SWIFTCRAFT
21-24 May 2024

Scaling Up with SwiftUI

Aryaman Sharda

👋 Hi, I'm Aryaman

@aryamansharda

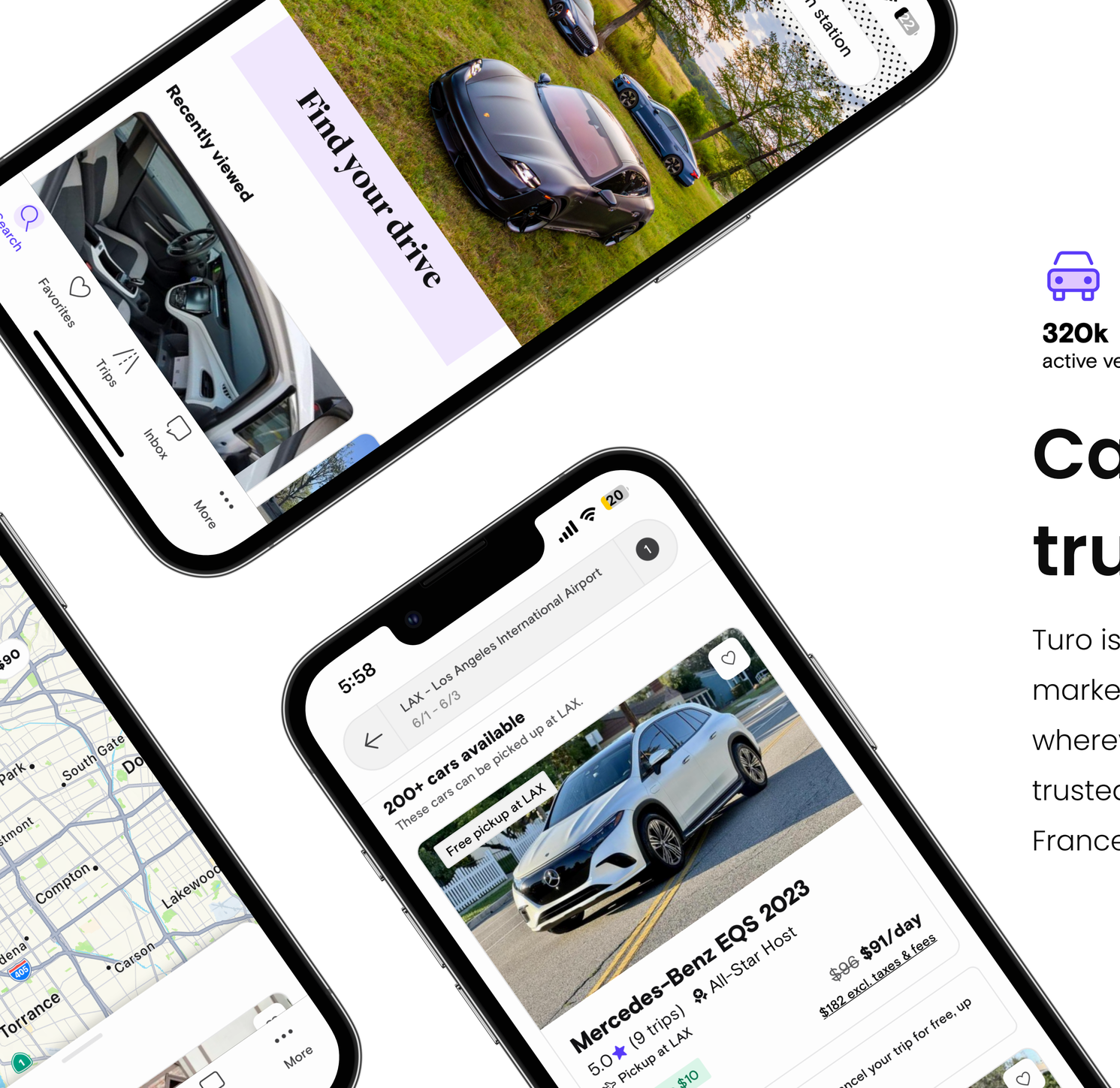
- 👨‍💻 Staff iOS Engineer at Turo
- 👨‍💻 Previously iOS & CarPlay at Porsche
- ✍️ Blog @ digitalbunker.dev
- ✉️ Newsletter @ indie.watch
- 🌉 San Francisco, CA

Overview

01

iOS @ TURO

Starting June 2012



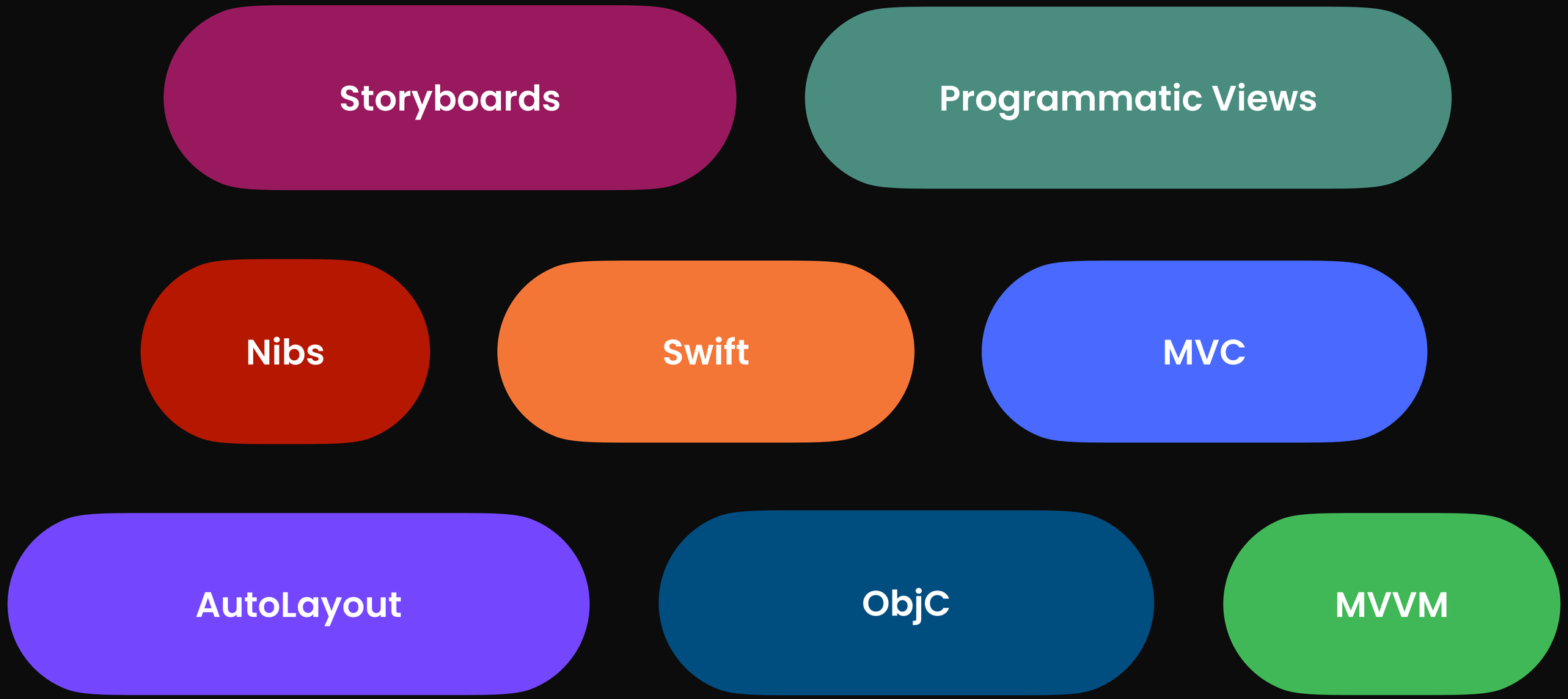
320k
active vehicles

1,400+
makes and models

11,000+
cities with active vehicles

Car rentals from trusted, local hosts

Turo is the world's largest peer-to-peer car sharing marketplace, where guests can book any car they want, wherever they want it, from a vibrant community of trusted hosts across the US, UK, Canada, Australia, and France.


Storyboards
Programmatic Views
Nibs
Swift
MVC
AutoLayout
ObjC
MVVM

Storyboards

Programmatic Views

Nibs

Swift

SwiftUI

MVC

AutoLayout

ObjC

MVVM

02

Transitioning to SwiftUI

March 2023

Creating Pods

Specialized working groups to investigate these topics and integrate developments from WWDC.

SwiftUI

Modularization

Performance

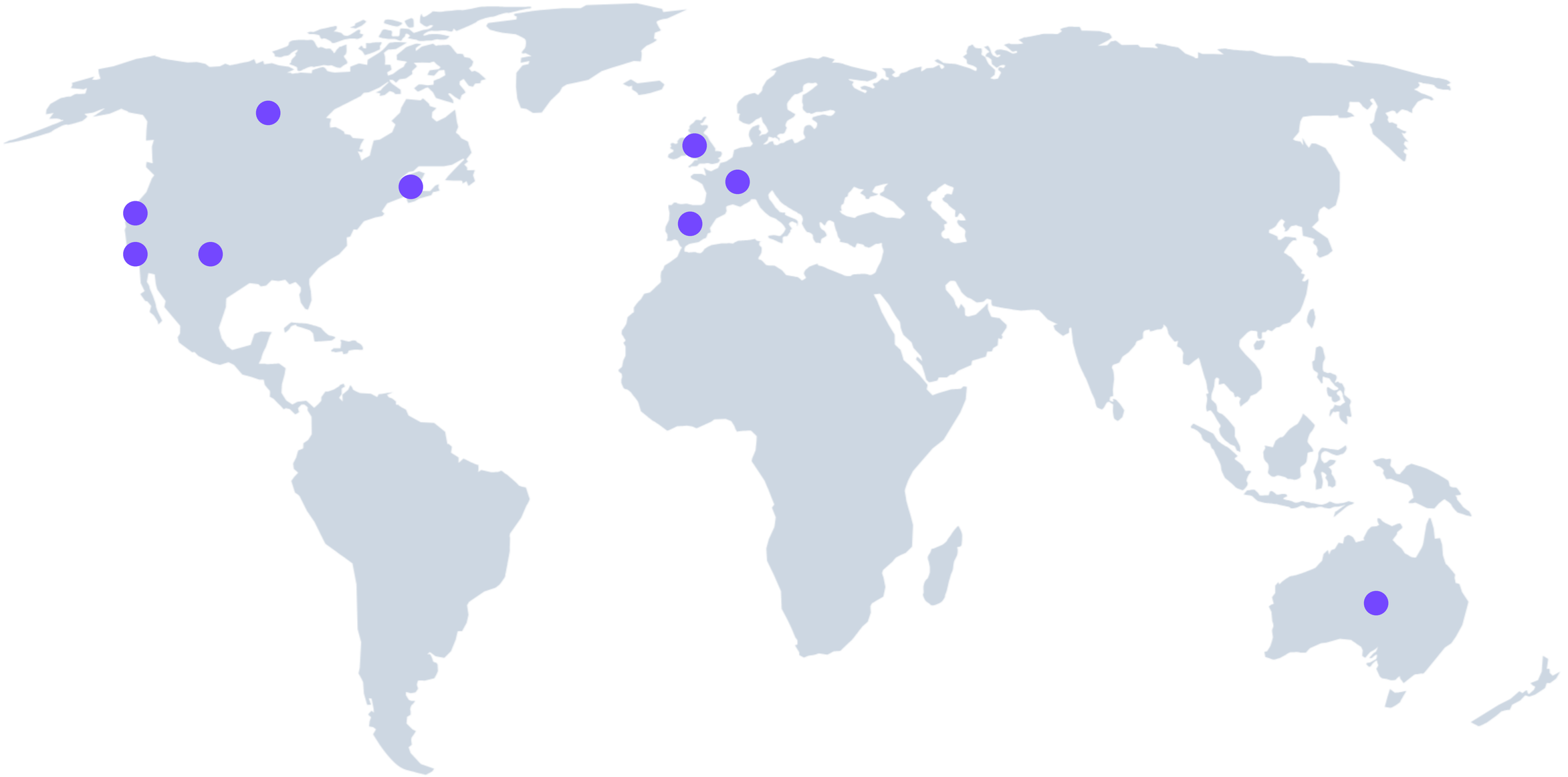
Networking & Model Generation

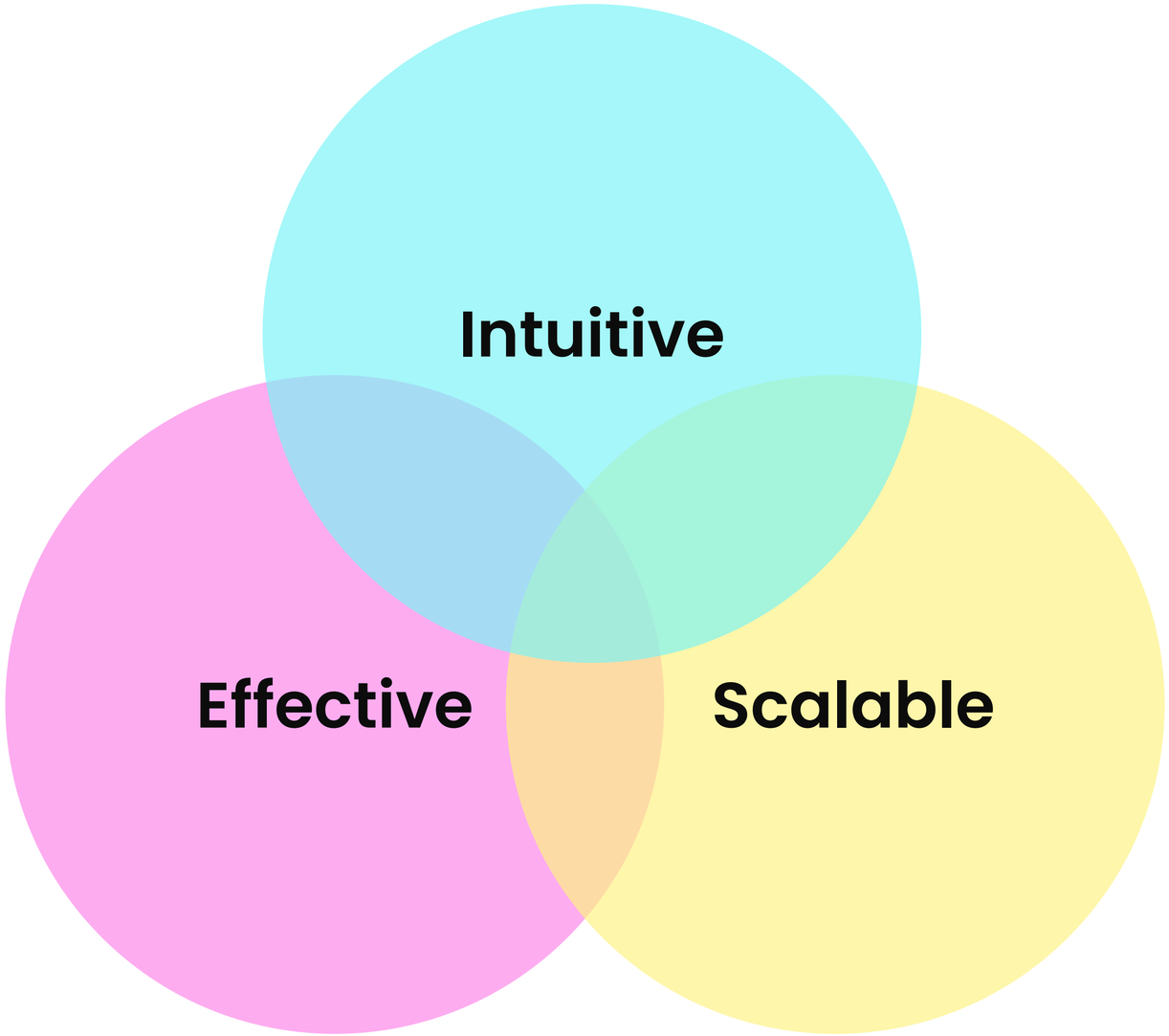
App Store Release Process

Internationalization

Goal

To establish the best practices and conventions for SwiftUI at Turo, along with the essential tools, documentation, and support necessary to make SwiftUI the standard for all new development.




