AppCraft

Faster than a speeding release train

#MobiusConf @dammitandy

zalando

Outline

Outline

1. The problem

Outline

1. The problem
2. Our solution

Outline

1. The problem
2. Our solution
3. Mobile client architecture

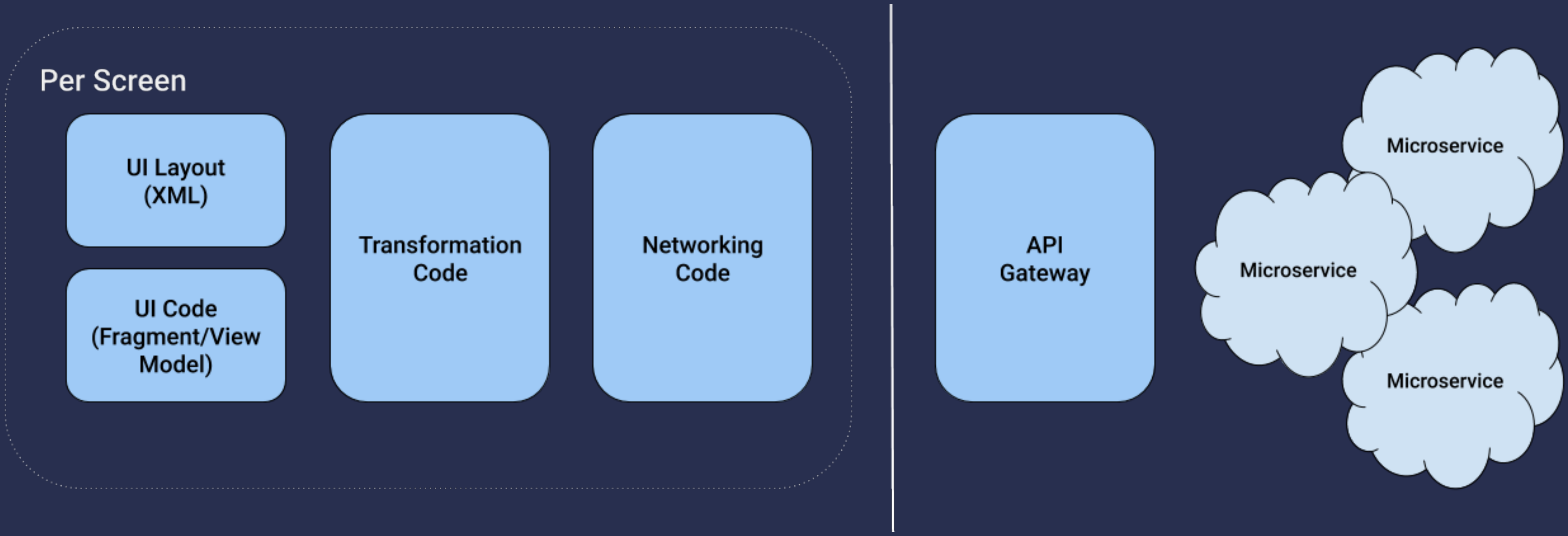
Outline

1. The problem
2. Our solution
3. Mobile client architecture
4. Demo

Outline

1. The problem
2. Our solution
3. Mobile client architecture
4. Demo
5. Future plans & lessons learned

Traditional App



What is the lifecycle of an app change?

Meeting

Meeting, ticket

Meeting, ticket, code

Meeting, ticket, code, test

Meeting, ticket, code, test, pull request

Meeting

ticket, code, test, pull request, discussion

Meeting, ticket, code, test, pull request, discussion, merge PR

Meeting, ticket, code, test, pull request, discussion, merge PR, regression/QA

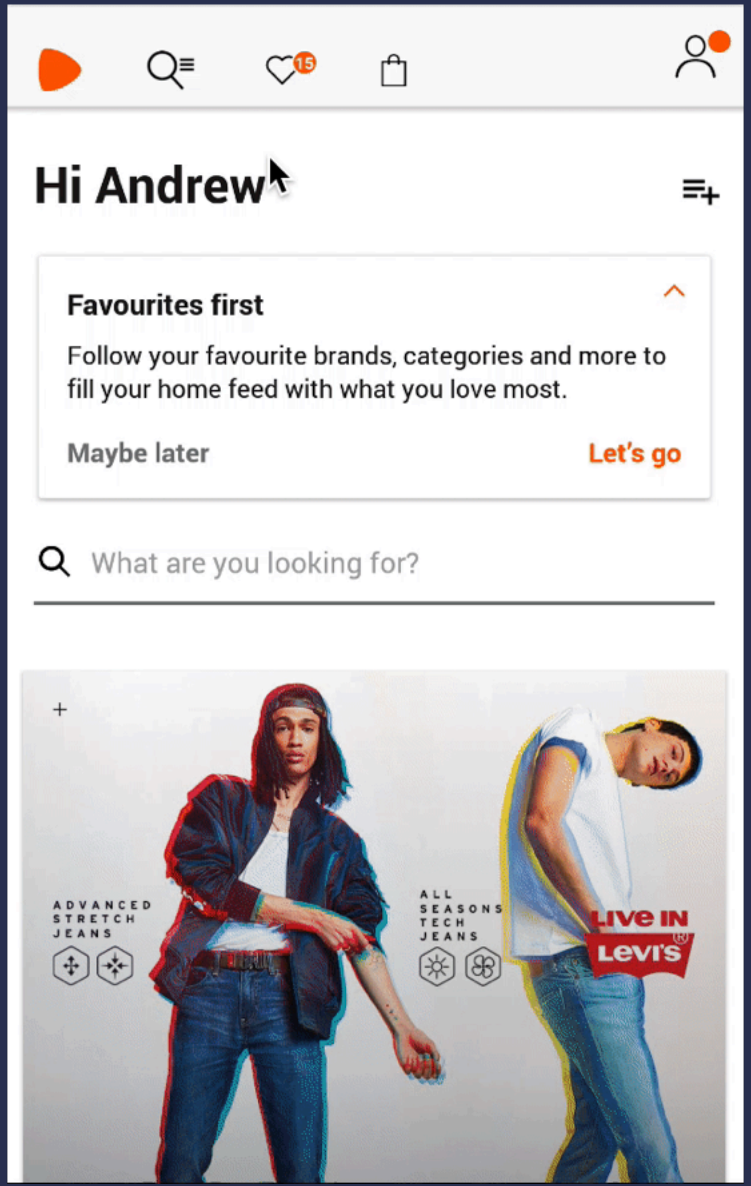
Meeting, ticket, code, test, pull request, discussion,

merge PR, regression/QA, app release

React Native + CodePush

Cross-platform integration with existing apps is hard.

