



# Быстрый запуск фичей с Server Driven UI

Александр Свиридов

# Опрос 1

Насколько вы следите за размером передаваемого по API JSON?

1. Строго (минимум данных, передаю только данные, Rest и прочее)
2. Средне (следим когда появляются проблемы, периодически делаем ревизию)
3. Не следите

# Опрос 2

Насколько вы следите за оптимальностью layout в верстке?

1. Строго (минимальное вложенность, 1 ConstraintLayout и все такое)
2. Средне (следим когда появляются визуальные тормоза, периодически делаем ревизию)
3. Не следите

# Спойлеры

1. Браузер виджетов (Composer)
2. Навигация между страницами (Flow)
3. Кастомизация виджета (TileBuilder/Atoms)

# Советы по созданию фреймворков

1. Ограничьте область применения
2. Схожая реализация Android/iOS/Mobile Web
3. Удобство пользования фреймворком крайне важно
4. Выберите красивое название
5. Визуализируйте!

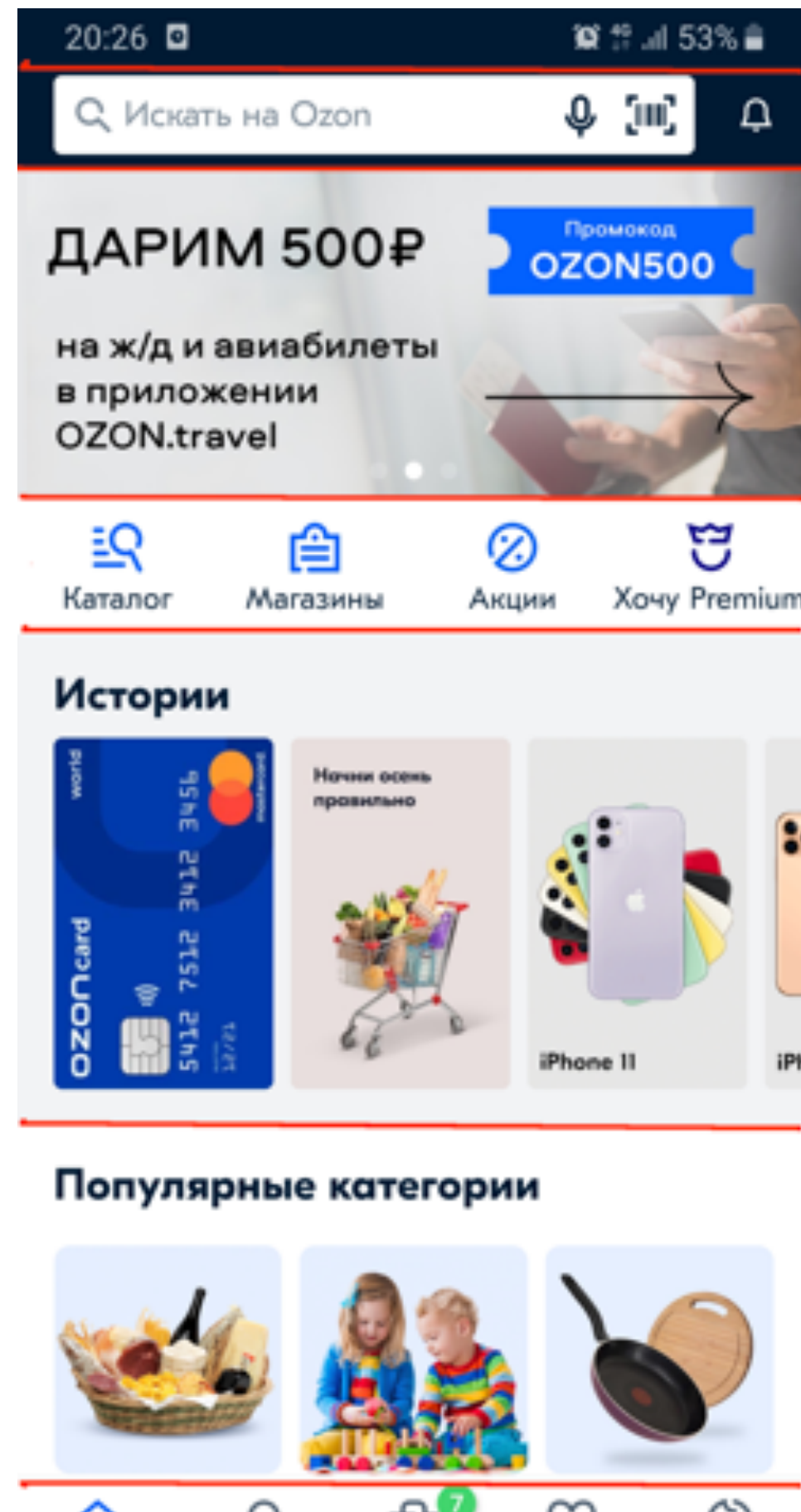
# Задачи

- Быстро запускать акции, промоутить товары и продавцов  
eCommerce очень динамичный рынок
- BE команда разделена на business units  
Microservices architecture
- У каждого сервиса свой владелец  
контроль от и до
- Нет команды MobileAPI  
узкое место
- АВ тесты и аналитика  
быстрый фидбэк

# Composer

## браузер виджетов

# Composer - браузер виджетов



Service

storefront

marketing

storefront

stories

shelf

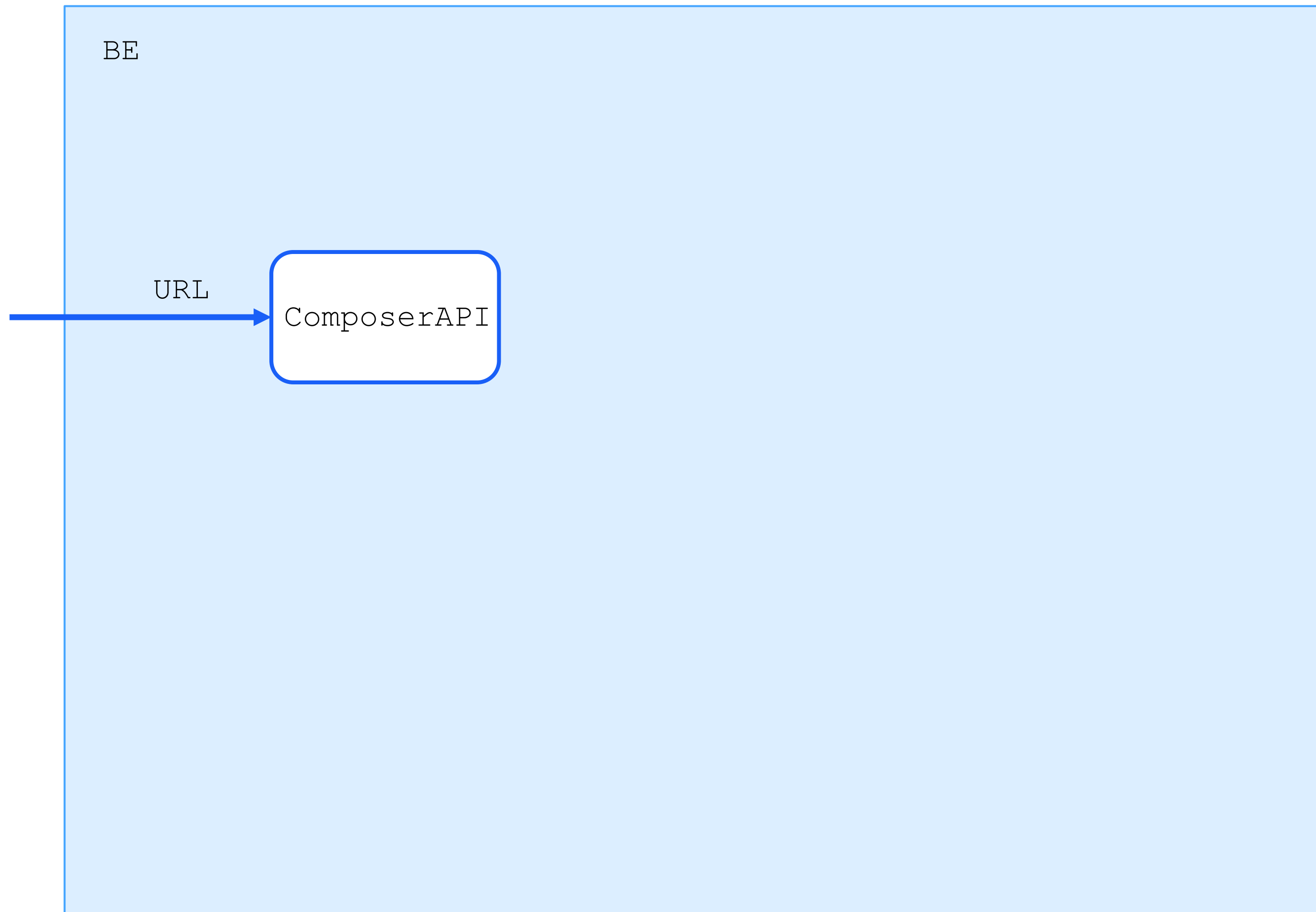




Apps



Desktop



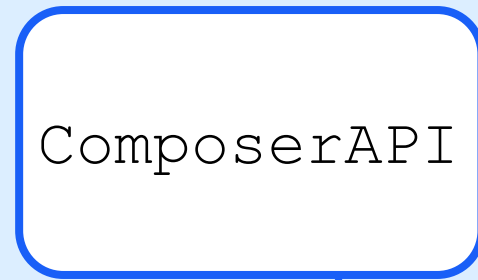


Apps

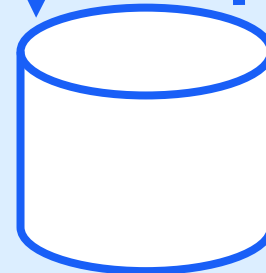


Desktop

BE



URL



Layout API

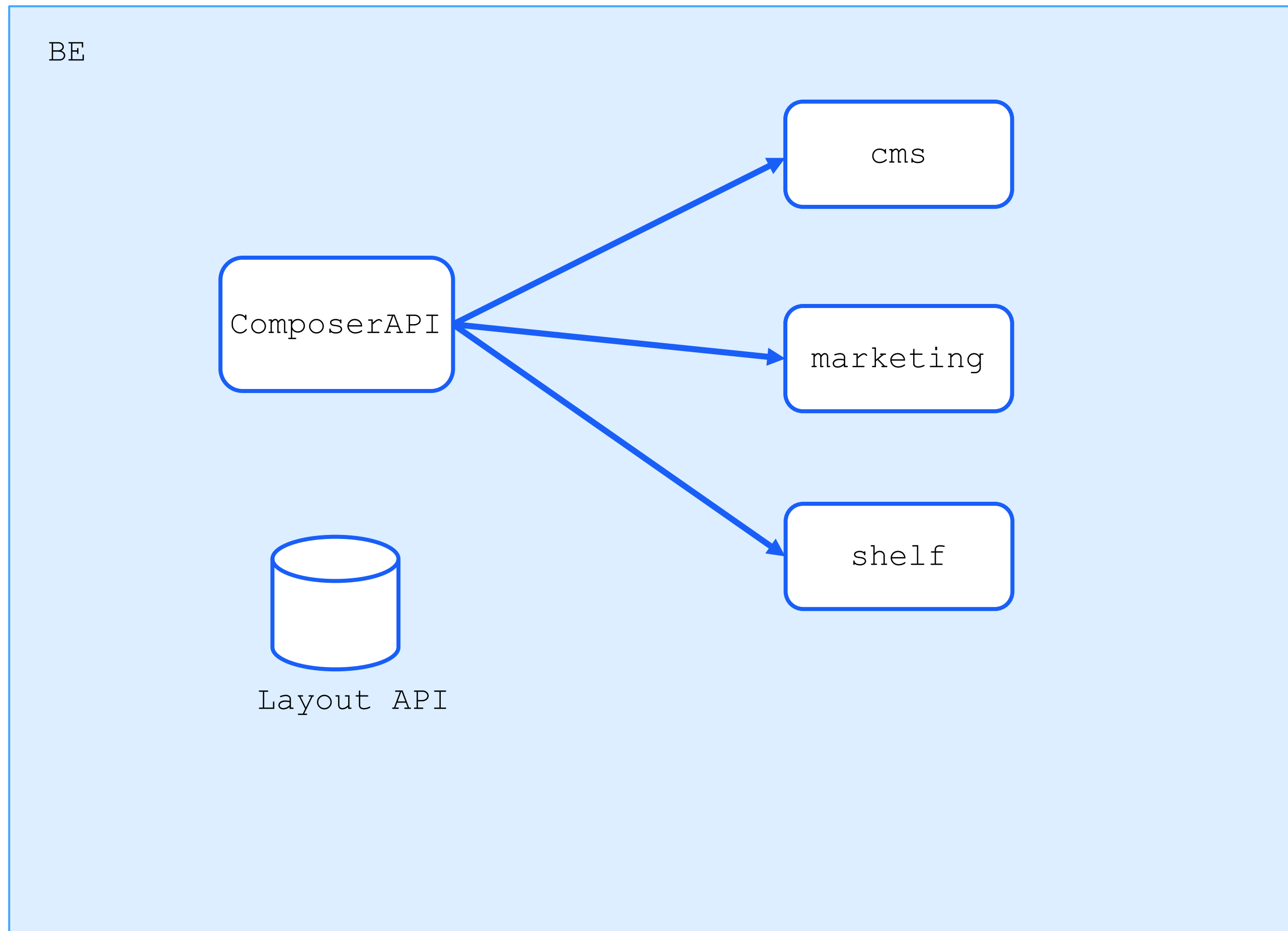
layout



Apps



Desktop



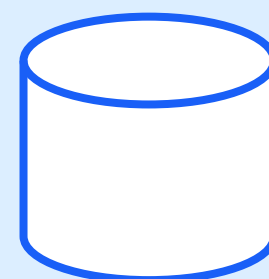
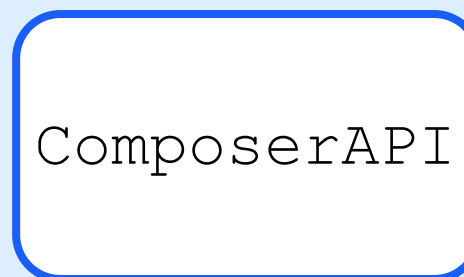


Apps

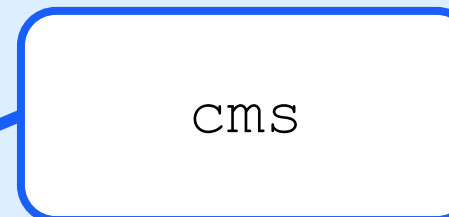


Desktop

BE



Layout API





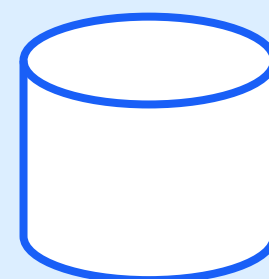
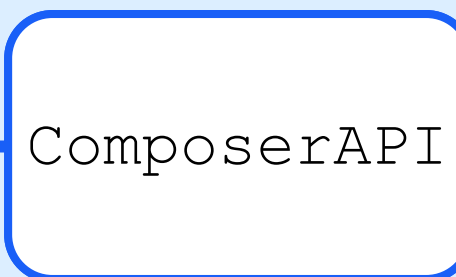
Apps



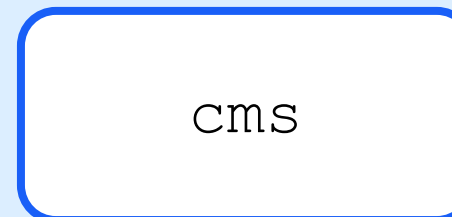
Desktop

BE

response



Layout API



# Страницы управляются с ВЕ

Редактирование шаблона «[DEFAULT] Pdp template-mobile»  
ID: 1995 ВЕРСИЯ: 210 PLATFORM: MOBILE  
АВТОР: ANASTASIYA MATVEEVA ИЗМЕНЕНО: 1 ДЕНЬ НАЗАД

