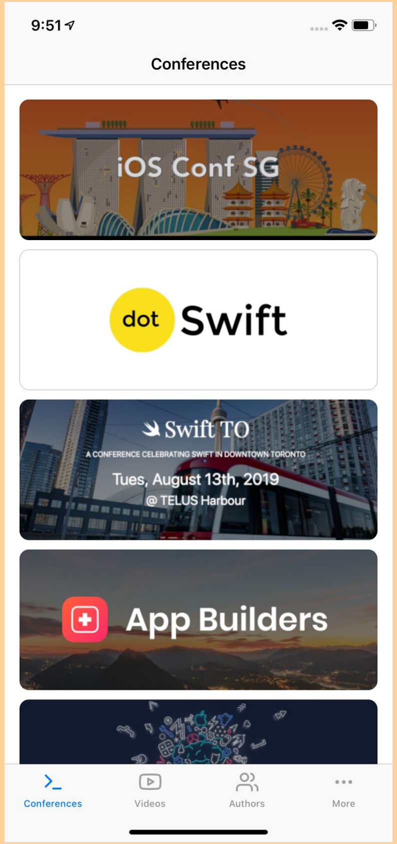
SwiftCommunity


9:51

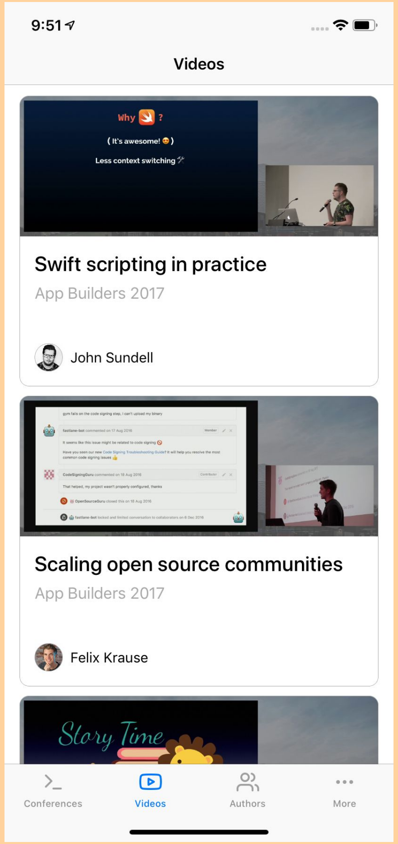
Conferences

iOS Conf SG

dot Swift

Swift TO
A CONFERENCE CELEBRATING SWIFT IN DOWNTOWN TORONTO
Tues, August 13th, 2019
@ TELUS Harbour

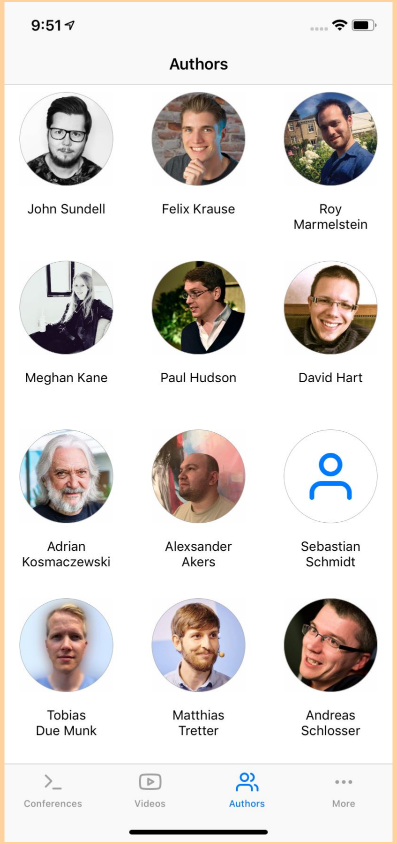
App Builders


9:51

Videos

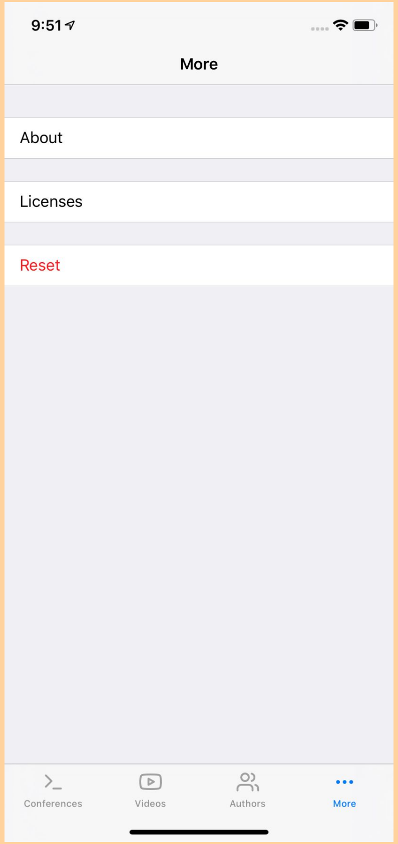
Swift scripting in practice
App Builders 2017
John Sundell