Home Screen


Hi Andrew

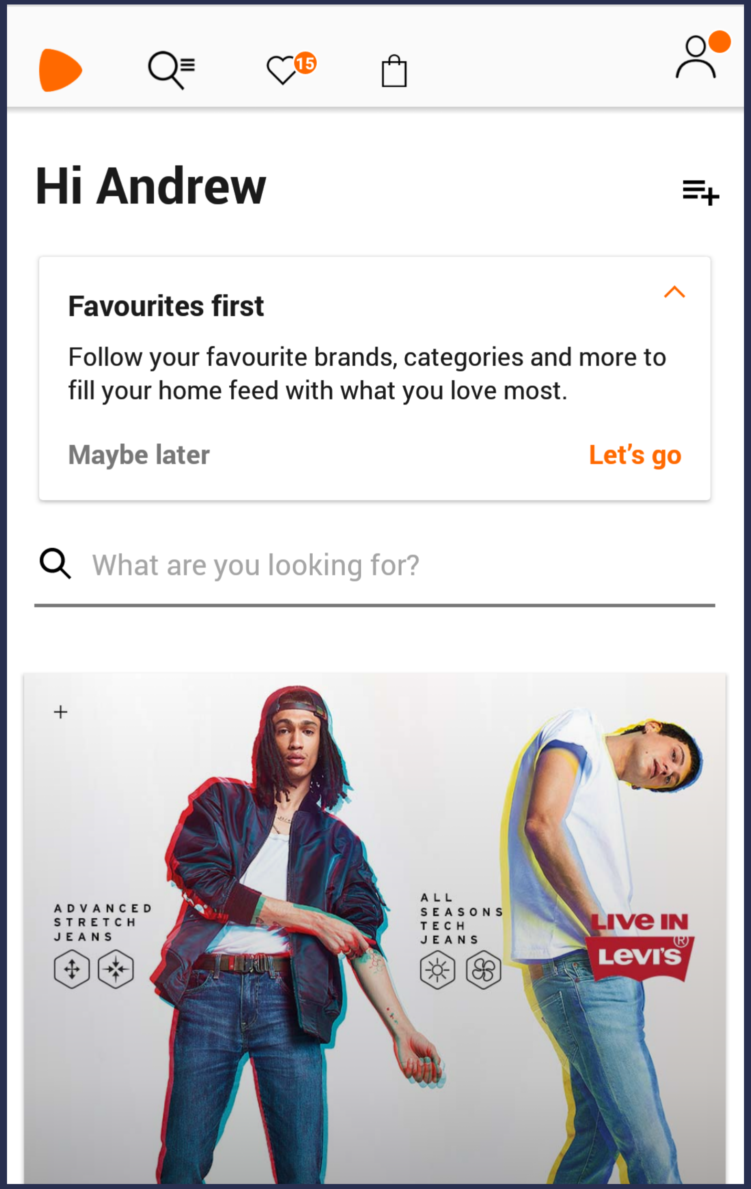
Favourites first

Follow your favourite brands, categories and more to fill your home feed with what you love most.

Maybe later Let's go

What are you looking for?

Home Screen



Hi Andrew

Favourites first

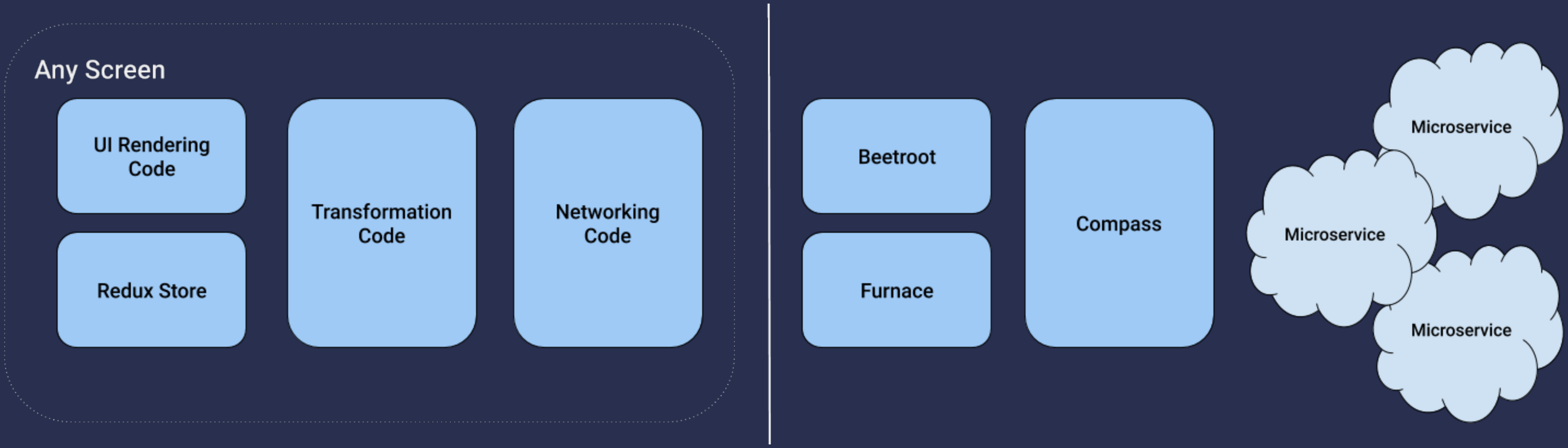
Follow your favourite brands, categories and more to fill your home feed with what you love most.

Maybe later

Let's go

What are you looking for?

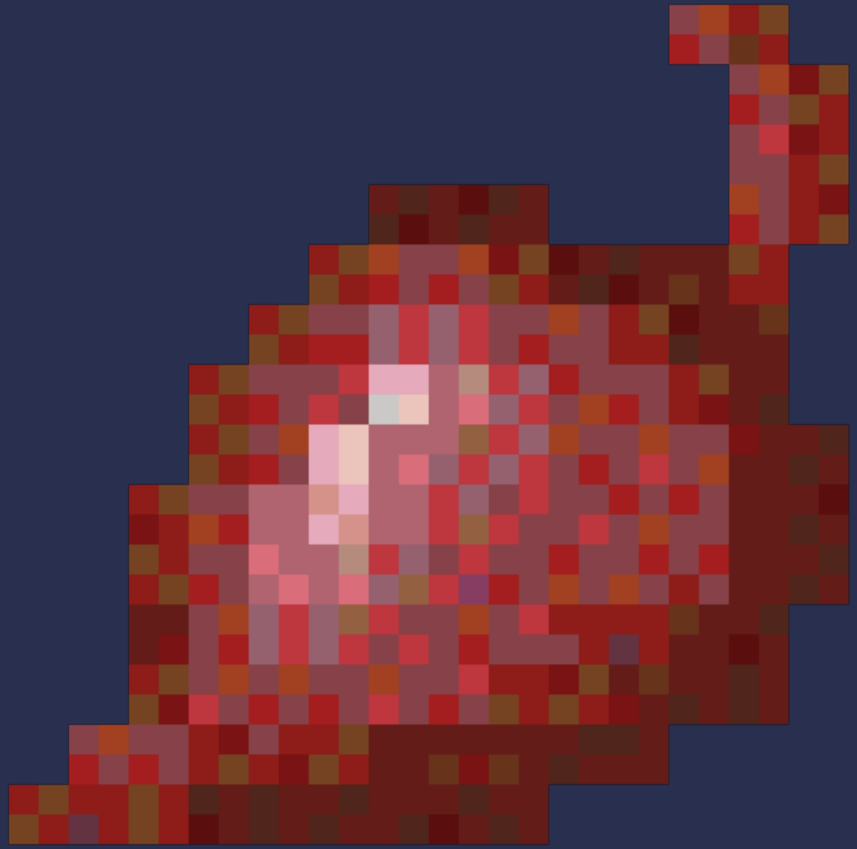
AppCraft



Any Screen
UI Rendering Code
Redux Store
Transformation Code
Networking Code
Beetroot
Furnace
Compass
Microservice
Microservice
Microservice