Intuitive
Effective
Scalable

03

Milestones

Learning The Basics

01

Getting Started

Observability

View Modifiers vs View Builders

`.task` vs `.onAppear`

Breaking Up Large SwiftUI Views

SwiftUI Lifecycle

Localization

SwiftUI ⟷ UIKit

Declarative Programming

01

Late March 2023

Learning The Basics

Exploring SwiftUI Architectures

02

DC iOS: SwiftUI Architecture and Best Practices

16K views • 7 months ago

Dev Community

SwiftUI Architecture and Best Practices Mohammad Azam In this talk Azam will share his experience of when working with SwiftUI ...

CC

What Is The Best Architecture For SwiftUI? 👀

7.3K views • 1 year ago

tundsdev

What Is The Best **Architecture** For **SwiftUI**? ********** *Timestamps:* 00:00 - Intro 00:33 - What is an ...

4K

Intro | What is an architecture | Why do you developers care about architectures | Making things scalab... 7 chapters

VIM: A new SwiftUI Architecture for iOS 17

3.6K views • 7 months ago

Flo writes Code

We all know MVVM, MVC, and perhaps even TCA. But might it be time to consider a new **architecture**? Let's explore how we can ...

Intro | App Intents | VIM Architecture 3 chapters

STOP using MVVM for SwiftUI | Clean iOS Architecture

28K views • 2 years ago

Rebeloper - Rebel Developer

STOP using MVVM for **SwiftUI** | Clean iOS **Architecture** Want to build an app with Design Patterns for iOS and clean **architecture**?

CC

INTRO | WHY MVVM | IVO BASICS | IVO EXAMPLE | IDENTIFIABLE | OBSERVED | VIEW CONNECTIONS ... 9 chapters

SwiftUI Architectures

Plenty of options to pick from...

MVP

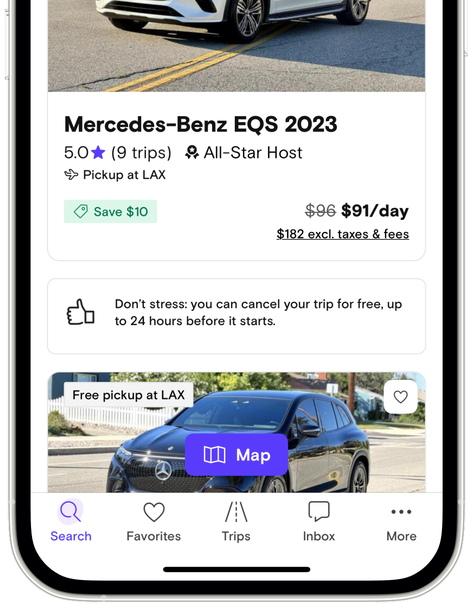
MVVM

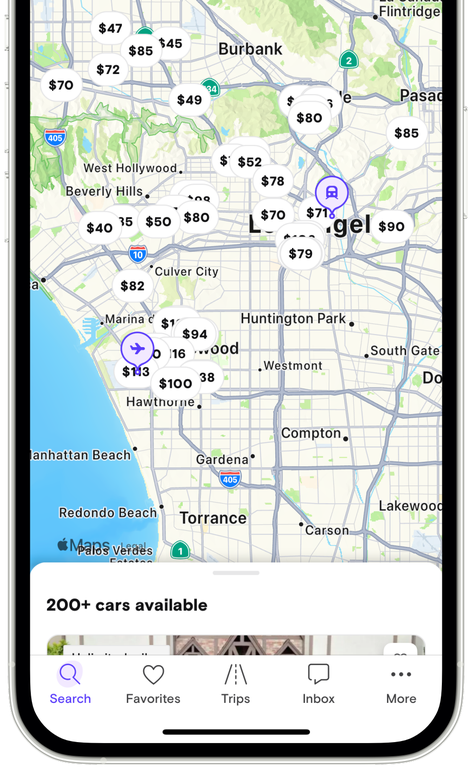
MVVM-C

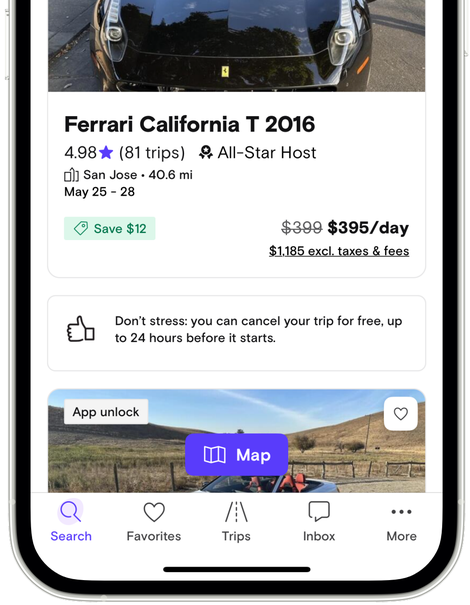
TCA (The Composable Architecture)

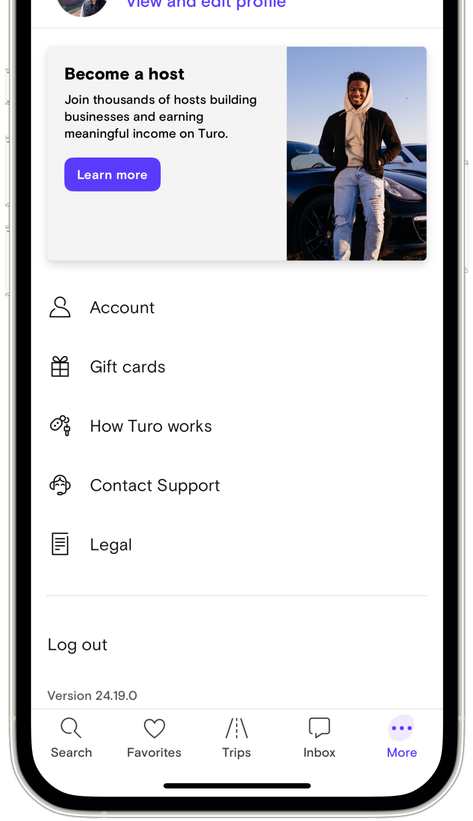
MV

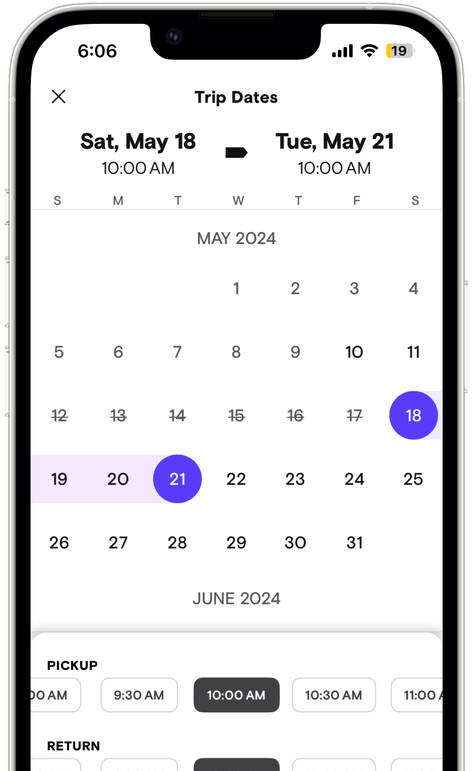
and more...

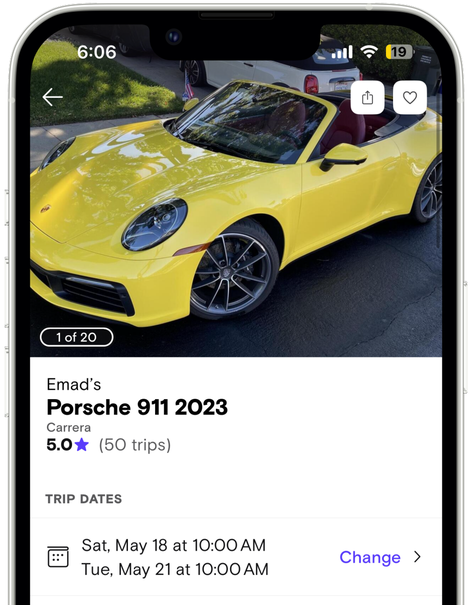

Mercedes-Benz EQS 2023
5.0★ (9 trips) All-Star Host
Pickup at LAX
Save $10
$96 $91/day
$182 excl. taxes & fees
Don't stress: you can cancel your trip for free, up to 24 hours before it starts.
Free pickup at LAX
Map
Search
Favorites
Trips
Inbox
More

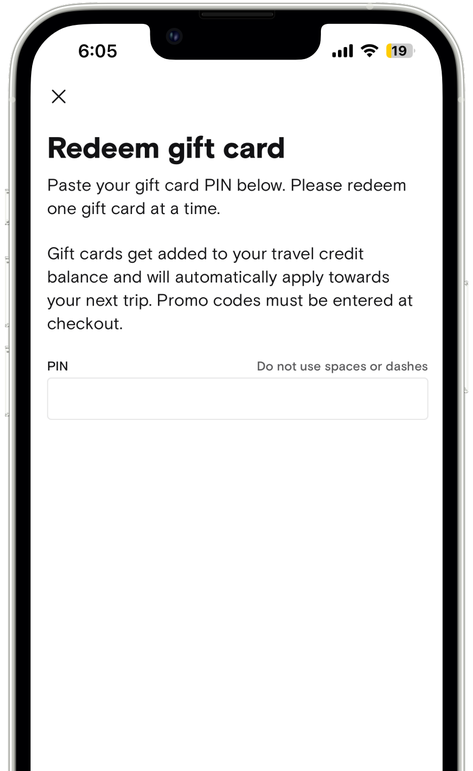

$47
$85
$45
Burbank
$72
$70
$49
$80
Pasad
$85
West Hollywood
$52
Beverly Hills
$78
$40
$50
$80
$70
$90
$79
Culver City
$82
Huntington Park
$94
South Gate
$100
Westmont
Hawthorne
Compton
Gardena
Redondo Beach
Torrance
Carson
Lakewood
Palos Verdes
200+ cars available
Search
Favorites
Trips
Inbox
More

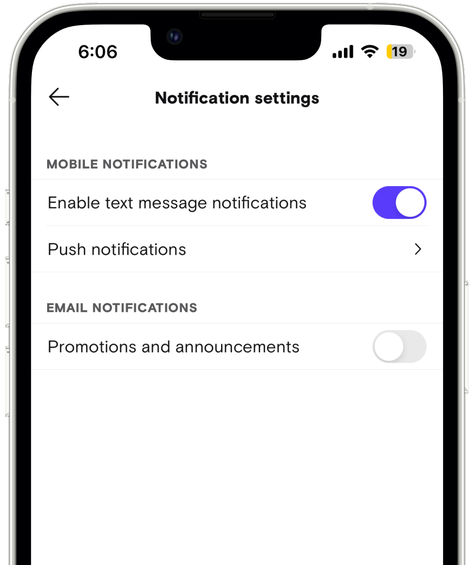

Ferrari California T 2016
4.98★ (81 trips) All-Star Host
San Jose • 40.6 mi
May 25 - 28
Save $12
$399 $395/day
$1,185 excl. taxes & fees
Don't stress: you can cancel your trip for free, up to 24 hours before it starts.
App unlock
Map
Search
Favorites
Trips
Inbox
More


View and edit profile
Become a host
Join thousands of hosts building businesses and earning meaningful income on Turo.
Learn more
Account
Gift cards
How Turo works
Contact Support
Legal
Log out
Version 24.19.0
Search
Favorites
Trips
Inbox
More

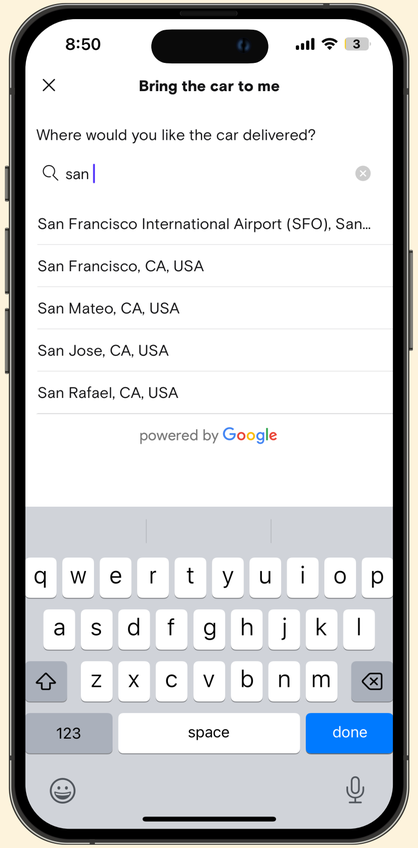

6:06
Trip Dates
Sat, May 18
10:00 AM
Tue, May 21
10:00 AM
S M T W T F S
MAY 2024
1 2 3 4
5 6 7 8 9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30 31
JUNE 2024
PICKUP
9:30 AM
10:00 AM
10:30 AM
RETURN


6:06
1 of 20
Emad's
Porsche 911 2023
Carrera
5.0★ (50 trips)
TRIP DATES
Sat, May 18 at 10:00 AM
Tue, May 21 at 10:00 AM
Change

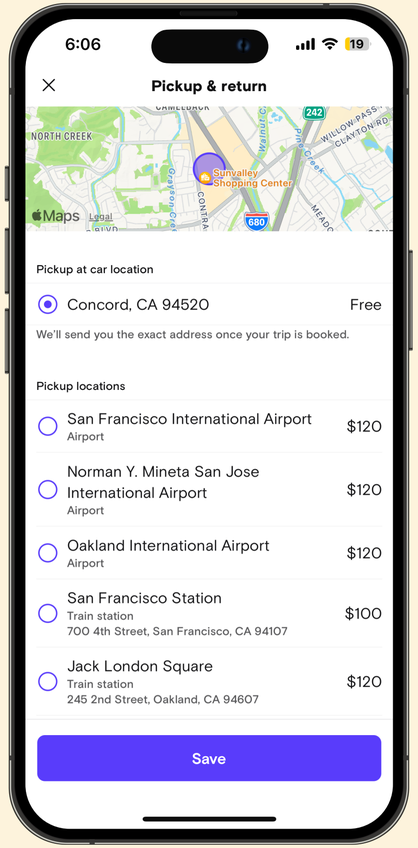

6:05
Redeem gift card
Paste your gift card PIN below. Please redeem one gift card at a time.
Gift cards get added to your travel credit balance and will automatically apply towards your next trip. Promo codes must be entered at checkout.
PIN
Do not use spaces or dashes


