Gray, Iron Gray

Backgrounds
Палитра дополнительных цветов: Pink, Purple, Mint, Yellow, Light Yellow

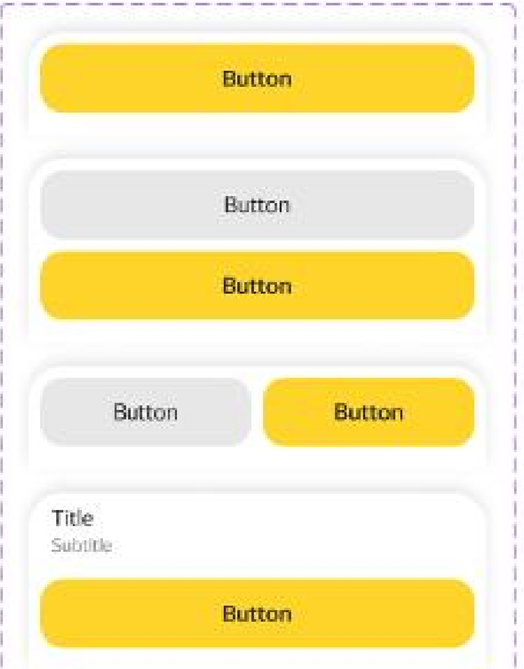


Search field

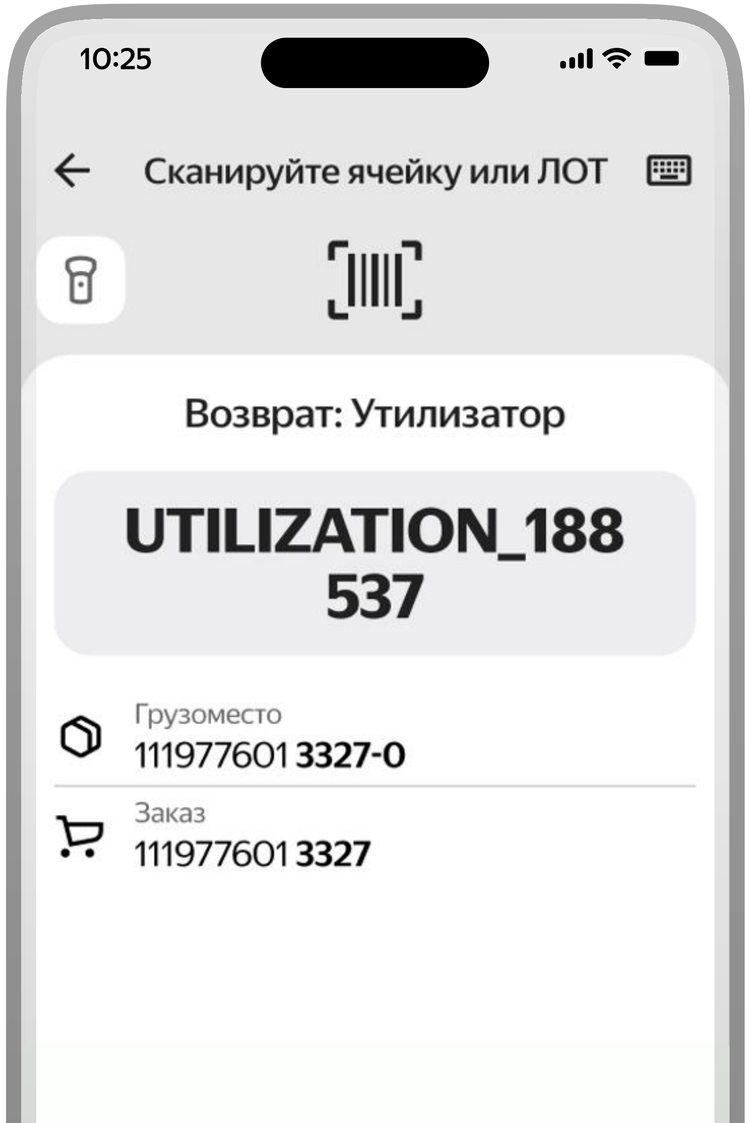


Counter

ButtonFooter



ПРИМЕР ЭКРАНА




```
{
  “type” : “Input",
  “metadata” :
    “inputRequest" : {
      ”url”: “atomic/next-scan”,
      ”type”: “Post”,
      ”variables”: [],
    },
    “backRequest”:{
      ”url”: “atomic/close”,
      ”type”: “Post”
    },
    “supportedInputs”: [“Scanner”]
    “title”: “Сканируйте Ячейку или ЛОТ”,
    “hint”: “Введите вручную”
  },
  “body”:{
    “type”: “Column”,
    “content”: [
      {
        “type”: “Text”,
        “text”: “Возврат: Утилизатор”,
        “font-size”: 27,
        “text_color”: “#FF000000”,
        “font_weight”: “W700”,
        “paddings”: {“top”: 4, “bottom”: 4, “start”: 4, “end”: 4}
       },
```