Compass

Furnace & Beetroot



Apps Don't Care

Apps Don't Care

- Layout variants - phone vs. tablet, country, A/B tests

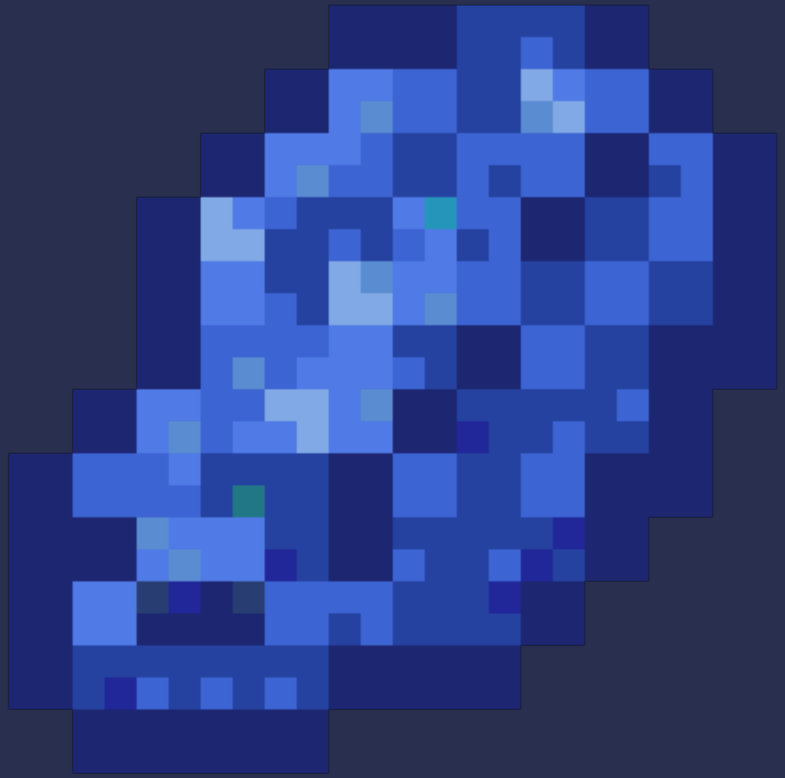
Apps Don't Care

- Layout variants - phone vs. tablet, country, A/B tests
- Localization

Apps Don't Care

- Layout variants - phone vs. tablet, country, A/B tests
- Localization
- Tracking & analytics

Golem & Lapis



API Endpoints

API Endpoints

1. Configuration

API Endpoints

1. Configuration
2. Layout

API Endpoints

1. Configuration
2. Layout
3. Data

API Endpoints

1. Configuration
2. Layout
3. Data
4. Component Data

Layout Response

```
{
  "screen_id": "example-screen",
  "component": {
    "component_id": "root-component",
    "type": "layout",
    "children": [
      {
        "component_id": "hello-text",
        "type": "text",
        "options": {
          "text": "Placeholder text"
        },
        "style": { ... },
        "events: [ ... ]
      }
    ]
  }
}
```

Layout Response

```
{
  "screen_id": "example-screen",
  "component": {
    "component_id": "root-component",
    "type": "layout",
    "children": [
      {
        "component_id": "hello-text",
        "type": "text",
        "options": {
          "text": "Placeholder text"
        },
        "style": { ... },
        "events: [ ... ]
      }
    ]
  }
}
```

Layout Response

```
{
  "screen_id": "example-screen",
  "component": {
    "component_id": "root-component",
    "type": "layout",
    "children": [
      {
        "component_id": "hello-text",
        "type": "text",
        "options": {
          "text": "Placeholder text"
        },
        "style": { ... },
        "events: [ ... ]
      }
    ]
  }
}
```

Layout Response

```
{
  "screen_id": "example-screen",
  "component": {
    "component_id": "root-component",
    "type": "layout",
    "children": [
      {
        "component_id": "hello-text",
        "type": "text",
        "options": {
          "text": "Placeholder text"
        },
        "style": { ... },
        "events: [ ... ]
      }
    ]
  }
}
```

Layout Response

```
{
  "screen_id": "example-screen",
  "component": {
    "component_id": "root-component",
    "type": "layout",
    "children": [
      {
        "component_id": "hello-text",
        "type": "text",
        "options": {
          "text": "Placeholder text"
        },
        "style": { ... },
        "events: [ ... ]
      }
    ]
  }
}
```

Layout Response

```
{
  "screen_id": "example-screen",
  "component": {
    "component_id": "root-component",
    "type": "layout",
    "children": [
      {
        "component_id": "hello-text",
        "type": "text",
        "options": {
          "text": "Placeholder text"
        },
        "style": { ... },
        "events: [ ... ]
      }
    ]
  }
}
```

Layout Response

```
{
  "screen_id": "example-screen",
  "component": {
    "component_id": "root-component",
    "type": "layout",
    "children": [
      {
        "component_id": "hello-text",
        "type": "text",
        "options": {
          "text": "Placeholder text"
        },
        "style": { ... },
        "events: [ ... ]
      }
    ]
  }
}
```

Layout Response

```
{
  "screen_id": "example-screen",
  "component": {
    "component_id": "root-component",
    "type": "layout",
    "children": [
      {
        "component_id": "hello-text",
        "type": "text",
        "options": {
          "text": "Placeholder text"
        },
        "style": { ... },
        "events: [ ... ]
      }
    ]
  }
}
```

Layout Response

```
{
  "screen_id": "example-screen",
  "component": {
    "component_id": "root-component",
    "type": "layout",
    "children": [
      {
        "component_id": "hello-text",
        "type": "text",
        "options": {
          "text": "Placeholder text"
        },
        "style": { ... },
        "events: [ ... ]
      }
    ]
  }
}
```

Layout Response

```
{
  "screen_id": "example-screen",
  "component": {
    "component_id": "root-component",
    "type": "layout",
    "children": [
      {
        "component_id": "hello-text",
        "type": "text",
        "options": {
          "text": "Placeholder text"
        },
        "style": { ... },
        "events: [ ... ]
      }
    ]
  }
}
```

Layout Response

```
{
  "screen_id": "example-screen",
  "component": {
    "component_id": "root-component",
    "type": "layout",
    "children": [
      {
        "component_id": "hello-text",
        "type": "text",
        "options": {
          "text": "Placeholder text"
        },
        "style": { ... },
        "events: [ ... ]
      }
    ]
  }
}
```

Layout Response

```
{
  "screen_id": "example-screen",
  "component": {
    "component_id": "root-component",
    "type": "layout",
    "children": [
      {
        "component_id": "hello-text",
        "type": "text",
        "options": {
          "text": "Placeholder text"
        },
        "style": { ... },
        "events: [ ... ]
      }
    ]
  }
}
```