200

gallery [PDP]

199

caption price per unit CAPTION [PDP]

198

caption premium CAPTION [PDP]

197

price premium PRICE [PDP]

196

marketingMarks [PDP]

195

price simple PRICE [PDP]

194

price per unit PRICE [PDP]

193

crosslink [PDP]

192

title [PDP]

191

reviewProductScore [RPPRODUCT]

190

productMobile [PDP]

189

188

187


186

185

184

Билайн 00:12 66 %

←



↑

♡

-36%

Цена с Ozon Premium  
**1 771 Р**

Бестселлер  
**1 884 Р** ~~2 990 Р~~  
Xiaomi  
Фитнес-браслет Xiaomi Mi Band 3, черный  
★★★★★ 2100  
Цвет  
черный

В корзину 1 884 Р

Главная

Каталог

Корзина

Избранное

Кабинет

→ gallery [PDP]

→ price premium PRICE [PDP]

→ price simple PRICE [PDP]

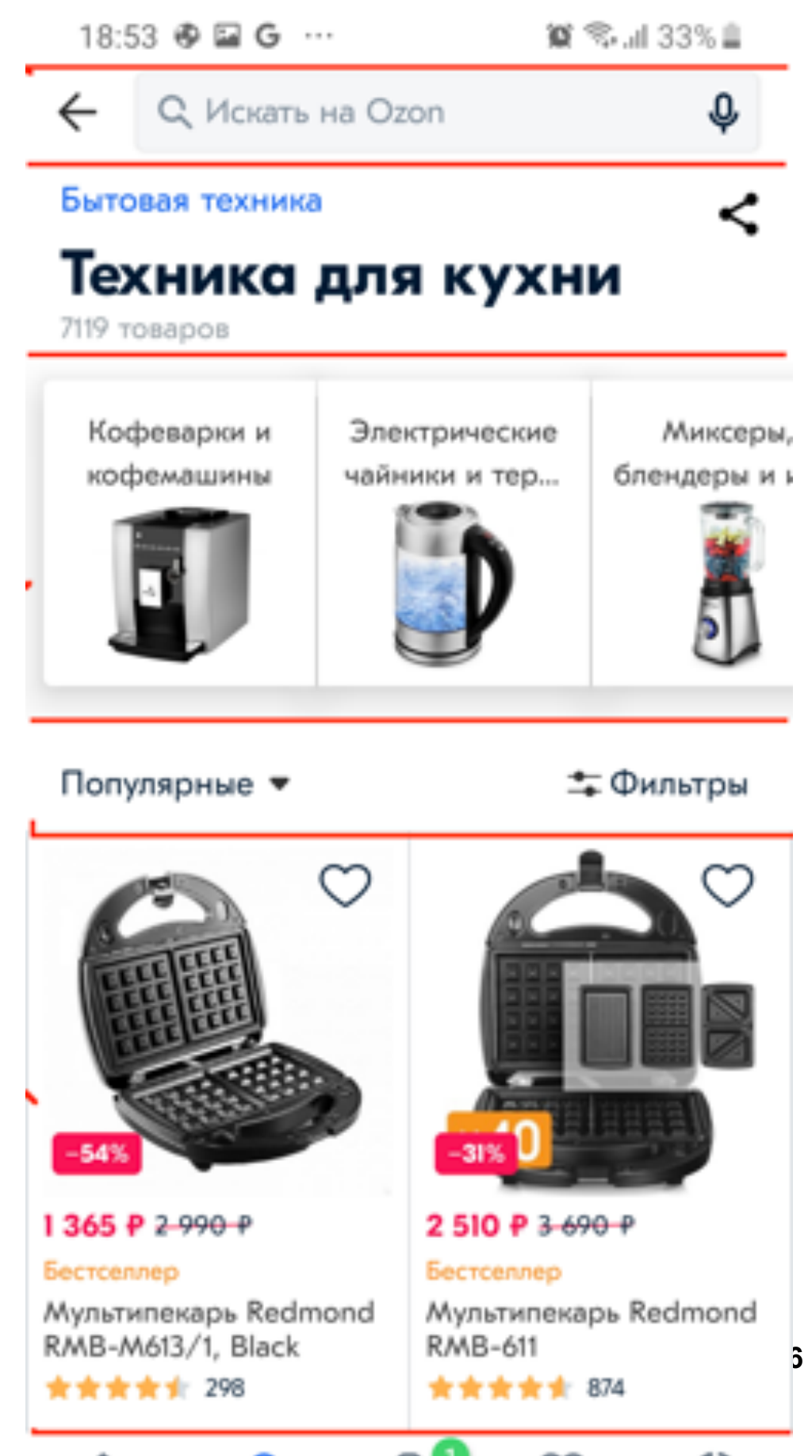
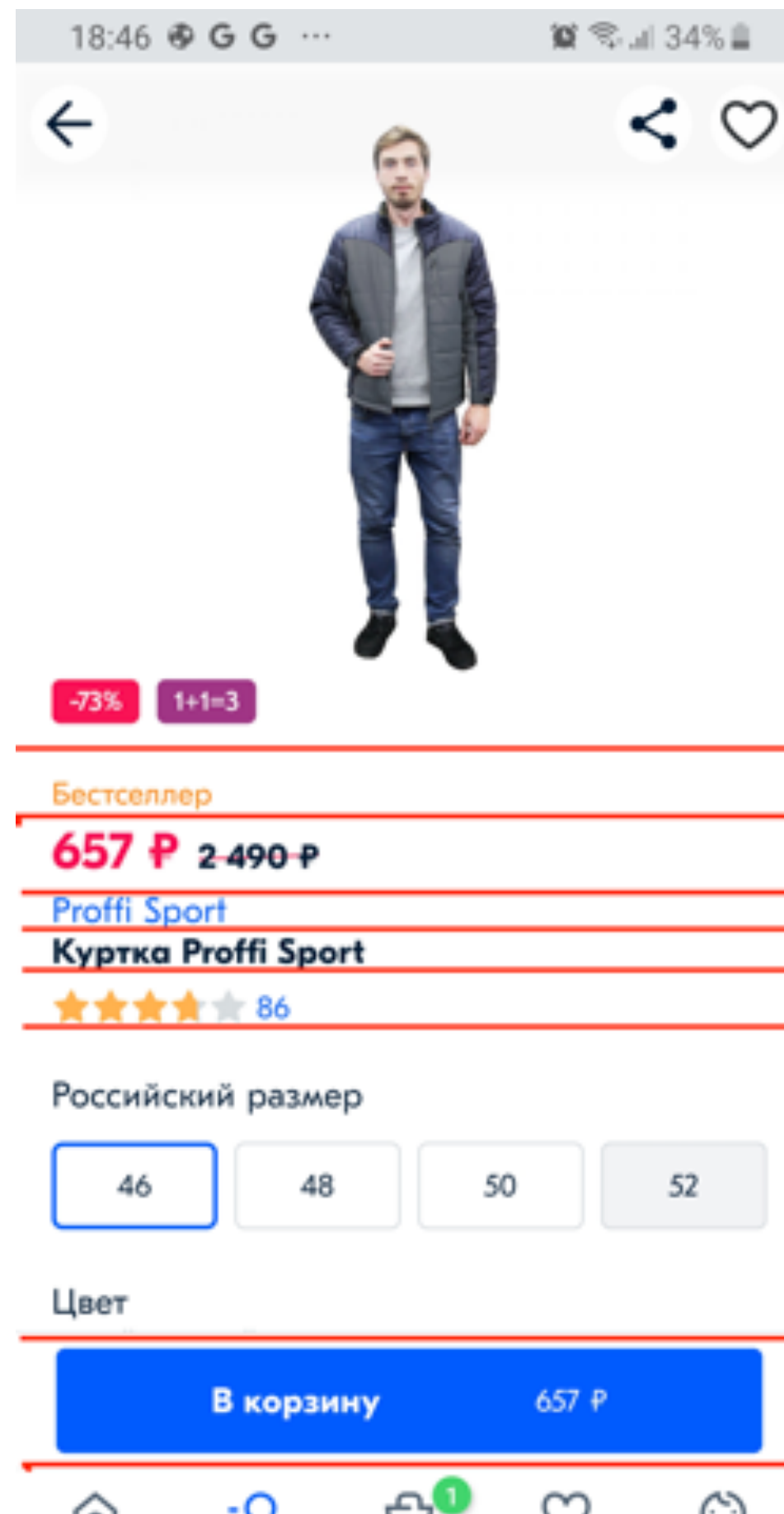
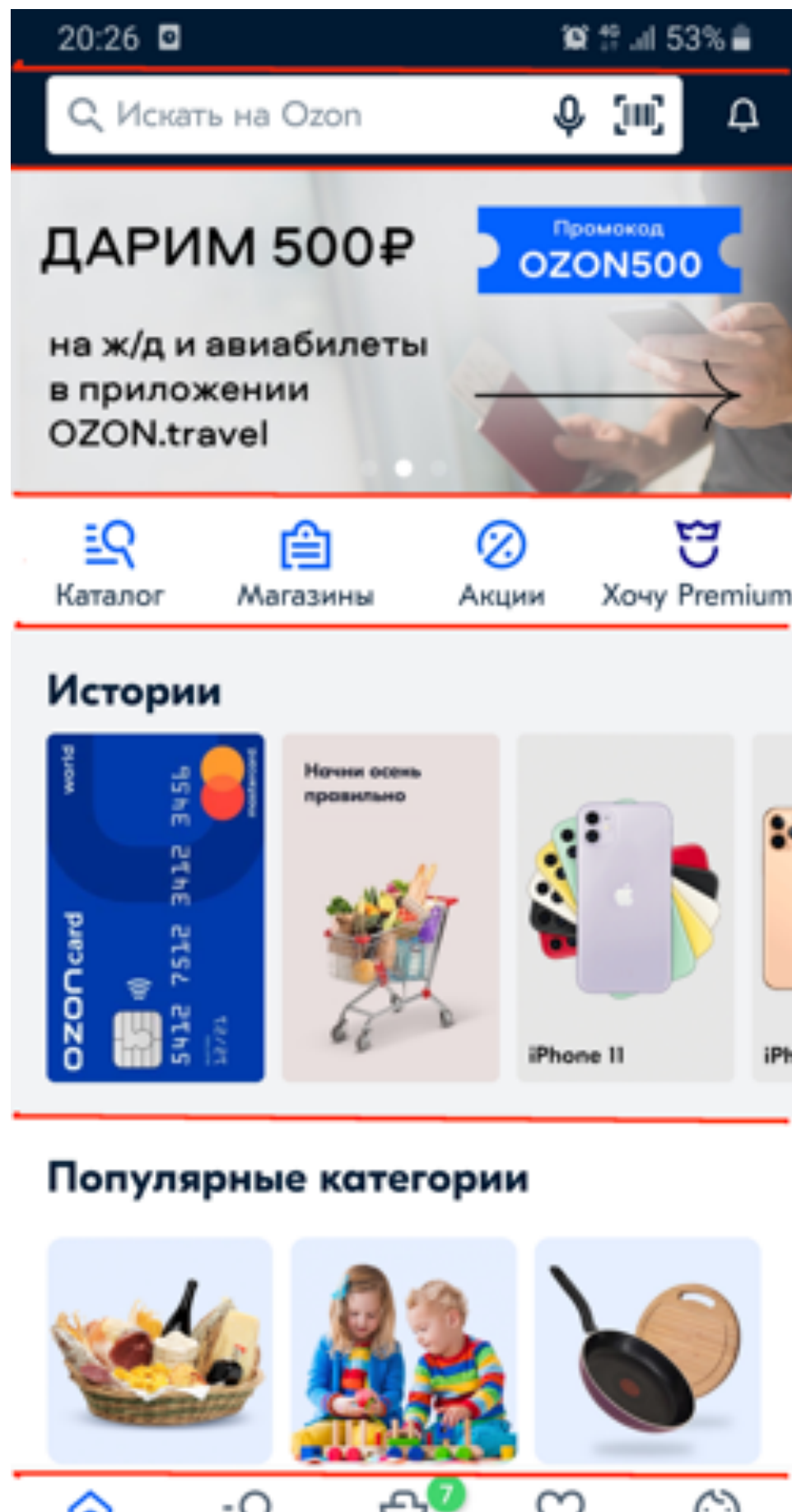
→ title [PDP]

→ reviewProductScore [RPPRODUCT]

