

The Quimera: SwiftUI Navigation

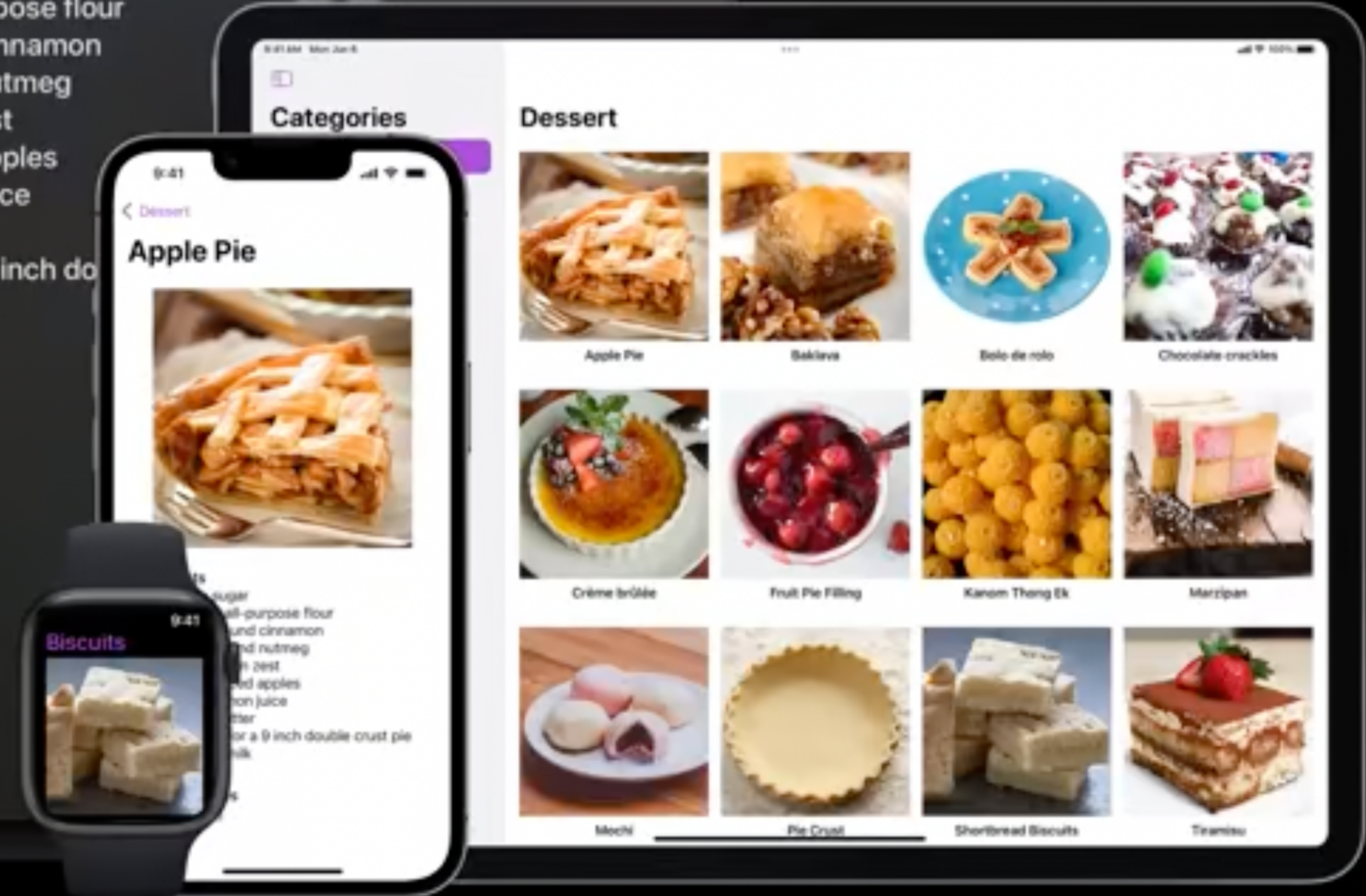
FrenchKit 2022

@lascorbe



Apple Pie

- ¾ cup white sugar
- 2 tablespoons all-purpose flour
- ½ teaspoon ground cinnamon
- ¼ teaspoon ground nutmeg
- ½ teaspoon lemon zest
- 7 cups thinly sliced apples
- 2 teaspoons lemon juice
- 1 tablespoon butter
- 1 recipe pastry for a 9 inch do
- 4 tablespoons milk



Luis Ascorbe

@lascorbe

Agenda



Agenda

- Navigation with UIKit



Agenda

- Navigation with UIKit
- Navigation with SwiftUI



Agenda

- Navigation with UIKit
- Navigation with SwiftUI
- Which one to choose



Navigation with UIKit

UIHostingController<Content>

```
class UIHostingController<Content> where Content: View
```

```
import SwiftUI
```

```
struct FrenchView: View {  
    var body: some View {  
        Text("Bonjour")  
    }  
}
```

```
class UIHostingController<Content> where Content: View
```

```
import SwiftUI
```

```
struct FrenchView: View {  
    var body: some View {  
        Text("Bonjour")  
    }  
}
```

```
UIHostingController rootView: FrenchView())
```

```
class UIHostingController<Content> where Content: View
```

```
import SwiftUI
```

```
struct FrenchView: View {  
    @ObservedObject var viewModel: ViewModel  
    var body: some View {  
        Text("Bonjour")  
    }  
}
```

```
UIHostingController(  
    rootView: FrenchView(viewModel: ViewModel())  
)
```

```
class UIHostingC
```

```
class ViewModel: ObservableObject {
```

```
}
```

```
import SwiftUI
```

```
struct FrenchView: View {
```

```
    @ObservedObject var viewModel: ViewModel
```

```
    var body: some View {
```

```
        Text("Bonjour")
```

```
    }
```

```
}
```

```
UIHostingController(
```

```
    rootView: FrenchView(viewModel: ViewModel())
```

```
)
```

```
class UIHostingC
```

```
class ViewModel: ObservableObject {  
    @Published var title: String  
}
```

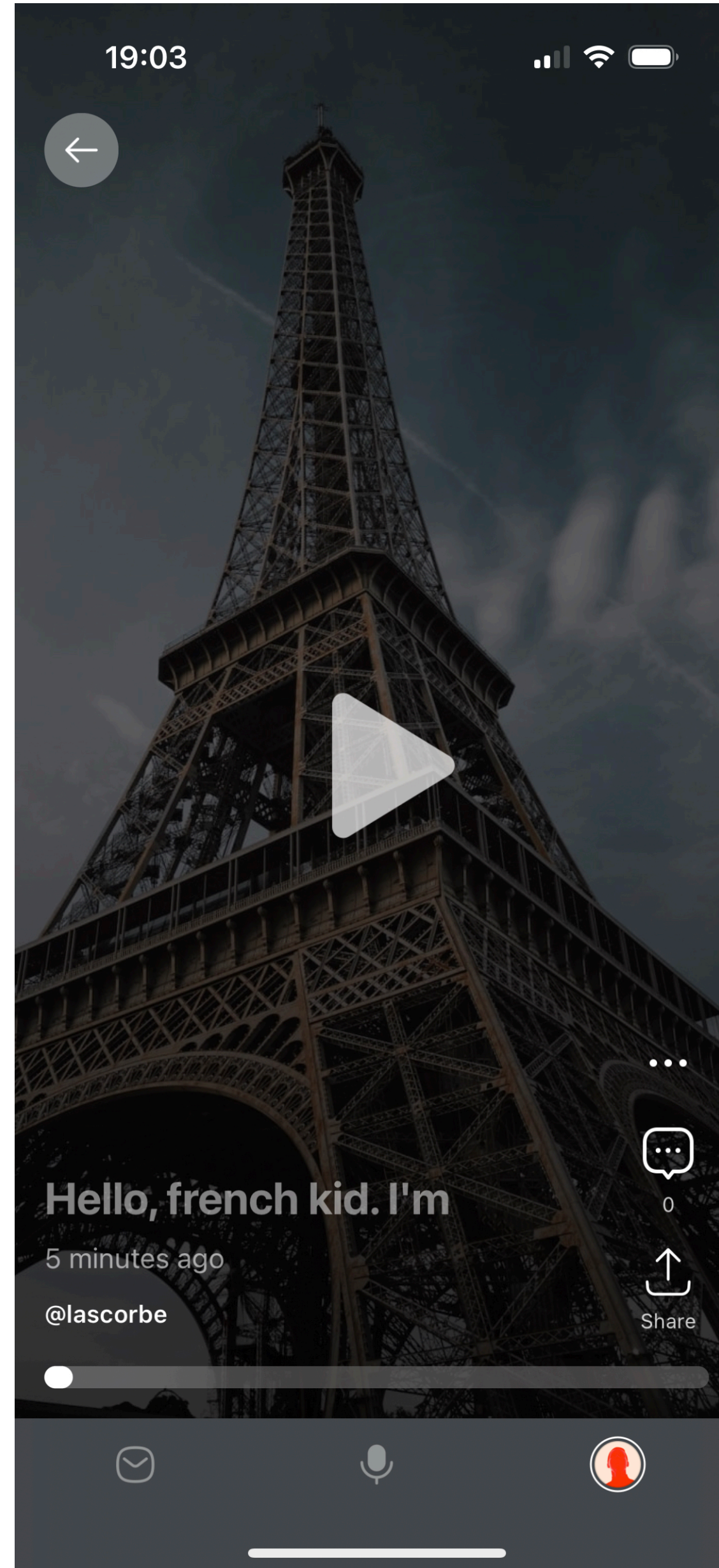
```
import SwiftUI
```

```
struct FrenchView: View {  
    @ObservedObject var viewModel: ViewModel  
    var body: some View {  
        Text(viewModel.title)  
    }  
}
```