Data Response

```
{
  "component": {
    "component_id": "root-component",
    "items": [
      {
        "component_id": "hello-text",
        "options": {
          "text": "Hello, MobiusConf!"
        }
      }
    ]
  }
}
```

Data Response

```
{
  "component": {
    "component_id": "root-component",
    "items": [
      {
        "component_id": "hello-text",
        "options": {
          "text": "Hello, MobiusConf!"
        }
      }
    ]
  }
}
```

Data Response

```
{
  "component": {
    "component_id": "root-component",
    "items": [
      {
        "component_id": "hello-text",
        "options": {
          "text": "Hello, MobiusConf!"
        }
      }
    ]
  }
}
```

Data Response

```
{
  "component": {
    "component_id": "root-component",
    "items": [
      {
        "component_id": "hello-text",
        "options": {
          "text": "Hello, MobiusConf!"
        }
      }
    ]
  }
}
```

Data Response

```
{
  "component": {
    "component_id": "root-component",
    "items": [
      {
        "component_id": "hello-text",
        "options": {
          "text": "Hello, MobiusConf!"
        }
      }
    ]
  }
}
```

Data Response

```
{
  "component": {
    "component_id": "root-component",
    "items": [
      {
        "component_id": "hello-text",
        "options": {
          "text": "Hello, MobiusConf!"
        }
      }
    ]
  }
}
```

Data Response

```
{
  "component": {
    "component_id": "root-component",
    "items": [
      {
        "component_id": "hello-text",
        "options": {
          "text": "Hello, MobiusConf!"
        }
      }
    ]
  }
}
```

Litho & Texture

Litho: Component Example

```
@LayoutSpec
public object LithoComponentSpec {

  @OnCreateLayout
  fun onCreateLayout(c: ComponentContext): Component =
    Text.create(c)
        .text("Hello, MobiusConf!")
        .textSizeSp(40))
        .build();
}
```

Litho: Component Example

```
@LayoutSpec
public object LithoComponentSpec {

  @OnCreateLayout
  fun onCreateLayout(c: ComponentContext): Component =
    Text.create(c)
        .text("Hello, MobiusConf!")
        .textSizeSp(40))
        .build();
}
```

Litho: Component Example

```
@LayoutSpec
public object LithoComponentSpec {

  @OnCreateLayout
  fun onCreateLayout(c: ComponentContext): Component =
    Text.create(c)
        .text("Hello, MobiusConf!")
        .textSizeSp(40))
        .build();
}
```

Litho: Component Example

```
@LayoutSpec
public object LithoComponentSpec {

  @OnCreateLayout
  fun onCreateLayout(c: ComponentContext): Component =
    Text.create(c)
        .text("Hello, MobiusConf!")
        .textSizeSp(40))
        .build();
}
```

Litho: Component Example

```
@LayoutSpec
public object LithoComponentSpec {

  @OnCreateLayout
  fun onCreateLayout(c: ComponentContext): Component =
    Text.create(c)
        .text("Hello, MobiusConf!")
        .textSizeSp(40))
        .build();
}
```

Litho: Component Example

```
@LayoutSpec
public object LithoComponentSpec {

  @OnCreateLayout
  fun onCreateLayout(c: ComponentContext): Component =
    Text.create(c)
        .text("Hello, MobiusConf!")
        .textSizeSp(40))
        .build();
}
```

Litho: Component Example

```
@LayoutSpec
public object LithoComponentSpec {

  @OnCreateLayout
  fun onCreateLayout(c: ComponentContext): Component =
    Text.create(c)
        .text("Hello, MobiusConf!")
        .textSizeSp(40))
        .build();
}
```

Litho: Wrapping a Native View

```
@MountSpec
public object ColorComponentSpec {

  @OnCreateMountContent
  fun onCreateMountContent(Context c): ColorDrawable = ColorDrawable();

  @OnMount
  fun onMount(
      context: ComponentContext,
      colorDrawable: ColorDrawable,
      @Prop colorName: String) {
    colorDrawable.color = Color.parseColor(colorName)
  }
}
```

Litho: Wrapping a Native View

```
@MountSpec
public object ColorComponentSpec {

  @OnCreateMountContent
  fun onCreateMountContent(Context c): ColorDrawable = ColorDrawable();

  @OnMount
  fun onMount(
      context: ComponentContext,
      colorDrawable: ColorDrawable,
      @Prop colorName: String) {
    colorDrawable.color = Color.parseColor(colorName)
  }
}
```

Litho: Wrapping a Native View

```
@MountSpec
public object ColorComponentSpec {

  @OnCreateMountContent
  fun onCreateMountContent(Context c): ColorDrawable = ColorDrawable();

  @OnMount
  fun onMount(
     context: ComponentContext,
     colorDrawable: ColorDrawable,
     @Prop colorName: String) {
    colorDrawable.color = Color.parseColor(colorName)
  }
}
```

Litho: Wrapping a Native View

```
@MountSpec
public object ColorComponentSpec {

  @OnCreateMountContent
  fun onCreateMountContent(Context c): ColorDrawable = ColorDrawable();

  @OnMount
  fun onMount(
     context: ComponentContext,
     colorDrawable: ColorDrawable,
     @Prop colorName: String) {
    colorDrawable.color = Color.parseColor(colorName)
  }
}
```

Litho: Wrapping a Native View

```
@MountSpec
public object ColorComponentSpec {

  @OnCreateMountContent
  fun onCreateMountContent(Context c): ColorDrawable = ColorDrawable();

  @OnMount
  fun onMount(
     context: ComponentContext,
     colorDrawable: ColorDrawable,
     @Prop colorName: String) {
    colorDrawable.color = Color.parseColor(colorName)
  }
}
```

Litho: Wrapping a Native View

```
@MountSpec
public object ColorComponentSpec {

  @OnCreateMountContent
  fun onCreateMountContent(Context c): ColorDrawable = ColorDrawable();

  @OnMount
  fun onMount(
     context: ComponentContext,
     colorDrawable: ColorDrawable,
     @Prop colorName: String) {
    colorDrawable.color = Color.parseColor(colorName)
  }
}
```

Litho: Wrapping a Native View

```
@MountSpec
public object ColorComponentSpec {

  @OnCreateMountContent
  fun onCreateMountContent(Context c): ColorDrawable = ColorDrawable();

  @OnMount
  fun onMount(
     context: ComponentContext,
     colorDrawable: ColorDrawable,
     @Prop colorName: String) {
    colorDrawable.color = Color.parseColor(colorName)
  }
}
```

Litho: Wrapping a Native View

```
@MountSpec
public object ColorComponentSpec {

  @OnCreateMountContent
  fun onCreateMountContent(Context c): ColorDrawable = ColorDrawable();

  @OnMount
  fun onMount(
      context: ComponentContext,
      colorDrawable: ColorDrawable,
      @Prop colorName: String) {
    colorDrawable.color = Color.parseColor(colorName)
  }
}
```

Litho: Wrapping a Native View

```
@MountSpec
public object ColorComponentSpec {

  @OnCreateMountContent
  fun onCreateMountContent(Context c): ColorDrawable = ColorDrawable();

  @OnMount
  fun onMount(
     context: ComponentContext,
     colorDrawable: ColorDrawable,
     @Prop colorName: String) {
    colorDrawable.color = Color.parseColor(colorName)
  }
}
```