# Что получилось?

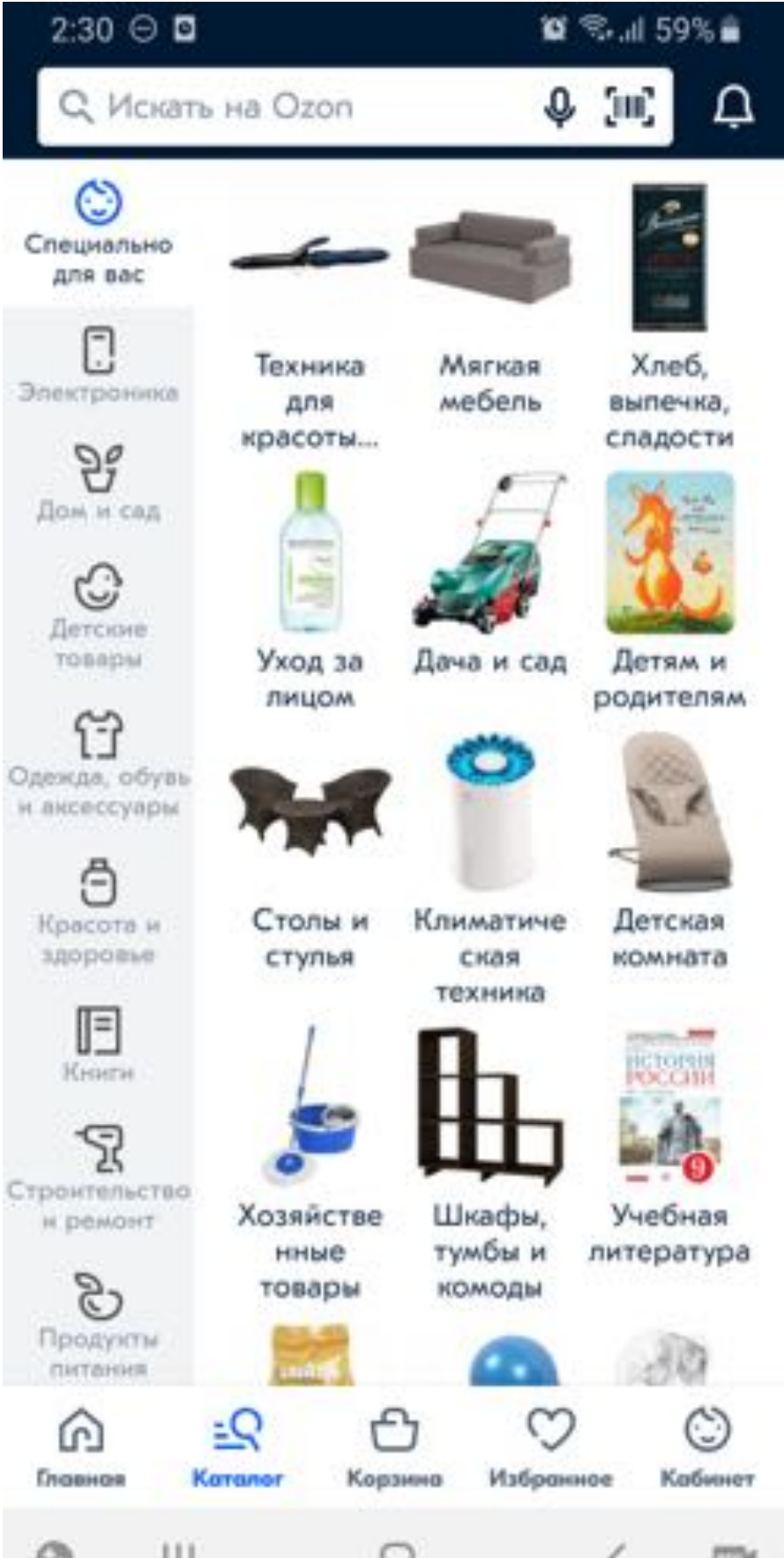
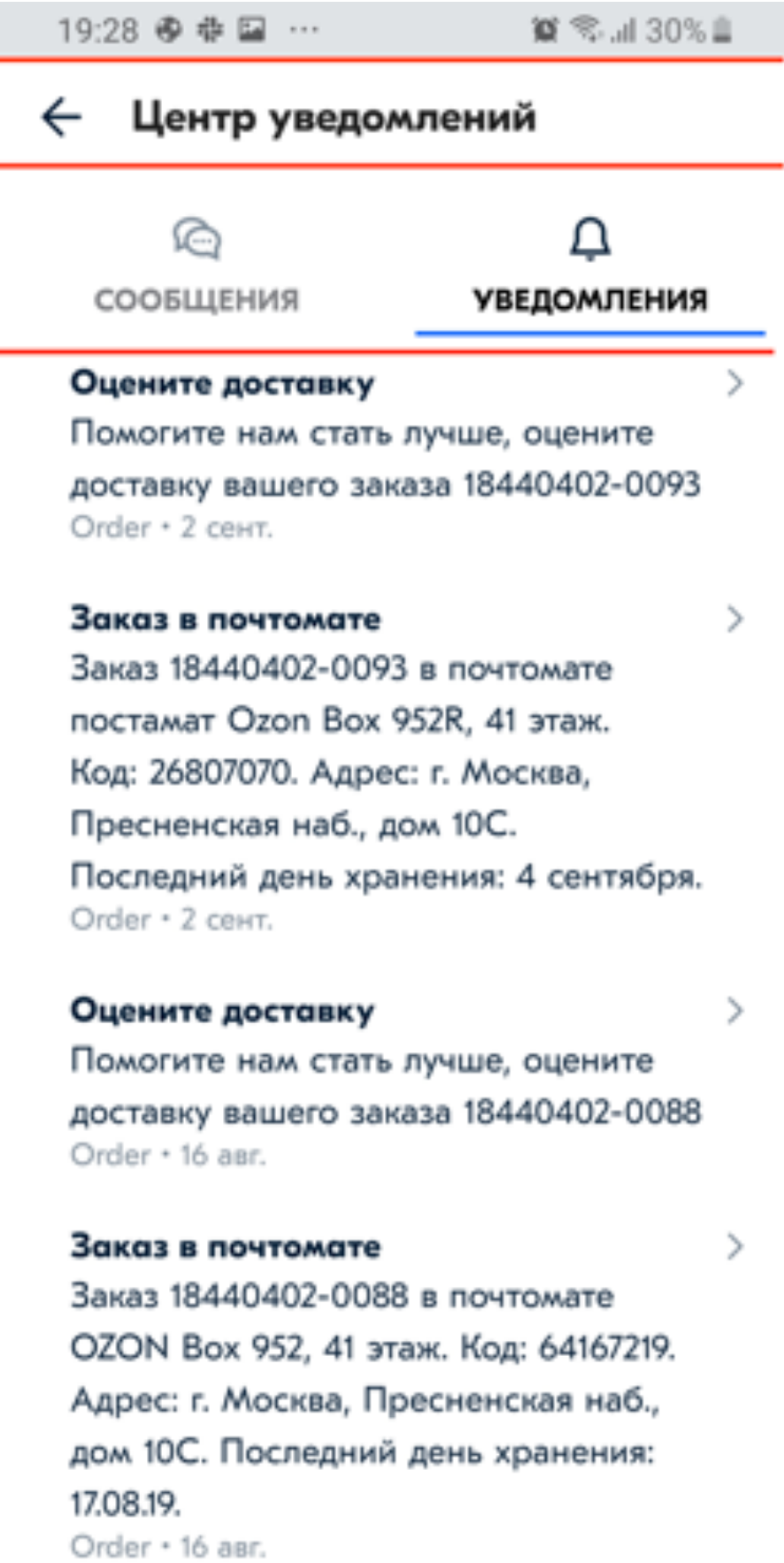
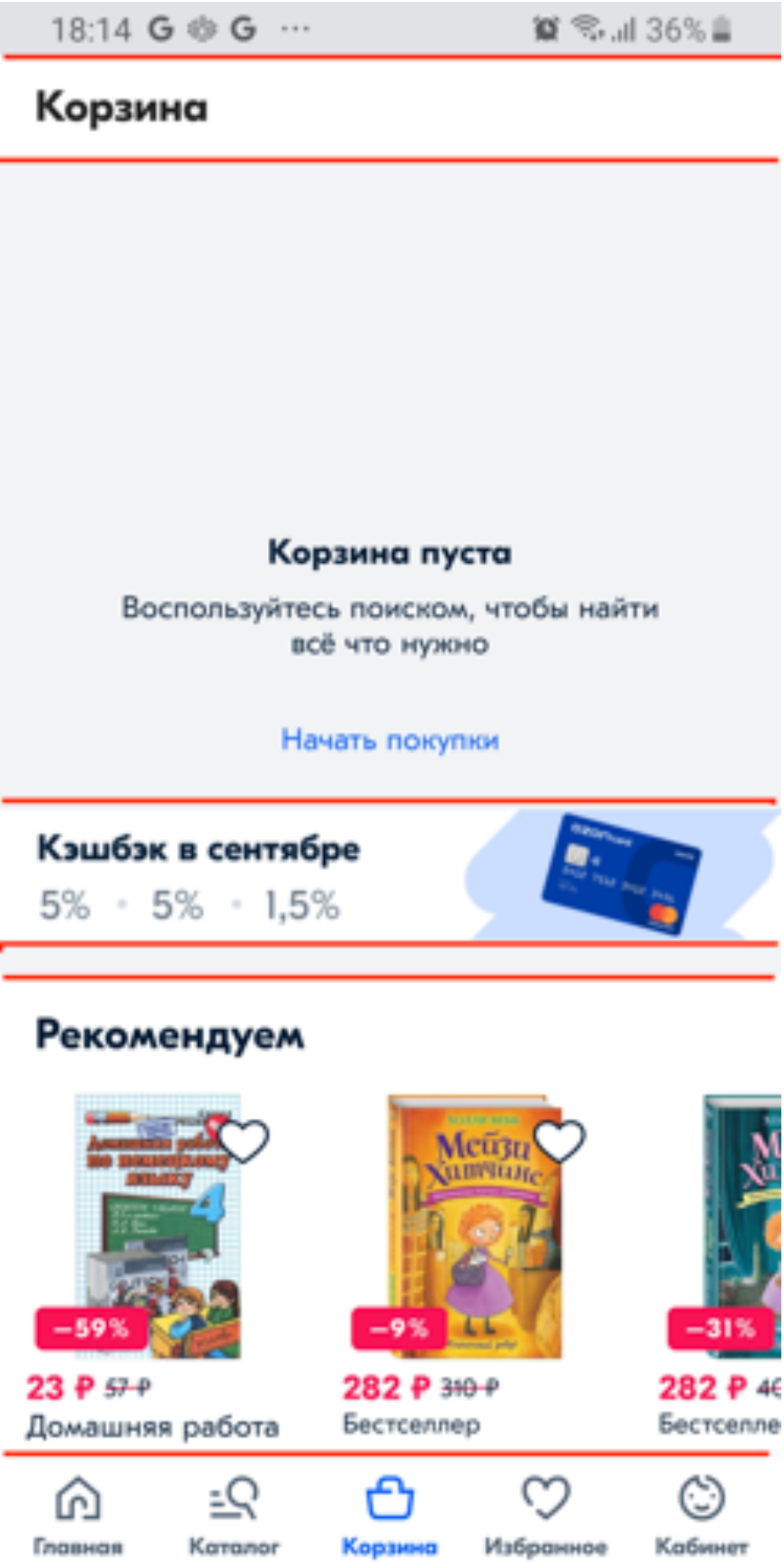
1. Один запрос на страницу
2. Логика виджета полностью у business unit
3. Можно управлять набором и порядком виджетов на странице

# Composer SDK - RecyclerView с ViewHolders

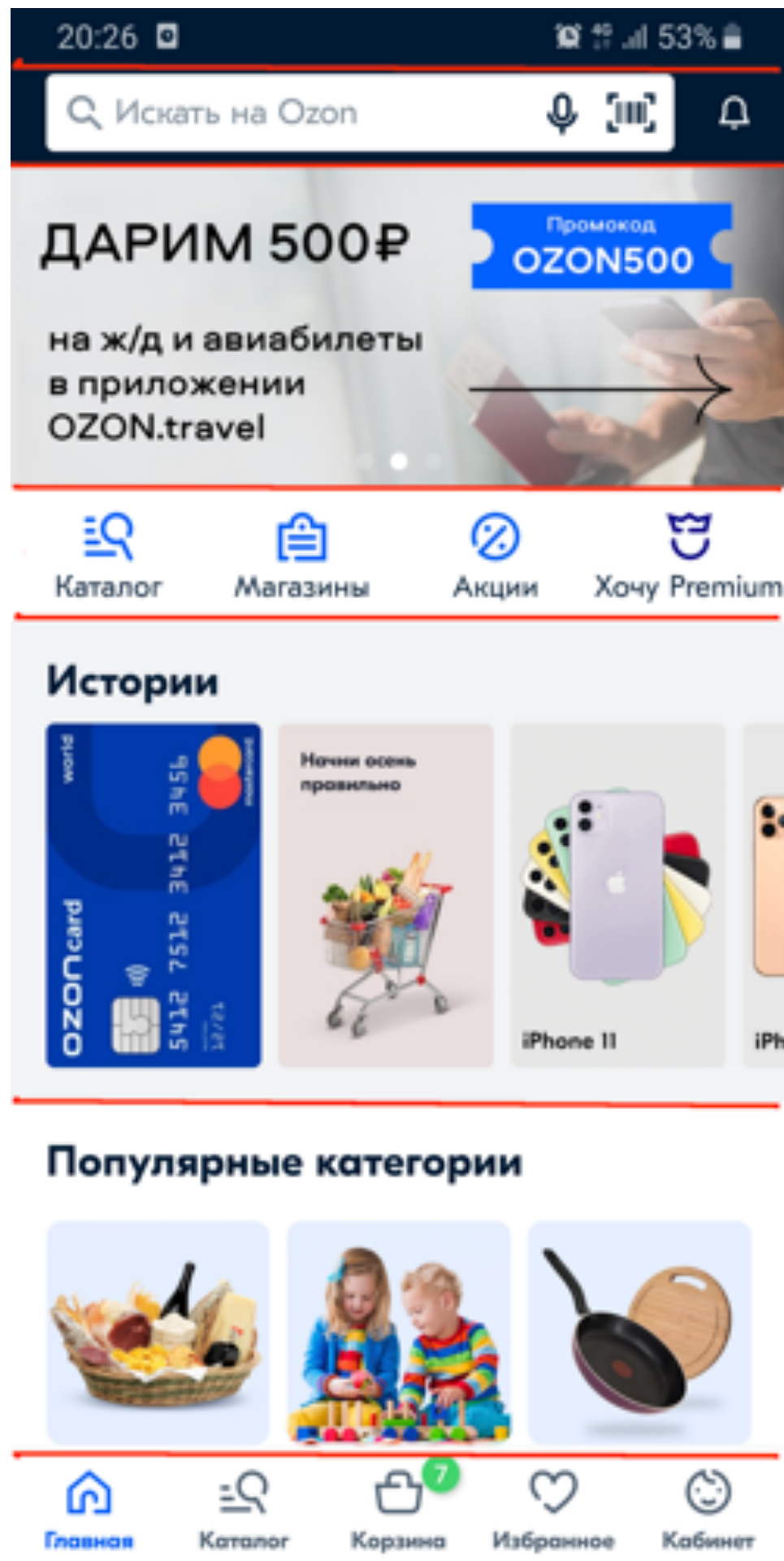




# Примеры

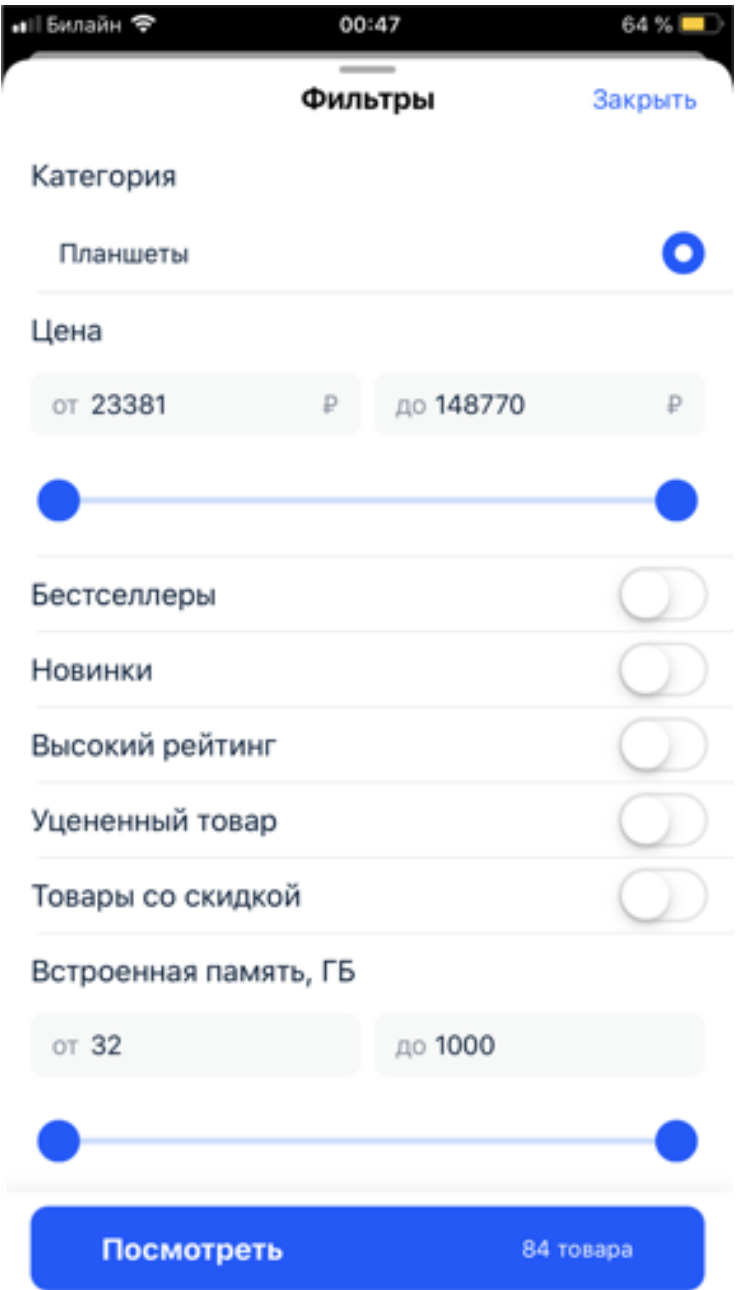
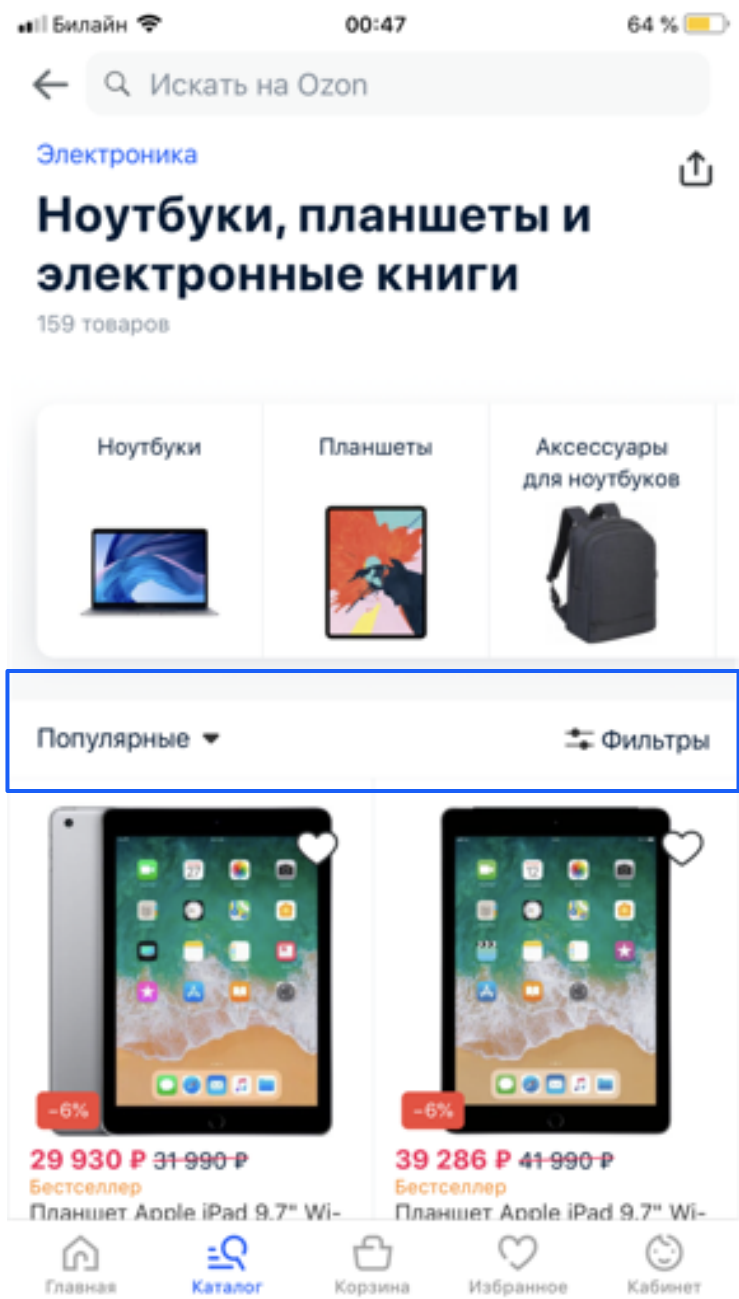