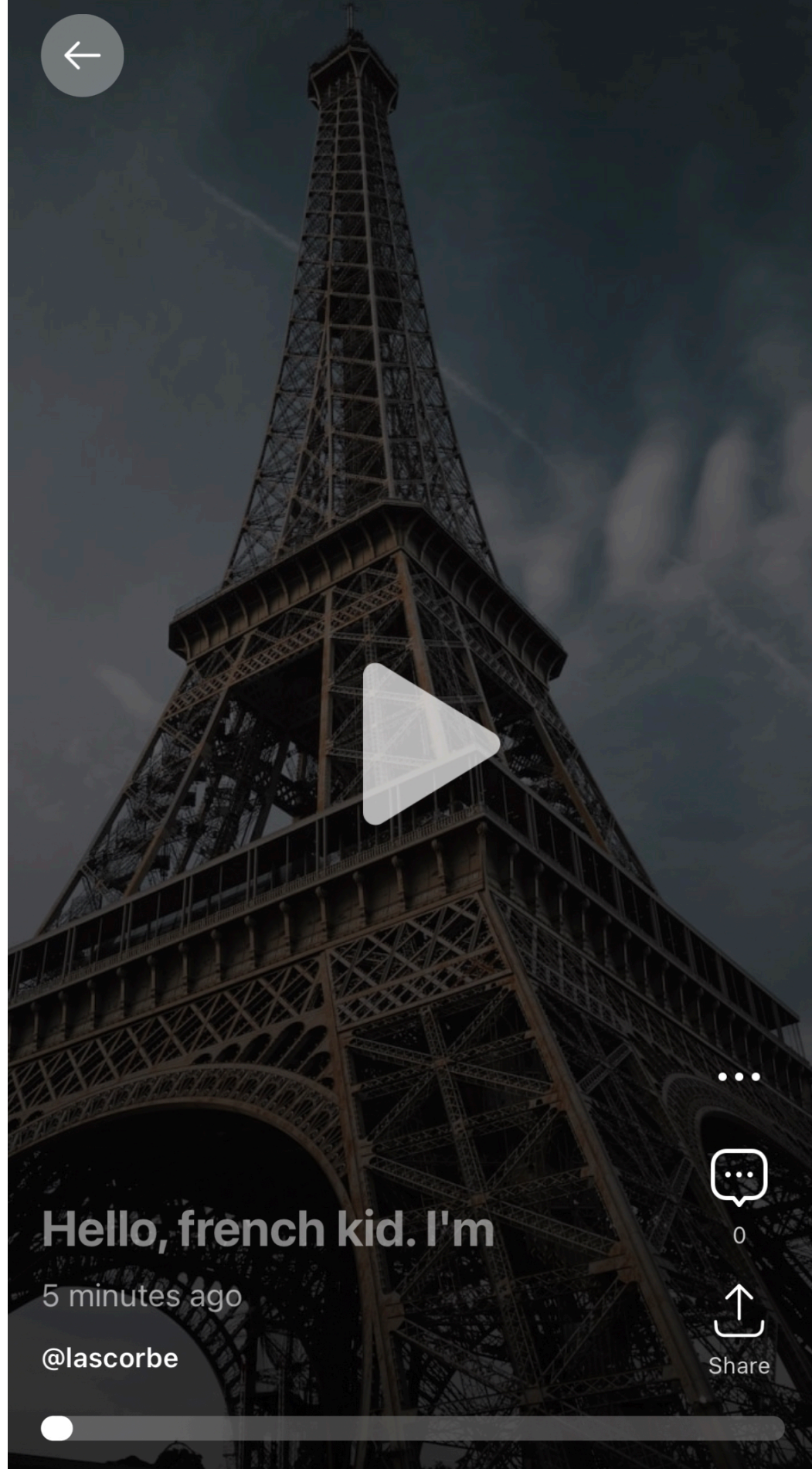
```
UIHostingController(  
    rootView: FrenchView(viewModel: ViewModel())  
)
```