Scaling open source communities
App Builders 2017
Felix Krause


9:51

Authors

John Sundell
Felix Krause
Roy Marmelstein
Meghan Kane
Paul Hudson
David Hart
Adrian Kosmaczewski
Alexsander Akers
Sebastian Schmidt
Tobias Due Munk
Matthias Tretter
Andreas Schlosser


9:51

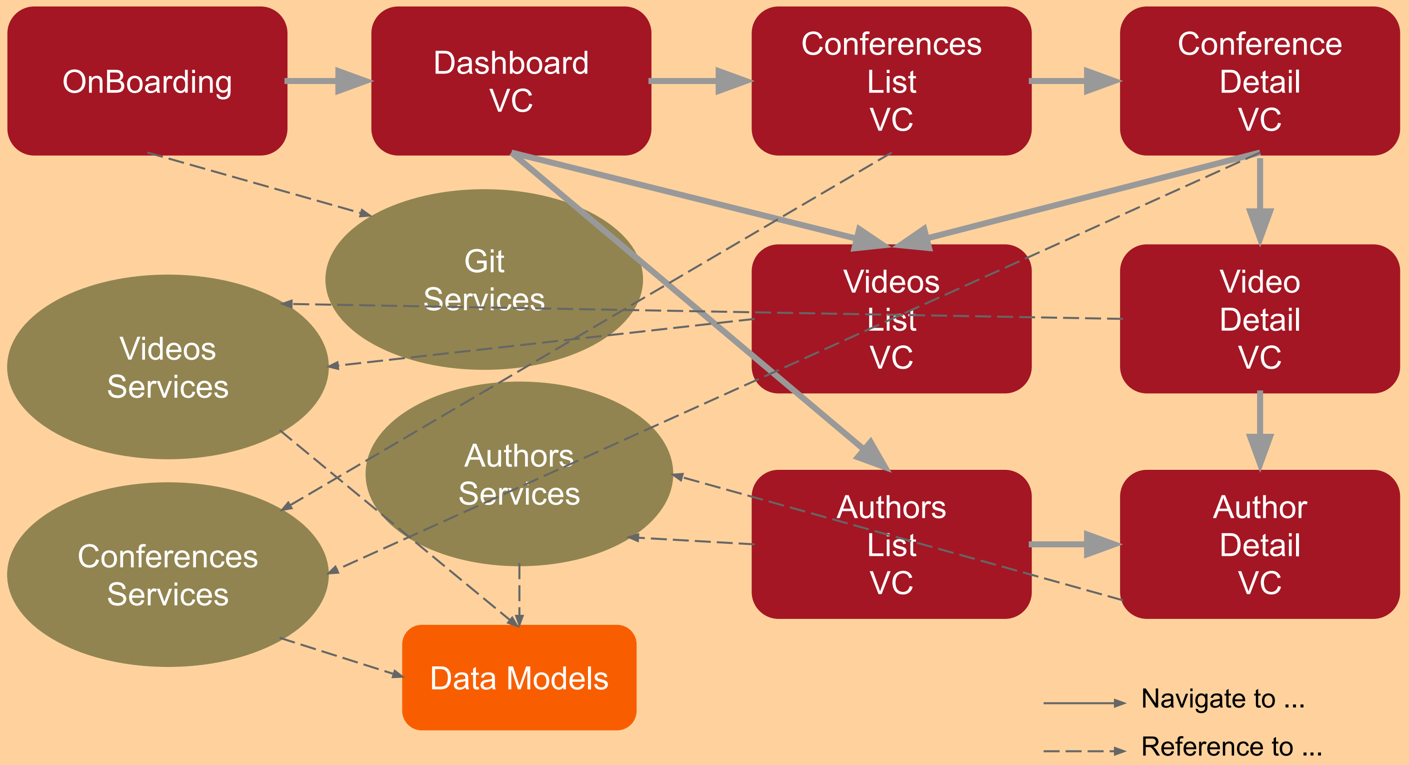
More

About

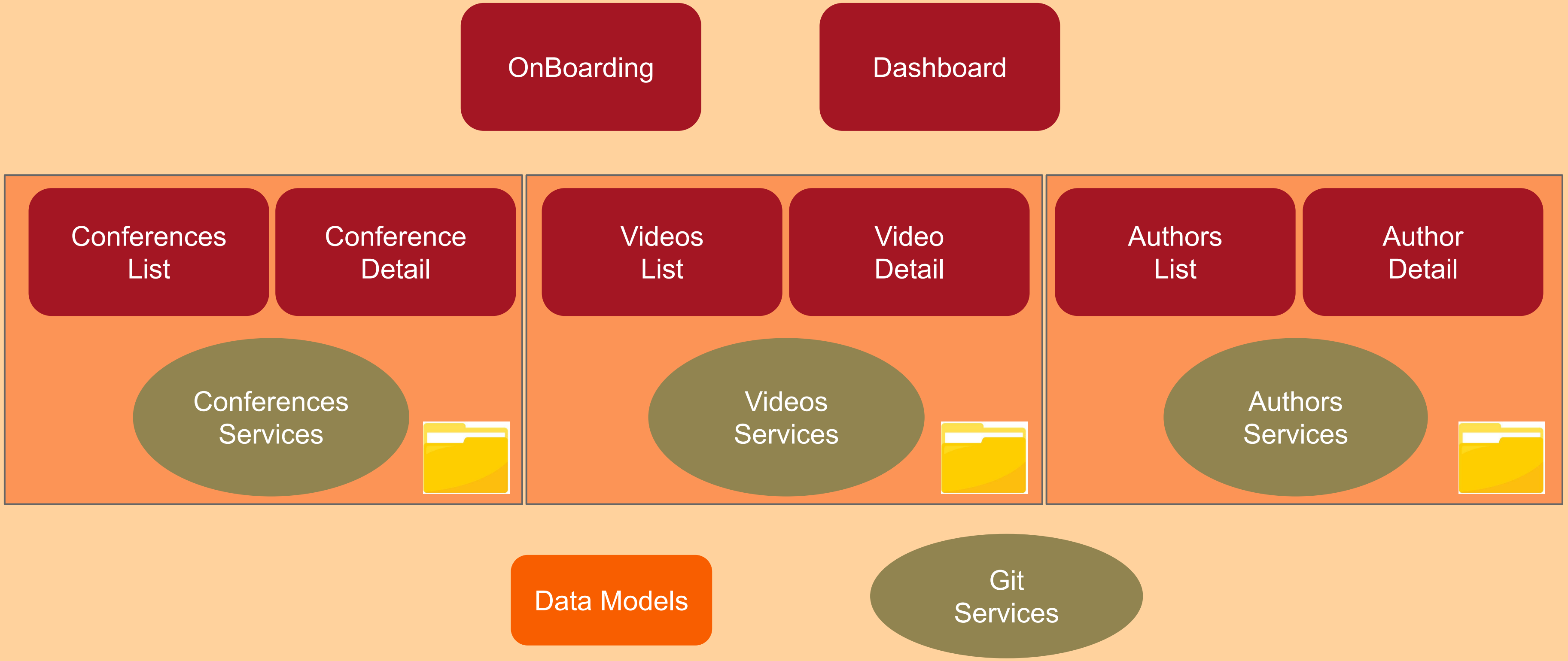
Licenses

Reset

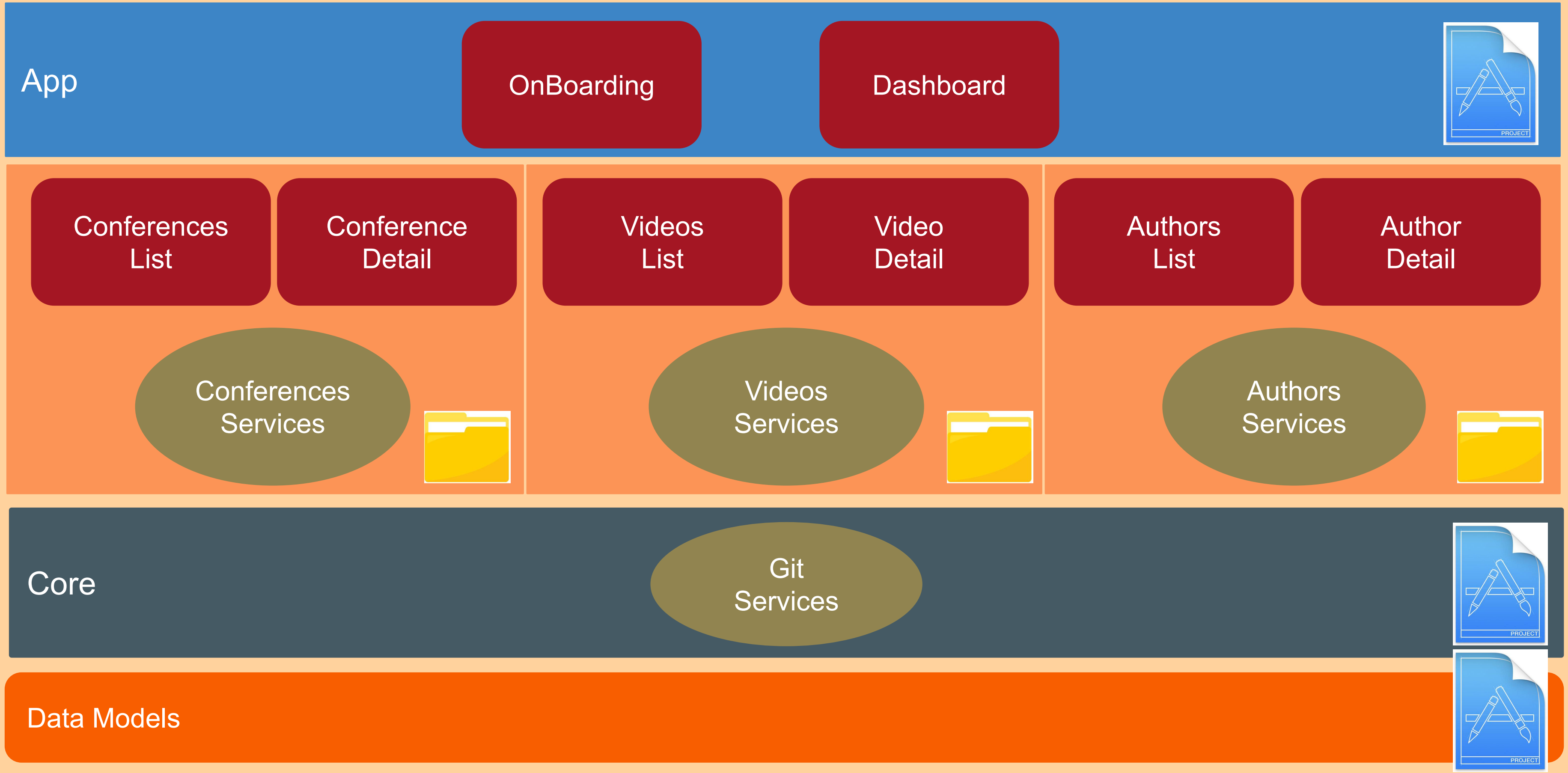
Monolith


OnBoarding
Dashboard VC
Conferences List VC
Conference Detail VC
Git Services
Videos Services
Videos List VC
Video Detail VC