Litho: Wrapping a Native View

```
@MountSpec
public object ColorComponentSpec {

  @OnCreateMountContent
  fun onCreateMountContent(Context c): ColorDrawable = ColorDrawable();

  @OnMount
  fun onMount(
     context: ComponentContext,
     colorDrawable: ColorDrawable,
     @Prop colorName: String) {
    colorDrawable.color = Color.parseColor(colorName)
  }
}
```

Using a Litho Component

```
fun createView(context: Context): View {
  val c = ComponentContext(context)
  val component =
    LithoComponent.create(c)
      .background(
        ColorComponent.create(c).colorName("blue")
      )
      .build()

  return LithoView.create(c, component))
}
```

Using a Litho Component

```
fun createView(context: Context): View {
  val c = ComponentContext(context)
  val component =
    LithoComponent.create(c)
      .background(
        ColorComponent.create(c).colorName("blue")
      )
      .build()

  return LithoView.create(c, component))
}
```

Using a Litho Component

```
fun createView(context: Context): View {
  val c = ComponentContext(context)
  val component =
    LithoComponent.create(c)
      .background(
        ColorComponent.create(c).colorName("blue")
      )
      .build()

  return LithoView.create(c, component))
}
```

Using a Litho Component

```
fun createView(context: Context): View {
  val c = ComponentContext(context)
  val component =
    LithoComponent.create(c)
      .background(
        ColorComponent.create(c).colorName("blue")
      )
      .build()

  return LithoView.create(c, component))
}
```

Using a Litho Component

```
fun createView(context: Context): View {
  val c = ComponentContext(context)
  val component =
    LithoComponent.create(c)
      .background(
        ColorComponent.create(c).colorName("blue")
      )
      .build()

  return LithoView.create(c, component))
}
```

Using a Litho Component

```
fun createView(context: Context): View {
  val c = ComponentContext(context)
  val component =
    LithoComponent.create(c)
      .background(
        ColorComponent.create(c).colorName("blue")
      )
      .build()

  return LithoView.create(c, component))
}
```

Using a Litho Component

```
fun createView(context: Context): View {
  val c = ComponentContext(context)
  val component =
    LithoComponent.create(c)
      .background(
        ColorComponent.create(c).colorName("blue")
      )
      .build()

  return LithoView.create(c, component))
}
```

Using a Litho Component

```
fun createView(context: Context): View {
  val c = ComponentContext(context)
  val component =
    LithoComponent.create(c)
      .background(
        ColorComponent.create(c).colorName("blue")
      )
      .build()

  return LithoView.create(c, component))
}
```

Using a Litho Component

```
fun createView(context: Context): View {
  val c = ComponentContext(context)
  val component =
    LithoComponent.create(c)
      .background(
        ColorComponent.create(c).colorName("blue")
      )
      .build()

  return LithoView.create(c, component))
}
```

Basic Components

Basic Components

- Text

Basic Components

- Text
- Button

Basic Components

- Text
- Button
- Image

Basic Components (cont.)

Basic Components (cont.)

- Layout

Basic Components (cont.)

- Layout
- Repeater

Basic Components (cont.)