Real life use case

beams.fm



19:03



Hello, french kid. I'm

5 minutes ago

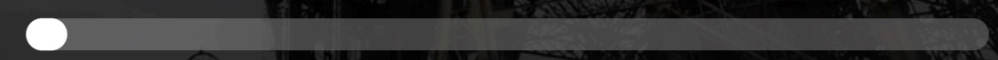
@lascorbe

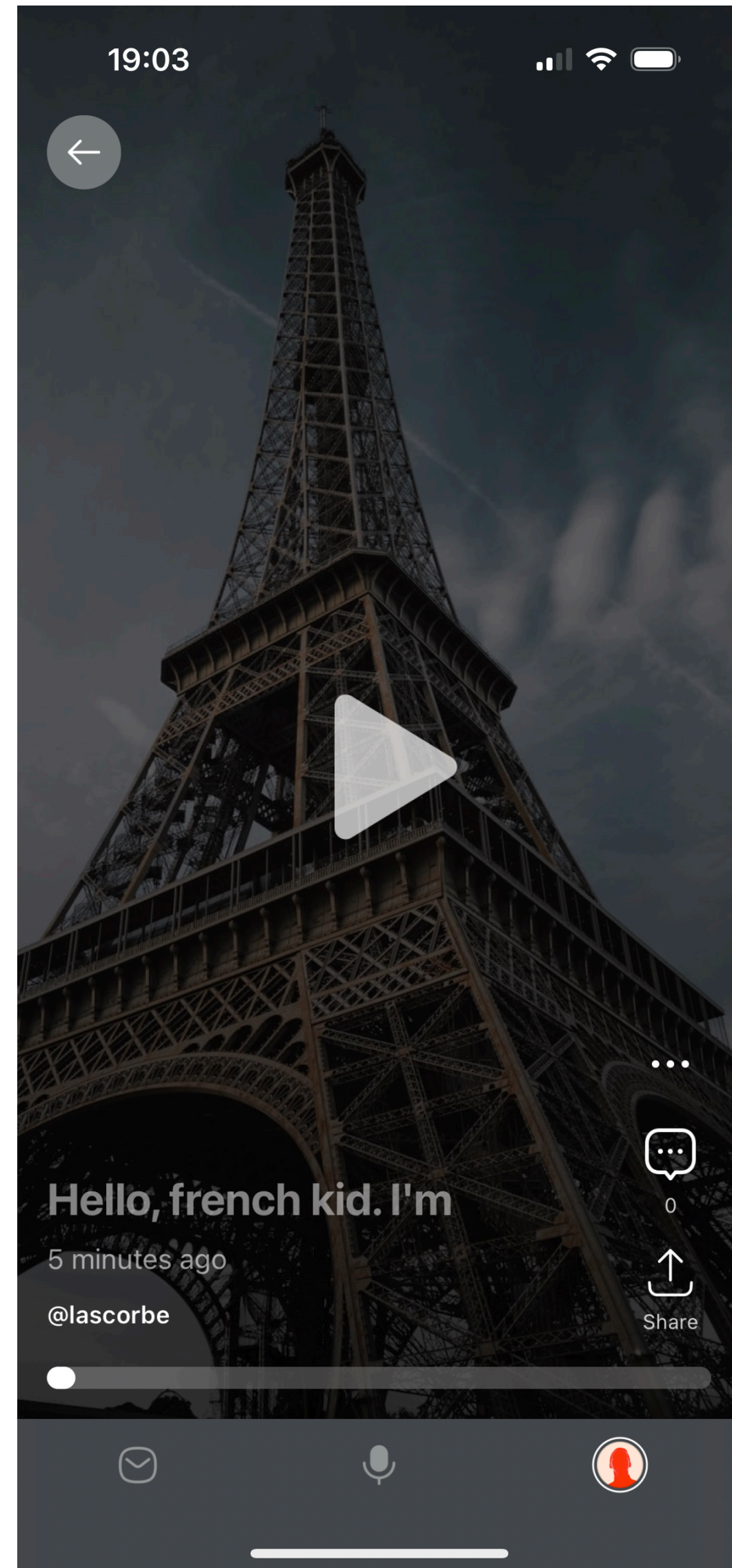


0

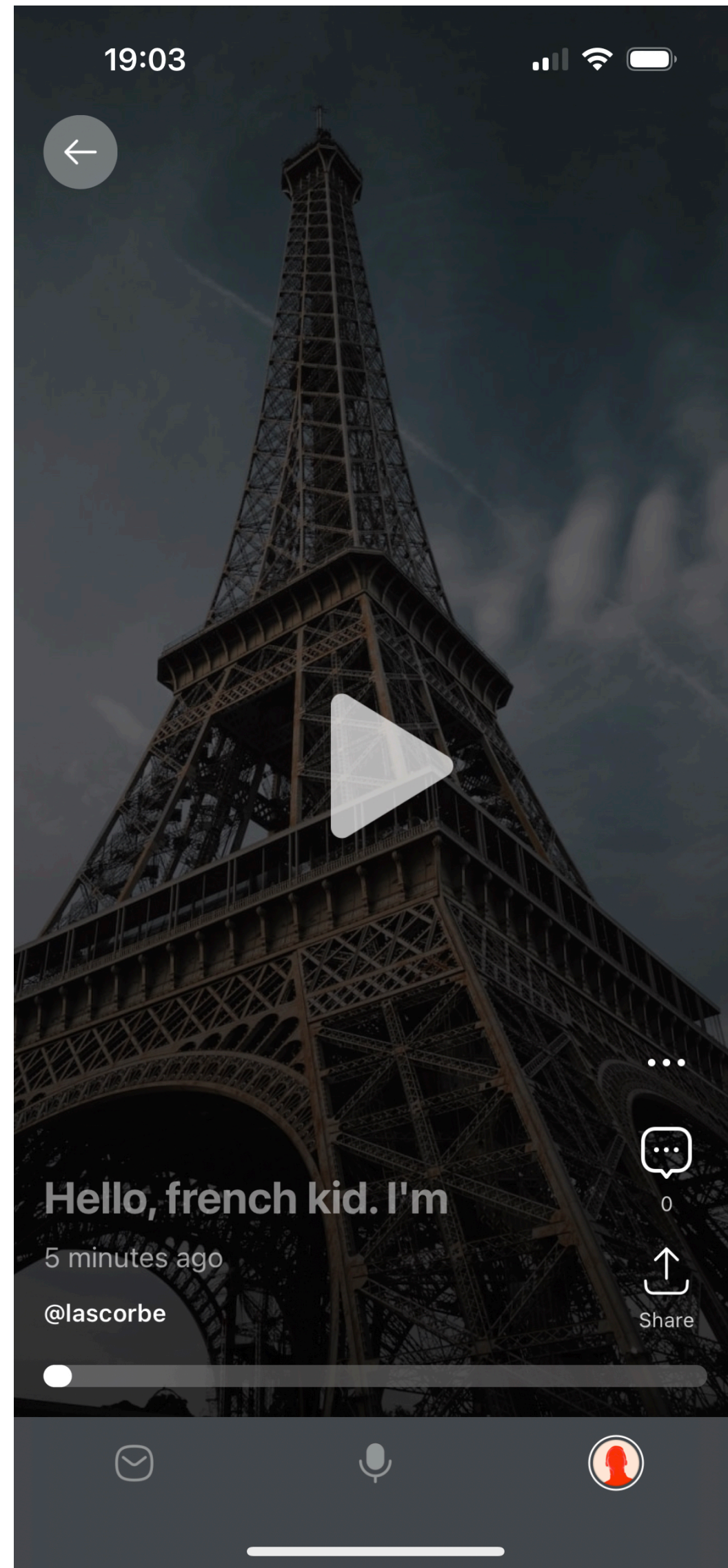