6:06
Notification settings
MOBILE NOTIFICATIONS
Enable text message notifications
Push notifications
EMAIL NOTIFICATIONS
Promotions and announcements

UIKit to SwiftUI Conversions



List



Map

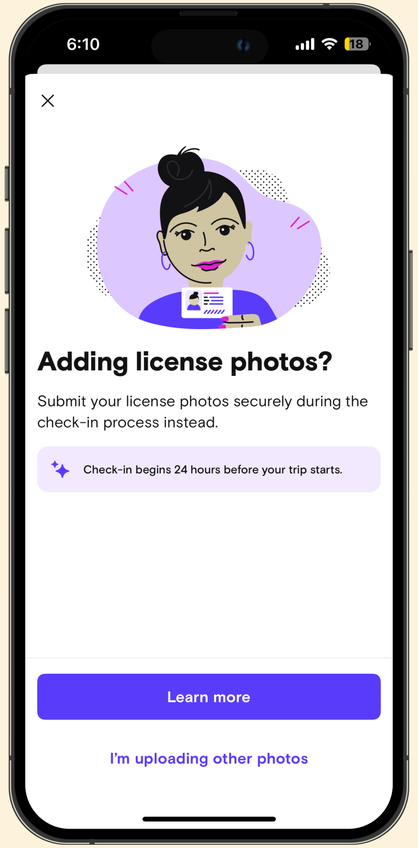
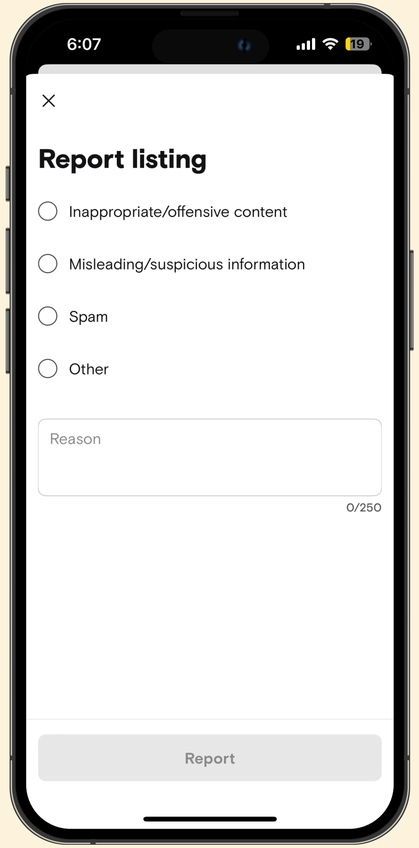
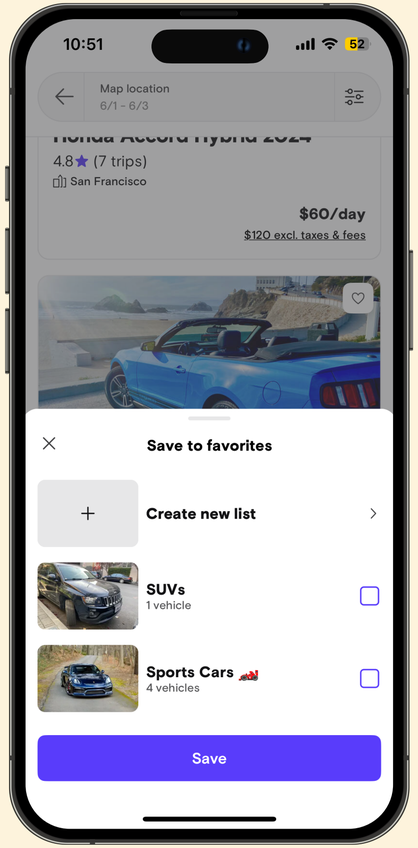


Photo Upload

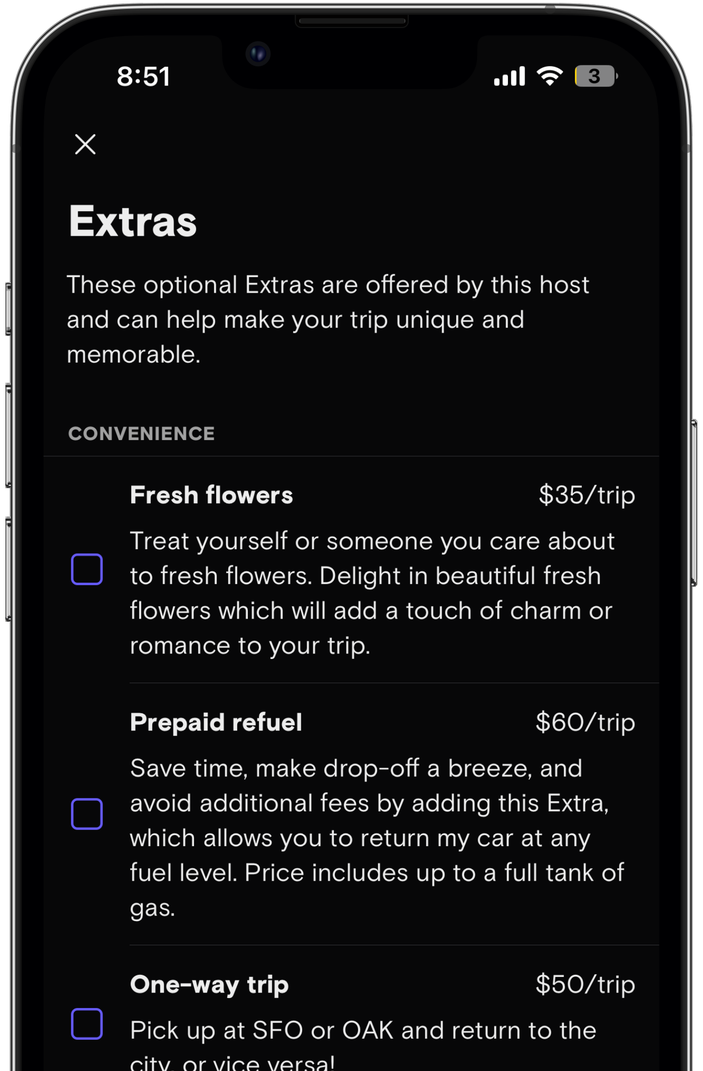


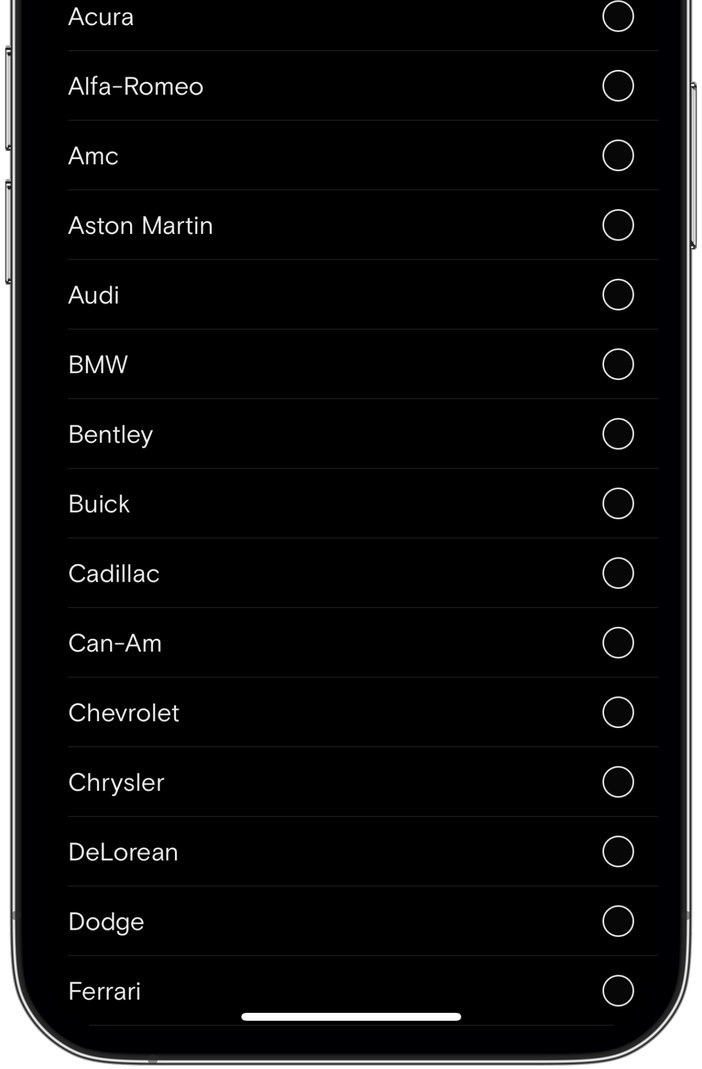
Form



Custom Sheet

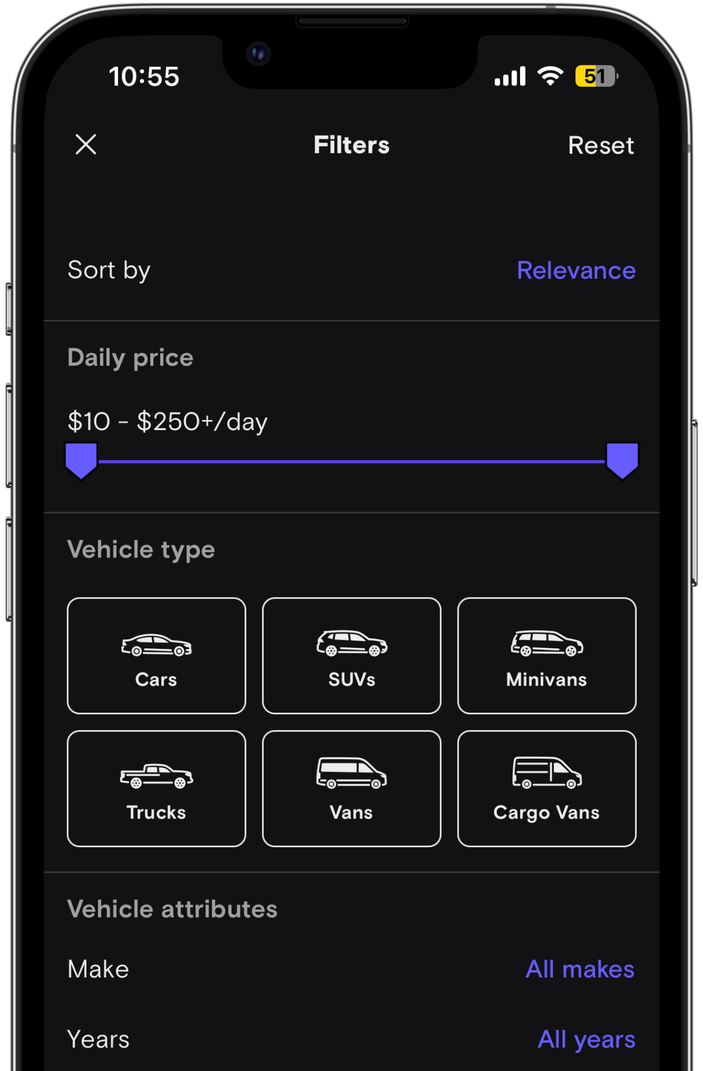
MVVM & VIM





TCA & MV

MVP & MVVM



02

Late March - Mid April 2023

Exploring SwiftUI Architectures

Evaluating SwiftUI Architectures

03

What are we looking for?

An approach that would allow for:

01 **Separation Between Business Logic & View**

02 **Testable**

03 **Structured Data Flow**

04 **Scalable**

05 **Supports Modularization**

06 **Interoperability Between UIKit & SwiftUI**

What are we looking for?

An approach that would allow for:

01 Separation Between Business Logic & View

02 **Testable**

03 Structured Data Flow

04 Scalable

05 Supports Modularization

06 Interoperability Between UIKit & SwiftUI

What are we looking for?

An approach that would allow for:

01 **Separation Between Business Logic & View**

02 **Testable**

03 **Structured Data Flow**

04 **Scalable**

05 **Supports Modularization**

06 **Interoperability Between UIKit & SwiftUI**

What are we looking for?

An approach that would allow for:

01 Separation Between Business Logic & View

02 Testable

03 Structured Data Flow

04 Scalable

05 Supports Modularization

06 Interoperability Between UIKit & SwiftUI

What are we looking for?

An approach that would allow for:

01 Separation Between Business Logic & View

02 Testable

03 Structured Data Flow

04 Scalable

05 Supports Modularization

06 Interoperability Between UIKit & SwiftUI

What are we looking for?

An approach that would allow for:

01 Separation Between Business Logic & View

02 Testable

03 Structured Data Flow

04 Scalable

05 Supports Modularization

06 Interoperability Between UIKit & SwiftUI

07 **Separation Between View & Backend Response**

08 **Supports Dependency Injection**

09 **Easy Onboarding**

10 **Cross-Platform Support**

07 Separation Between View & Backend Response

09 Easy Onboarding

08 Supports Dependency Injection

10 Cross-Platform Support

07 Separation Between View & Backend Response

09 Easy Onboarding

08 Supports Dependency Injection

10 Cross-Platform Support

07 Separation Between View & Backend Response

09 Easy Onboarding

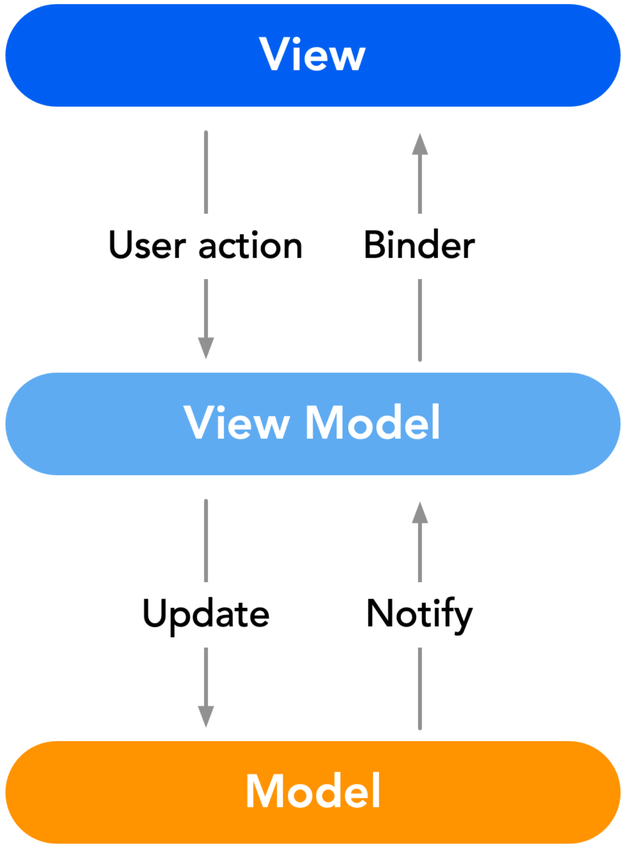
08 Supports Dependency Injection

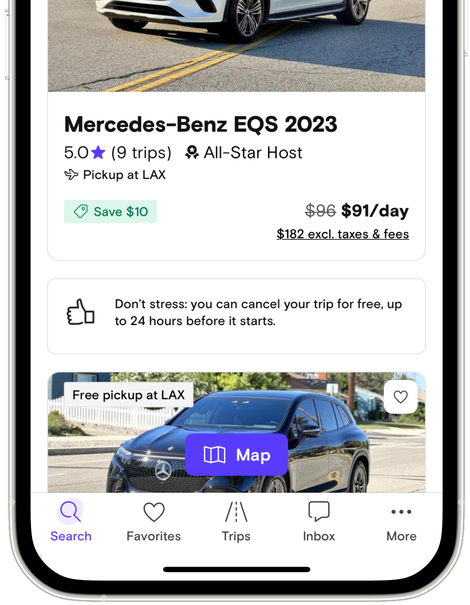
10 Cross-Platform Support

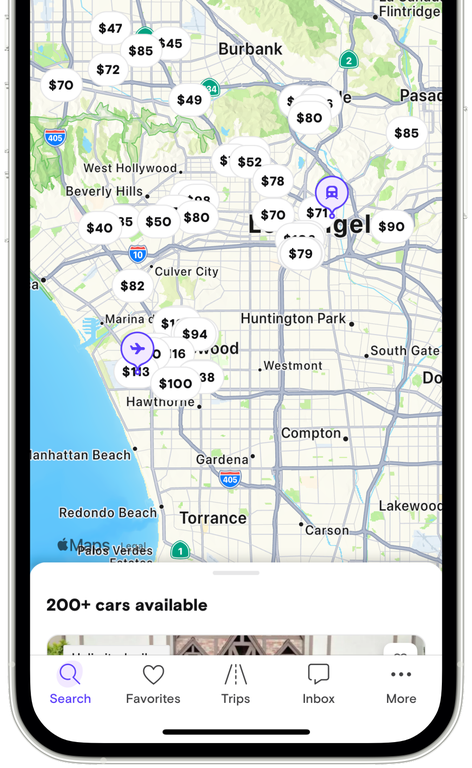
MVVM

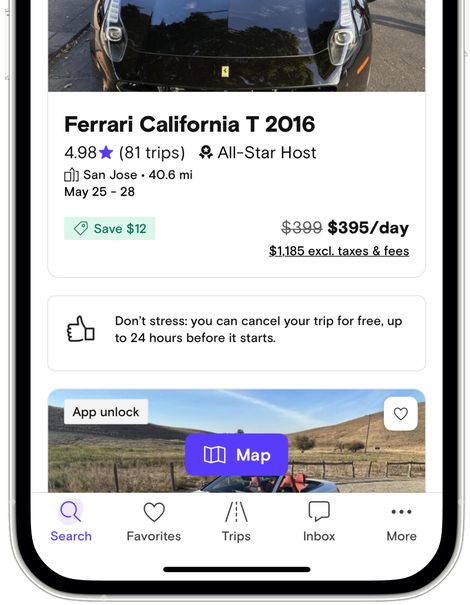
Model-View-View Model

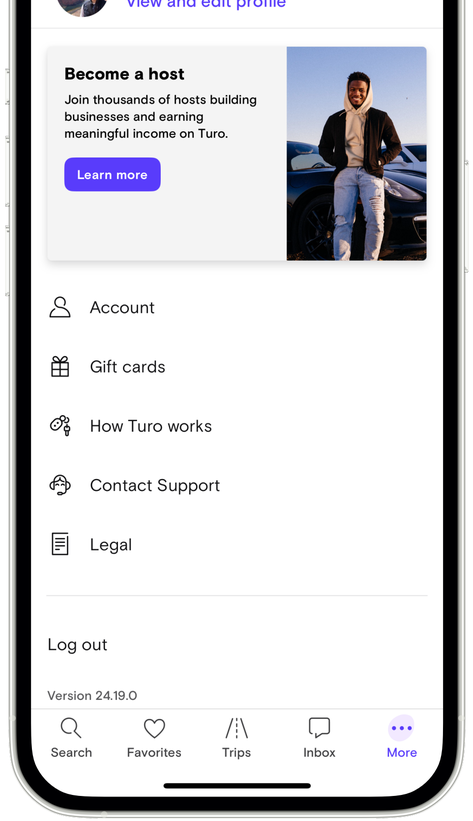
The Structure of the MVVM Pattern

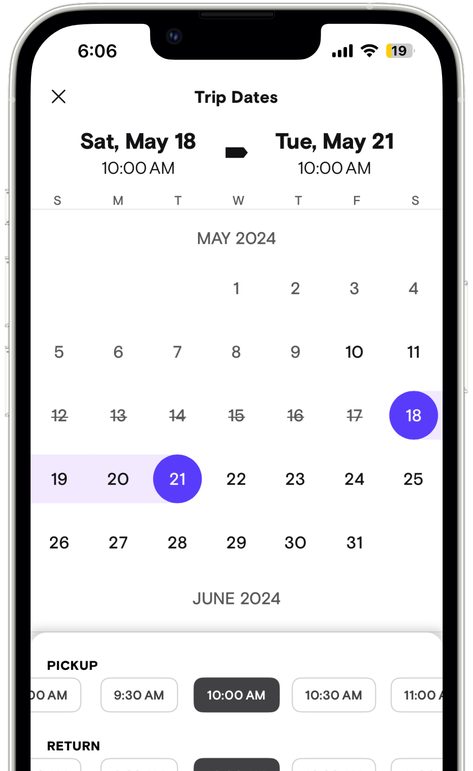


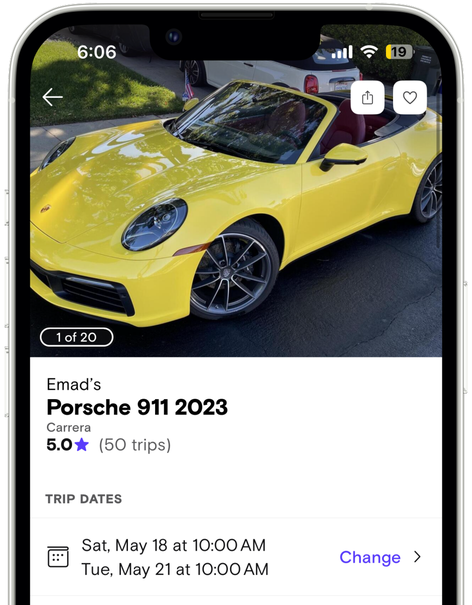

Mercedes-Benz EQS 2023
5.0★ (9 trips) All-Star Host
Pickup at LAX
Save $10
$96 $91/day
$182 excl. taxes & fees
Don't stress: you can cancel your trip for free, up to 24 hours before it starts.
Free pickup at LAX
Map
Search
Favorites
Trips
Inbox
More

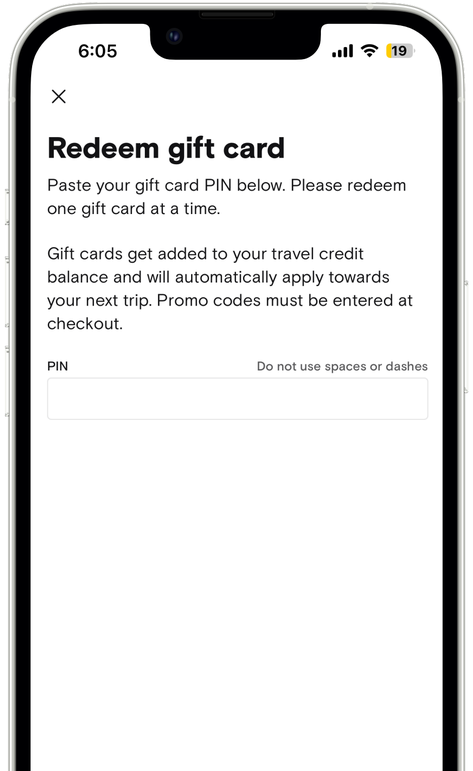

$47
$85
$45
Burbank
$72
$70
$49
$80
Pasad
$85
West Hollywood
$52
Beverly Hills
$78
$40
$50
$80
$70
$90
$79
Culver City
$82
Huntington Park
$94
South Gate
$100
Westmont
Hawthorne
Compton
Gardena
Redondo Beach
Torrance
Carson
Lakewood
Palos Verdes
200+ cars available
Search
Favorites
Trips
Inbox
More

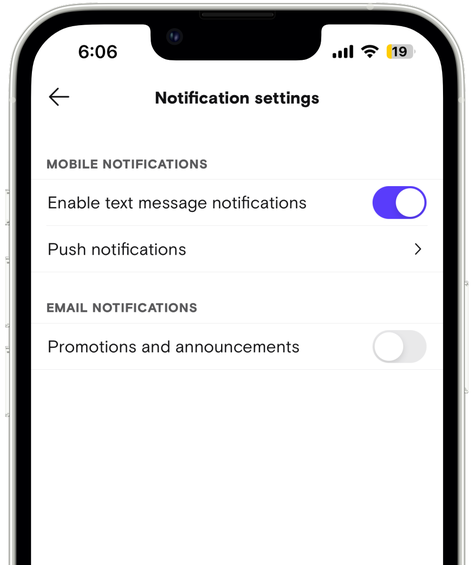

Ferrari California T 2016
4.98★ (81 trips) All-Star Host
San Jose • 40.6 mi
May 25 - 28
Save $12
$399 $395/day
$1,185 excl. taxes & fees
Don't stress: you can cancel your trip for free, up to 24 hours before it starts.
App unlock
Map
Search
Favorites
Trips
Inbox
More


View and edit profile
Become a host
Join thousands of hosts building businesses and earning meaningful income on Turo.
Learn more
Account
Gift cards
How Turo works
Contact Support
Legal
Log out
Version 24.19.0
Search
Favorites
Trips
Inbox
More


6:06
Trip Dates
Sat, May 18
10:00 AM
Tue, May 21
10:00 AM
S M T W T F S
MAY 2024
1 2 3 4
5 6 7 8 9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30 31
JUNE 2024
PICKUP
9:30 AM
10:00 AM
10:30 AM
RETURN

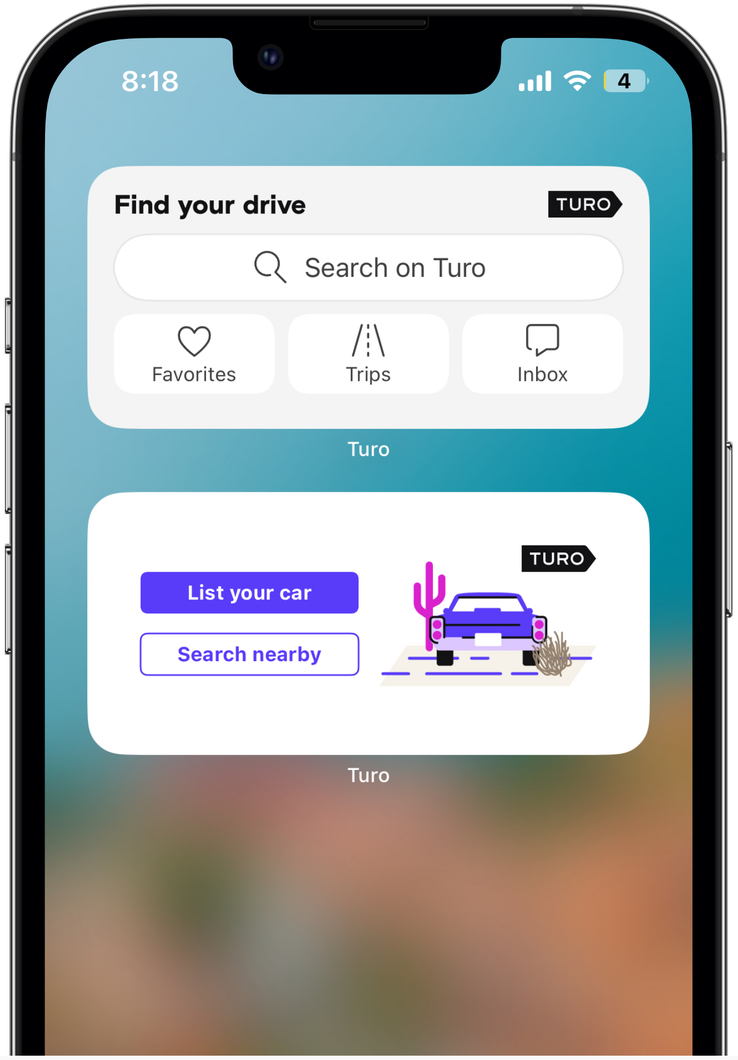

6:06
1 of 20
Emad's
Porsche 911 2023
Carrera
5.0★ (50 trips)
TRIP DATES
Sat, May 18 at 10:00 AM
Tue, May 21 at 10:00 AM
Change


6:05
Redeem gift card
Paste your gift card PIN below. Please redeem one gift card at a time.
Gift cards get added to your travel credit balance and will automatically apply towards your next trip. Promo codes must be entered at checkout.
PIN
Do not use spaces or dashes

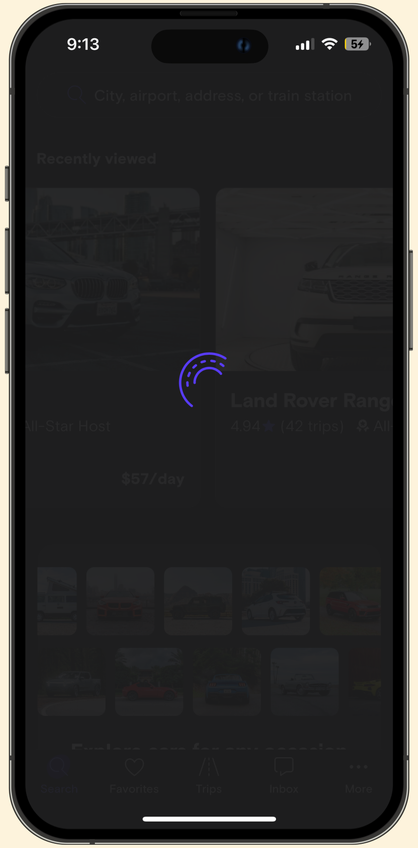

6:06
Notification settings
MOBILE NOTIFICATIONS
Enable text message notifications
Push notifications
EMAIL NOTIFICATIONS
Promotions and announcements

Moving to MVVM

- Clear Separation Between UI & Business Logic
- Easy Unit & Snapshot Testing
- Easy Dependency Injection
- Minor Learning Curve
- Structured Data Flow