Authors Services
Conferences Services
Authors List VC
Author Detail VC
Data Models
Navigate to ...
Reference to ...

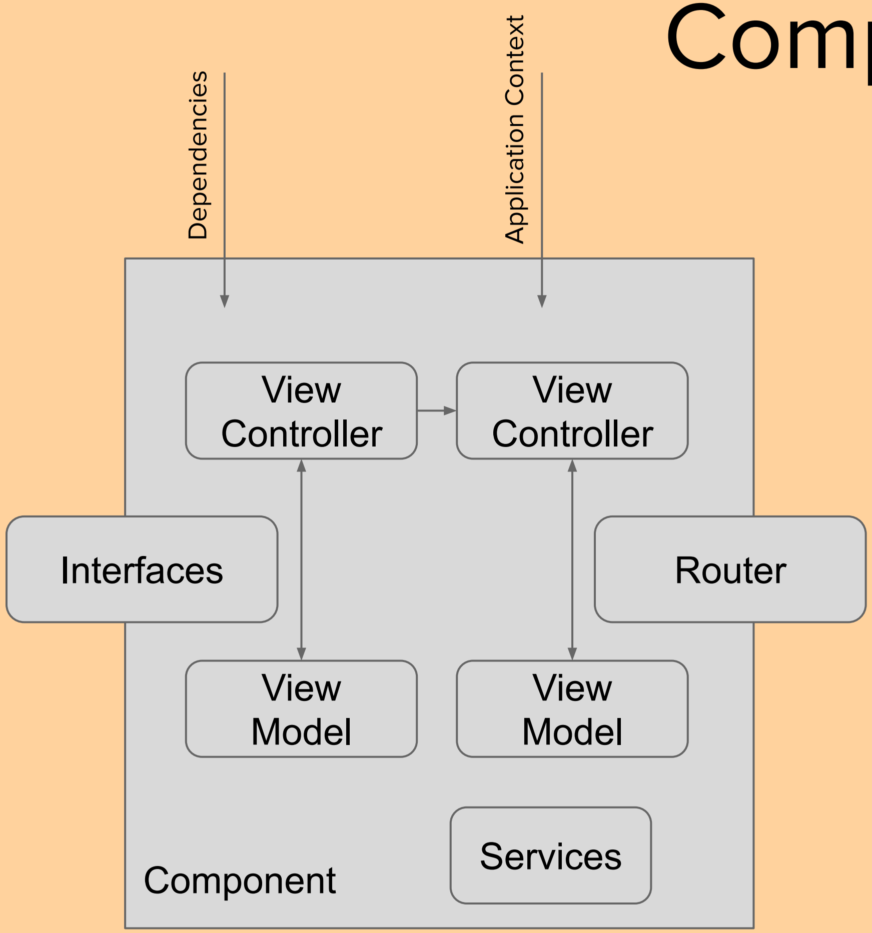
Monolith - Layers



Modularisation



Component



Dependencies: all dependencies needed to intialize the Component.