Шаг 2

ATOMICOPS

01 Kotlin Multiplatfom

02 KotlinX serialization
Backend + Frontend

03 Coroutines, Ktor
Frontend only

04 Compose multiplatform
Android, Web?

05 Общие модели, протокол общения и клиент — AtomicOps (AO)

Шаг 3

BACKEND FOR FRONTEND

Клиент
Дай моё состояние

Бэк
Покажи камеру и «Сканируй рабочую зону»

Клиент
Я отсканировал камерой код «308»

Бэк
Покажи список кнопок: «Предсортировка», «Сортировка»

Клиент
Я нажал «Предсортировка»

Бэк
Покажи камеру и «Сканируйте отправление»

Клиент
Я отсканировал «123»

Бэк
Покажи камеру и «Лот 123, положите в секцию 12, Повторите сканирование»

Клиент
Я набрал в камере код «1234»

Бэк
Покажи текст «Код не распознан» и доступные операции выбрать код 12345», убрать в аномалии

Клиент
Я нажал «Назад»

Бэк
Покажи список кнопок: «Предсортировка», «Сортировка»

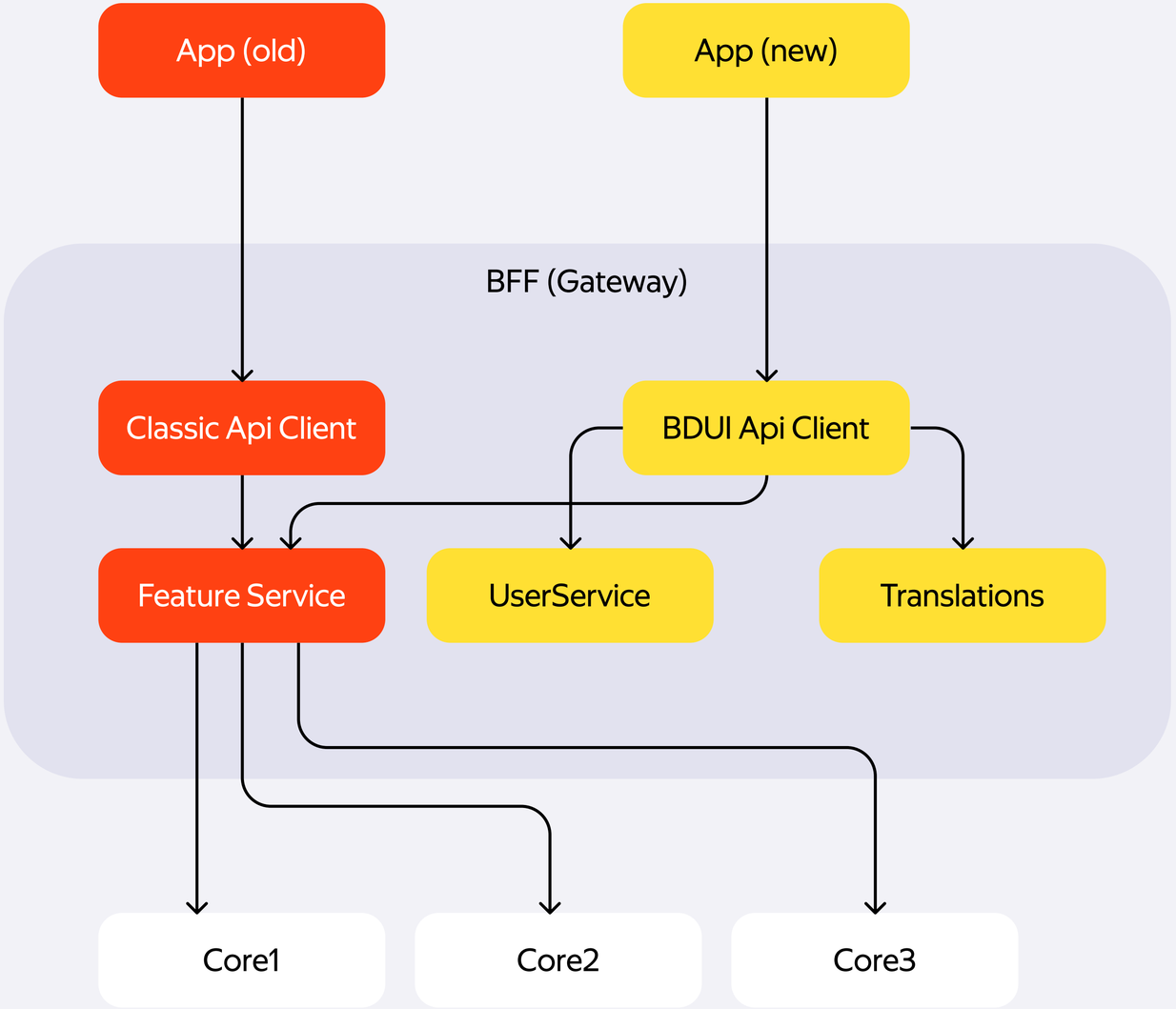
АРХИТЕКТУРА БЭКЕНДА

01 Бэкенд отличается от старого мапперами BusinessDTO – ScreenDTO

02 Хранение состояния ниже BFF, перевод – на BFF

03 BDUI Api Client пишет фронтенд

Там по сути только верстка



АРХИТЕКТУРА ЗАПРОСА

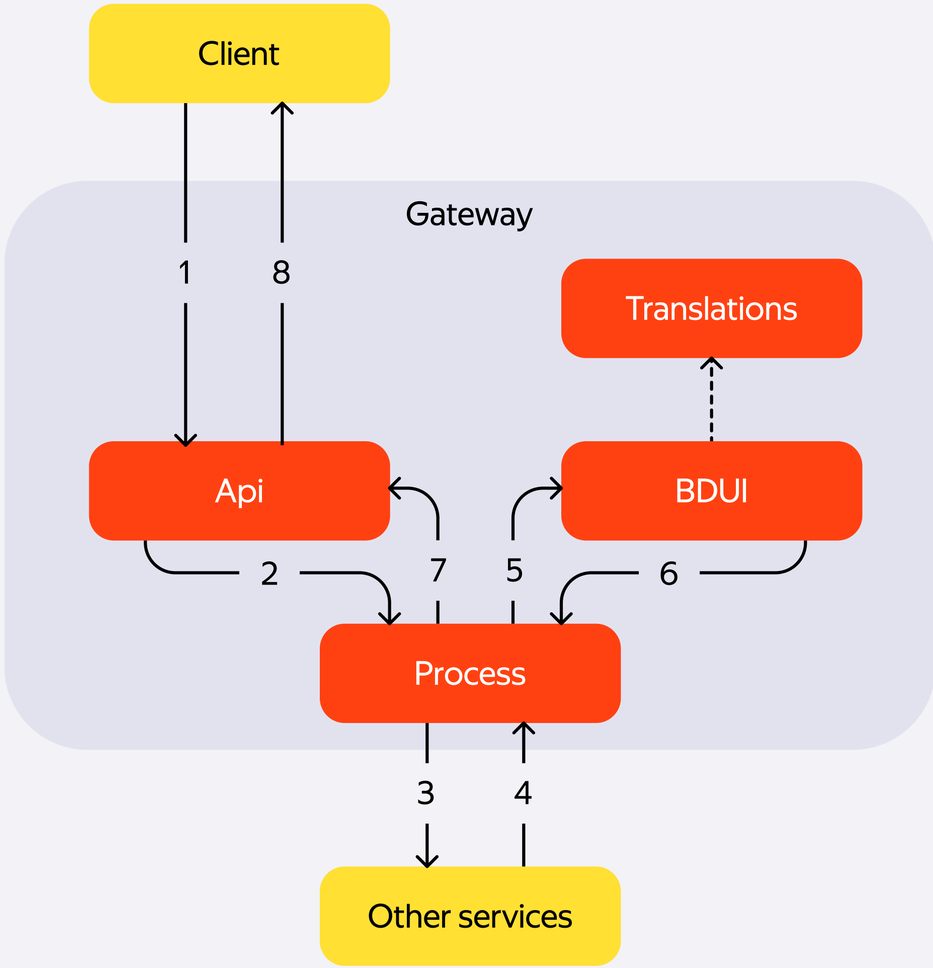
1 Клиент стучится в апи

2 Апи идет в блок процесса

3,4 Процесс ходит в другие сервисы, собирает ответы

5,6 Мапим ответ в верстку
используя кэш переводов

7,8 Отдаем ответ клиенту



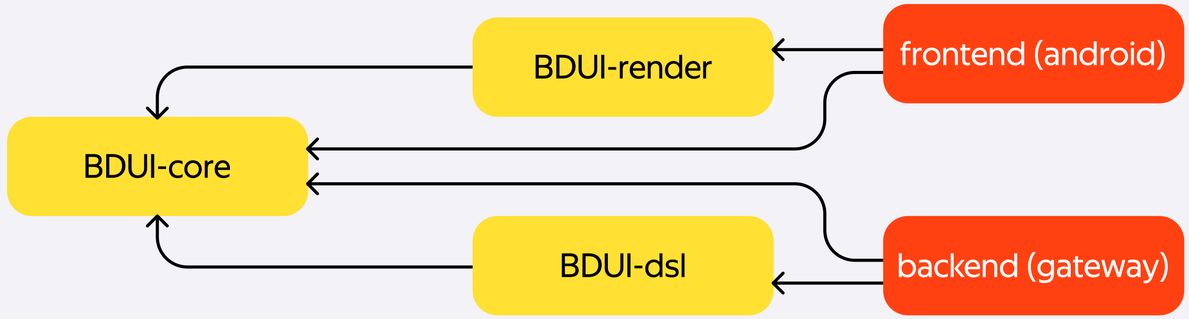
Шаг 4

АРХИТЕКТУРА

BDUI-core
модели, сериализация (KMP, kotlinx serialization)

BDUI-dsl
дизайн система, экстеншены,
билдер экранов и т.п. (Core)