- Layout
- Repeater
- Parallax Layout

Specialized Components

Specialized Components

- Wish List button

Specialized Components

- Wish List button
- Cart button

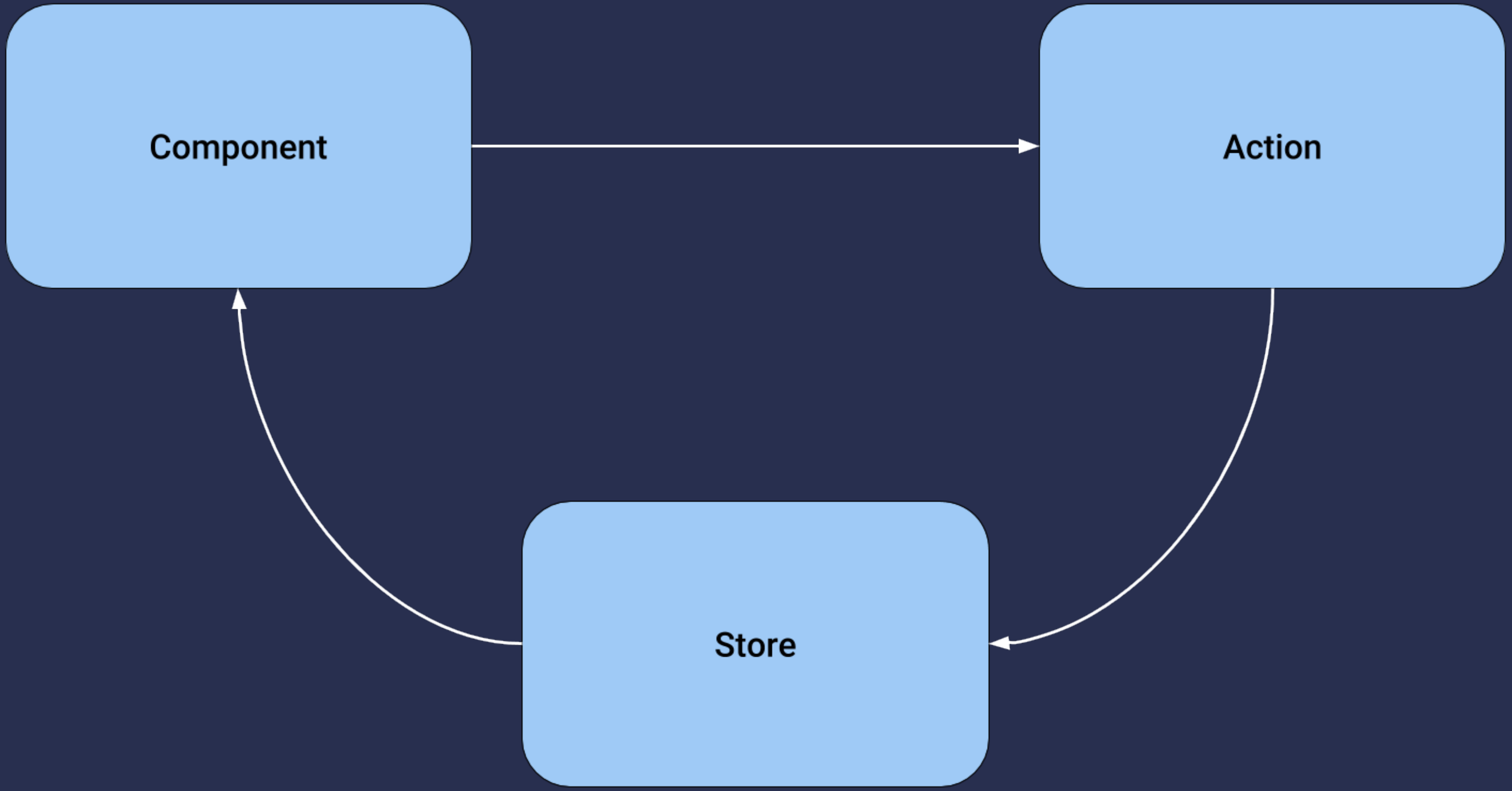
Specialized Components

- Wish List button
- Cart button
- Follow button

Specialized Components

- Wish List button
- Cart button
- Follow button
- Price

Redux Architecture



Redux/MVI Android Libraries

Redux/MVI Android Libraries

- RxRedux

Redux/MVI Android Libraries

- RxRedux
- Mobius

Redux/MVI Android Libraries

- RxRedux
- Mobius
- MvRx

Rendering a Screen

Rendering a Screen

1. Layout API request

Rendering a Screen

1. Layout API request
2. Render layout with placeholders and/or defaults

Rendering a Screen

1. Layout API request
2. Render layout with placeholders and/or defaults
3. Data API request

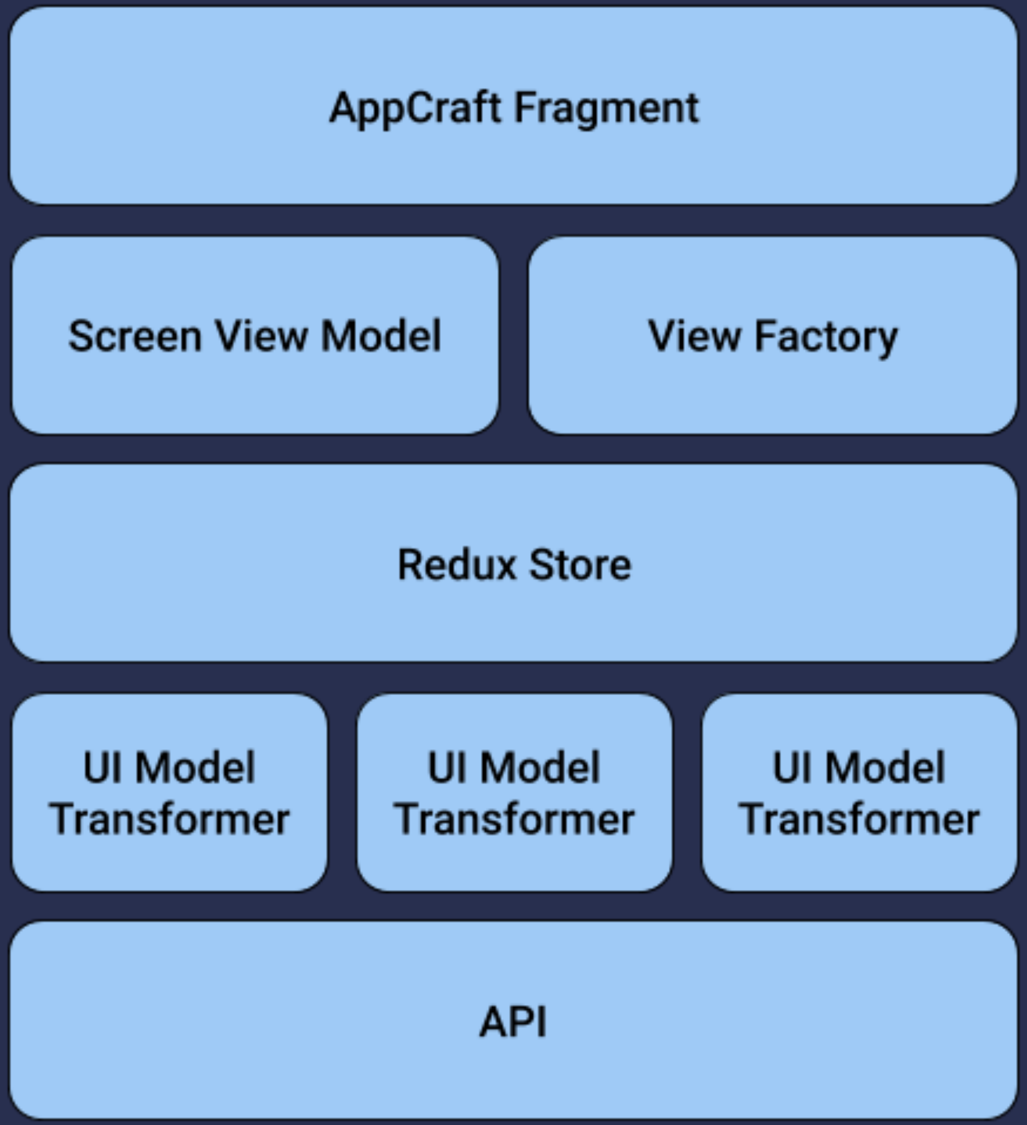
Rendering a Screen

1. Layout API request
2. Render layout with placeholders and/or defaults
3. Data API request
4. Rerender layout bound with data and events

Rendering a Screen

1. Layout API request
2. Render layout with placeholders and/or defaults
3. Data API request
4. Rerender layout bound with data and events
5. Additional data requests, if necessary

Android Architecture



Testing & CI

Testing & CI

- Static analysis - code styles, lint, public API, etc.