Application Context: contains all dependencies required for processing application logic and driving UIs. Only used to provide dependencies for the next components.

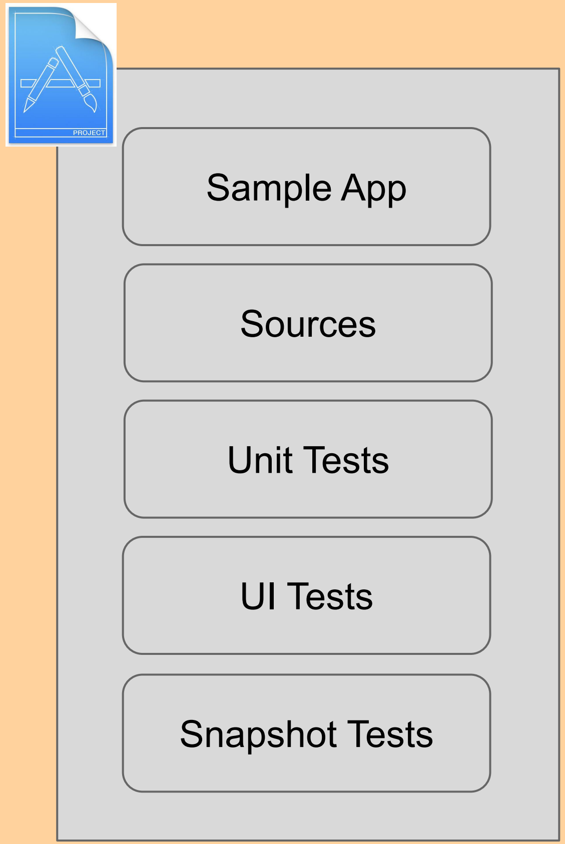
Interfaces: public APIs to interact with the Component.

Router: internal APIs to interact with outside Components.

ViewControllers/ViewModels: UI elements.

Services: performing business logics.