Finding Blockers via SwiftUI Conversions



Activity Indicator

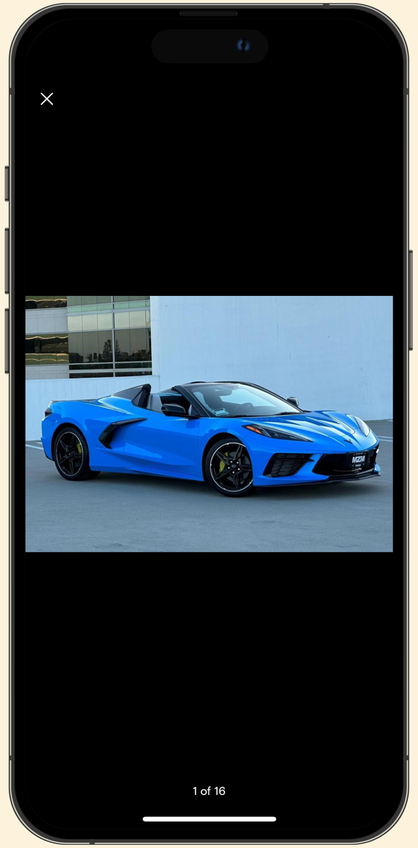
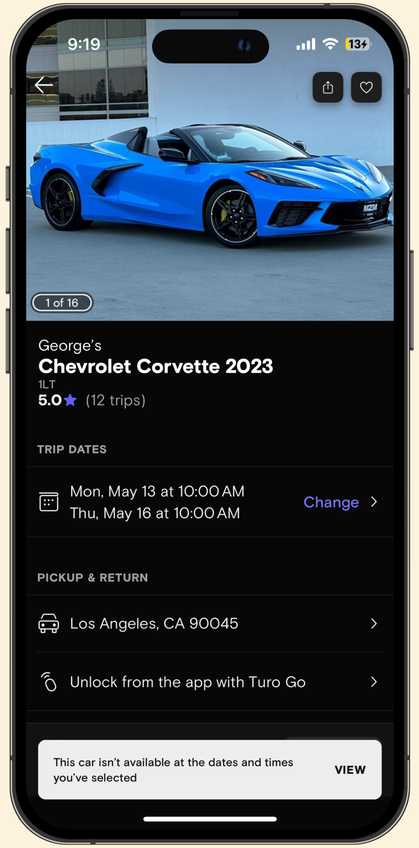
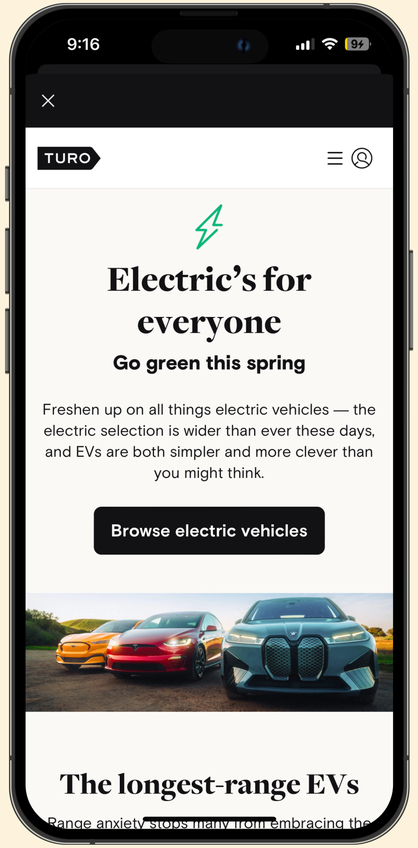


Image Carousel

Bottom Sheet



Toast



Web View

03

Mid April - Early July 2023

Evaluating SwiftUI Architectures

Team Onboarding

04

Style Guide

Onboarding Support

1.

Weekly Team Meetings

A chance to discuss concerns, clarify our decisions, and work through the new guidelines with the team.

2.

Office Hours

Zoom rooms were set up where team members could receive extra help and collaborate with the Pod on SwiftUI issues, with multiple sessions available each week.

3.

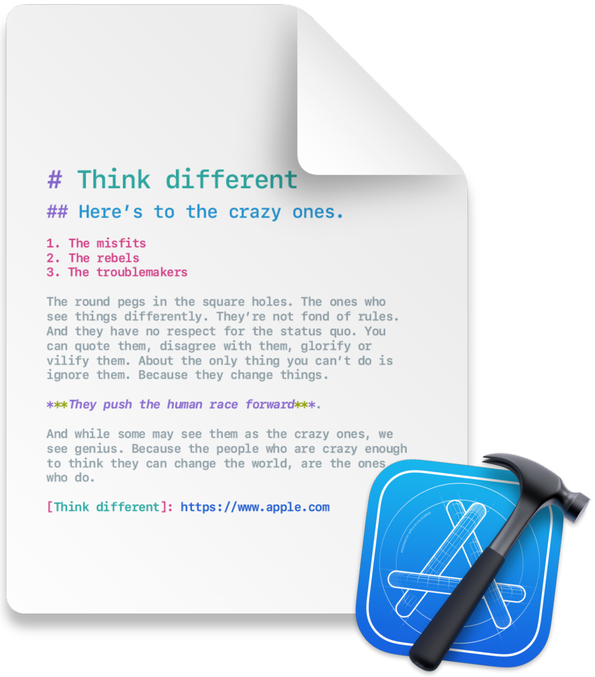
1:1 Mentoring

We supported those new to SwiftUI by pairing them with experienced developers for more hands-on guidance.

4.

Learning Resources

Documentation, tutorial videos, articles, code samples, workshops, group code reviews, etc.



SwiftUI Guidelines.md

- Accessible Within Xcode
- Single Source Of Truth For Best Practices And Conventions
- Less Context-Switching And Faster Onboarding
- Easy To Search For Common Helpers, Utilities, Useful Links, etc.
- Fewer Styling And Formatting Errors In Pull Requests
- Automatic Updates Through Version Control

04

Early July - Early September 2023

Team Onboarding

Developer Tooling

05

Getting Rid of Storyboards

Storyboards are prone to merge conflicts, slow down large projects, are difficult to code review, and are just generally a pain to use.

StoryboardToSwiftUI

Public

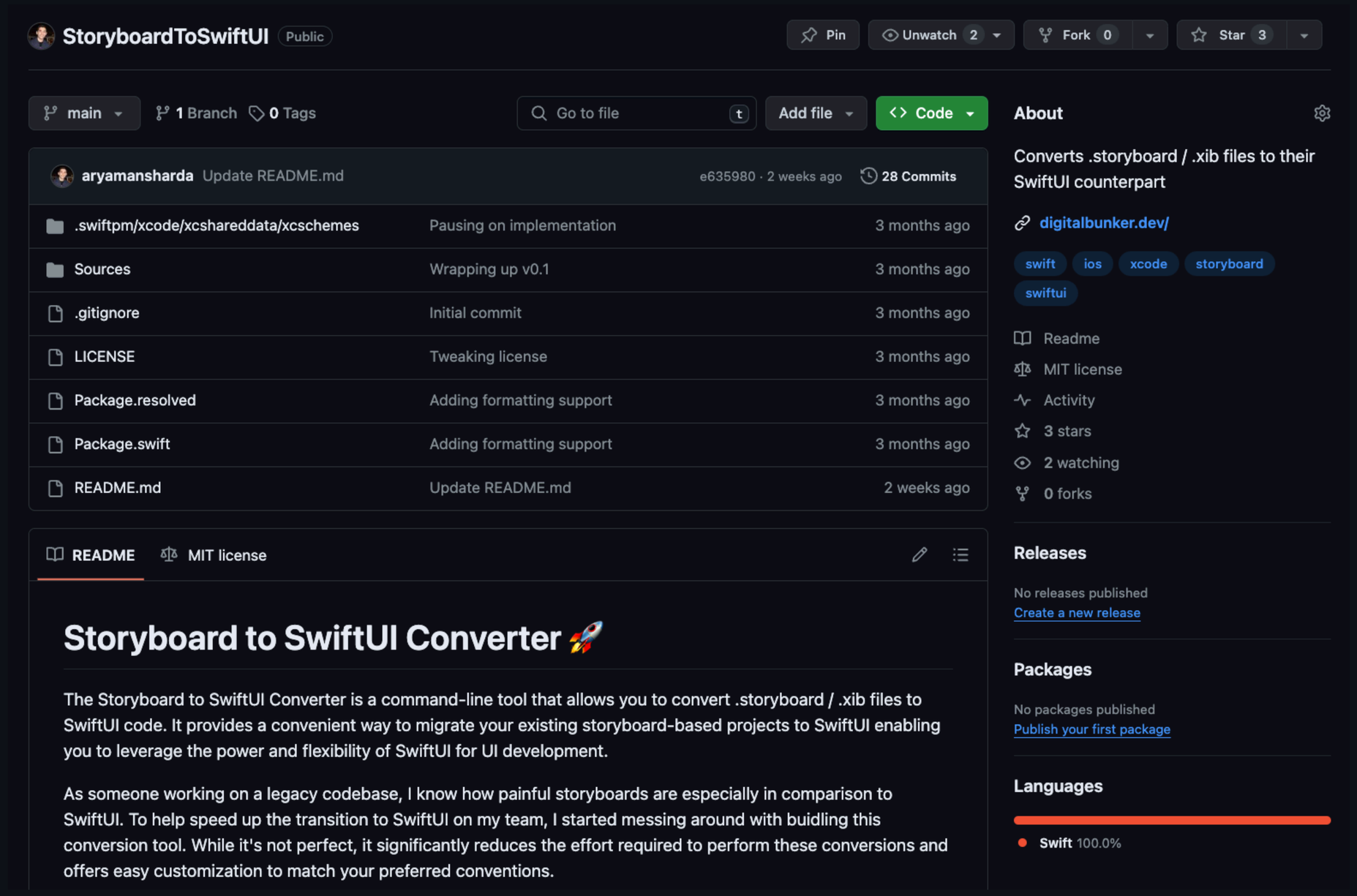
Converts .storyboard / .xib files to their SwiftUI counterpart

swift ios xcode storyboard swiftui

Swift 3 MIT License Updated 2 weeks ago

```
<?xml version="1.0" encoding="UTF-8"?>
<document type="…" … initialViewController="RMJ-Sm-gi6">
    <scenes>
        <scene sceneID="f4A-B1-2Ad">
            <objects>
                <viewController
                    storyboardIdentifier="EarningsViewController" … >
                    …
                </viewController>
            </objects>
        </scene>
    </scenes>
</document>
```

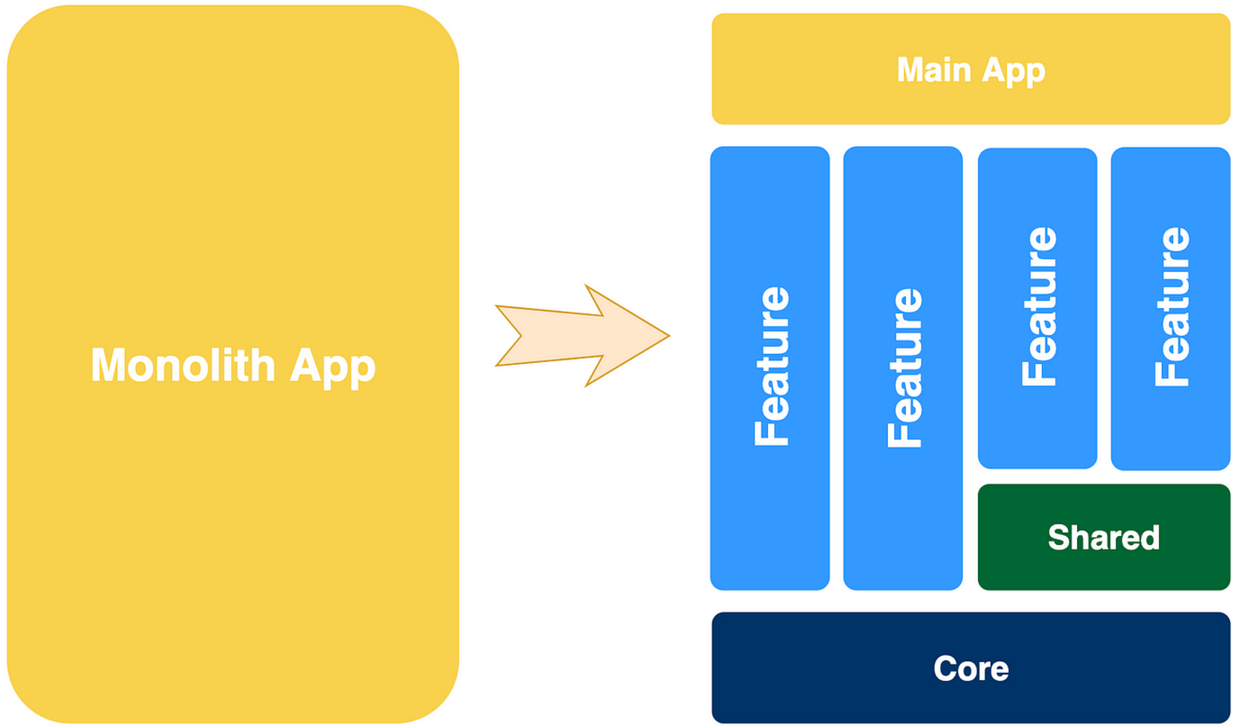
```swift
import SwiftUI

public struct EarningsScreen: View {
    public init() {}

    public var body: some View {
        ScrollView {
            VStack(alignment: .leading, spacing: .space16) {
                Text(netTripEarningsAmount)
                    .textToken(.body)

                VStack(alignment: .leading, spacing: .space16) {
                    Text(totalTripPrice)
                        .textToken(.headerXS)
                    Text(averageDailyVehiclePrice)
                        .textToken(.body)
                }
            }
        }
    }
}
```

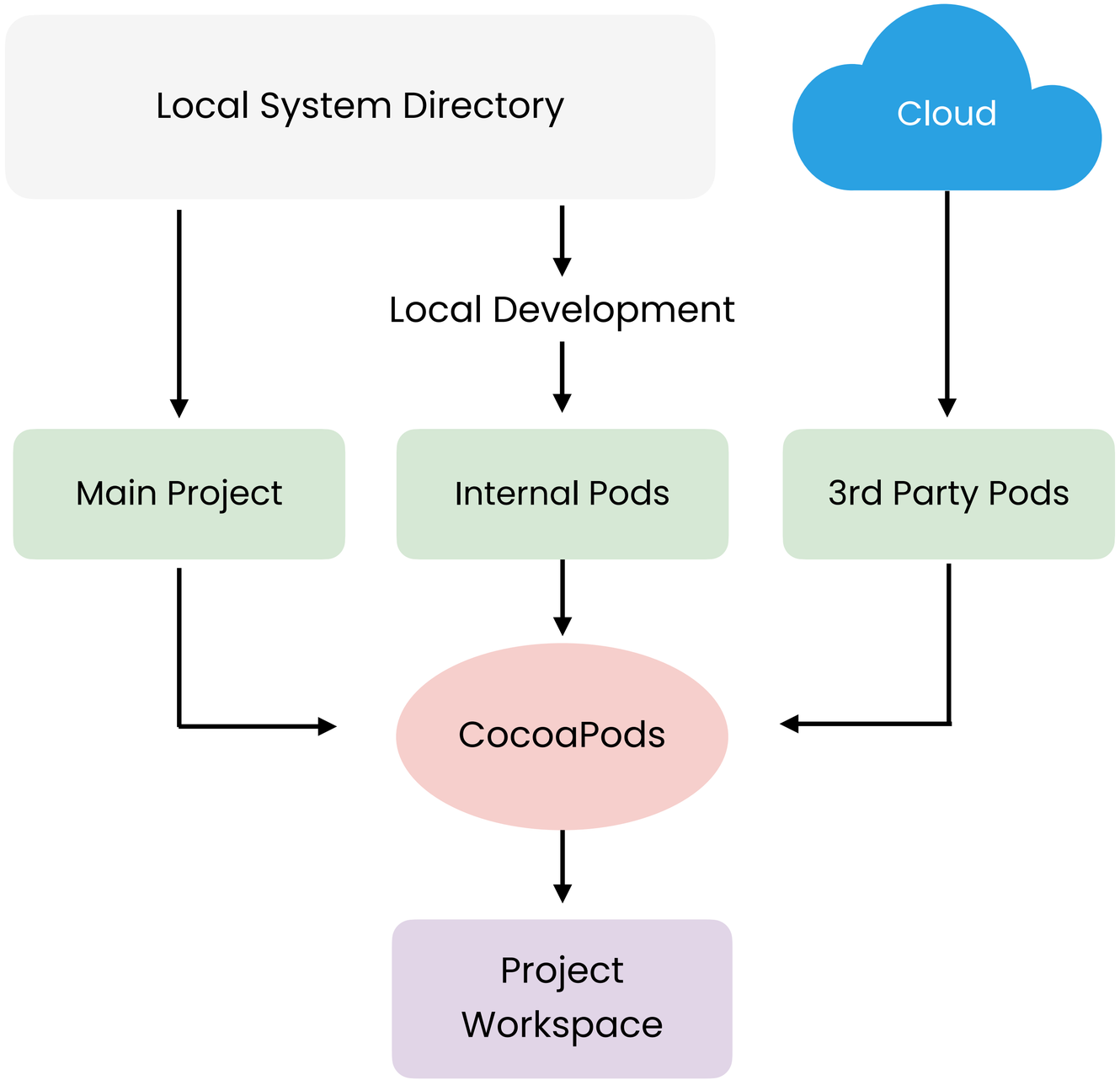


https://github.com/aryamansharda/StoryboardToSwiftUI

Modularization

Modularization involves breaking an application into smaller, independent modules. This reduces compile times and makes SwiftUI Previews faster and more responsive since each module can be compiled on its own.