Share



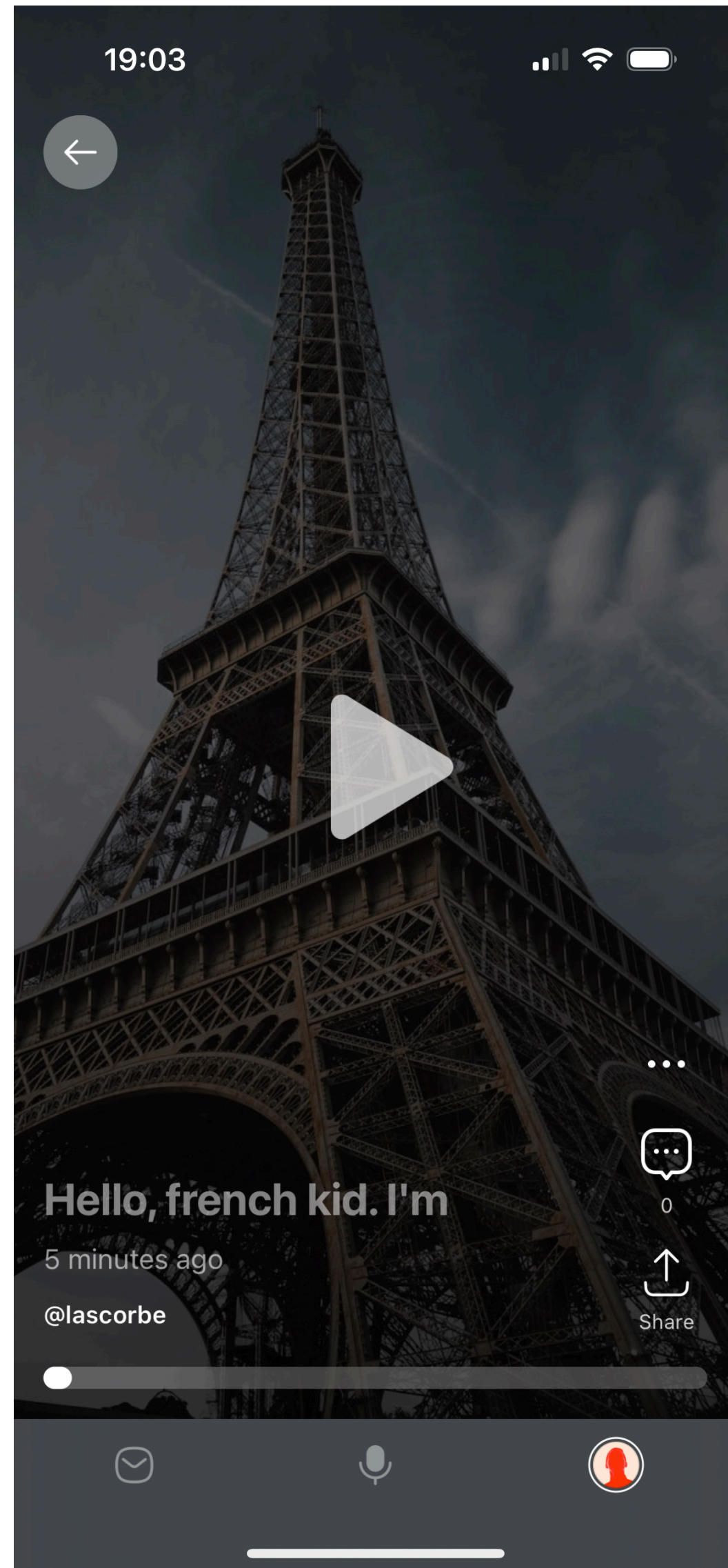


UITabBarController



UITabBarController

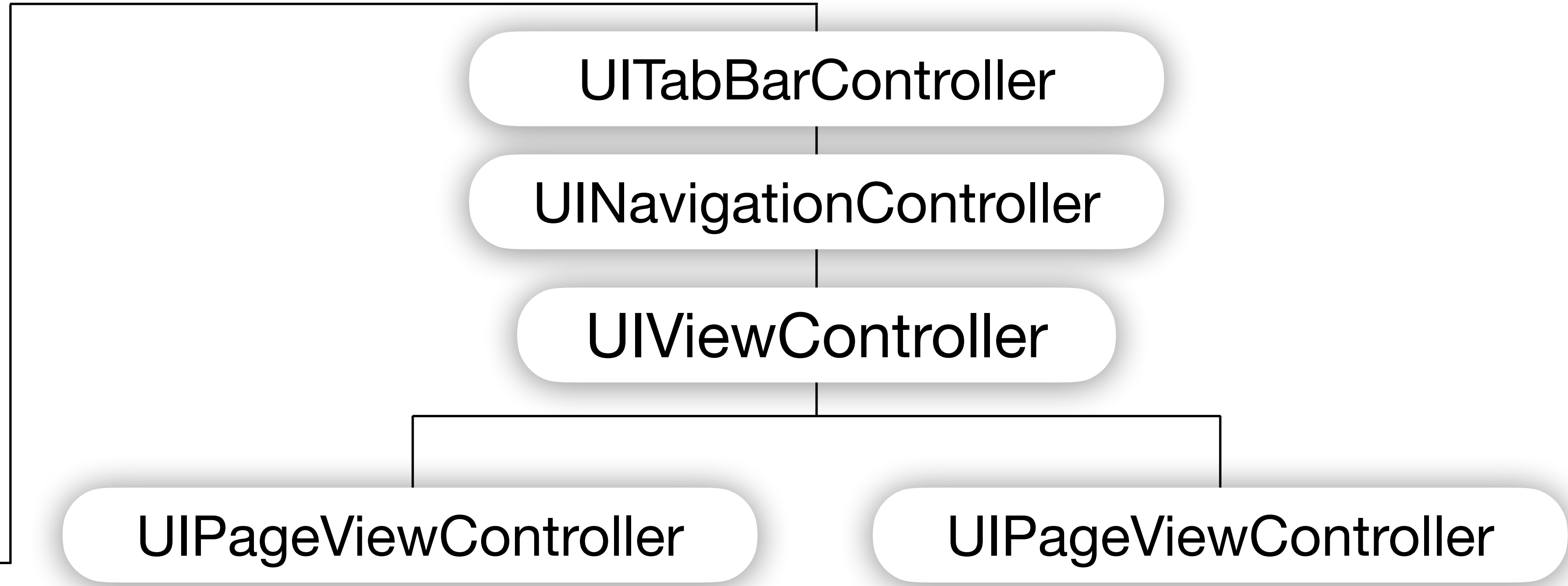
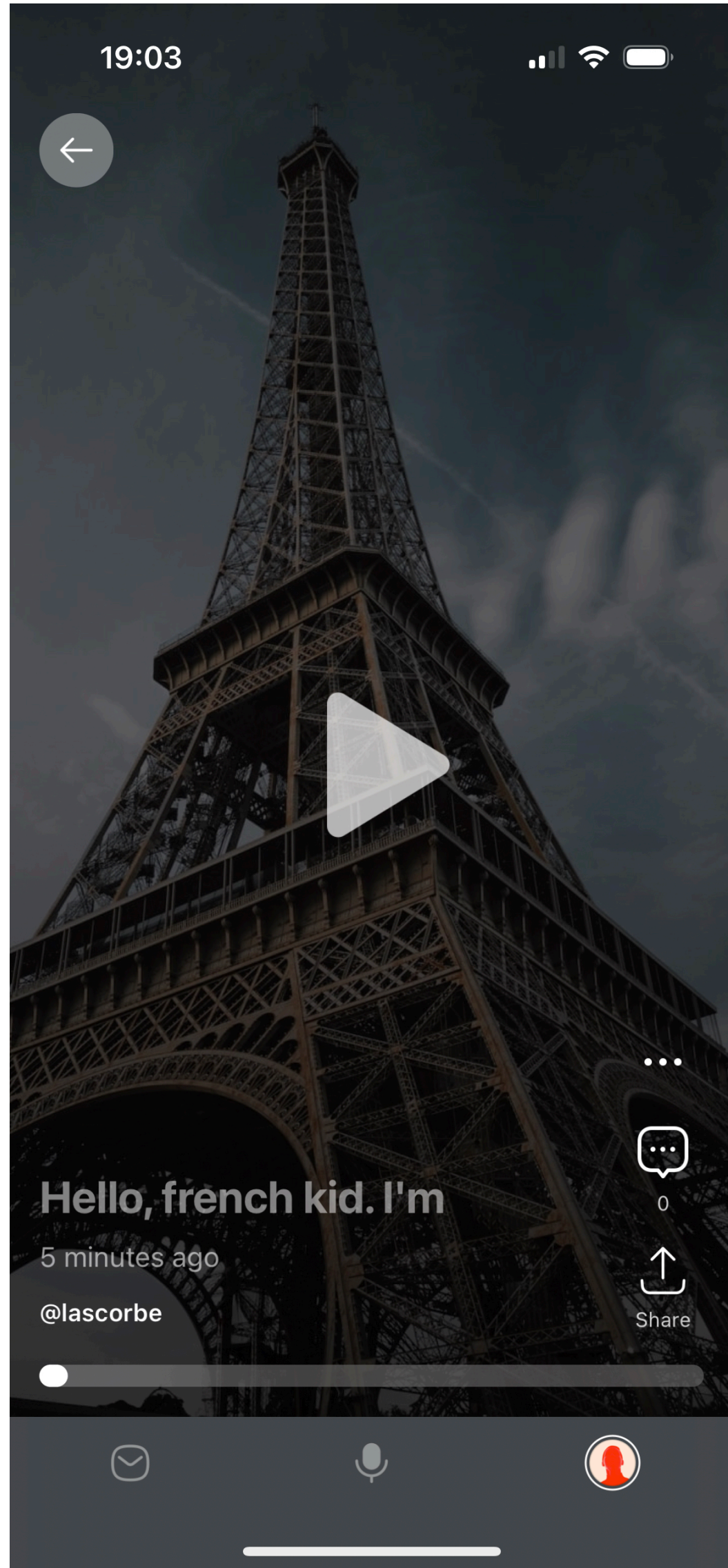
UINavigationController