Xcode project for a component

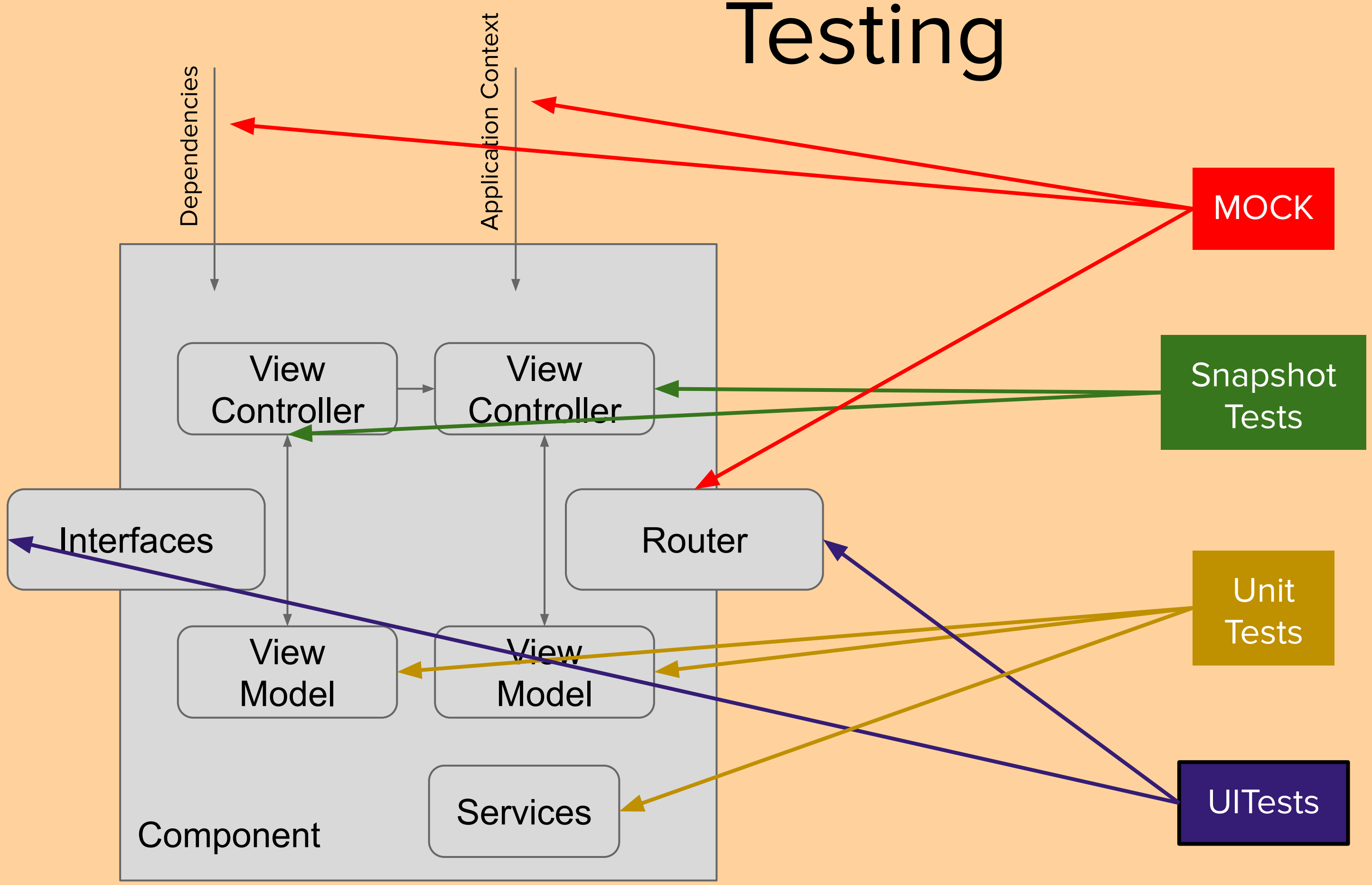


Sample App: used to test host the components for testing purpose.

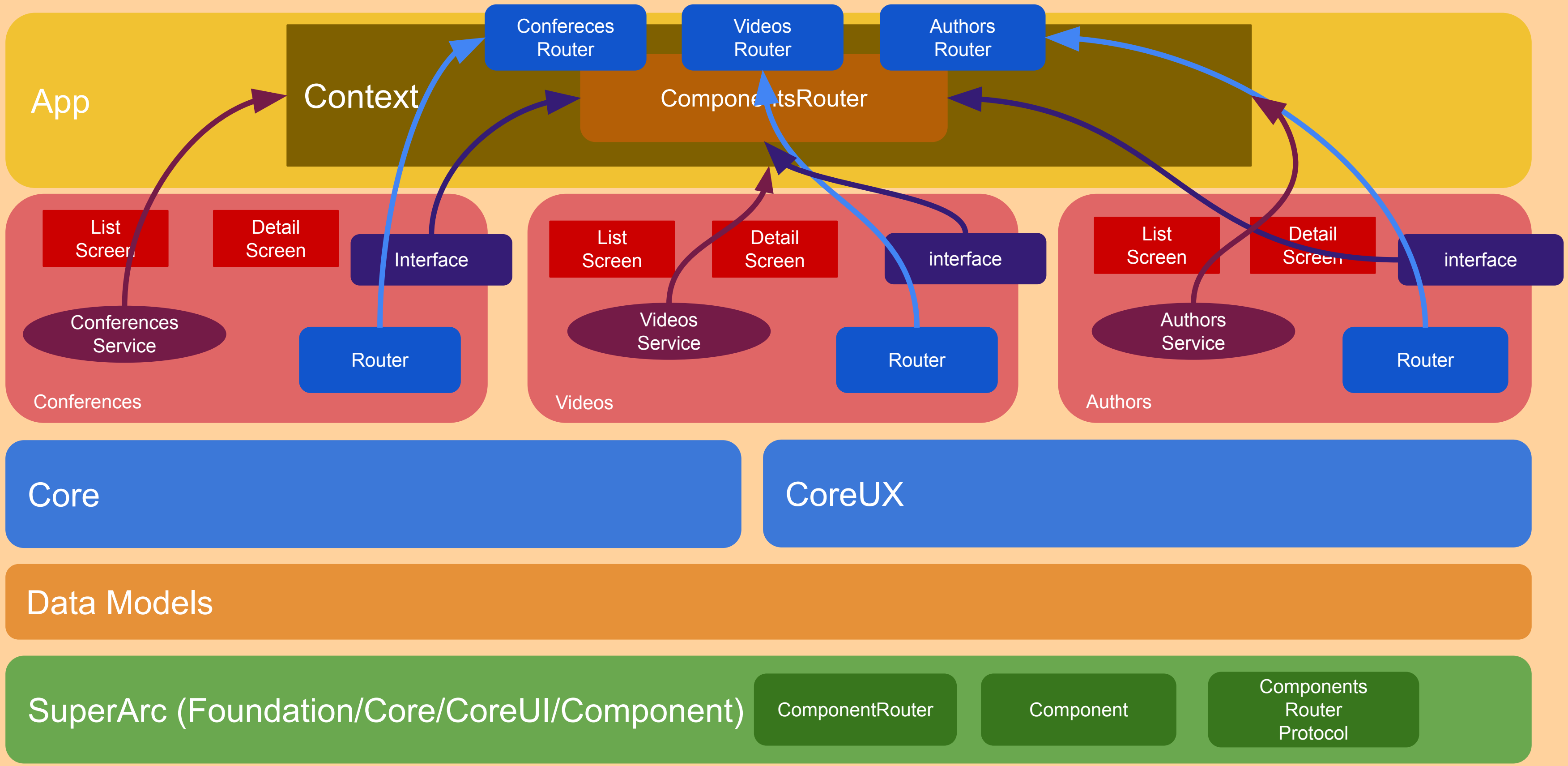
Sources: contains all source code of the component.