Local System Directory
Cloud
Local Development
Main Project
Internal Pods
3rd Party Pods
CocoaPods
Project
Workspace

cocoapods-generate Public

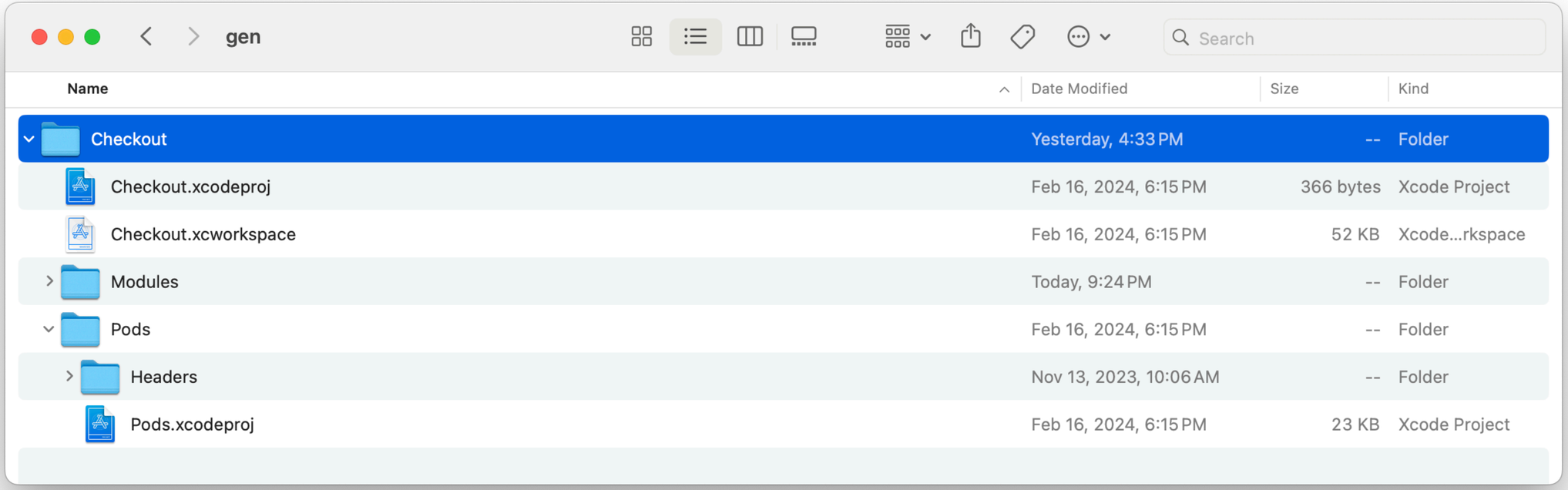
Plugin that allows you to easily generate a workspace from a Podspec.

swift ios cocoapods cocoapods-plugin

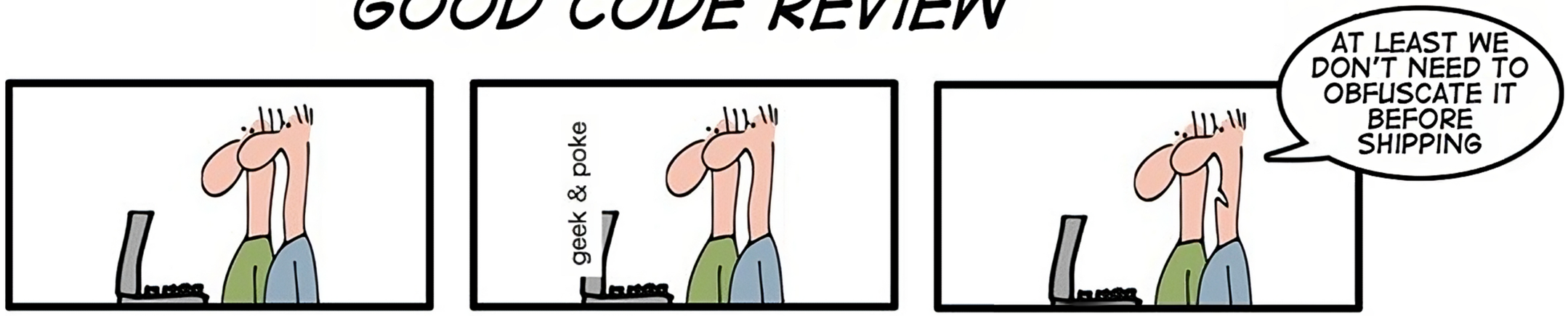
Ruby • MIT License • 43 • 272 • 17 • 2 • Updated on Mar 1

Checkout Module

An example workspace generated from the `cocapods-generate` tool.



HOW TO MAKE A GOOD CODE REVIEW



RULE 1: TRY TO FIND AT LEAST SOMETHING POSITIVE

SwiftLint Public

A tool to enforce Swift style and conventions.

swift linting linter static-analysis code-quality hacktoberfest

Swift • MIT License • 2.2k • 18k • 357 • 65 • Updated 5 hours ago

pre-commit Public

A framework for managing and maintaining multi-language pre-commit hooks.

refactoring git linter pre-commit python

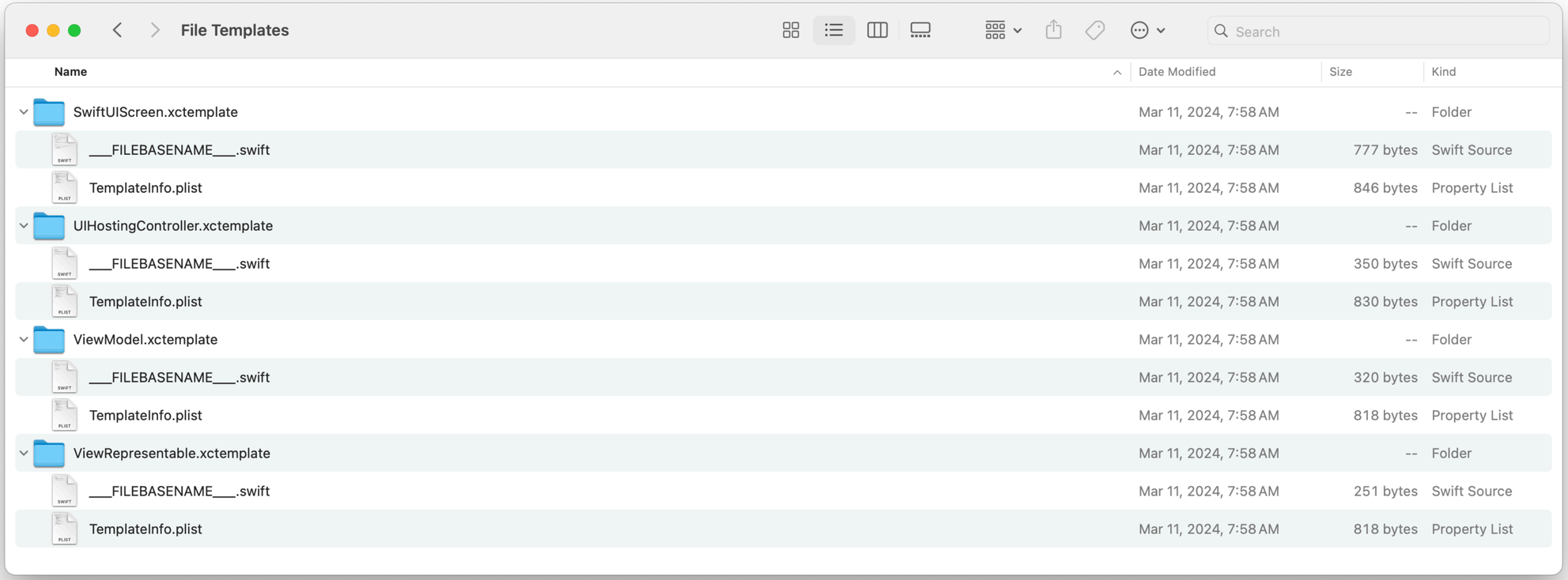
Python • MIT License • 766 • 12k • 18 • 8 • Updated 4 days ago

```yaml
repos:
  - repo: 'https://github.com/pre-commit/pre-commit-hooks'
    rev: v2.3.0
    hooks:
      - id: fix-trailing-whitespace
      - id: check-design-system
      - id: check-swiftui-localization
        name: check-swiftui-localization
        entry: python3 ./Scripts/check-swiftui-localization.py
        language: python
      - id: swift-lint
      - id: swift-format
```

Sample pre-commit hook configuration

File Templates & Code Snippets

File templates and shared code snippets helped avoid nitpicky comments in our code reviews.