# Аналитика автомагический трекинг страниц и виджетов



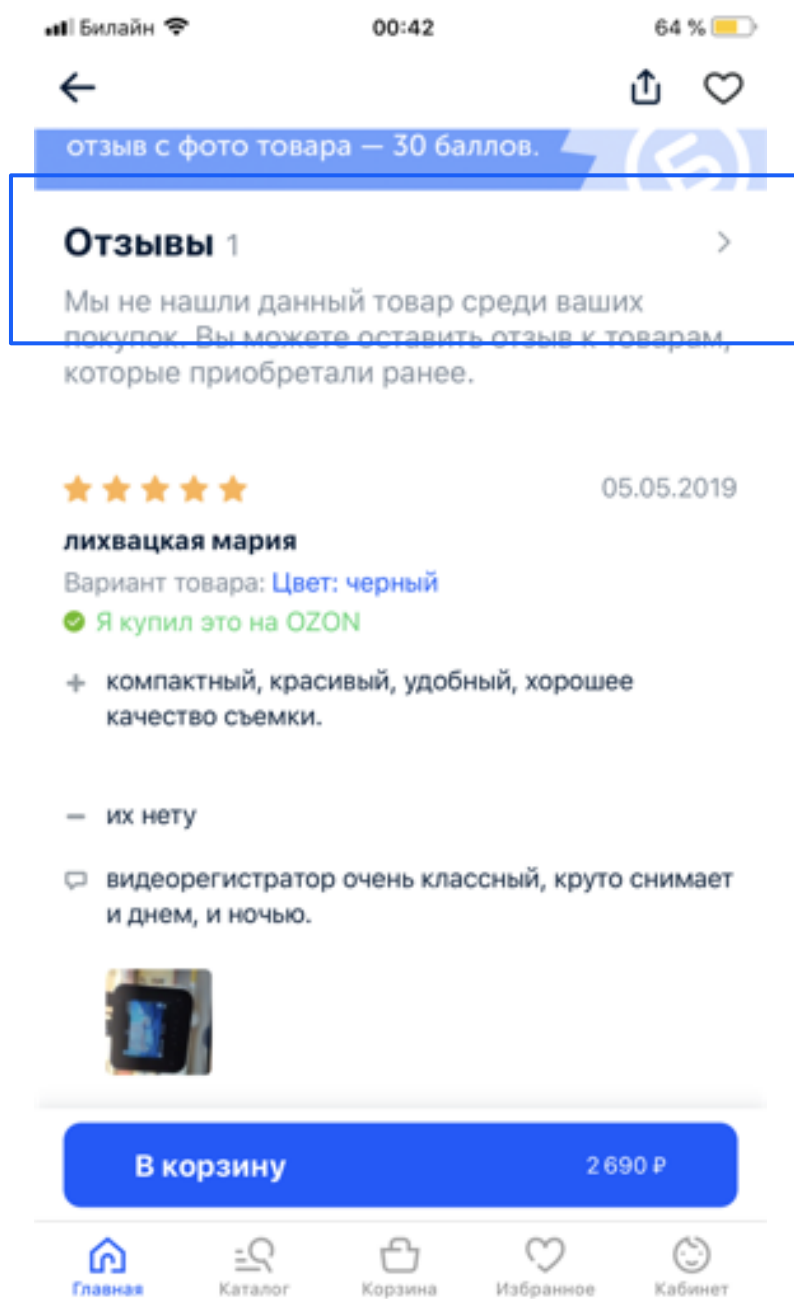
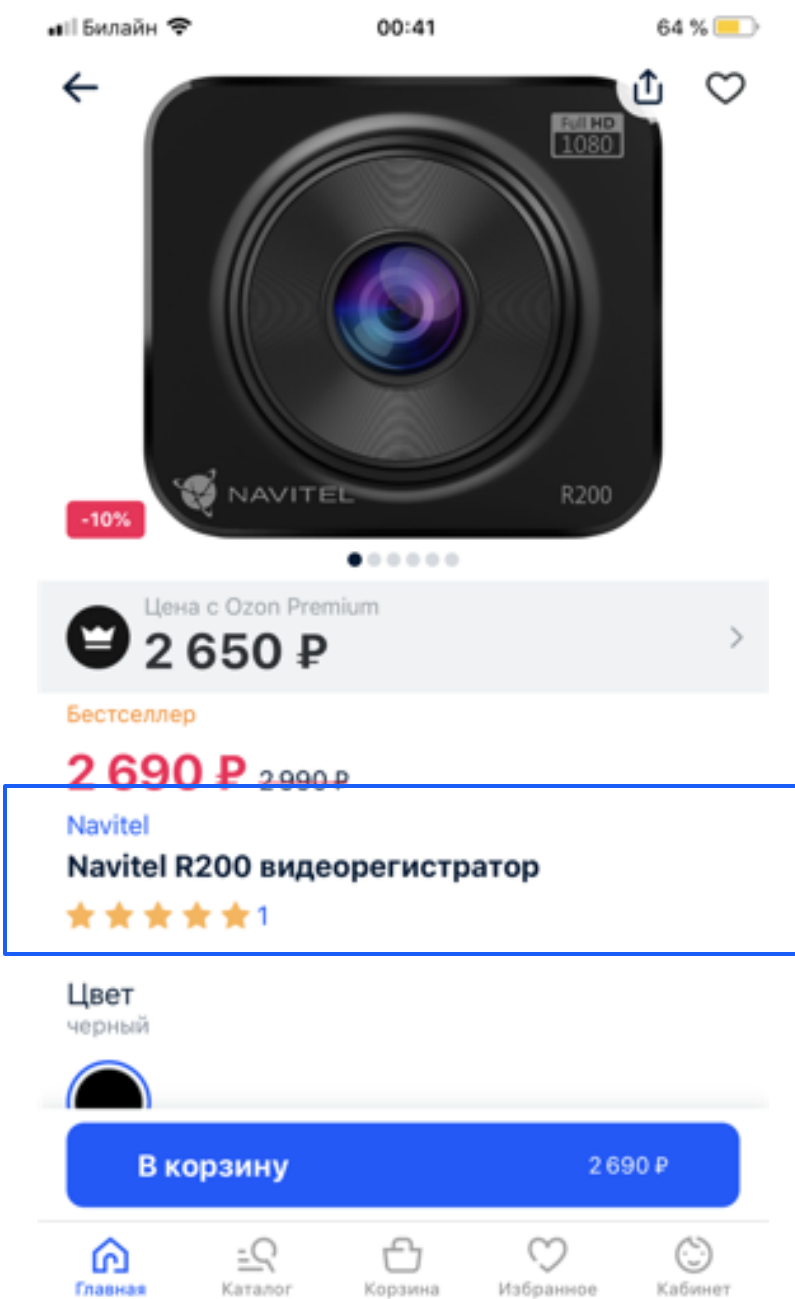
```
1  "layout": [  
2    {  
3      "vertical": "cms",  
4      "component": "setOfStories",  
5      "stateId": "setOfStories-213915-default-1",  
6      "version": 1,  
7      "widgetTrackingInfo": {  
8        "name": "cms.setOfStories",  
9        "vertical": "cms",  
10       "component": "setOfStories",  
11       "version": 1,  
12       "id": 213915,  
13       "revisionId": 154033,  
14       "index": 25  
15     }  
16   },  
17   {  
18     "vertical": "cms",  
19     "component": "uWidgetSKU",  
20     "stateId": "uWidgetSKU-216753-default-1",  
21     "version": 1,  
22     "widgetTrackingInfo": {  
23       "name": "cms.uWidgetSKU",  
24       "vertical": "cms",  
25       "component": "uWidgetSKU",  
26       "version": 1,  
27       "id": 216753,  
28       "configId": 1135,  
29       "configDtId": 49,  
30       "revisionId": 154031,  
31       "index": 33,  
32       "dtName": "sku.list"  
33     }  
34   }  
35 ],  
36 "layoutTrackingInfo": {  
37   "pageType": "home",  
38   "ruleId": 2423,  
39   "layoutId": 2215,  
40   "layoutVersion": 37  
41 }  
42
```