BDUI-render
Библиотека отрисовки и интерактива BDUI
(Core, Coroutines, Compose Multiplatform)



КАК ПЕРЕДАЁМ ЗАПРОС

```
{
  "type": "TextButton",
  "text": ”Есть еще один ШК",
  "request": {
  "url": "atomic-ops/sort/one-more-code",
  "variables": [
      {"type": ”string”, “name”: ”operation","value": “sort” },
      {"type": ”int”, “name”: ”firstScanId","value": "42” },
    ]
  }
}
```

Шаг 5

СТРАТЕГИЯ

01 Пишем UI фреймворк и модели

02 Запускаем офлайн демо

03 Запускаем онлайн демо

04 Переводим первый процесс на BDUI

05 В приложении делаем механизм переходов

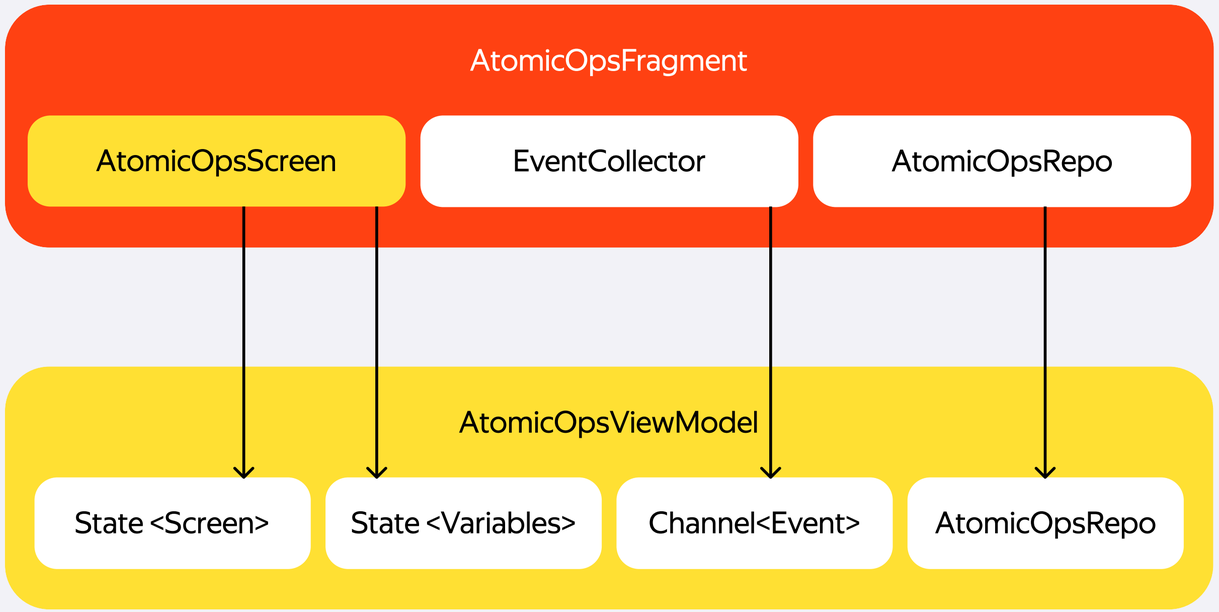
АРХИТЕКТУРА МОБИЛКИ