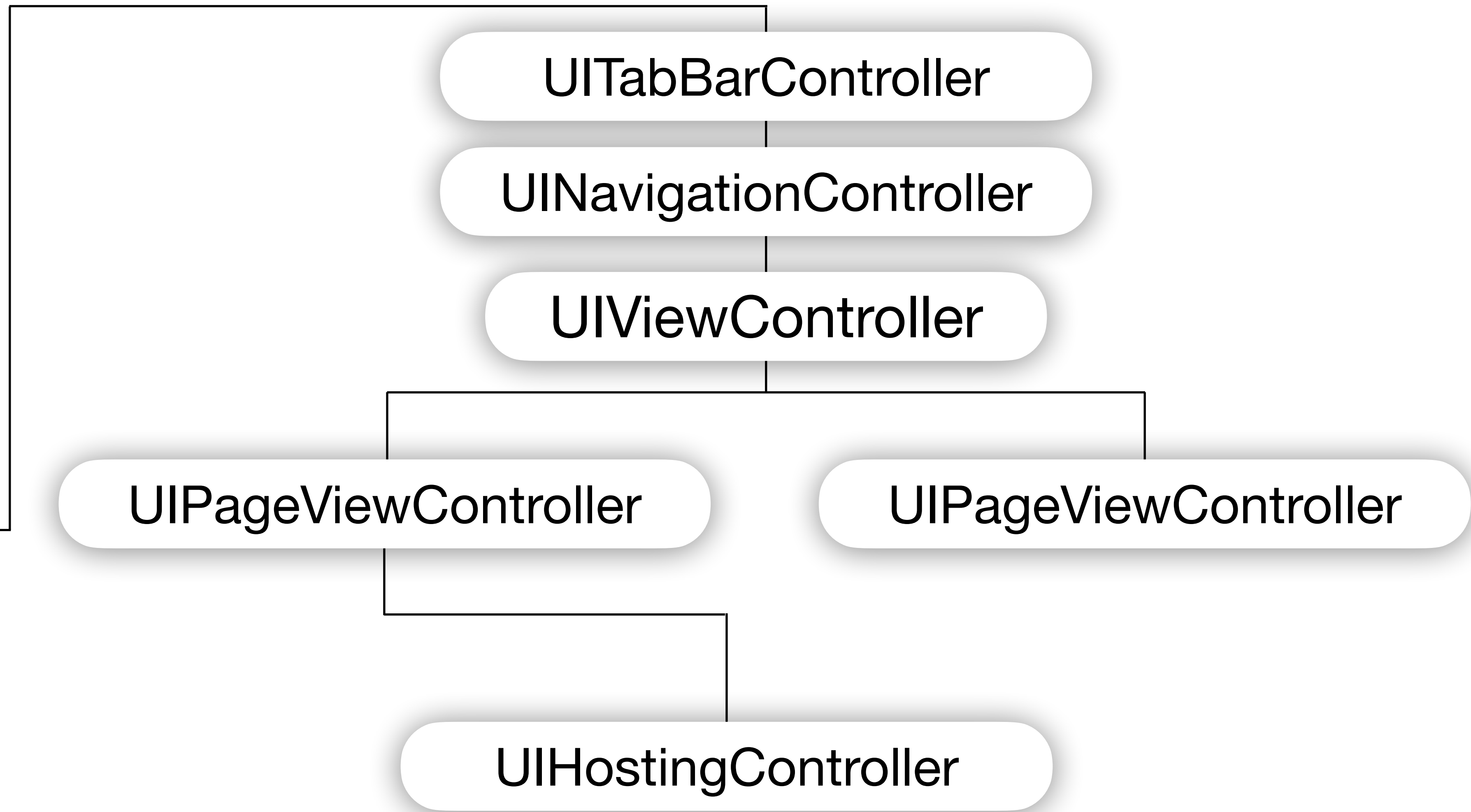
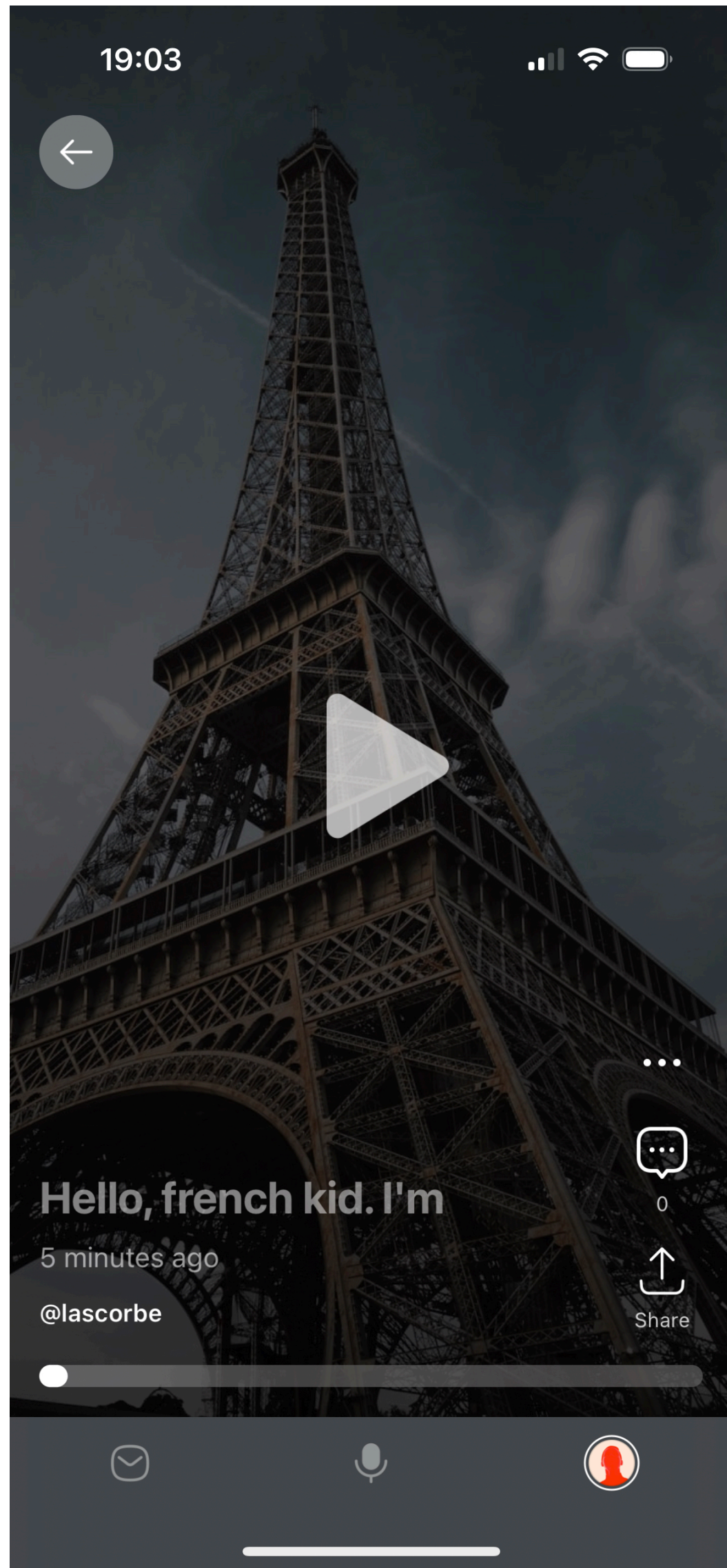