# Проблемы. Связанные виджеты

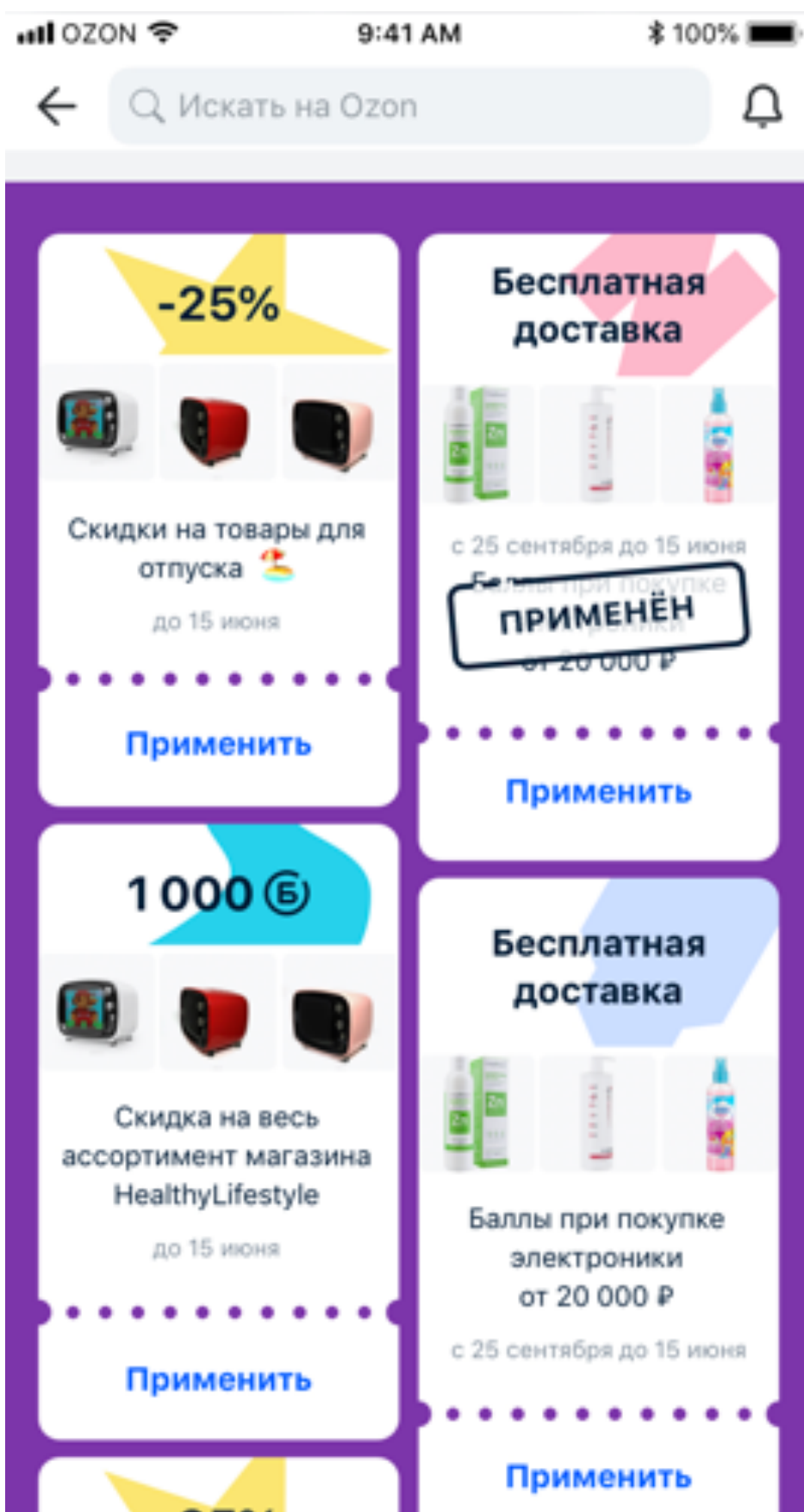




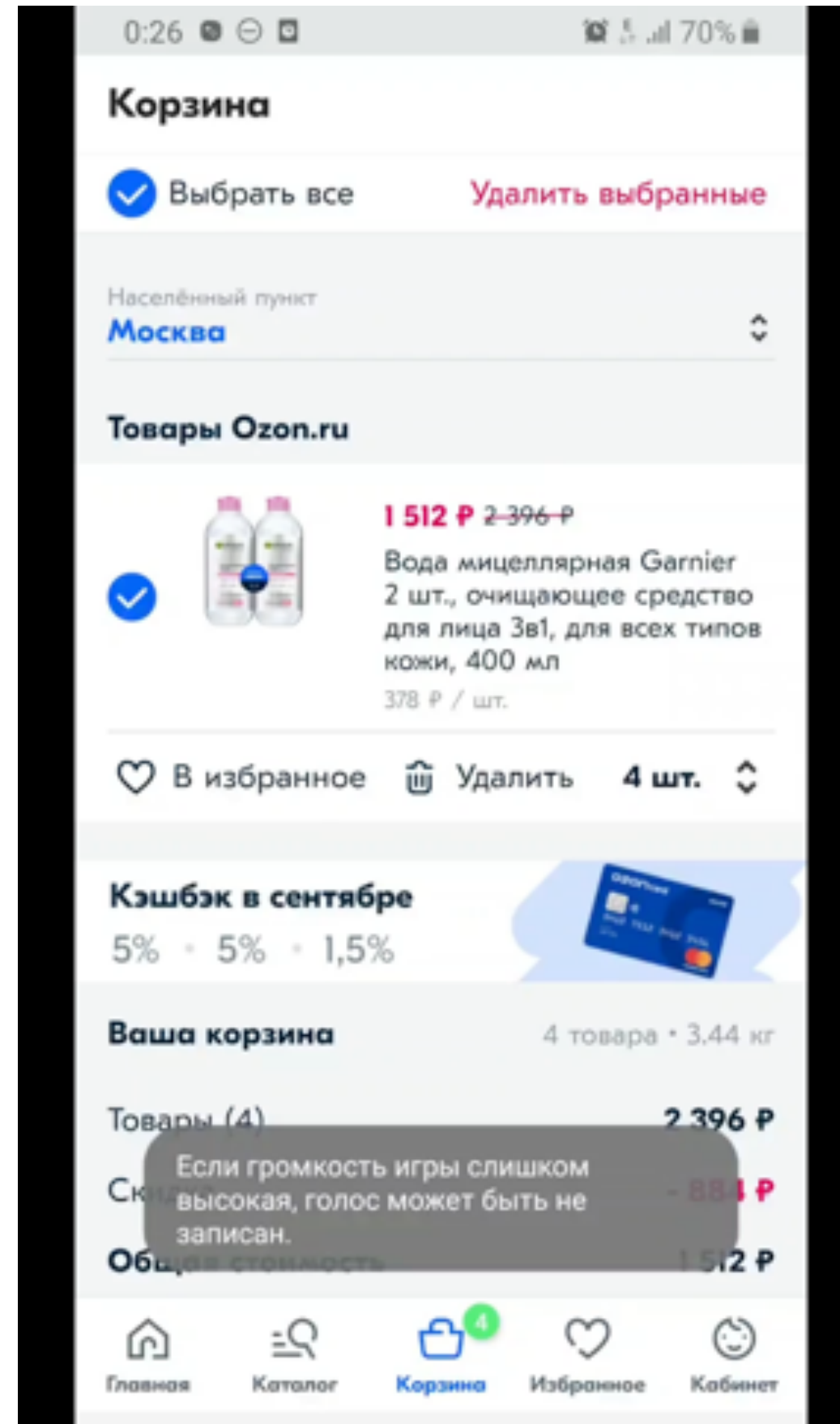
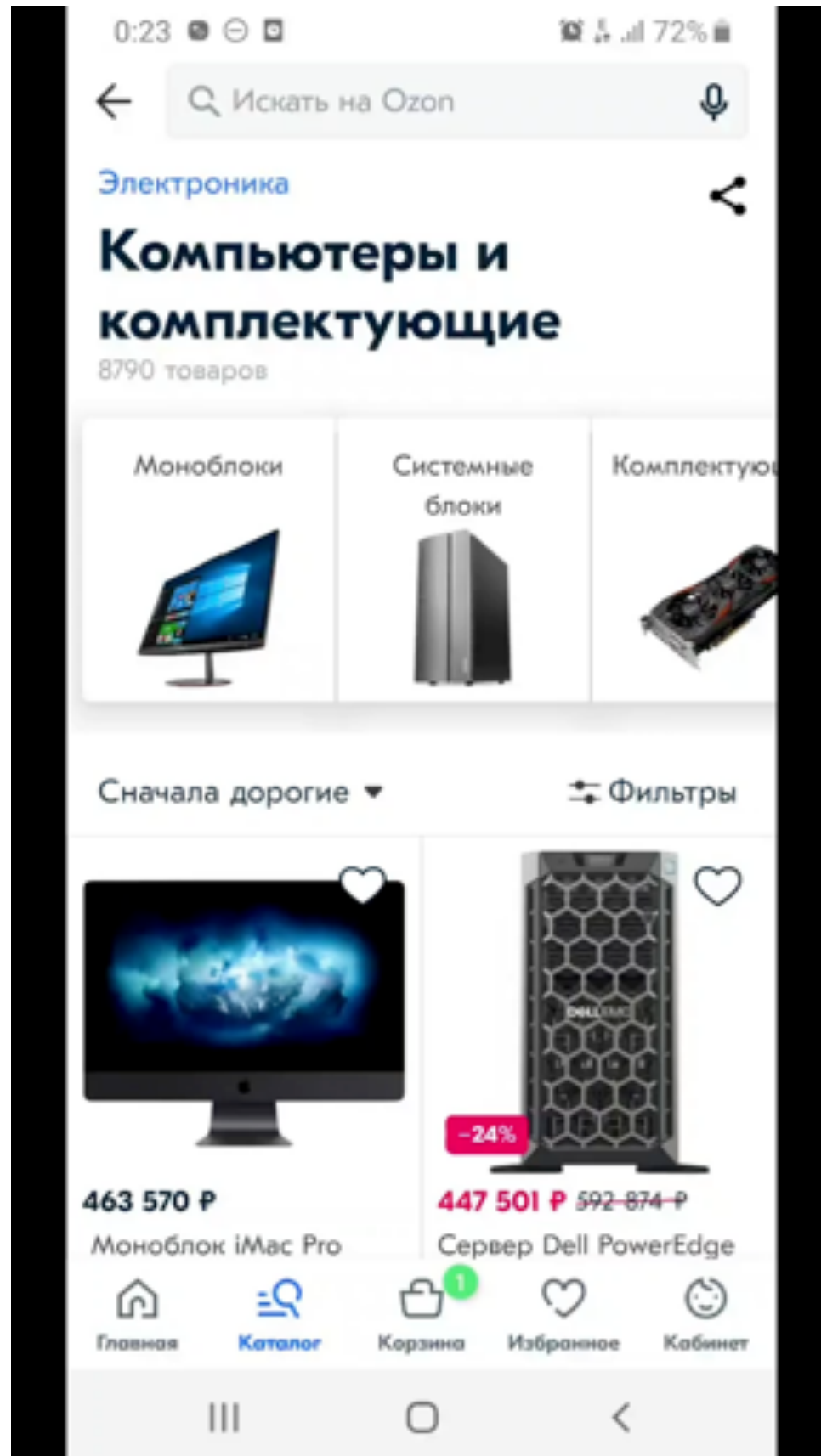
# Проблемы. Избыточные данные



# Проблемы. «Сложные» layouts

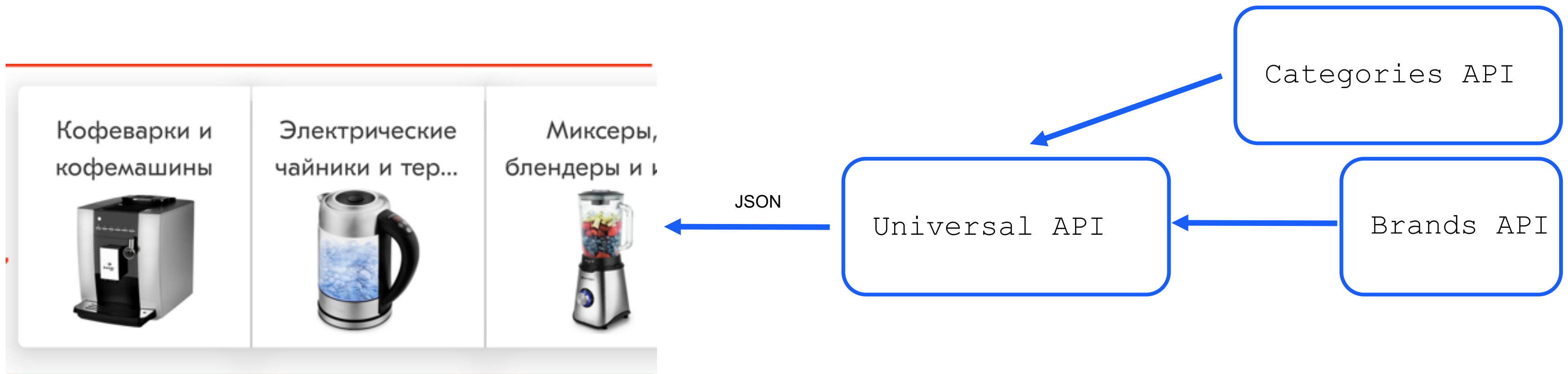


# Проблемы. Браузер есть браузер



# Проблемы. Сложность переиспользования

- Для переиспользования виджета нужно интегрировать несвязанные API (не делайте так!)
- Нужно: библиотека переиспользуемых виджетов



# Flow

навигация между страницами

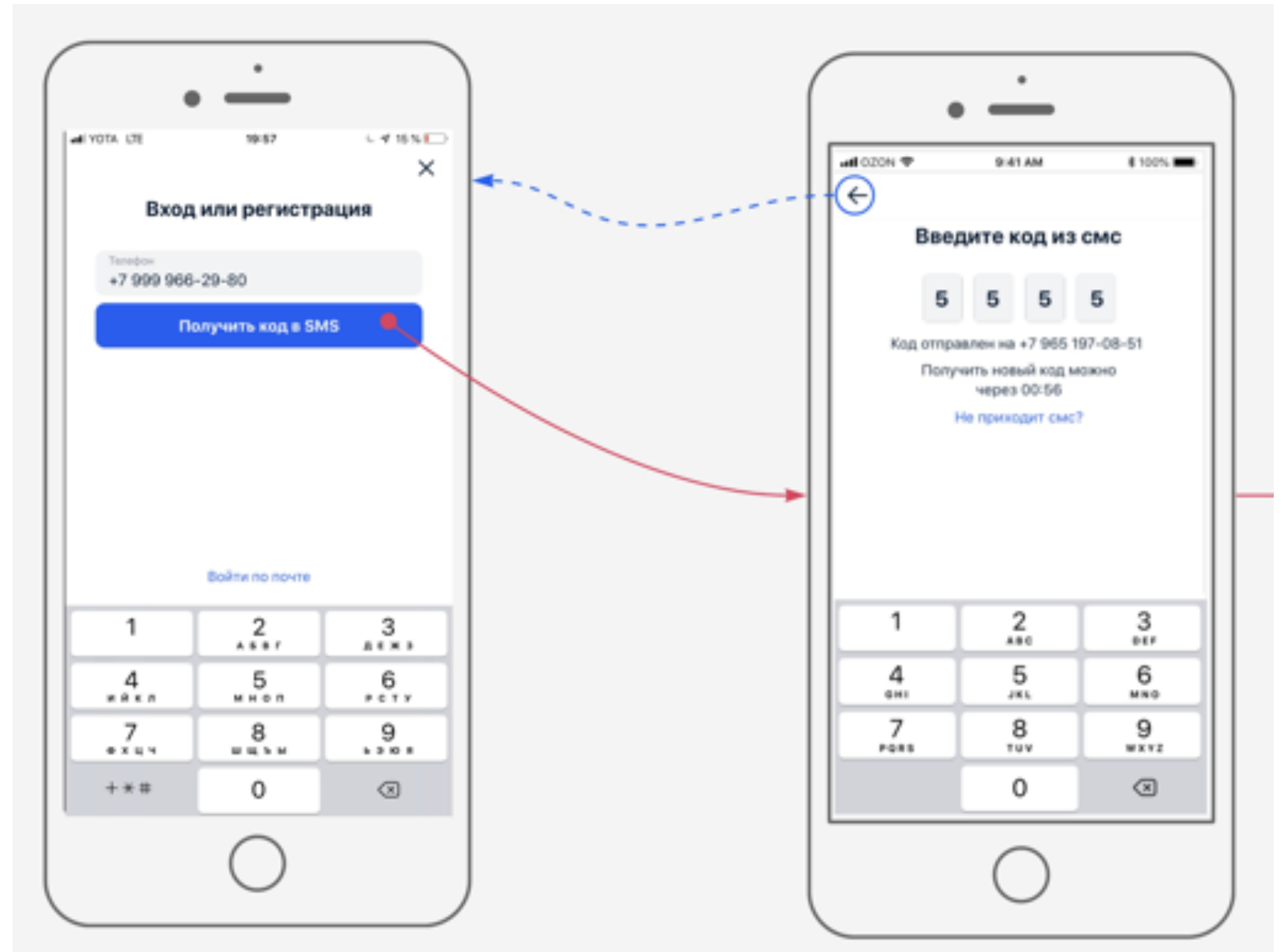


# Управление навигацией между страницами



# Flow

1. Просто 1 Activity
2. FlowRouter управляет переходами.
3. Переход на предыдущий шаг – очистка стека.
4. Финальное событие – закрытие Activity



# Tile Builder

конструктор поисковой «плитки»

# Tile Builder. Кастомизация поисковой плитки

- Конфигурирование внутренностей 1го виджета
- Разный контент в разных категориях
- Разная высота «плитки»
- Разное image ratio.