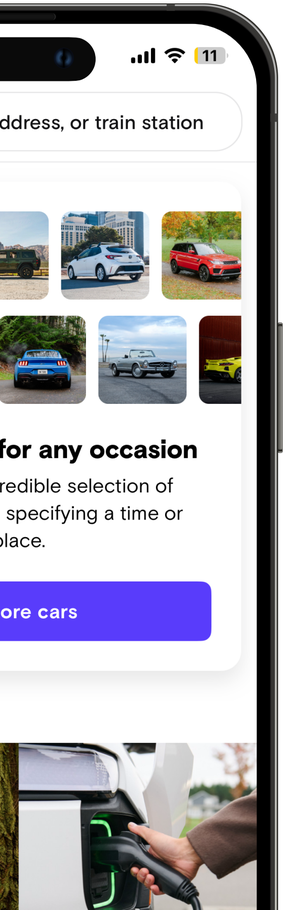


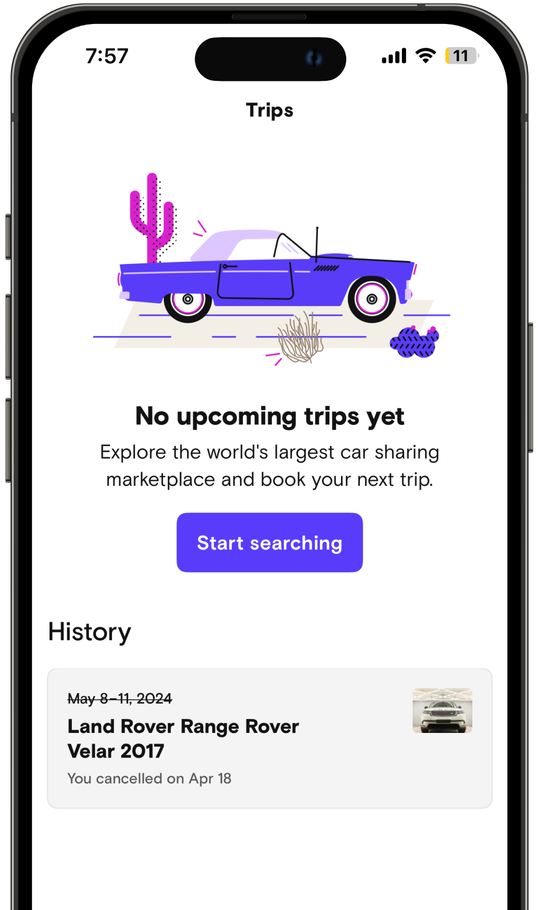
05

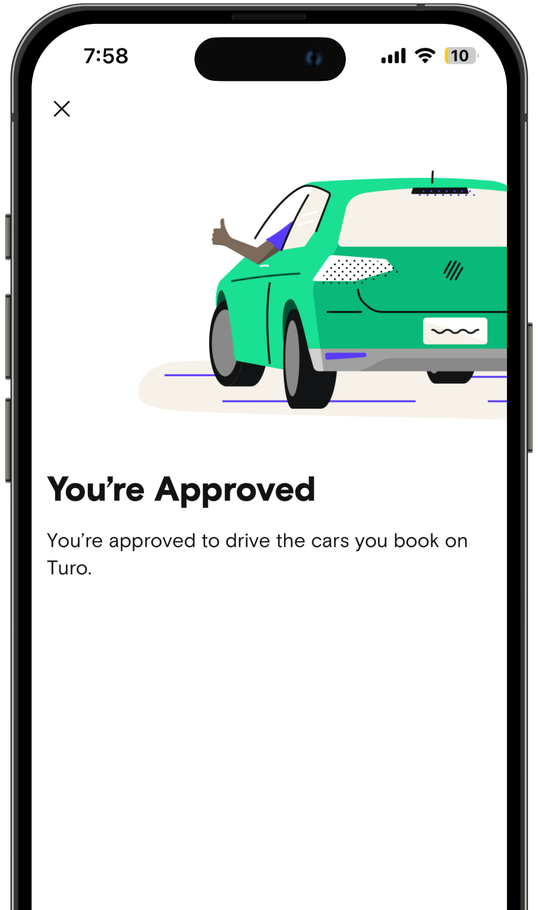
Early September - Mid October 2023

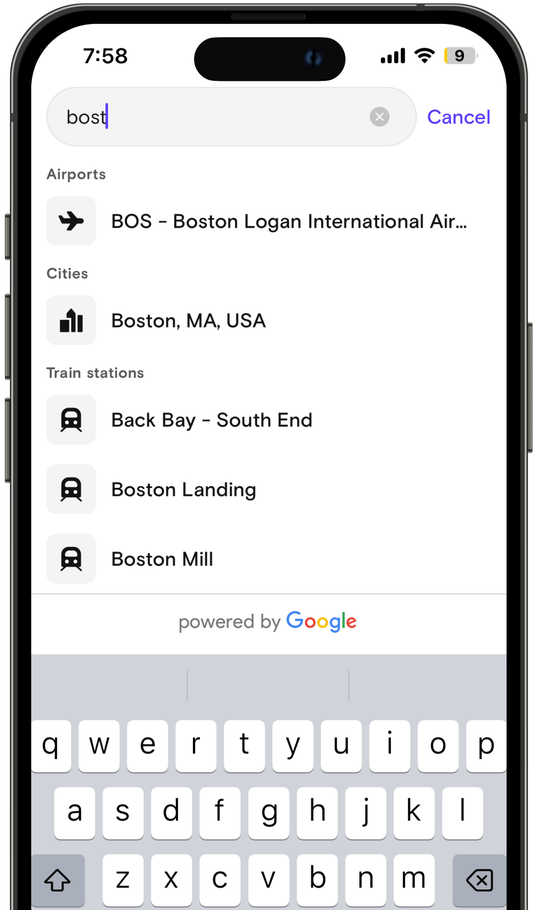
Developer Tooling

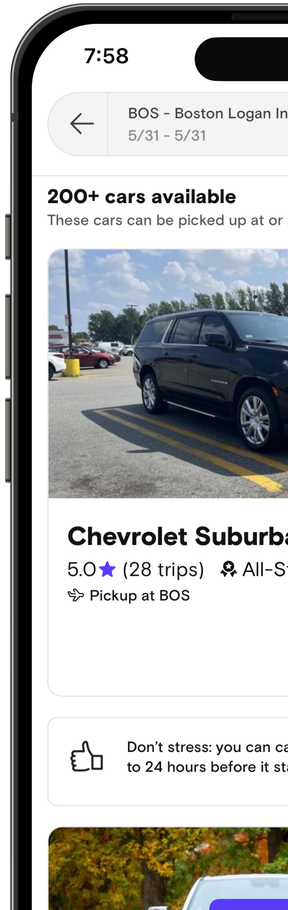
All New Features In SwiftUI 🚀







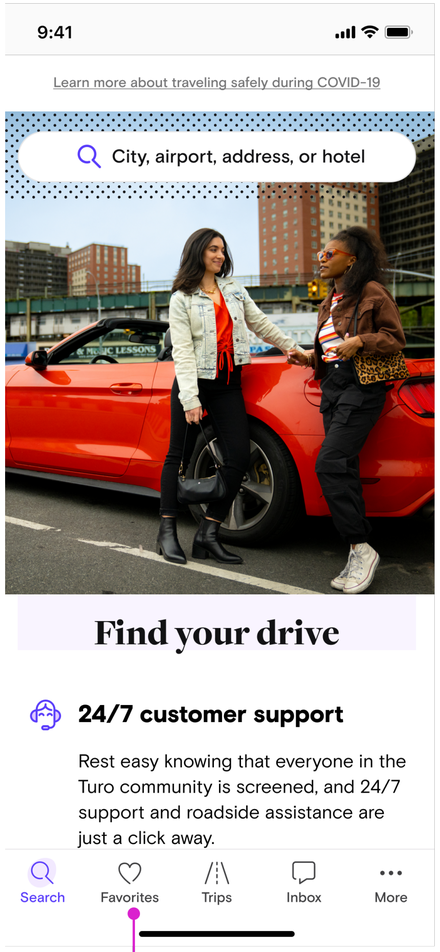




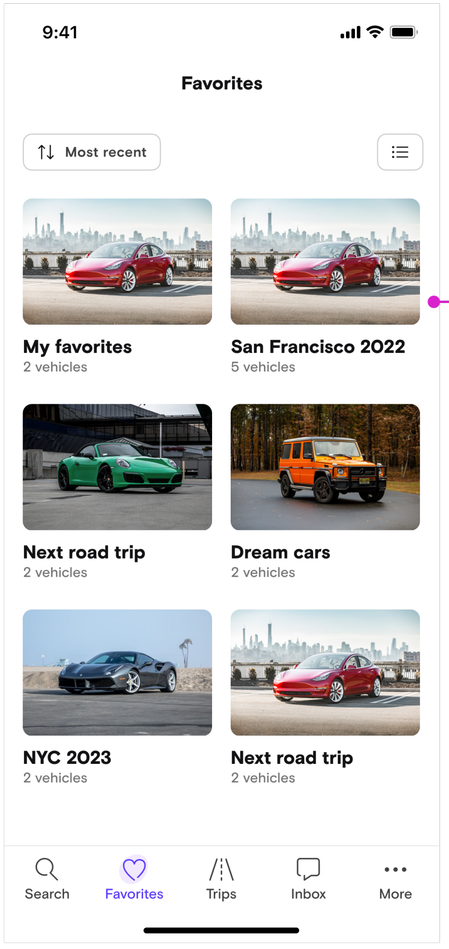
Navigation

05

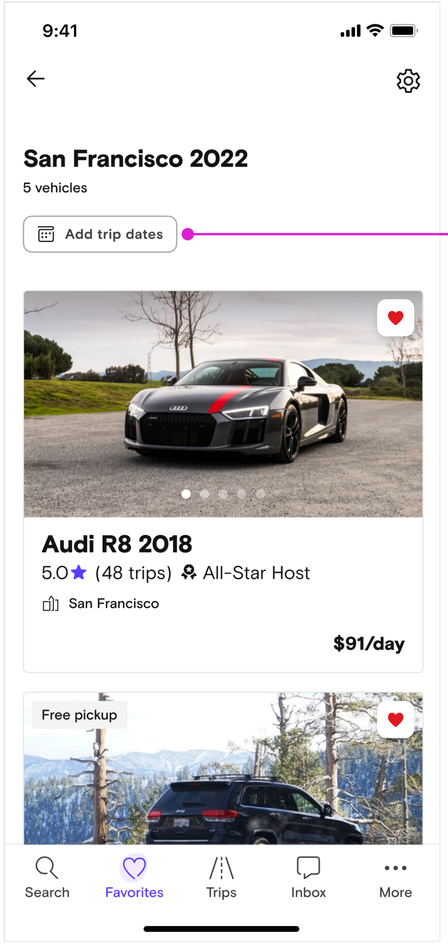
UIKit



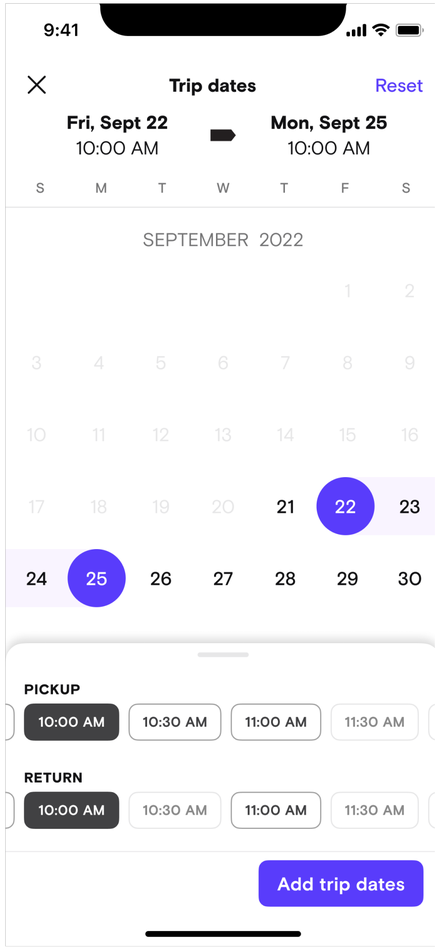
SwiftUI



SwiftUI



UIKit



Navigation Requirements

An approach that would allow for:

01 **Screens Decoupled From Navigation**

02 **MVVM Support**

03 **Flexible Navigation**

04 **Simple & Scalable**

05 **Separate Flow Control**

06 **Deep Linking Ready**

Navigation Requirements

An approach that would allow for:

01 Screens Decoupled From Navigation

02 **MVVM Support**

03 Flexible Navigation

04 Simple & Scalable

05 Separate Flow Control

06 Deep Linking Ready

Navigation Requirements

An approach that would allow for:

01 **Screens Decoupled From Navigation**

02 **MVVM Support**

03 **Flexible Navigation**

04 **Simple & Scalable**

05 **Separate Flow Control**

06 **Deep Linking Ready**

Navigation Requirements

An approach that would allow for:

01 Screens Decoupled From Navigation

02 MVVM Support

03 Flexible Navigation

04 Simple & Scalable

05 Separate Flow Control

06 Deep Linking Ready

Navigation Requirements

An approach that would allow for:

01 Screens Decoupled From Navigation

02 MVVM Support

03 Flexible Navigation

04 Simple & Scalable

05 Separate Flow Control

06 Deep Linking Ready

Navigation Requirements

An approach that would allow for:

01 Screens Decoupled From Navigation

02 MVVM Support

03 Flexible Navigation

04 Simple & Scalable

05 Separate Flow Control

06 Deep Linking Ready

Navigation Requirements

An approach that would allow for:

07 **Nested Navigation**

08 **Multiple Presentation Styles**

09 **Integrate With UIKit**

10 **Supports Dependency Injection**

Navigation Requirements

An approach that would allow for:

07 **Nested Navigation**

08 **Multiple Presentation Styles**

09 **Integrate With UIKit**

10 **Supports Dependency Injection**

Navigation Requirements

An approach that would allow for:

07 Nested Navigation

08 Multiple Presentation Styles

09 Integrate With UIKit

10 Supports Dependency Injection

Navigation Requirements

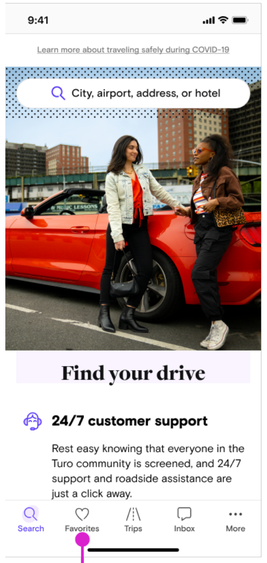
An approach that would allow for:

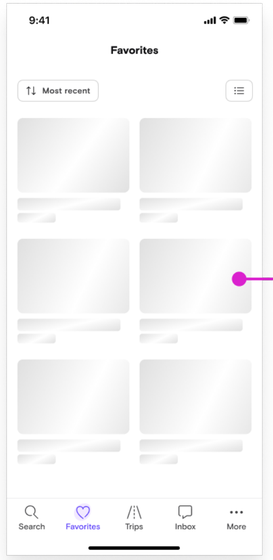
07 Nested Navigation

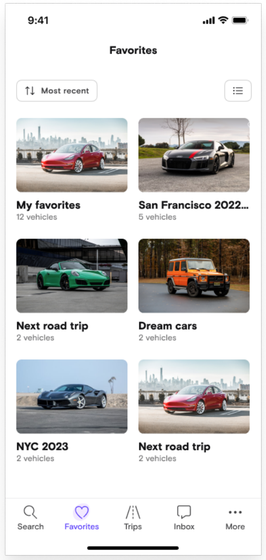
08 Multiple Presentation Styles

09 Integrate With UIKit

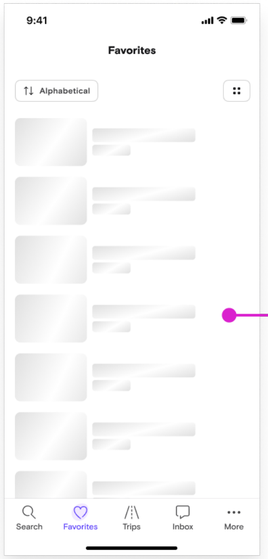
10 Supports Dependency Injection

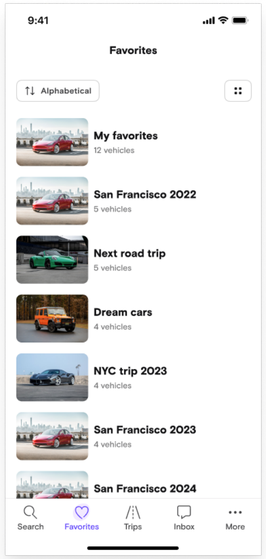






Experiment





Control

How to use SwiftfulRouting in SwiftUI | Swift Packages #5

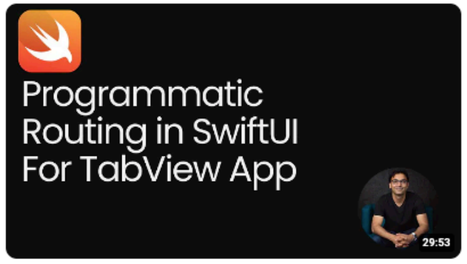
3.4K views • 1 month ago

Swiftful Thinking

Streamline navigation in your **SwiftUI** apps with SwiftfulRouting. This concise tutorial demonstrates how to implement ...

4K

Decoupled Routing from View 16 chapters



Programmatic Routing in SwiftUI for TabView Apps in SwiftUI

2K views • 4 months ago

azamsharp

In this video, Mohammad Azam will demonstrate how to perform programmatic **routing** for TabView apps in **SwiftUI**. Azam will ...

Intro | Create Tabs | Create Label | Create Binding | Create Navigation Stack | Create Navigation Stacks ... 9 chapters

NavigationStack | This is how SwiftUI's navigation should've been since day one!

21K views • 1 year ago

Swift and Tips

In this video, we will explore the difference between the old NavigationView vs NavigationStack announced at #WWDC22, and ...

Programmatic Navigation in SwiftUI

3.1K views • 8 months ago

azamsharp

In this video, Mohammad Azam will demonstrate how to perform programmatic navigation in **SwiftUI**. Azam will explain how to ...

STOP Using Navigation Stack in SwiftUI! Use THIS Instead...

9.2K views • 1 year ago

Rebeloper - Rebel Developer

STOP Using Navigation Stack in **SwiftUI**! Use this instead... Navigation **Router** https://store.rebeloper.com/navigation-**router** In ...