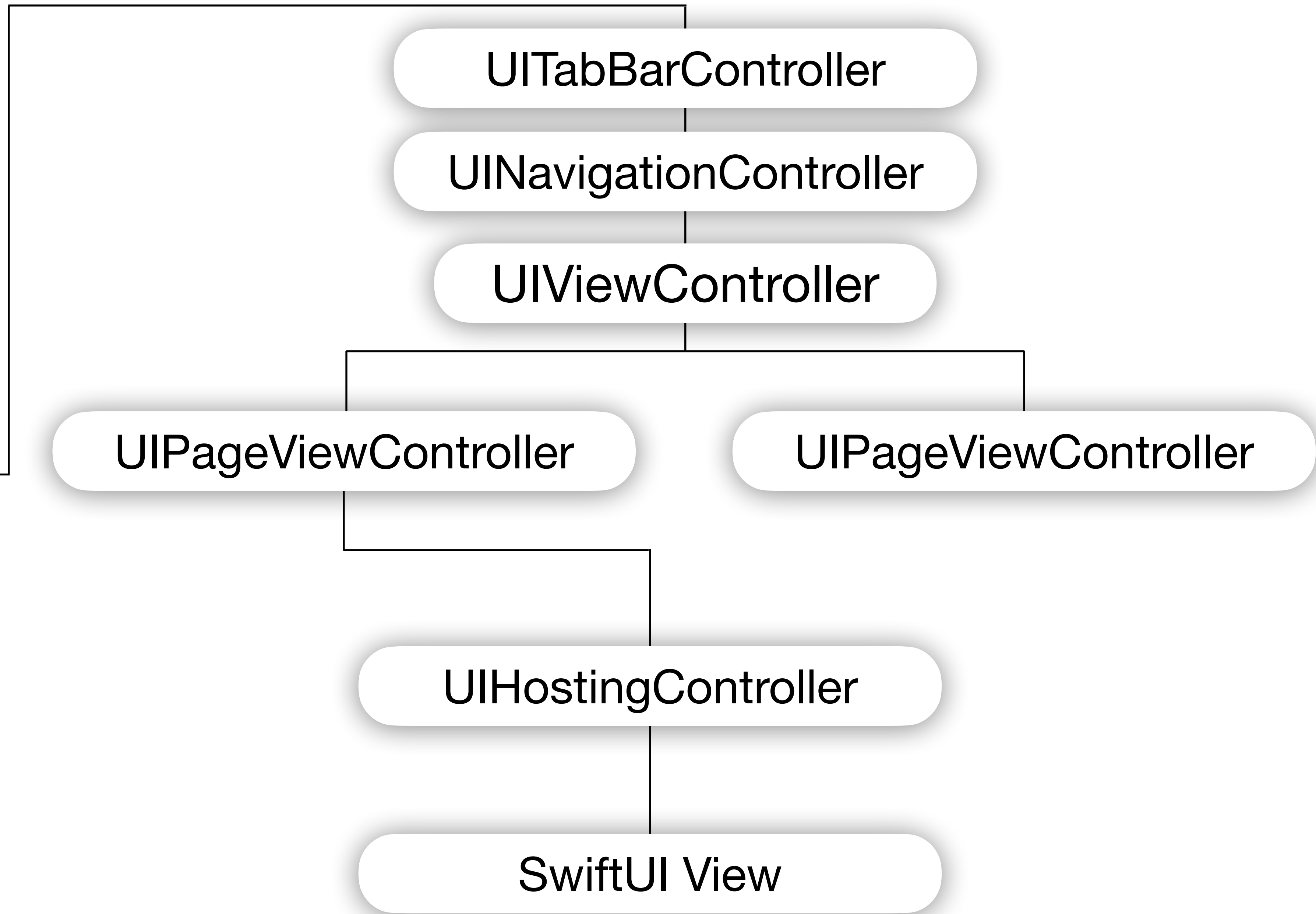
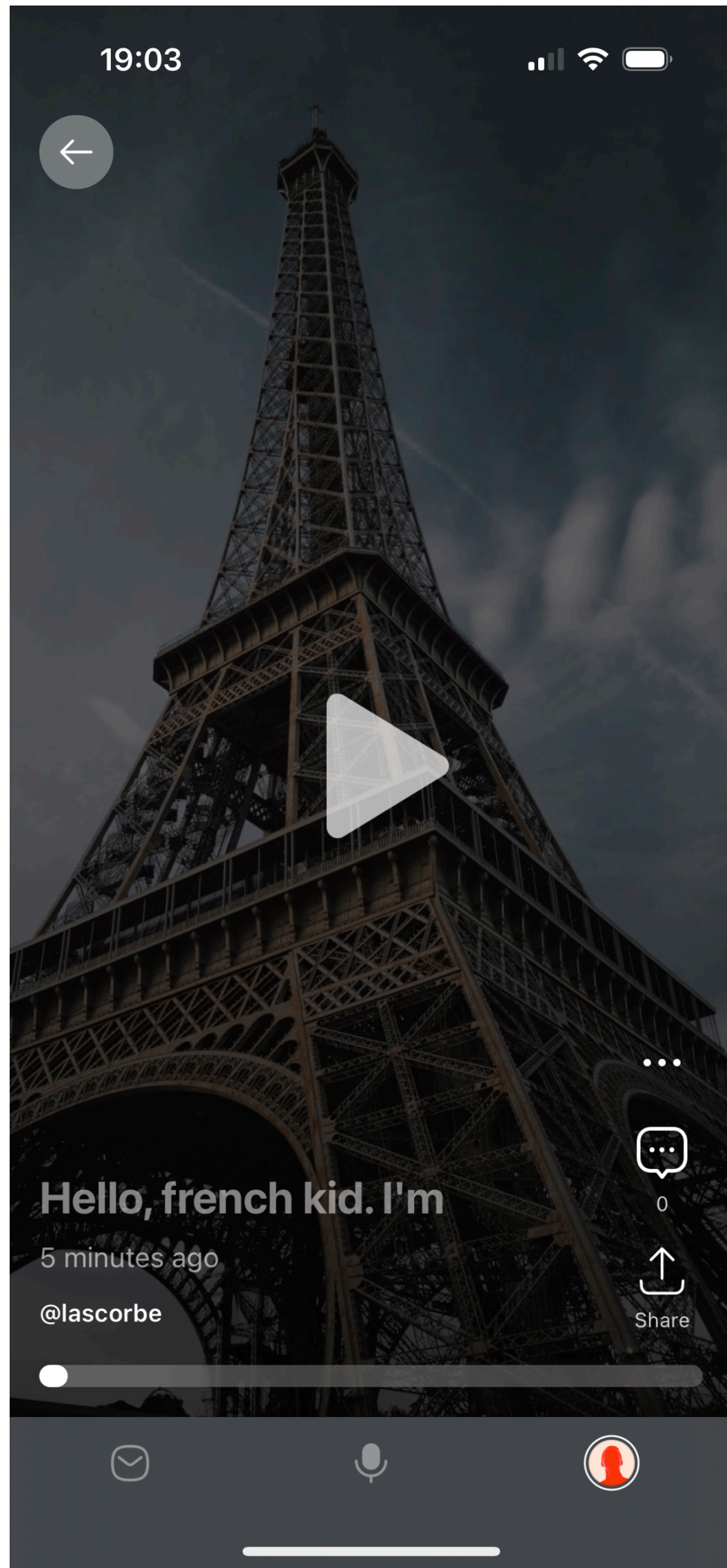
UITabBarController

UINavigationController

UIViewController







Why?

Why?

- All our navigation was working with UIKit

Why?

- All our navigation was working with UIKit
- No vertical page view on SwiftUI

Why?

- All our navigation was working with UIKit
- No vertical page view on SwiftUI
- Development on SwiftUI is a joy

IntrinsicSize-SwiftUI main iPhone 13 Running IntrinsicSize-SwiftUI on iPhone 13

View hierarchy for IntrinsicSize-SwiftUI

IntrinsicSize-SwiftUI > UIWindowScene - (Foreground Active) > UIWindow > UITransitionView > UIDropShadowView > RootViewController > UIView > SwiftUIViewController > Hosting view

Object

Class Name `_UIHostingView<SwiftUIView>`
Address `0x7fee14f09a30`

View

Layer `<CALayer: 0x60000090afc0>`
Layer Class `CALayer`

Content Mode `Scale To Fill`
Tag `0`

Interaction `User Interaction Enabled On Multiple Touch Off`

Alpha `1`
Background `R:0 G:0 B:1 A:1 blueColor`
Tint `R:0 G:0,48 B:1 A:1 systemBlueColor`

Drawing `Opaque On Hidden Off Clears Graphics Context On Clip To Bounds Off Autoresize Subviews On`

Stretching `x 0 y 0 width 1 height 1`

Trait Collection

`Light User Interface Style Regular Vertical Size Class Compact Horizontal Size Class Left To Right Layout Direction`

Accessibility

`Not Accessibility Element Value <null> Traits None Elements <null> Description <null> Hint <null> Identifier <null> Actions <null> Not Focused`

Description `<_TtGC7SwiftUI14_UIHostingViewV21IntrinsicSize_SwiftUI11SwiftUIView_: 0x7fee14f09a30; frame = (0 691.667; 390 69.3333); gestureRecognizers = <NSArray: 0x600000757660>; layer = <CALayer: 0x60000090afc0>`