01 BDUI не знает про приложение и бизнес процессы

02 Приложение хостит экран в отдельном модуле и фрагменте/Compose скрине

03 Зависимости предоставляем имплементацией и инжектом репозитория

04 AtomicOps включает в себя все необходимое для автономной работы



ИТОГО

01 Прекондишны – переводы на бэкенде, BFF

02 Делаем библиотеки моделей и рендера
с VM «под ключ» для BFF

03 Переносим на AtomicOps инфоскан
большой процесс в приложении

04 Запускаем, смотрим метрики,
плавно переносим все остальное

ЗАПУСК ОСЕНЬ 2023

01 **Запуск в проде**

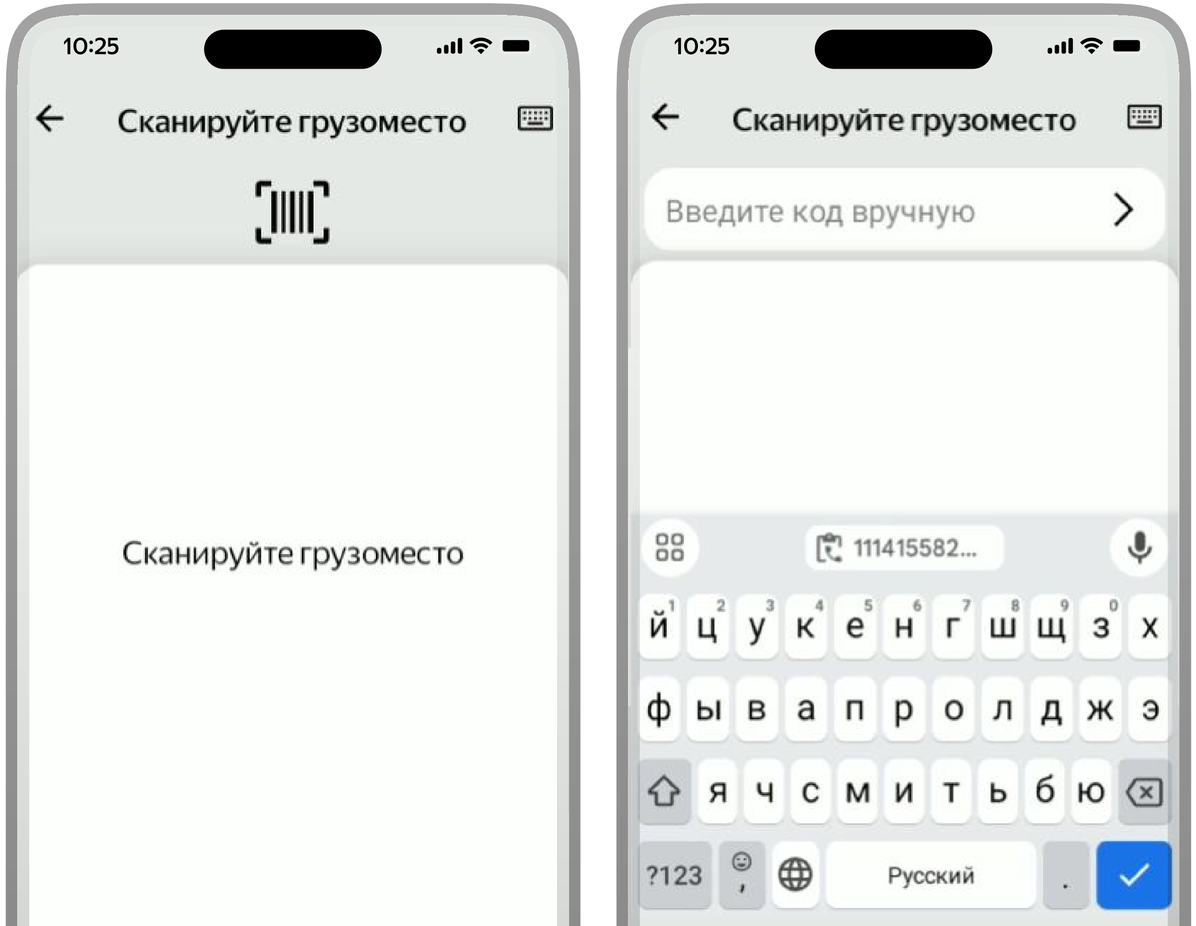
02 **Фронтенд «летает»**
парсинг + отрисовка меньше 1 кадра