Testing & CI

- Static analysis - code styles, lint, public API, etc.
- Unit tests

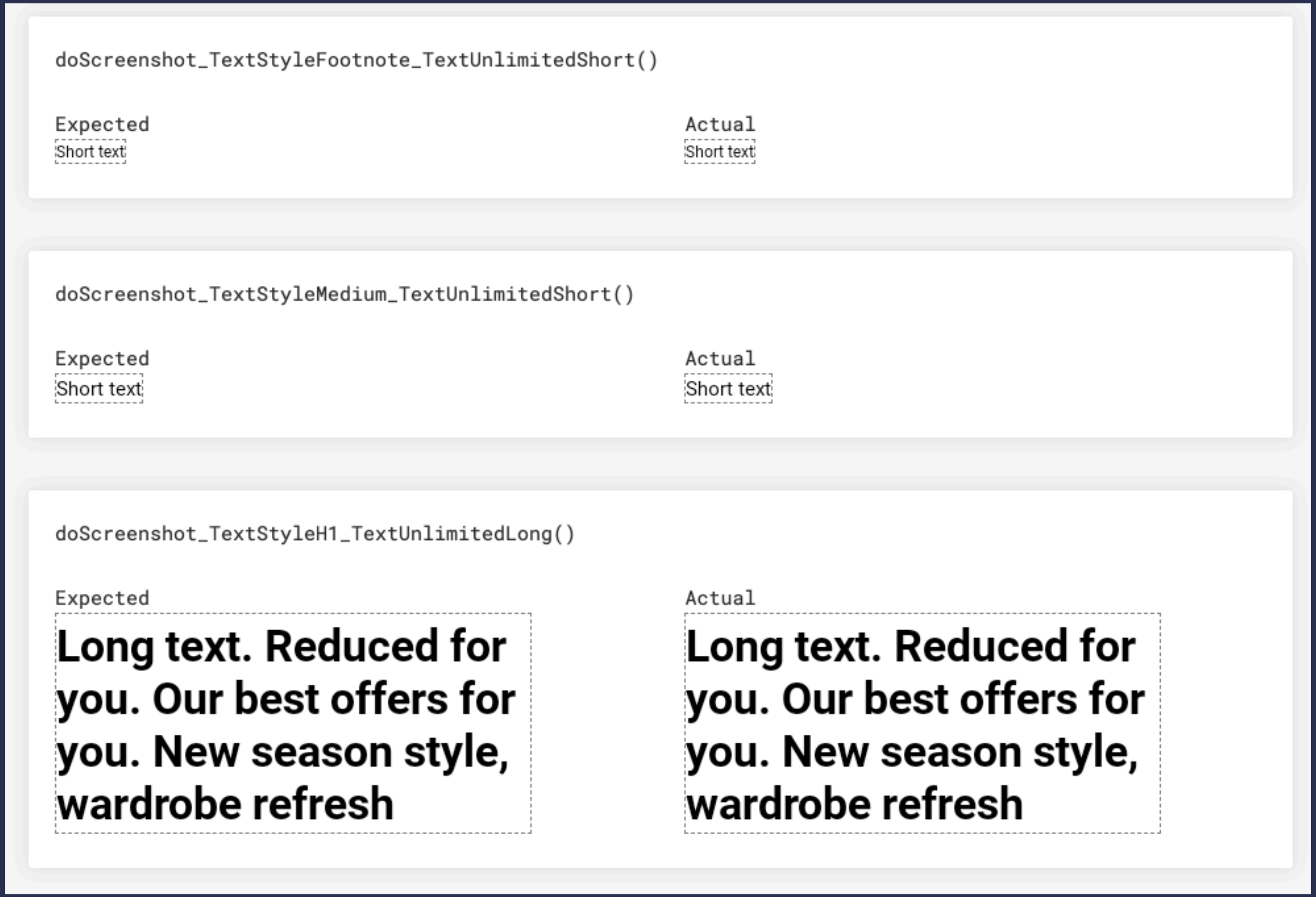
Testing & CI

- Static analysis - code styles, lint, public API, etc.
- Unit tests
- Screenshot tests

Testing & CI

- Static analysis - code styles, lint, public API, etc.
- Unit tests
- Screenshot tests
- End-to-end/integration tests

TestCraft



doScreenshot_TextStyleFootnote_TextUnlimitedShort()

Expected
Short text

Actual
Short text

doScreenshot_TextStyleMedium_TextUnlimitedShort()

Expected
Short text

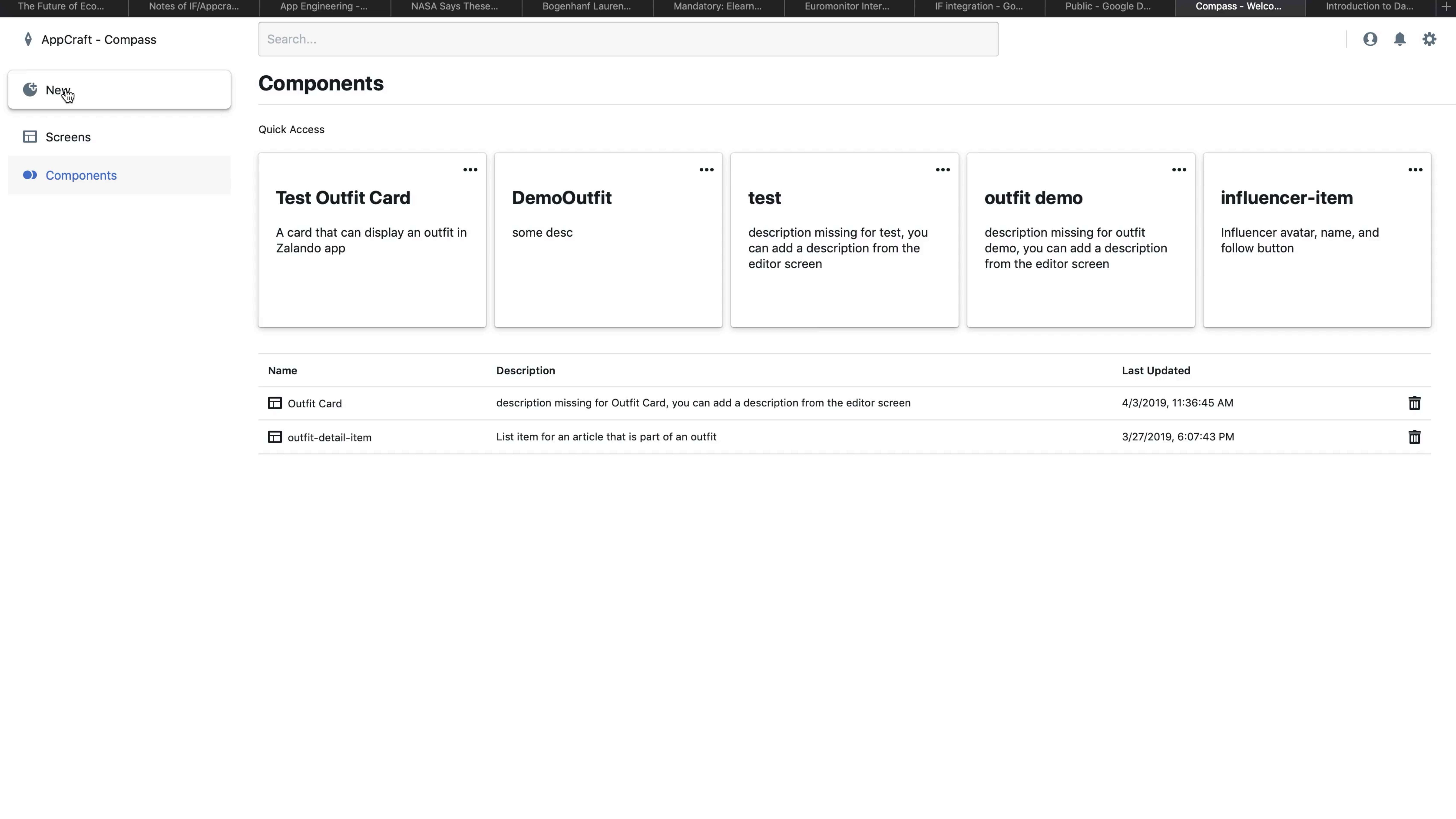
Actual
Short text

doScreenshot_TextStyleH1_TextUnlimitedLong()

Expected
Long text. Reduced for you. Our best offers for you. New season style, wardrobe refresh

Actual
Long text. Reduced for you. Our best offers for you. New season style, wardrobe refresh

Demo



The Future of Eco... | Notes of IF/Appcra... | App Engineering -... | NASA Says These... | Bogenhanf Lauren... | Mandatory: Elearn... | Euromonitor Inter... | IF integration - Go... | Public - Google D... | Compass - Welco... | Introduction to Da...

AppCraft - Compass

Search...