Hierarchy

`_UIHostingView<SwiftUIView> UIView UIResponder NSObject`

Backtrace

`Malloc stack logging is not enabled for this process.`

```

viewDidLoad(): Bounds: (0.0, 0.0, 0.0, 0.0)
viewDidLoad(): IntrinsicContentSize: (245.66666666666666, 22.333333333333332)
viewWillAppear(_:): Bounds: (0.0, 0.0, 0.0, 0.0)
viewWillAppear(_:): IntrinsicContentSize: (245.66666666666666, 22.333333333333332)
viewWillLayoutSubviews(): Bounds: (0.0, 0.0, 390.0, 69.33333333333333)
viewWillLayoutSubviews(): IntrinsicContentSize: (390.0, 22.333333333333332)
viewDidLayoutSubviews(): Bounds: (0.0, 0.0, 390.0, 69.33333333333333)
viewDidLayoutSubviews(): IntrinsicContentSize: (390.0, 22.333333333333332)
viewWillAppear(_:): Bounds: (0.0, 0.0, 390.0, 69.33333333333333)
viewWillAppear(_:): IntrinsicContentSize: (390.0, 22.333333333333332)
viewWillLayoutSubviews(): Bounds: (0.0, 0.0, 390.0, 69.33333333333333)
viewWillLayoutSubviews(): IntrinsicContentSize: (390.0, 22.333333333333332)
viewDidLayoutSubviews(): Bounds: (0.0, 0.0, 390.0, 69.33333333333333)
viewDidLayoutSubviews(): IntrinsicContentSize: (390.0, 22.333333333333332)
viewWillAppear(_:): Bounds: (0.0, 0.0, 390.0, 69.33333333333333)
viewWillAppear(_:): IntrinsicContentSize: (390.0, 22.333333333333332)
viewWillLayoutSubviews(): Bounds: (0.0, 0.0, 390.0, 69.33333333333333)
viewWillLayoutSubviews(): IntrinsicContentSize: (390.0, 22.333333333333332)
viewDidLayoutSubviews(): Bounds: (0.0, 0.0, 390.0, 69.33333333333333)
viewDidLayoutSubviews(): IntrinsicContentSize: (390.0, 22.333333333333332)
viewWillAppear(_:): Bounds: (0.0, 0.0, 390.0, 69.33333333333333)
viewWillAppear(_:): IntrinsicContentSize: (390.0, 22.333333333333332)
viewWillLayoutSubviews(): Bounds: (0.0, 0.0, 390.0, 69.33333333333333)
viewWillLayoutSubviews(): IntrinsicContentSize: (390.0, 22.333333333333332)
viewDidLayoutSubviews(): Bounds: (0.0, 0.0, 390.0, 69.33333333333333)
viewDidLayoutSubviews(): IntrinsicContentSize: (390.0, 22.333333333333332)
(lldb)
  
```

Thread 1 > 0 mach_msg_trap

Filter

```
final class HostingController<Content: View>: UIHostingController<Content> {  
    override func viewDidLoadSubviews() {  
        super.viewDidLoadSubviews()  
        view.setNeedsUpdateConstraints()  
    }  
}
```

```
final class HostingController<Content: View>: UIHostingController<Content> {  
    override func viewDidLoadSubviews() {  
        super.viewDidLoadSubviews()  
        view.setNeedsUpdateConstraints()  
    }  
}
```

Navigation with SwiftUI

Navigation with SwiftUI

.navigationDestination

.popover

NavigationLink

NavigationSplitView

NavigationView

NavigationStack

.sheet

.alert

.confirmationDialog

.fullScreenCover

Navigation with SwiftUI

.navigationDestination

.popover

NavigationLink

NavigationSplitView

NavigationView

NavigationStack

.sheet

.alert

.confirmationDialog

.fullScreenCover

Navigation with SwiftUI

.navigationDestination

.popover

NavigationLink

NavigationSplitView

NavigationView

NavigationStack

.sheet

.alert

.confirmationDialog

.fullScreenCover

Navigation with SwiftUI

.navigationDestination

.popover

NavigationLink

NavigationSplitView

DEPRECATED

NavigationView

NavigationStack

.sheet

.alert

.confirmationDialog

.fullScreenCover

Navigation with SwiftUI

.navigationDestination

.popover

NavigationLink

NavigationSplitView

DEPRECATED
NavigationView

NavigationStack

.sheet

.alert

.confirmationDialog

.fullScreenCover

NavigationLink

```
struct FrenchView: View {
    var body: some View {
        NavigationStack { // (prev NavigationView)
            NavigationLink("Push view") {
                Text("Bonjour")
            }
        }
    }
}
```

NavigationLink

```
struct FrenchView: View {
  var body: some View {
    NavigationStack { // (prev NavigationView)
      NavigationLink("Push view") {
        BonjourView()
      }
    }
  }
}
```

```
struct BonjourView: View {
  init() { print("init") }
  var body: some View {
    Text("Bonjour")
  }
}
```

NavigationLink


```
struct FrenchView: View {  
    var body: some View {  
        NavigationStack { // (prev NavigationView)  
            NavigationLink("Push view") {  
                BonjourView()  
            }  
        }  
    }  
}
```

```
struct BonjourView: View {  
    init() { print("init") }  
    var body: some View {  
        Text("Bonjour")  
    }  
}
```

NavigationLink

struct
va

Blog About



Luis Ascorbe
Software Developer. Tech Lead. Speaker.
NSSpain Organizer.

Email
GitHub
Twitter
LinkedIn

MVP + Coordinators in SwiftUI (part 1)

April 27, 2020 · 15 minutes

swiftui coordinator mvp article series part1

```
1 //
2 // Created by Luis Ascorbe on 23/04/2020.
3 // Copyright © 2020 Luis Ascorbe. All rights reserved.
4 //
5
6 import SwiftUI
7
8 protocol BaseCoordinator: AssociatedObject {
9     associatedType ID: View
10    associatedType P: BaseCoordinator
11    func start() -> U
12
13
14    extension BaseCoordinator { // Mixin Extension: Check out AssociatedObject.swift
15        fileprivate var Identifier: UUID {
16            @inlinable
17            get {
18                guard let Identifier: UUID = associatedObject(forKey: &IdentifierKey) else {
19                    self.Identifier = UUID()
20                    return self.Identifier
21                }
22                return Identifier
23            }
24            set { setAssociatedObject(newValue, forKey: &IdentifierKey) }
25        }
26
27        var parent: P? {
28            get { associatedObject(forKey: &ParentKey) }
29            set { setAssociatedObject(newValue, forKey: &ParentKey) }
30        }
31
32        func coordinate(to coordinator: T) -> some View {
33            = coordinator.Identifier // generate Identifier
34            coordinator.parent = self as? T.P
35            return coordinator.start()
36        }
37
38
39    private var IdentifierKey: UUID = 0
40    private var ParentKey: UUID = 0
```

N)

}

}

NavigationStack

and `.navigationDestination(for:, destination:)`

NavigationStack

```
NavigationStack(path: $viewModel.path) {  
    List {  
        NavigationLink(value: Destination.bonjour) {  
            Text("To Bonjour")  
        }  
    }  
    .navigationDestination(for: Destination.self) { destination in  
        switch destination {  
        case .bonjour:  
            BonjourView()  
        }  
    }  
}
```

NavigationStack

```
NavigationStack(path: $viewModel.path) {  
    List {  
        NavigationLink(value: Destination.bonjour) {  
            Text("To Bonjour")  
        }  
    }  
    .navigationDestination(for: Destination.self) { destination in  
        switch destination {  
        case .bonjour:  
            BonjourView()  
        }  
    }  
}
```

NavigationStack

```
NavigationStack(path: $viewModel.path) {  
    List {  
        NavigationLink(value: Destination.bonjour) {  
            Text("To Bonjour")  
        }  
    }  
    .navigationDestination(for: Destination.self) { destination in  
        switch destination {  
        case .bonjour:  
            BonjourView()  
        }  
    }  
}
```