# Tile Builder. Каркас + список элементов

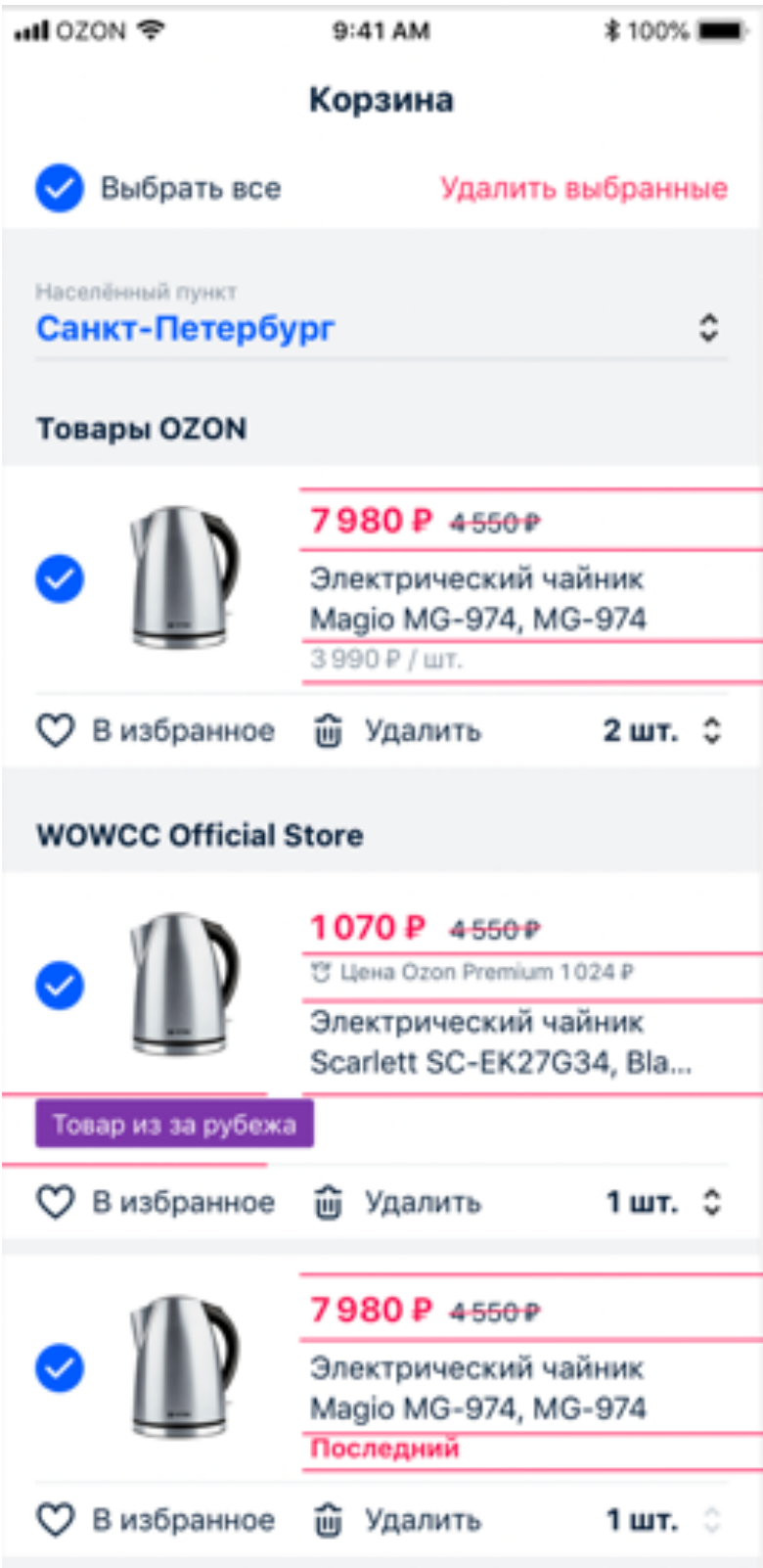
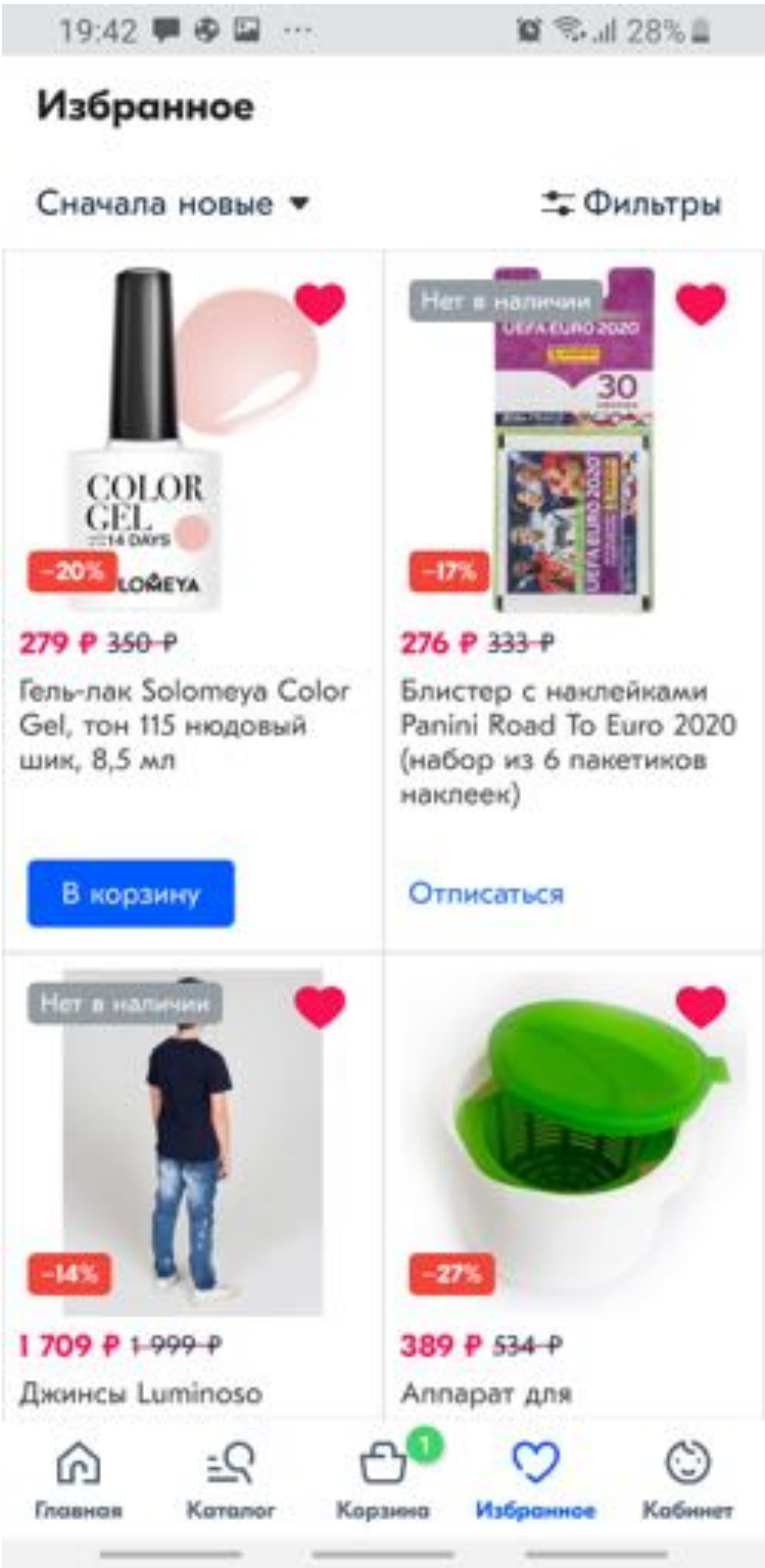
- Каркас: картинка, фиксированная высота плитки
- Кастомизация:
  - абстрактный список элементов,
  - абстрактные бейджики на картинке.
  - Image Ratio
  - Размер плитки



Черные линии – границы между элементами

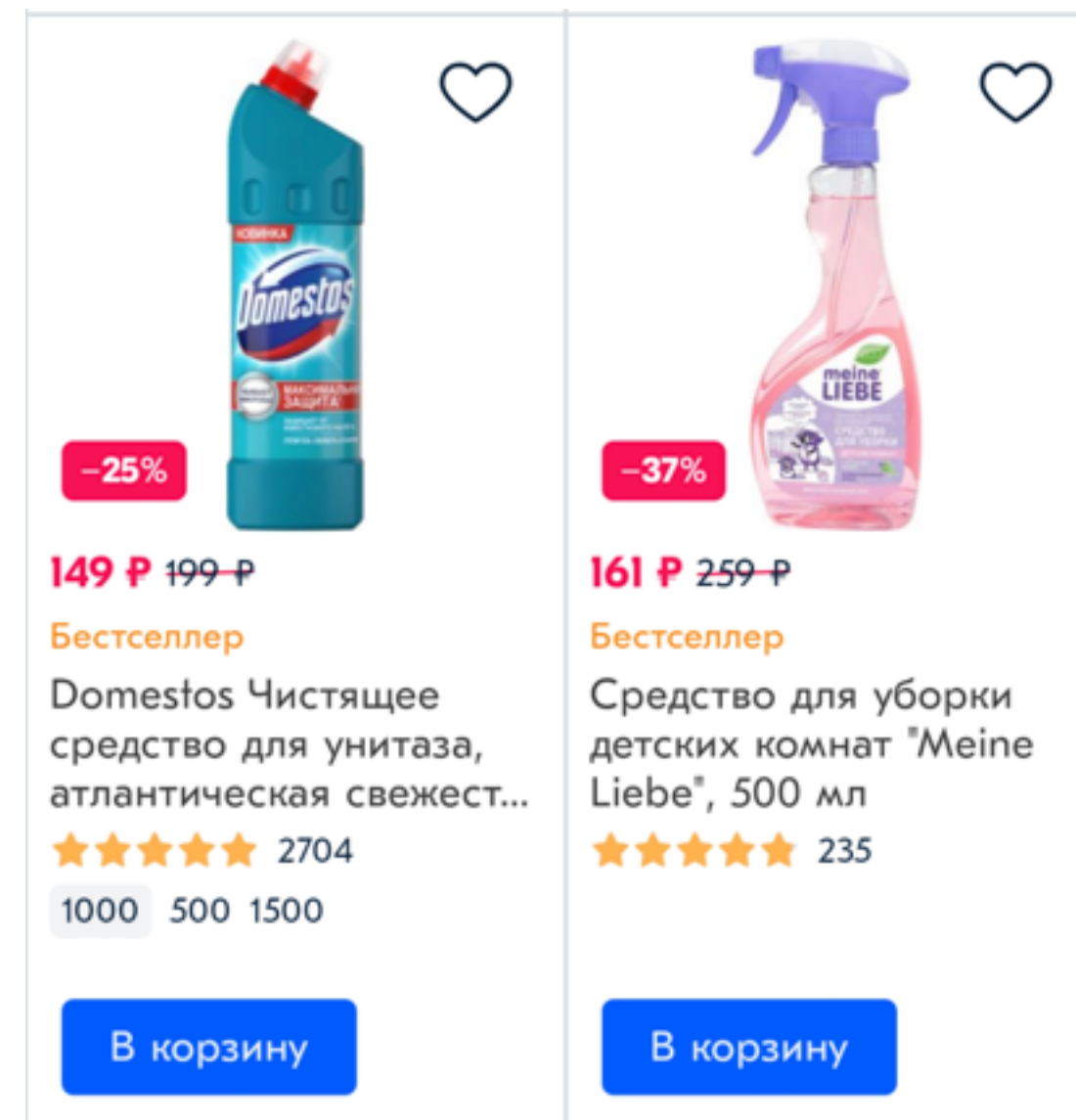


# Tile Builder+Composer. Фичи без релиза



# Tile Builder.Минусы

- Увеличивается response – не проблема
- Проблемы фиксированных отступов
- Проблема связанных элементов



# АТОМЫ

## конструктор виджетов



# Элементы TileBuilder —> Атомы

| Название         | Изображение   | JSON   | Примечание |
|------------------|---|--|------------|
| label            |    | <pre>{   "type": "label",   "items": [     {       "title": "89 000 P с Premium",       "backgroundColor?": "#012250",       "textColor?": "#ffffff"     }   ] }</pre> |            |
| price            |   | <pre>{   "type": "price",   "price": "112 490 P", //text   "finalPrice?": "or 92 490P", //text }</pre>   |            |
| unavailablePrice |  | <pre>{   "type": "unavailablePrice",   "price": "112 490 P", //text   "finalPrice?": "or 92 490P", //text }</pre>  |            |

# Style Guide

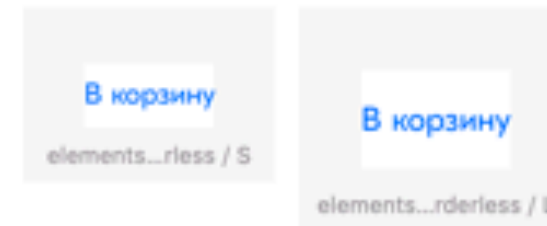
Каждый атом имеет

- Название
  - Состояния в Style Guide Zeplin
  - JSON
  - Статус (используется/в разработке/...)
- 
- Атомы – это xml + Configurator + Decorator