New

Screens

Components

Components

Quick Access

Test Outfit Card

A card that can display an outfit in Zalando app

DemoOutfit

some desc

test

description missing for test, you can add a description from the editor screen

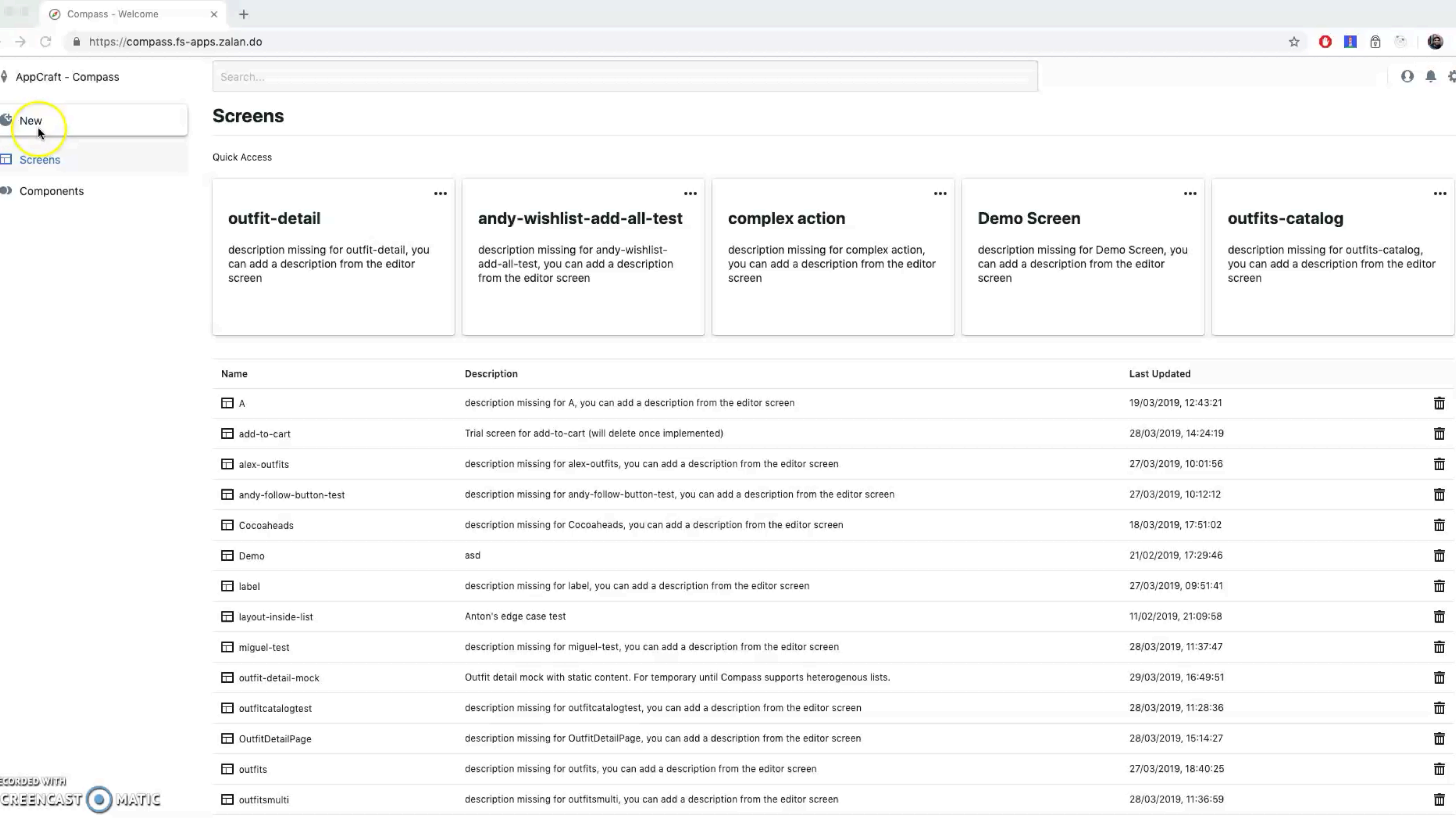
outfit demo

description missing for outfit demo, you can add a description from the editor screen

influencer-item

Influencer avatar, name, and follow button

Name	Description	Last Updated
Outfit Card	description missing for Outfit Card, you can add a description from the editor screen	4/3/2019, 11:36:45 AM
outfit-detail-item	List item for an article that is part of an outfit	3/27/2019, 6:07:43 PM



Compass - Welcome

https://compass.fs-apps.zalan.do

AppCraft - Compass

Search...

New

Screens

Components

Screens

Quick Access

outfit-detail

description missing for outfit-detail, you can add a description from the editor screen

andy-wishlist-add-all-test

description missing for andy-wishlist-add-all-test, you can add a description from the editor screen

complex action

description missing for complex action, you can add a description from the editor screen

Demo Screen

description missing for Demo Screen, you can add a description from the editor screen

outfits-catalog

description missing for outfits-catalog, you can add a description from the editor screen

Name	Description	Last Updated
A	description missing for A, you can add a description from the editor screen	19/03/2019, 12:43:21
add-to-cart	Trial screen for add-to-cart (will delete once implemented)	28/03/2019, 14:24:19
alex-outfits	description missing for alex-outfits, you can add a description from the editor screen	27/03/2019, 10:01:56
andy-follow-button-test	description missing for andy-follow-button-test, you can add a description from the editor screen	27/03/2019, 10:12:12
Cocoaheads	description missing for Cocoaheads, you can add a description from the editor screen	18/03/2019, 17:51:02
Demo	asd	21/02/2019, 17:29:46
label	description missing for label, you can add a description from the editor screen	27/03/2019, 09:51:41
layout-inside-list	Anton's edge case test	11/02/2019, 21:09:58
miguel-test	description missing for miguel-test, you can add a description from the editor screen	28/03/2019, 11:37:47
outfit-detail-mock	Outfit detail mock with static content. For temporary until Compass supports heterogenous lists.	29/03/2019, 16:49:51
outfitcatalogtest	description missing for outfitcatalogtest, you can add a description from the editor screen	28/03/2019, 11:28:36
OutfitDetailPage	description missing for OutfitDetailPage, you can add a description from the editor screen	28/03/2019, 15:14:27
outfits	description missing for outfits, you can add a description from the editor screen	27/03/2019, 18:40:25
outfitsmulti	description missing for outfitsmulti, you can add a description from the editor screen	28/03/2019, 11:36:59

Future Plans

Future Plans

- Initial production rollout

Future Plans

- Initial production rollout
- Fully dynamic home screen

Future Plans

- Initial production rollout
- Fully dynamic home screen
- Algorithmically created screens

Future Plans

- Initial production rollout
- Fully dynamic home screen
- Algorithmically created screens
- Flutter and/or Jetpack Compose

Lessons Learned

Lessons Learned

- The difference between a prototype and a production ready platform is immense.

Lessons Learned

- The difference between a prototype and a production ready platform is immense.
- Don’t cut corners. Paint both sides of the fence. YPGNI.

Lessons Learned

- The difference between a prototype and a production ready platform is immense.
- Don’t cut corners. Paint both sides of the fence. YPGNI.
- If something seems obvious but no one else is doing it, it’s either genius or foolish…maybe both.

Lessons Learned (cont.)

Lessons Learned (cont.)

- A Redux/MVI architecture has a lot of advantages.

Lessons Learned (cont.)

- A Redux/MVI architecture has a lot of advantages.
- Pair programming really is as great as people say it is.

Thank You !

Thank You !

- Slides: `bit.ly/2YCrQFb`

Thank You !

- Slides: `bit.ly/2YCrQFb`
- We're hiring! `jobs.zalando.com`