03 **Клиенты «не выкупают», что это BDUI**

04 **Бэкенд не чувствует нагрузки**

05 **...И новые требования**

- Надо много продумывать интерактив (диалоги)
- Нужен Flutter



ПЕРЕЕЗД НА DIVKIT ФОРМАТ

01 Умеет все, что нам нужно в будущем

02 Писать FlutterUI для дивкита проще – есть документация

03 Мы узнали, как обойти все блокеры раньше

04 Инвестировать в opensource выгоднее
оставив специфику продукта кастомной

СТРАТЕГИЯ 2024

01 Реализуем весь текущий функционал в формате DivKit

примерно 2 месяца

02 Пишем клиент DivKit-а под флаттер

03 Сходимся в одном формате и бэкенде

04 Переносим бэкенд на DivKit формат

НОВАЯ АРХИТЕКТУРА

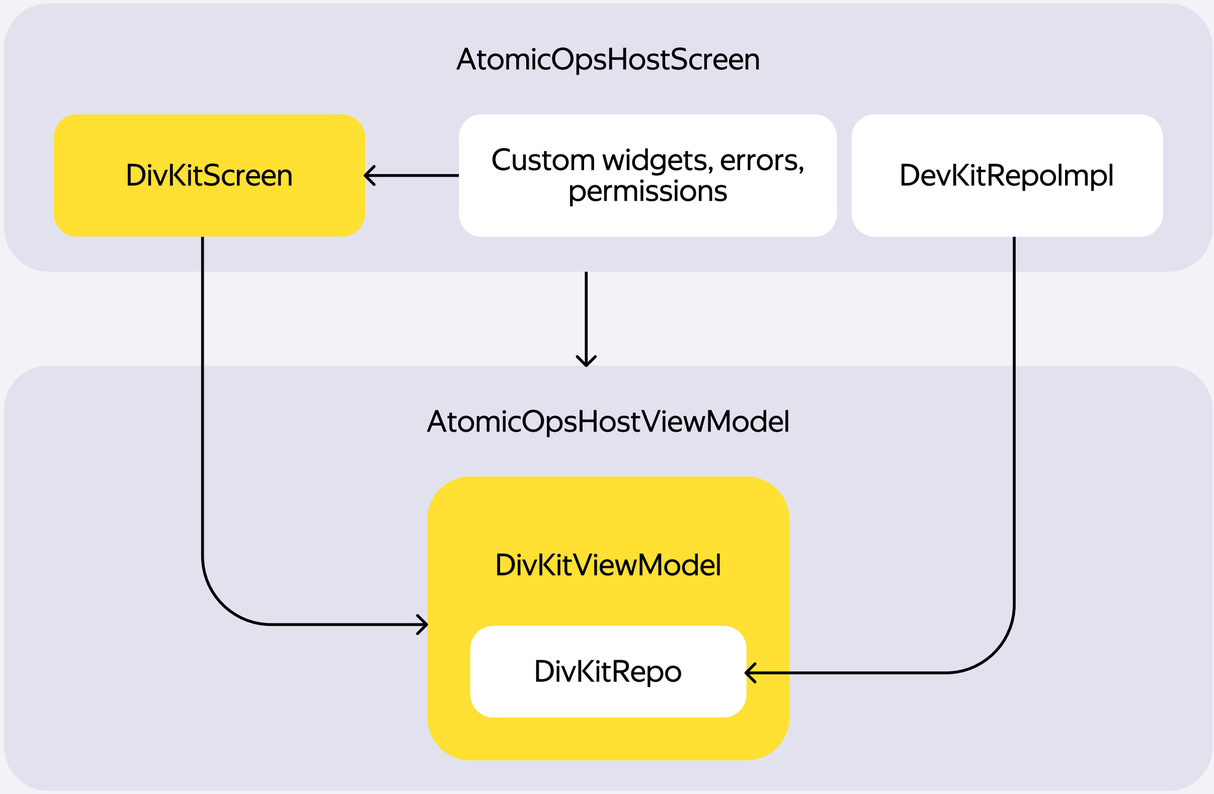
AtomicOpsHostScreen
Composable экран для отрисовки всего нашего приложения

DivKitScreen
рендер DivKit-а

AtomicOpsHostViewModel
VM для загрузки данных, шаблонов, обработки кастомных события и т. п.

DivKitViewModel
стейт машина дивкита

DivKitRepo
провайдер зависимостей для VM



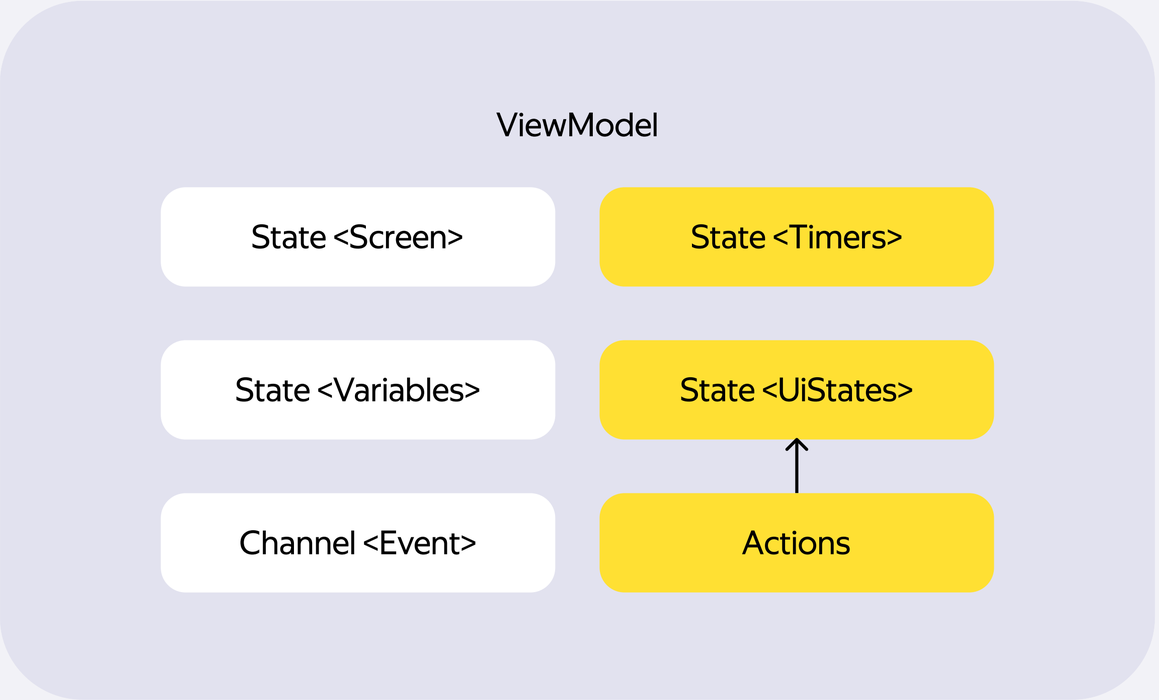
ЧТО ПОЯВИЛОСЬ ВО VIEWMODEL

01 Таймеры на старте приложения

02 **Машина состояний дивкита**
отображение части контента

03 **На старте возможны экшны**
таймеры, подсветка экранов, звуки и т. п.