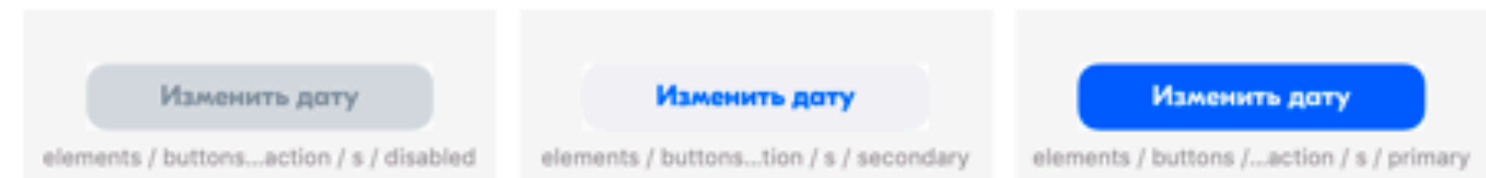
## Buttons

GET 1 component collapsed

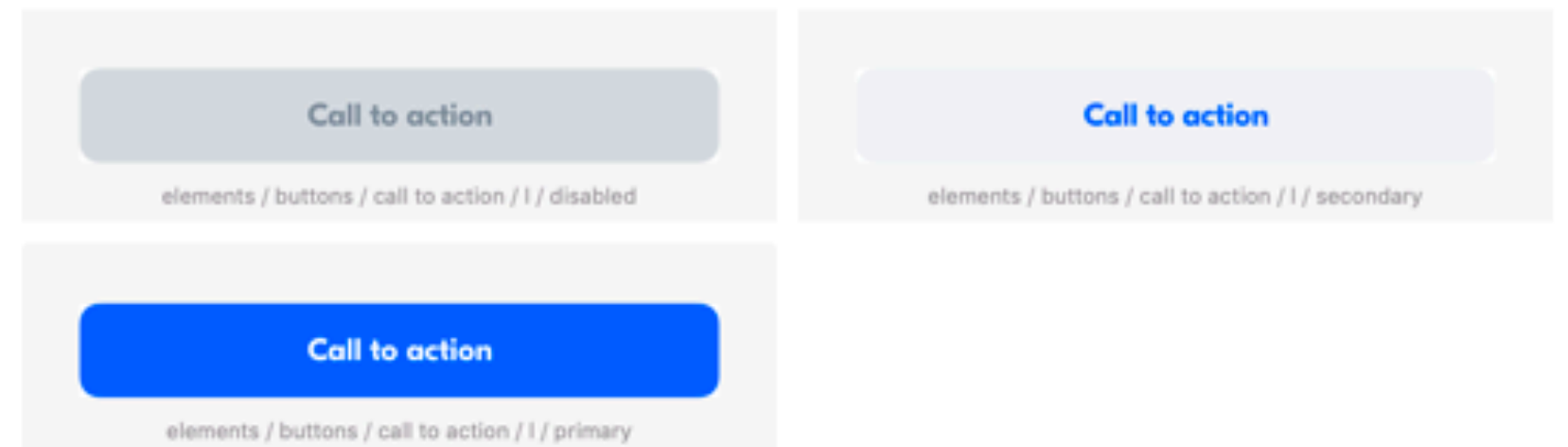
### Borderless



### S

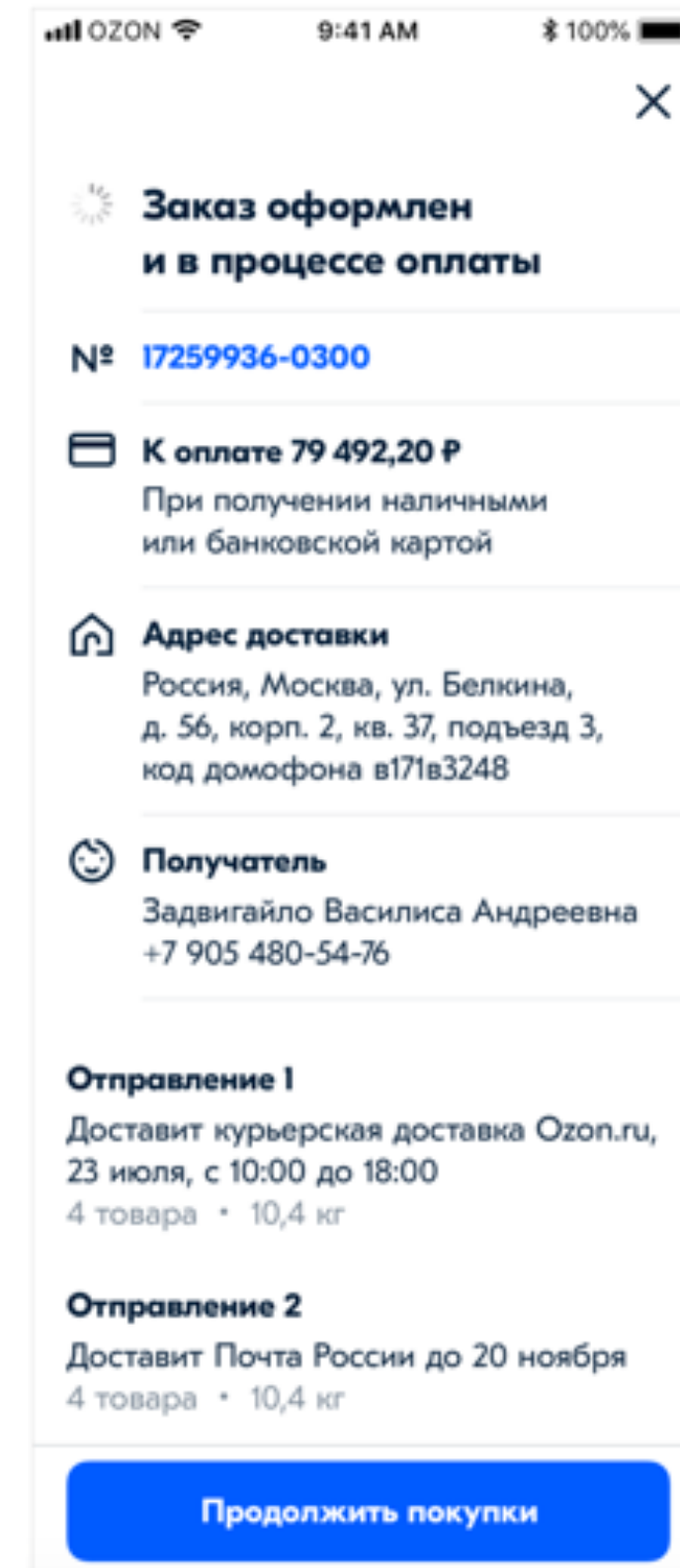


### L



# Атомы. Work in Progress

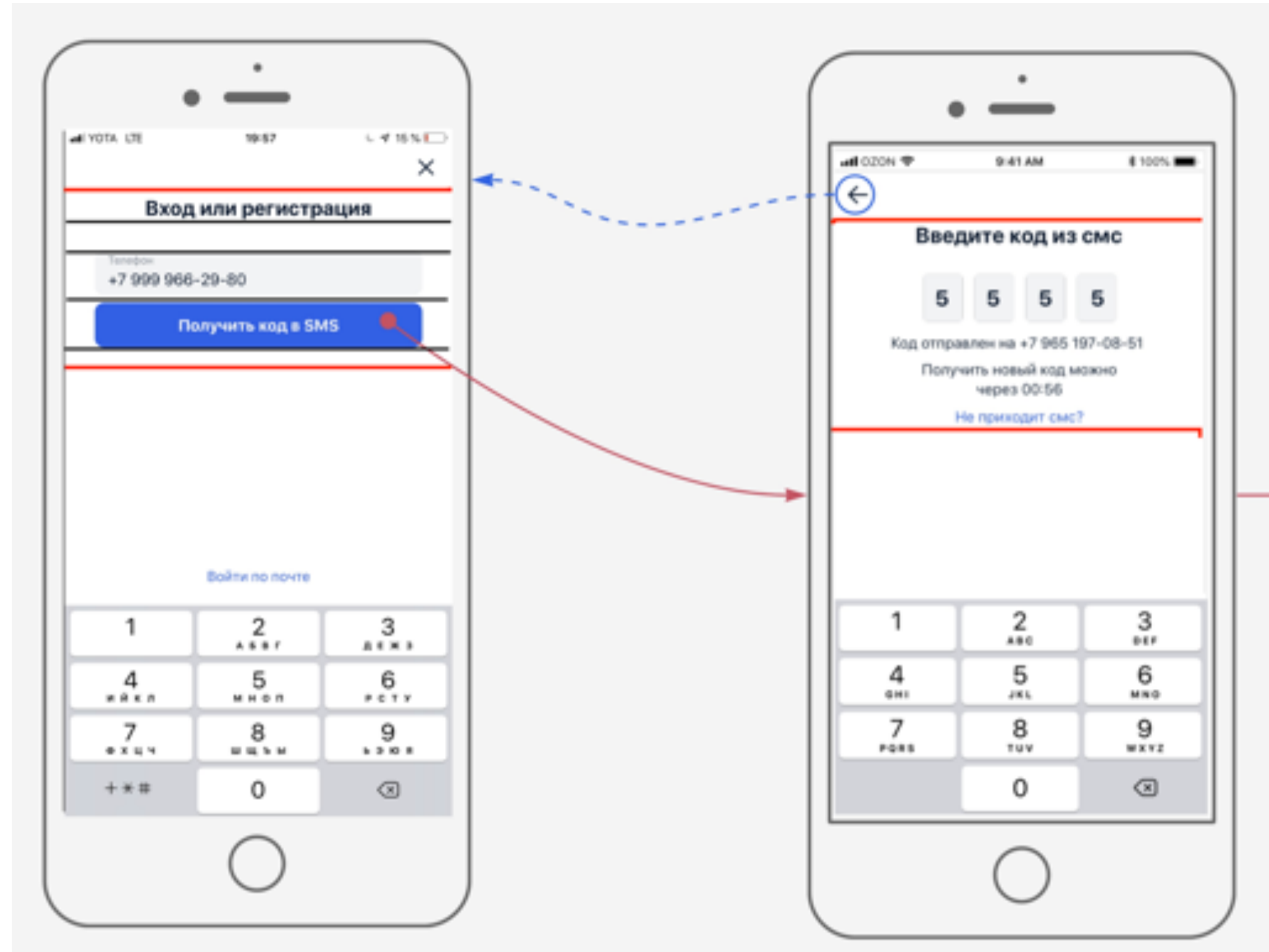
- Наполнение дизайн системы атомами
- Реализация переиспользуемых атомов
- UIKit с атомами, цветами и прочее.
- Нужны ли контейнеры?



**Собираем всё вместе**

# Всё вместе

- Атомы – примитивы дизайн системы
- Виджеты состоят из атомов
- Страницы из атомов
- Навигация из страниц



**OZON**

Спасибо за внимание!

Александр Свиридов



asviridov@ozon.ru



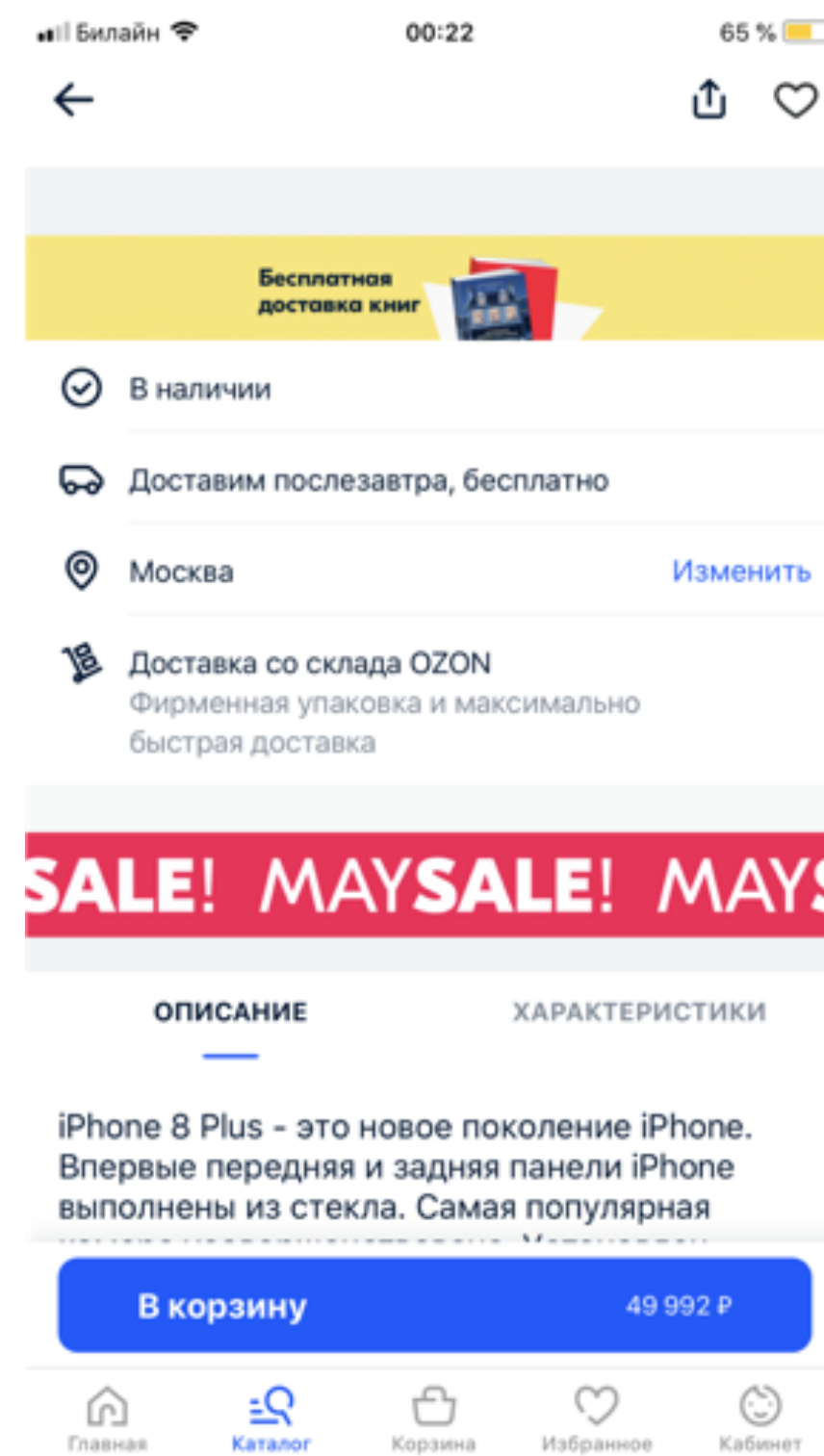
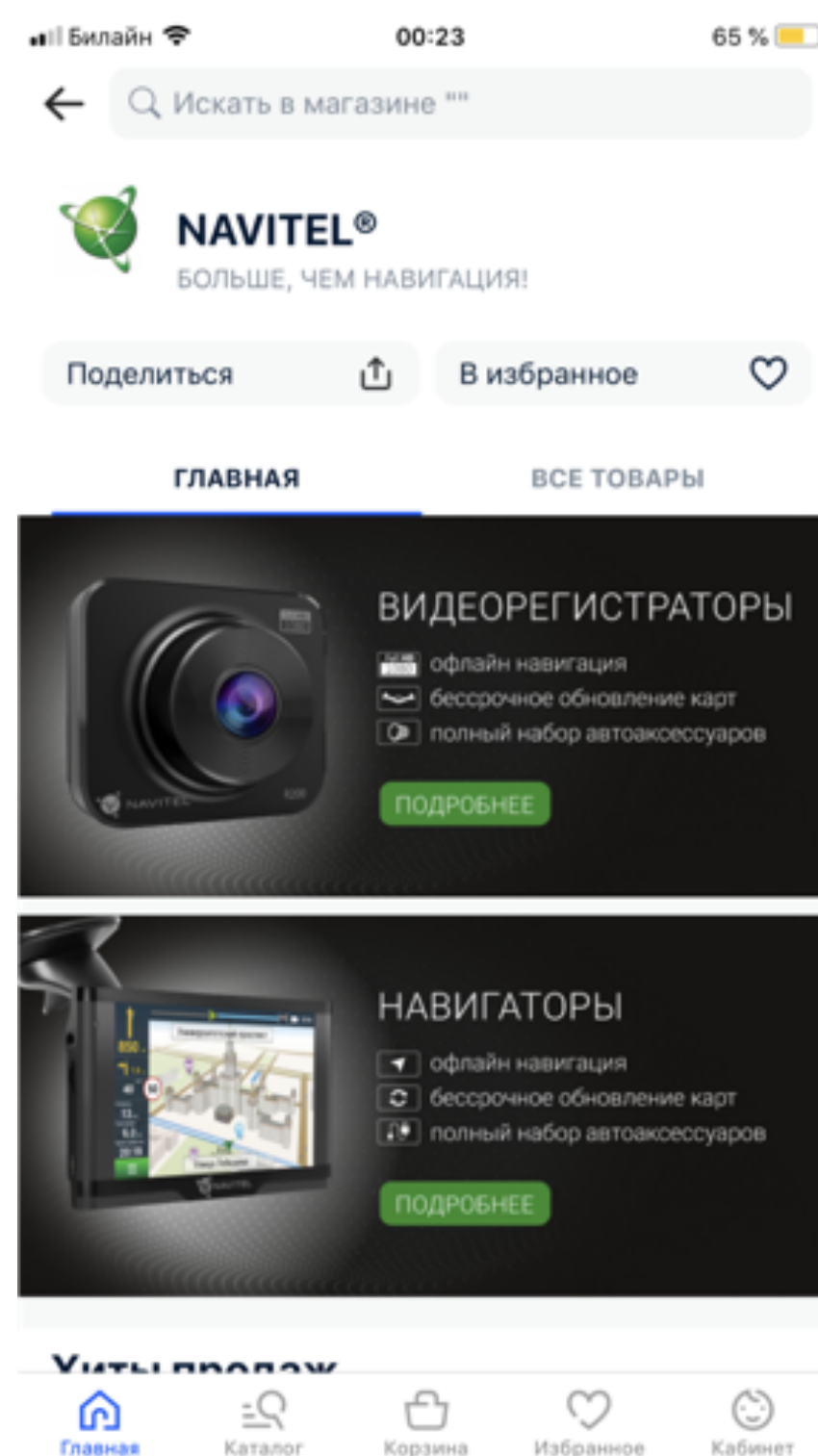
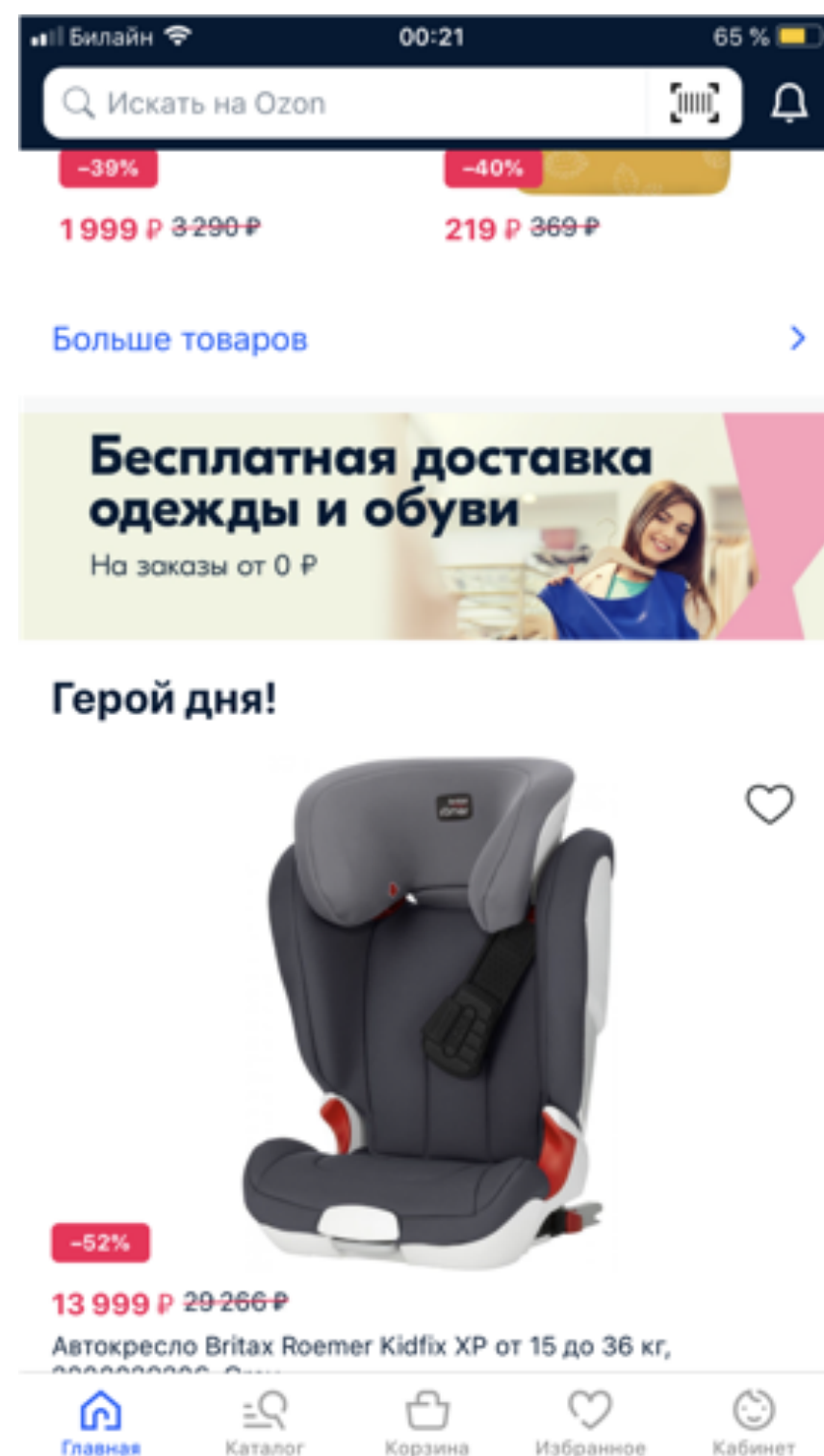
# Реализация на Android