Unit Tests / UI Tests / Snapshot Tests: used to test different aspects of the component.

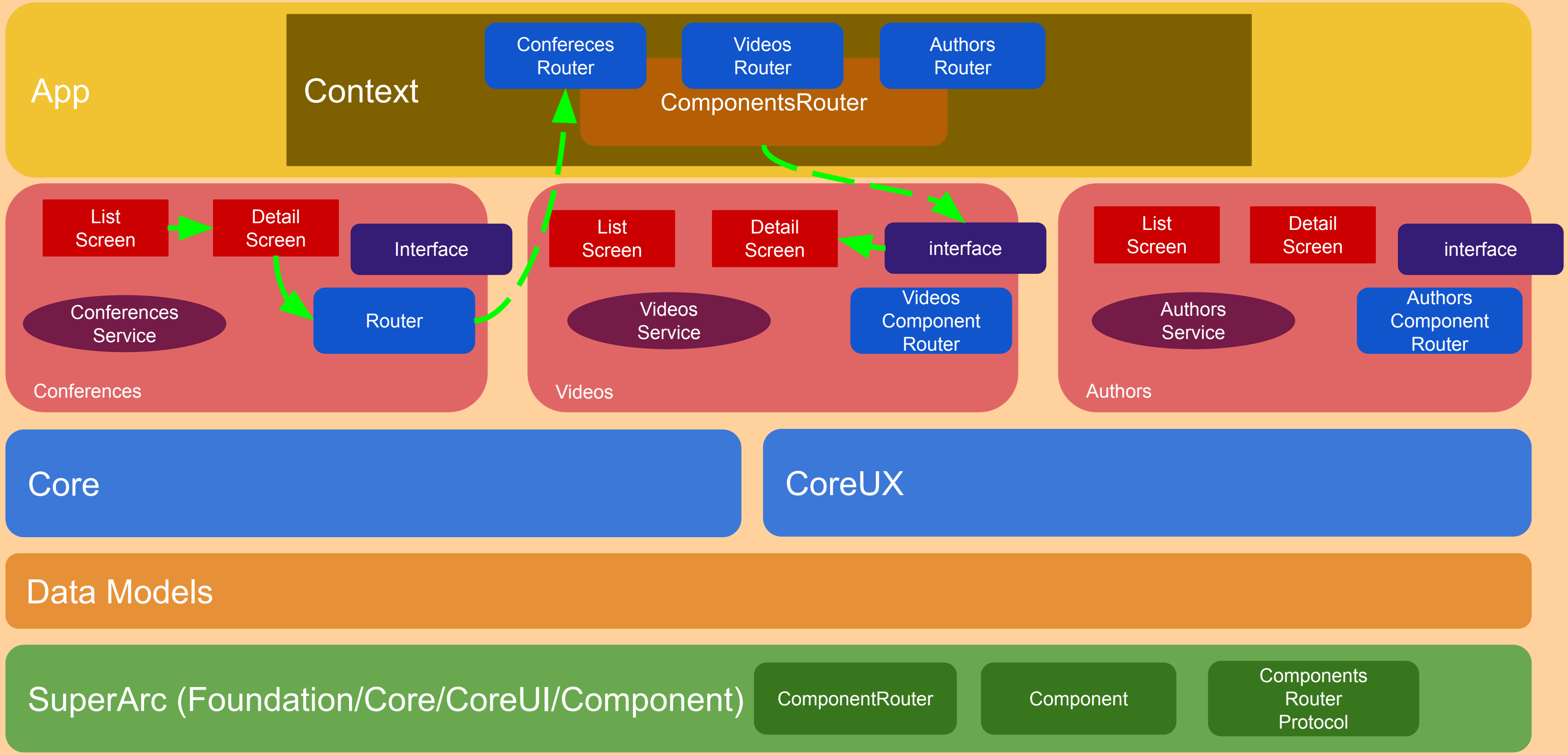
Testing


Dependencies
Application Context
MOCK
View Controller
View Controller
Snapshot Tests
Interfaces
Router
Unit Tests
View Model
View Model
Services
Component
UITests