NavigationStack

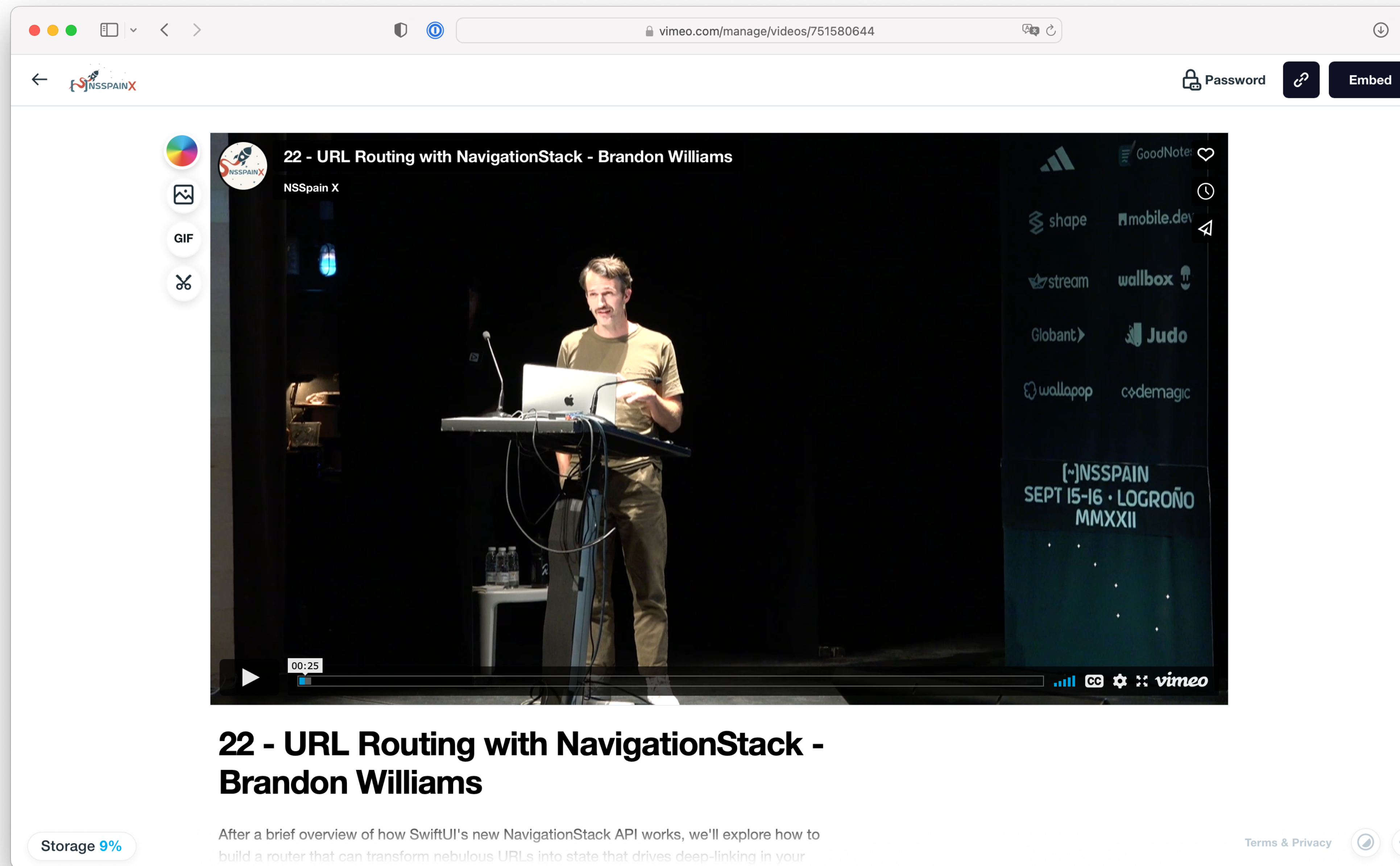
```
NavigationStack(path: $viewModel) {
  List {
    NavigationLink(value:
      Text("To Bonjour"),
    )
  }
}

.navigationDestination(for: Destination.self) { destination in
  switch destination {
  case .bonjour:
    BonjourView()
  }
}
}
```

```
enum Destination: Hashable {
  case bonjour
}
class ViewModel: ObservableObject {
  @Published var path: [Destination]
}
```

Brandon Williams

@mbrandonw



The screenshot shows a Vimeo video player interface. The video title is "22 - URL Routing with NavigationStack - Brandon Williams" by "NSSpain X". The video content shows a man, Brandon Williams, standing at a podium on a stage, presenting a slide. The slide displays a list of logos for various companies: GoodNote, shape, mobile.dev, stream, wallbox, Globant, Judo, wallpop, and codemagic. At the bottom of the slide, it says "[~]NSSPAIN SEPT 15-16 · LOGROÑO MMXXII". The video player includes a progress bar at the bottom showing 00:25, and a "Storage 9%" indicator in the bottom left corner. The Vimeo logo is visible in the bottom right corner of the video player.

22 - URL Routing with NavigationStack - Brandon Williams

After a brief overview of how SwiftUI's new NavigationStack API works, we'll explore how to build a router that can transform nebulous URLs into state that drives deep-linking in your

Storage 9%

Terms & Privacy

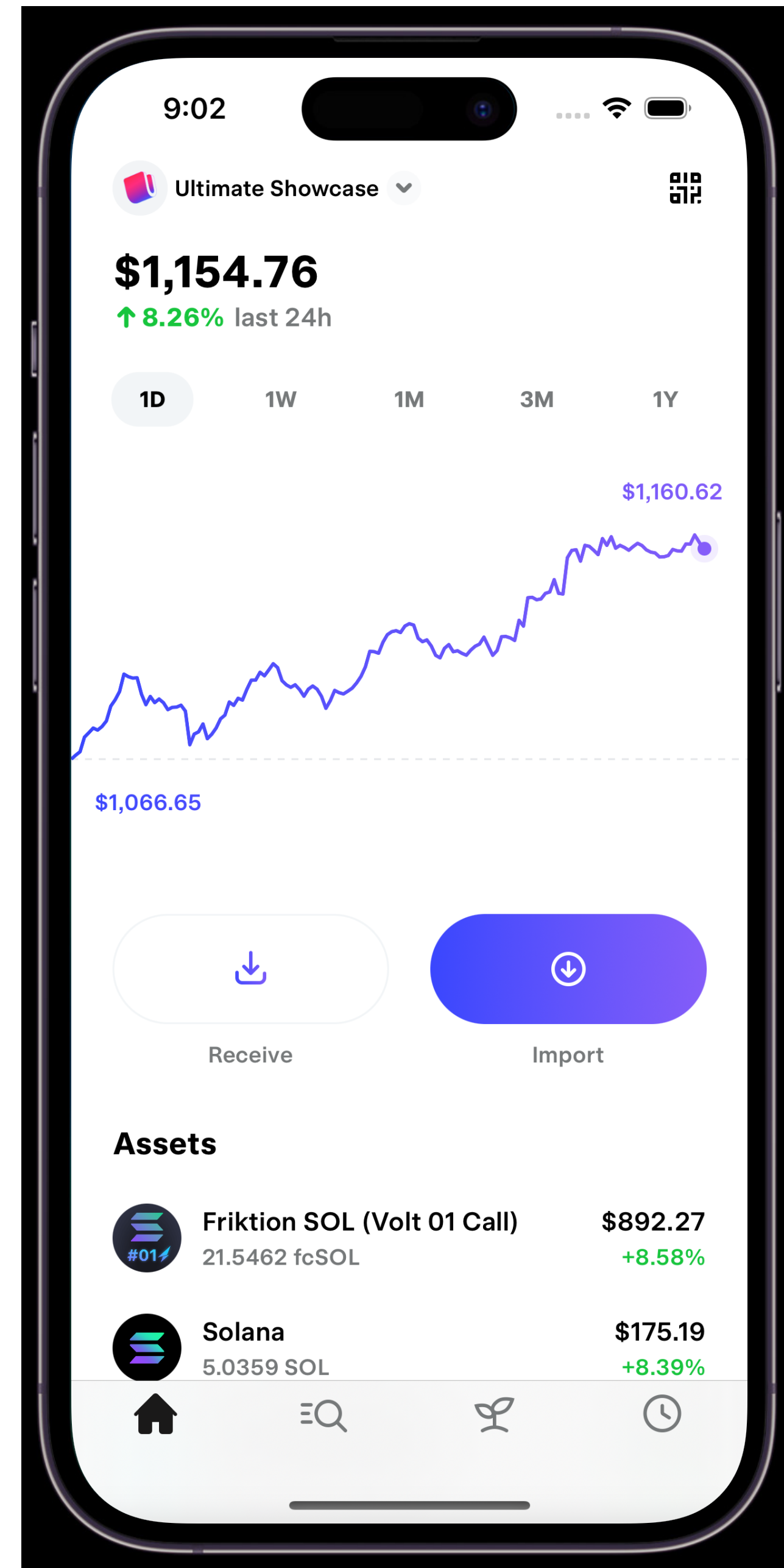
vimeo.com/nsspain/swiftui-navigation

The Composable Architecture

github.com/pointfreeco/swift-composable-architecture

Real life use case

ultimate.money



Which one to choose?

You should consider

You should consider

Navigation with UIKit if:

- Your project is UIKit
- You want to decouple the navigation logic from views
- You want to start using SwiftUI
- You can't switch your architecture

You should consider

Navigation with UIKit if:

- Your project is UIKit
- You want to decouple the navigation logic from views
- You want to start using SwiftUI
- You can't switch your architecture

Navigation with SwiftUI if:

- Your deployment target is iOS 14+
- You are starting a new project
- You want to use TCA

Merci

Luis Ascorbe @lascorbe