1. Composer framework
2. RecyclerView с GridLayoutManager
3. Большие виджеты могут разбиваться на несколько ViewHolders
4. Общий контейнер виджетов (через DI)

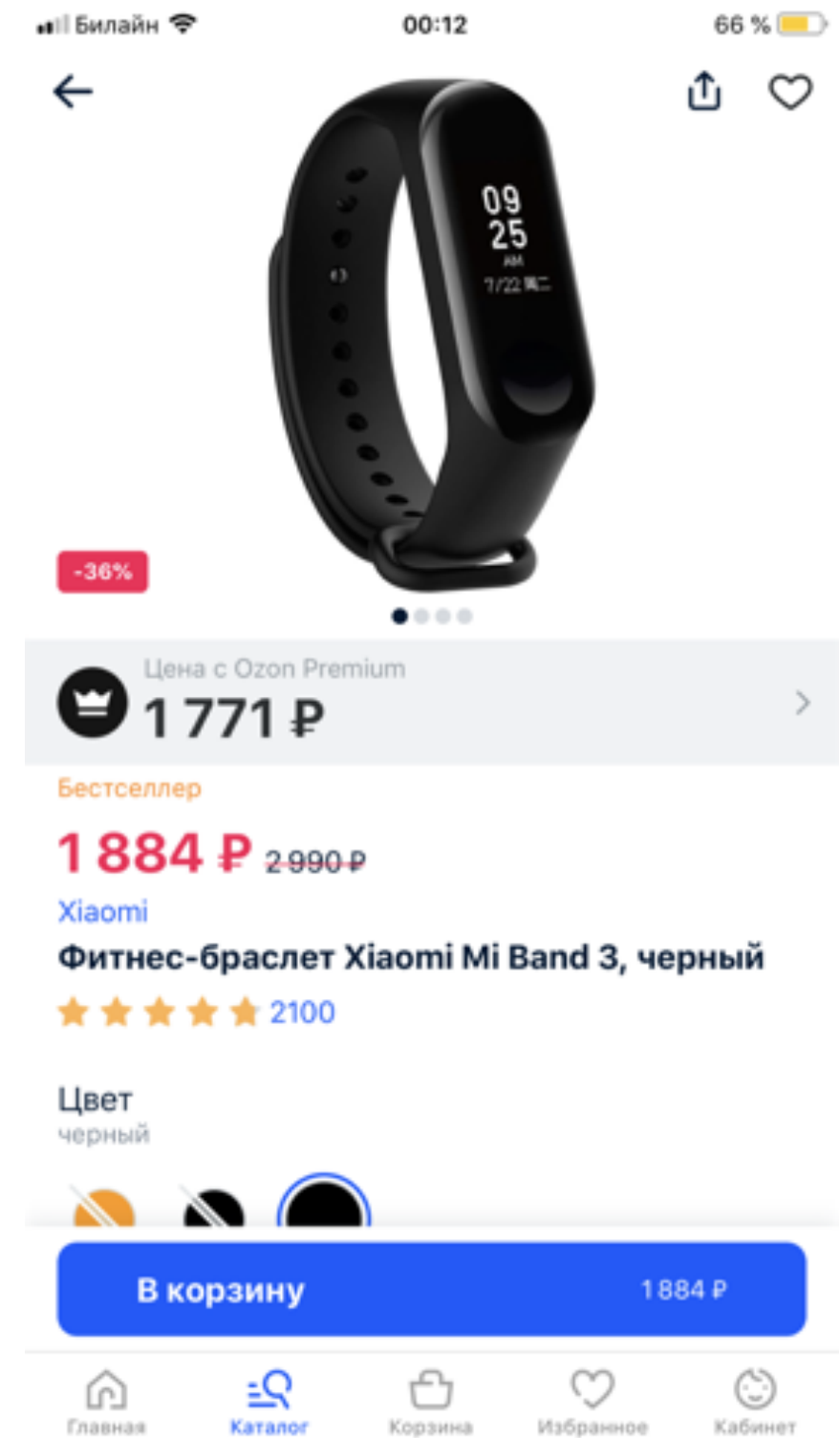
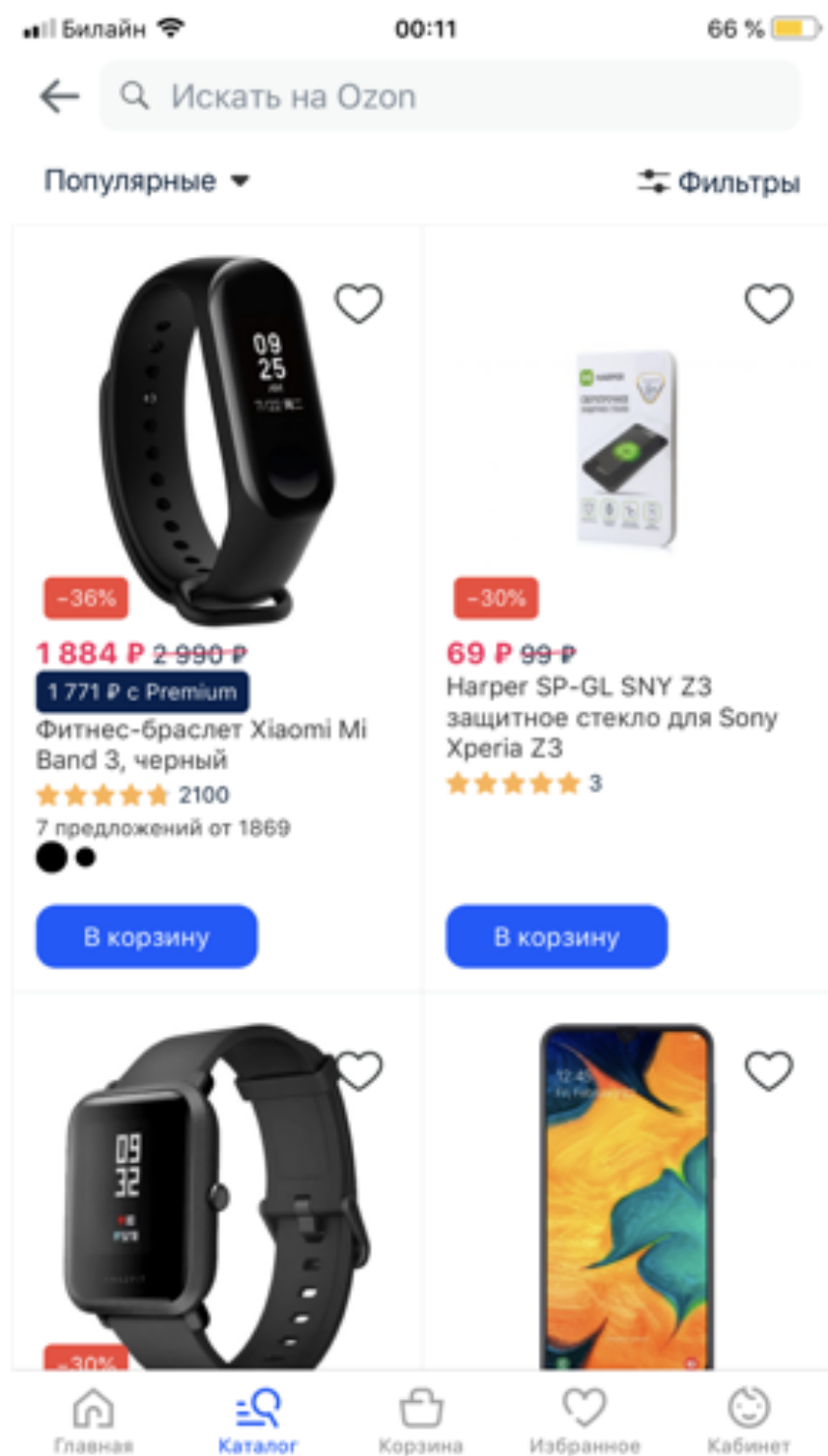
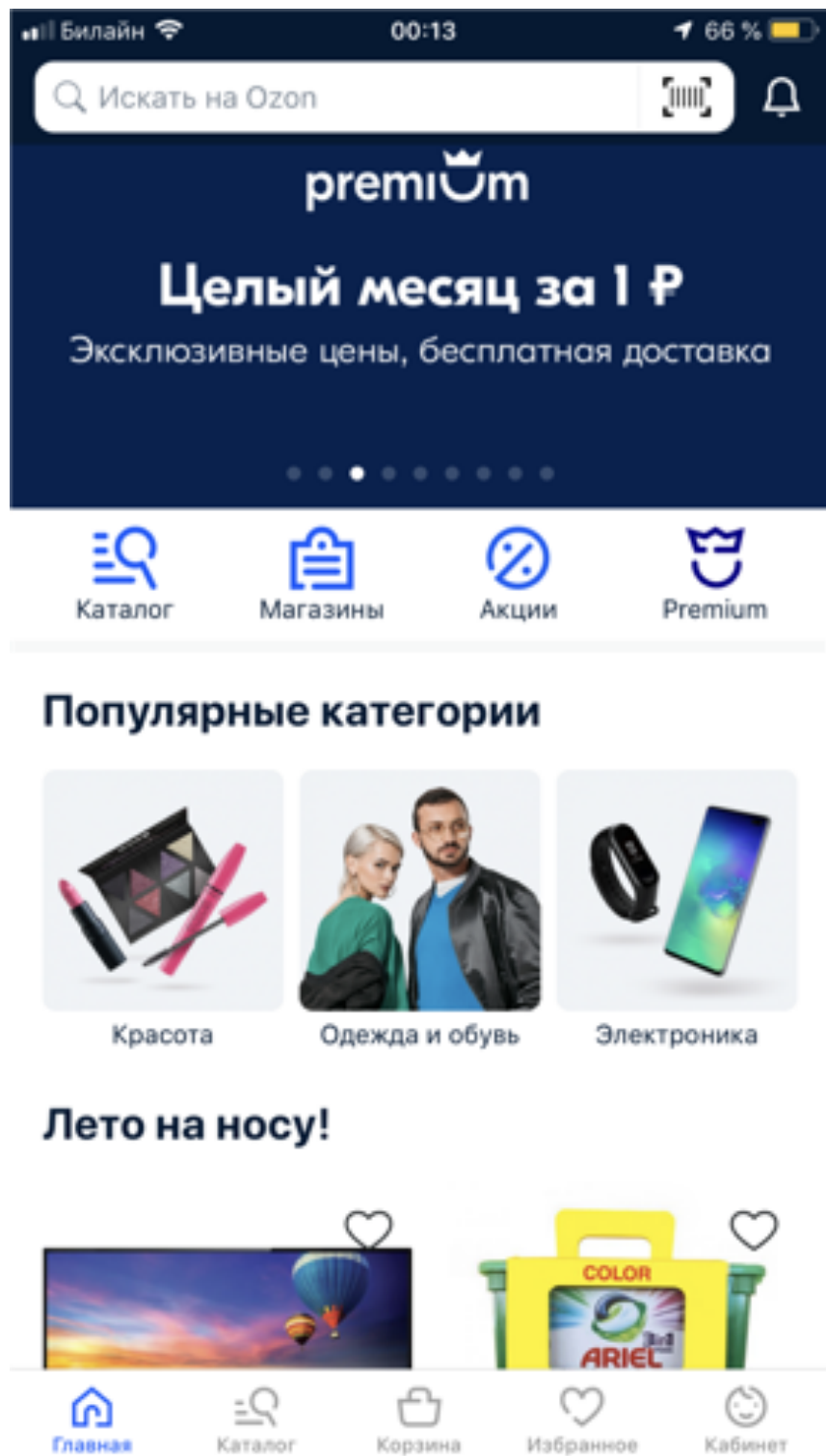


Development experience

# Переиспользование элементов



# Фичи без релиза



# Процесс разработки

1. Создание Config
2. Регистрация виджета в DI
3. Маппинг модели
4. Верстка и бизнес логика

# Создание Config

у нас есть Live Templates

```
interface Config<StateDTO> {  
    val parser: (params: String?, state: String?) -> StateDTO?  
}
```

```
interface ViewMapper<StateDTO : Any?, VO : ViewObject> {  
    val mapper: (StateDTO, WidgetInfo) -> List<VO>  
    val layout: Int  
    val holderProducer: (View, ComposerReferences) -> WidgetViewHolder<VO>?  
}
```

# Регистрируем виджет

```
@Module
abstract class ComposerCommonWidgetsModule {
    @Module
    companion object {

        @ElementsIntoSet
        @JvmStatic
        @Reusable
        @Provides
        fun provideWidget(
            bannerConfig: BannerConfig,
            bannerViewMapper: BannerViewMapper
        ): Set<Widget> =
            hashSetOf(
                Widget(
                    vertical = "cms",
                    component = "banner",
                    config = bannerConfig,
                    viewMappers = arrayOf(bannerViewMapper)
                )
            )
    }
}
```

# Мәппинг модели

```
class BannerMapper @Inject constructor() : WidgetMapper<BannerDTO, BannerVO> {  
  
    override fun invoke(dto: BannerDTO, widgetInfo: WidgetInfo) =  
        BannerVO(  
            id = "${dto.item.image}.${dto.width}.${dto.height}".hashCode().toLong(),  
            image = ImageHelper.getResizedImagePath(dto.item.image),  
            ratio = dto.height.toFloat() / dto.width.toFloat(),  
            url = dto.item.deeplink  
        )  
}
```

# Верстка и бизнес логика

```
class BannerViewHolder(  
    private val router: ScreenRouter,  
    private val analytics: WidgetAnalytics  
) : WidgetViewHolder<BannerVO>(), LayoutContainer {  
  
    override fun bind(item: BannerVO, info: WidgetInfo) {  
  
        (itemView as AspectRatioImageView).apply {  
            ratio = item.ratio  
            this.loadAsBitmap(item.image)  
        }  
  
        itemView.setOnClickListener { _ ->  
            analytics.bannerClick(info.component, item.url.orEmpty())  
            router.openDeeplink(deeplink = item.url!!, context = itemView.context)  
        }  
    }  
}
```



# Итог

1. Верхнеуровневый фреймворк
2. Простой контракт
3. Разработка до BE (Charles / Ghost Widget)
4. Onboarding