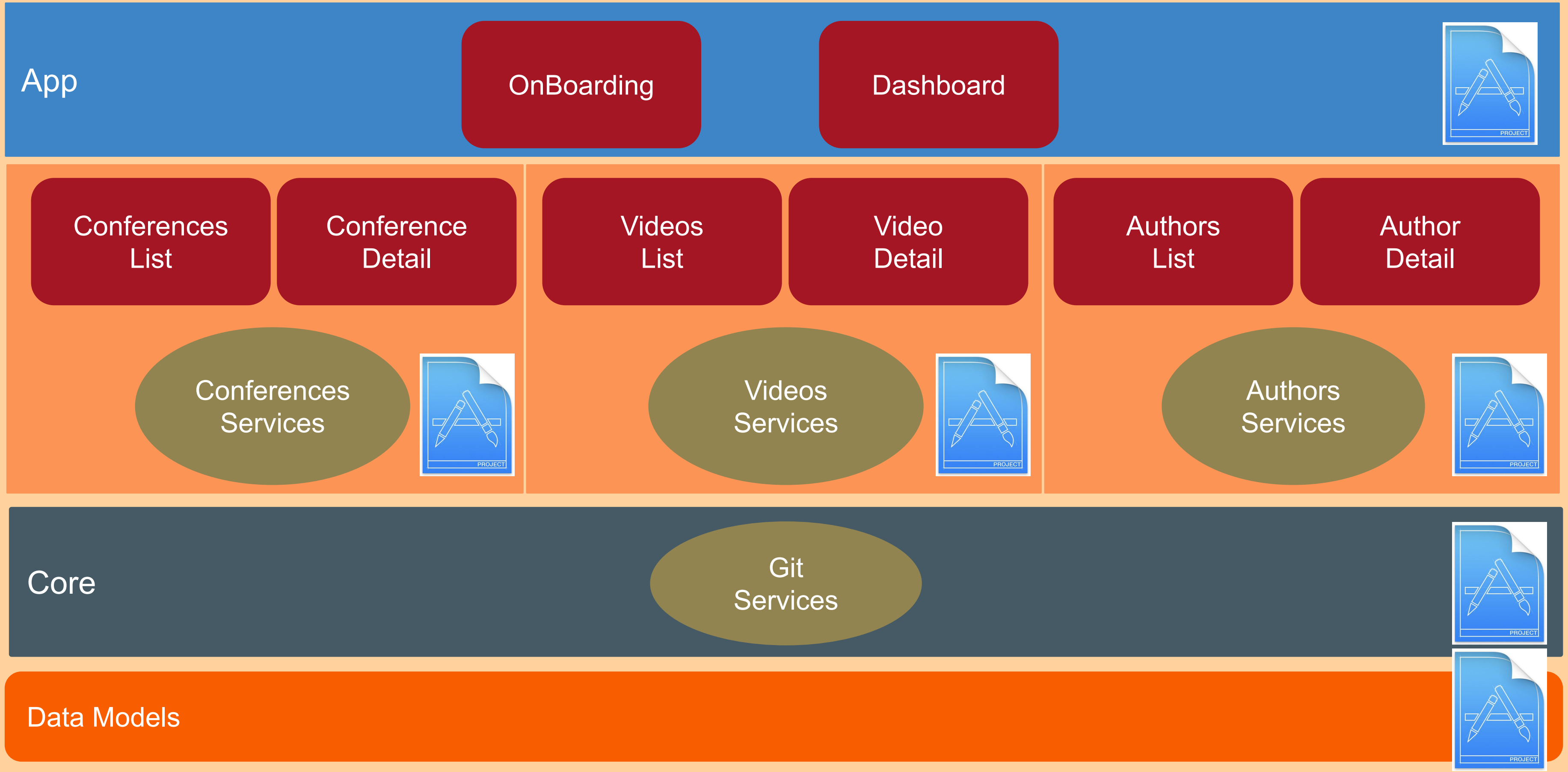
Routers+Interfaces registration



Navigation between components



Modularisation



Pros

Enforcing APIs separation/encapsulation.

Enforcing clear dependencies.

Improve team parallelism.

Improve testability/test coverage.

Improve compiling time.

Improve reusability

Cons

Many projects to manage.

More upfront planning.

Hard to share knowledge.