04 **Передаваемые переменные стали обязательными**
раньше могли доинициализироваться в процессе работы



ЧТО ПОЛУЧИЛОСЬ

01 **Проект усложнился**
зато сильно документирован дивкитом

02 **Визуально и интерактивно изменений нет**

03 **Часть вещей из старых экранов переехала на рельсы дивкита еще до выпила**

04 **JSON-ы выросли на 10-15 процентов**

Другая команда реализовала DivKit
на библиотеке Flutter

DIVKIT 0.1.3

библиотека для создания карточек
из элементов JSON-верстки
для приложений и сайтов

ЧТО ДАЛЬШЕ?

01 Выпиливаем старую верстку из АО

02 Отделяем Divkit от AtomicOps

03 Метрики!

04 Сильно упарываемся в шаблоны, чтобы экономить трафик

но надо мерить!

05 Встраиваем WebView

РЕТРОСПЕКТИВА

01 **Долго стартовали**
BFF, редизайн, переводы

02 **Все равно пришлось переписать :-)**

03 **Много нервов на релизе**
хотя различий со старым процессом немного

04 **Все мобильщики теперь бэкендеры**

ВЫВОДЫ

01

BackendForFrontend — обязательная штука

отделяем клиентские процессы
от доменнного уровня

02

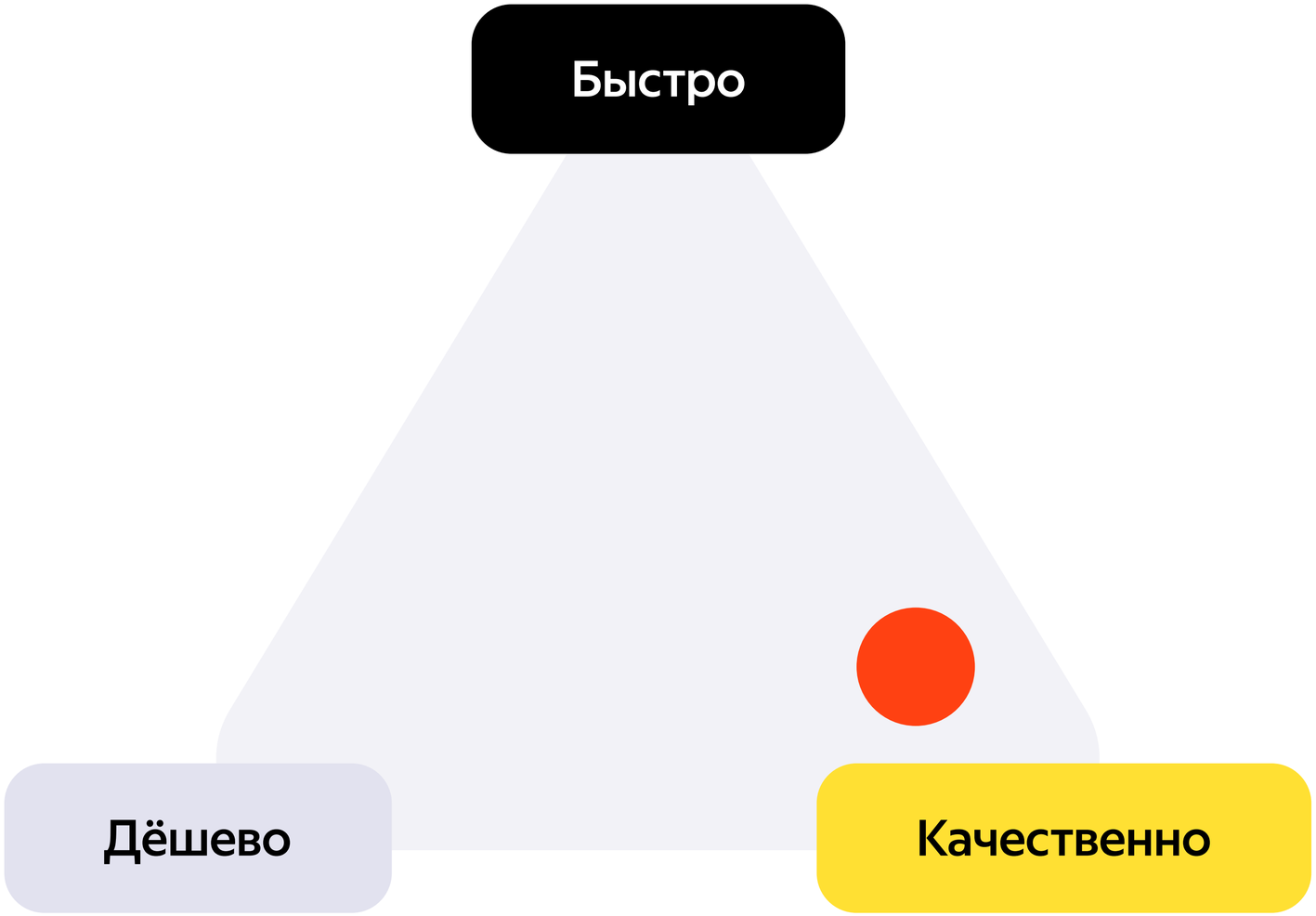
Frontend based API

не знаем про внутреннюю архитектуру
продукта, говорим на языке бизнес процесса

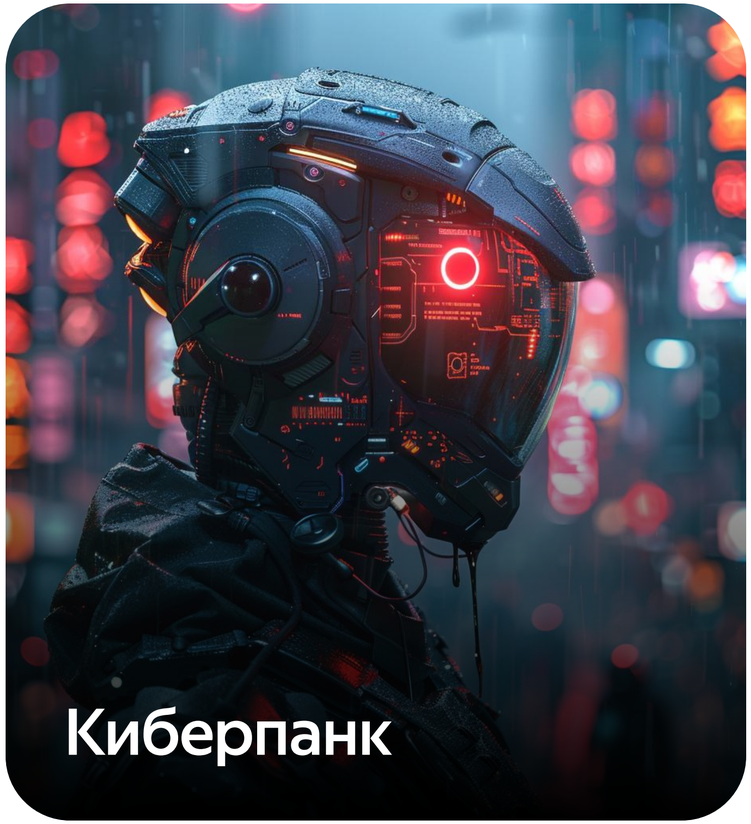
03

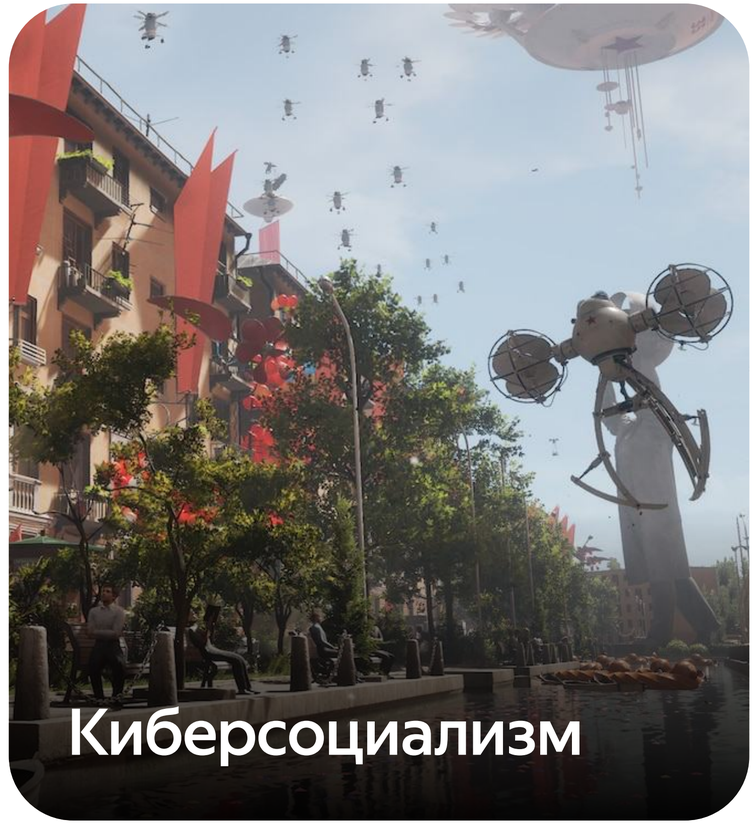
Ответы бека превращаются из 403, 404, 405 и т.п. в 200

BDUI ЭТО...



BDUI ЭТО...





ВОПРОС

А НУЖНА ЛИ ТРУ КРОСС ПЛАТФОРМА ДЛЯ BDUI?

ДЛЯ BDUI — НЕТ, ПОТОМУ ЧТО

01 Только платформно-специфичный код
камера, звуки

02 Минимальный менеджмент в приложении
одно окно

03 В кроссплатформе сложнее
бороться за перформанс

04 Но все равно хочется)
поэтому Flutter будет

ВОПРОСЫ?

@guitariz

DIVKIT КРУТОЙ!