CC

CREATE ROUTES IN SWIFTUI 19 chapters

```
enum OnboardingFlow {
    case basicInfo
    case profilePhoto
    case paymentInfo
    case locationInfo
}
```

```
class OnboardingCoordinator: ObservableObject {

}
```

```
class OnboardingCoordinator: ObservableObject {
    @Published var path = NavigationPath()

}
```

```
class OnboardingCoordinator: ObservableObject {
    @Published var path = NavigationPath()

    func showProfilePhotoPage() {
        path.append(OnboardingFlow.profilePhoto)
    }
}
```

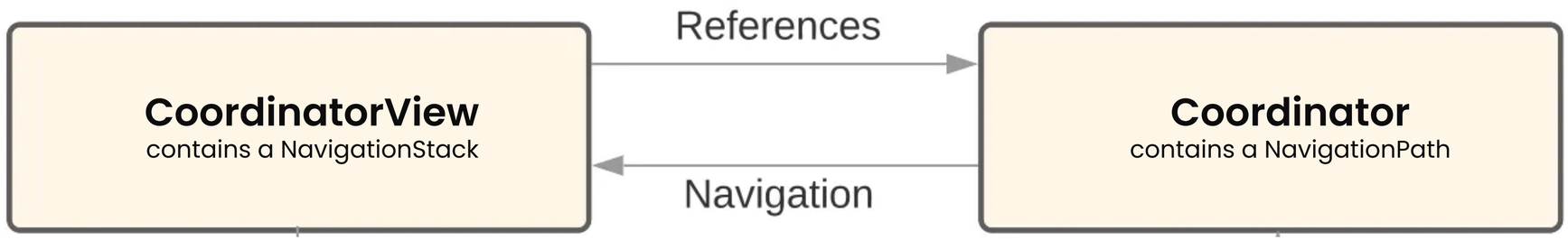
```
class OnboardingCoordinator: ObservableObject {
    @Published var path = NavigationPath()

    func showProfilePhotoPage() {
        path.append(OnboardingFlow.profilePhoto)
    }

    @ViewBuilder
    func create(screen: OnboardingFlow) -> some View {
       switch screen {
        case .basicInfo:
            BasicInfoView()
        case .profilePhoto:
            ProfilePhotoView()
        case .paymentInfo:
            PaymentInfoView()
        case .locationInfo:
            LocationInfoView()
        }
    }
}
```

```
struct OnboardingCoordinatorView: View {
    @StateObject var coordinator = OnboardingCoordinator()

    var body: some View {
        NavigationStack(path: $coordinator.path) {

        }
    }
}
```

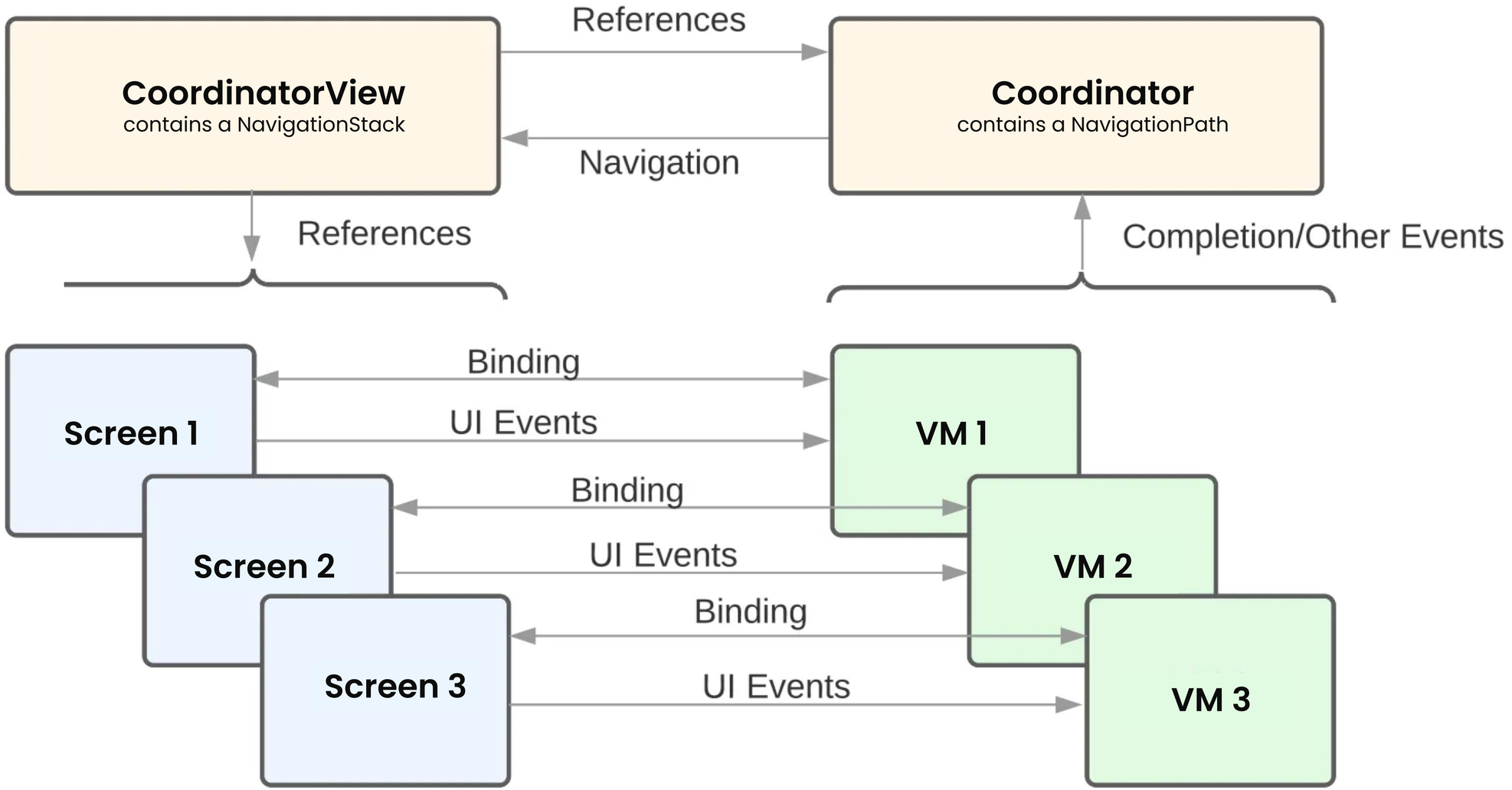
```swift
struct OnboardingCoordinatorView: View {
    @StateObject var coordinator = OnboardingCoordinator()

    var body: some View {
        NavigationStack(path: $coordinator.path) {
            // Create the root view (BasicInfoView)
            coordinator.create(screen: .basicInfo)
        }
        // Passes a reference to the coordinator to all child views
        .environmentObject(coordinator)
    }
}

struct BasicInfoView: View {
    @EnvironmentObject private var coordinator: OnboardingCoordinator

    var body: some View {
        VStack {
            Button(action: {
                // Calls path.append(OnboardingFlow.profilePhoto) in OnboardingCoordinator
                coordinator.showProfilePhotoPage()
            }, label: {
                Text("Add Profile Photo")
            })
        }
    }
}
```

```swift
struct OnboardingCoordinatorView: View {
    @StateObject var coordinator = OnboardingCoordinator()

    var body: some View {
        NavigationStack(path: $coordinator.path) {
            // Create the root view
            coordinator.create(screen: .basicInfo)

            // Handle future view transitions
                .navigationDestination(for: OnboardingFlow.self) { screen in
                    // Build the requested screen
                    coordinator.create(screen: screen)
                }
        }
        // Passes a reference to the coordinator to all child views
        .environmentObject(coordinator)
    }
}

struct BasicInfoView: View {
    @EnvironmentObject private var coordinator: OnboardingCoordinator

    var body: some View {
        VStack {
            Button(action: {
                // Calls path.append(OnboardingFlow.profilePhoto) in OnboardingCoordinator
                coordinator.showProfilePhotoPage()
            }, label: {
                Text("Add Profile Photo")
            })
        }
    }
}
```

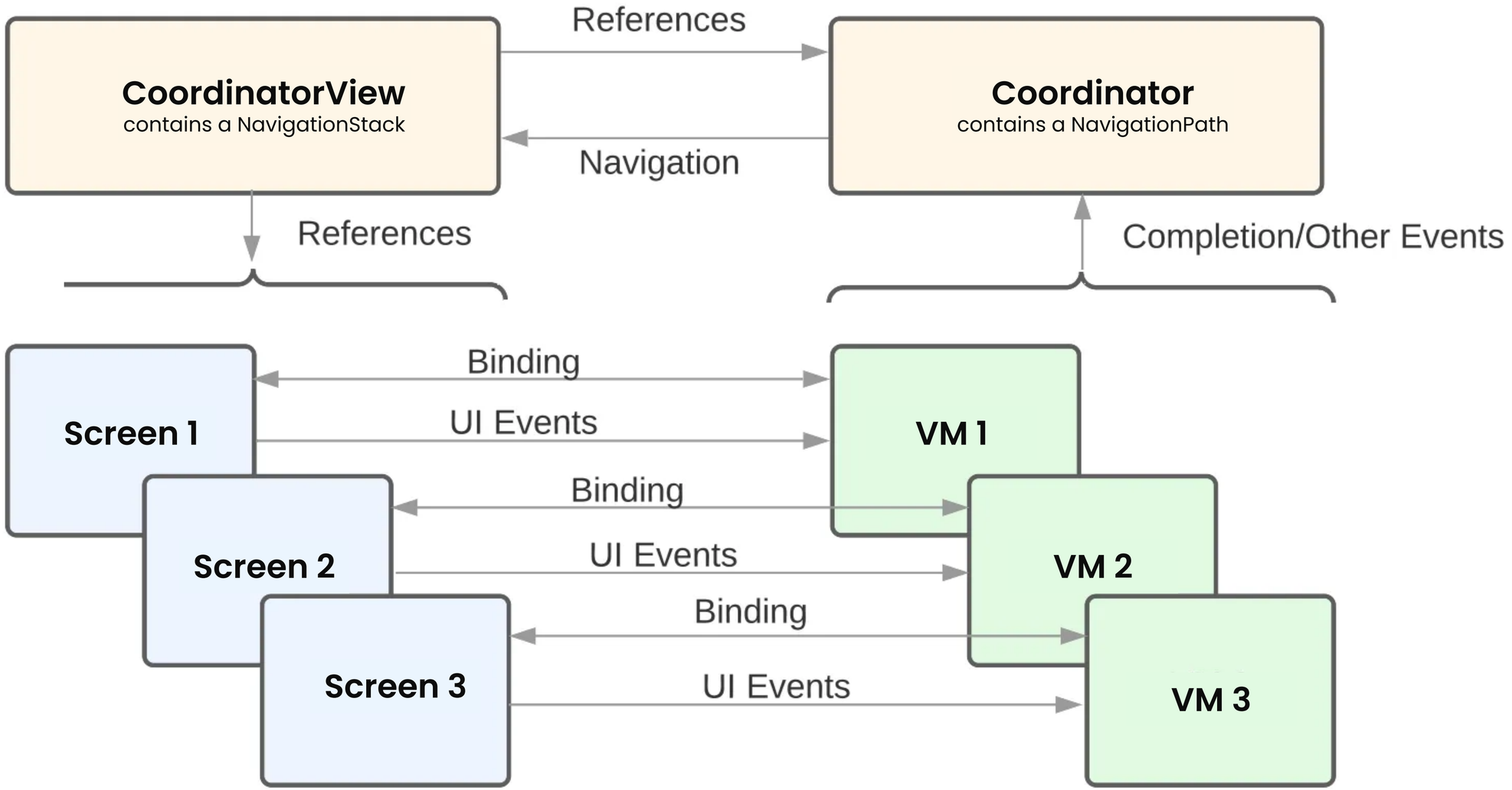
```swift
struct OnboardingCoordinatorView: View {
    @StateObject var coordinator = OnboardingCoordinator()

    var body: some View {
        NavigationStack(path: $coordinator.path) {
            // Create the root view
            coordinator.create(screen: .basicInfo)

            // Handle future view transitions
                .navigationDestination(for: OnboardingFlow.self) { screen in
                    // Build the requested screen
                    coordinator.create(screen: screen)
                }
        }
        // Passes a reference to the coordinator to all child views
        .environmentObject(coordinator)
    }
}

struct BasicInfoView: View {
    @EnvironmentObject private var coordinator: OnboardingCoordinator

    var body: some View {
        VStack {
            Button(action: {
                // Calls path.append(OnboardingFlow.profilePhoto) in OnboardingCoordinator
                coordinator.showProfilePhotoPage()
            }, label: {
                Text("Add Profile Photo")
            })
        }
    }
}
```



Flow Navigation



Flow Navigation



Flow Navigation

05

Present Day

Navigation

For more details about our navigation approach, check out:

Flow Navigation (https://betterprogramming.pub/flow-navigation-with-swiftui-4-e006882c5efa)

04

What Worked Well

What Worked Well

01 Code Reviews

02 Getting Quick Wins

03 Balancing Deadlines

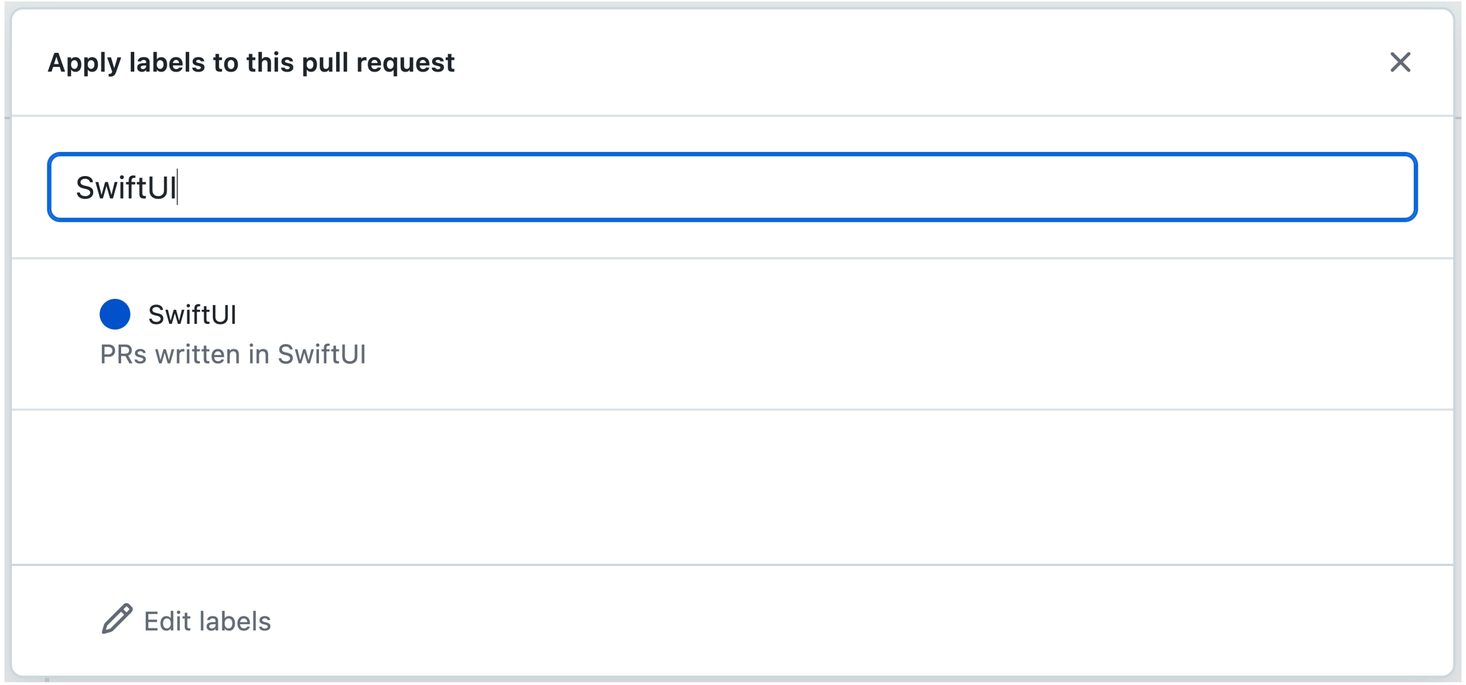
04 Improving Test Coverage

05 Accessibility From The Ground Up

06 Flexible Onboarding

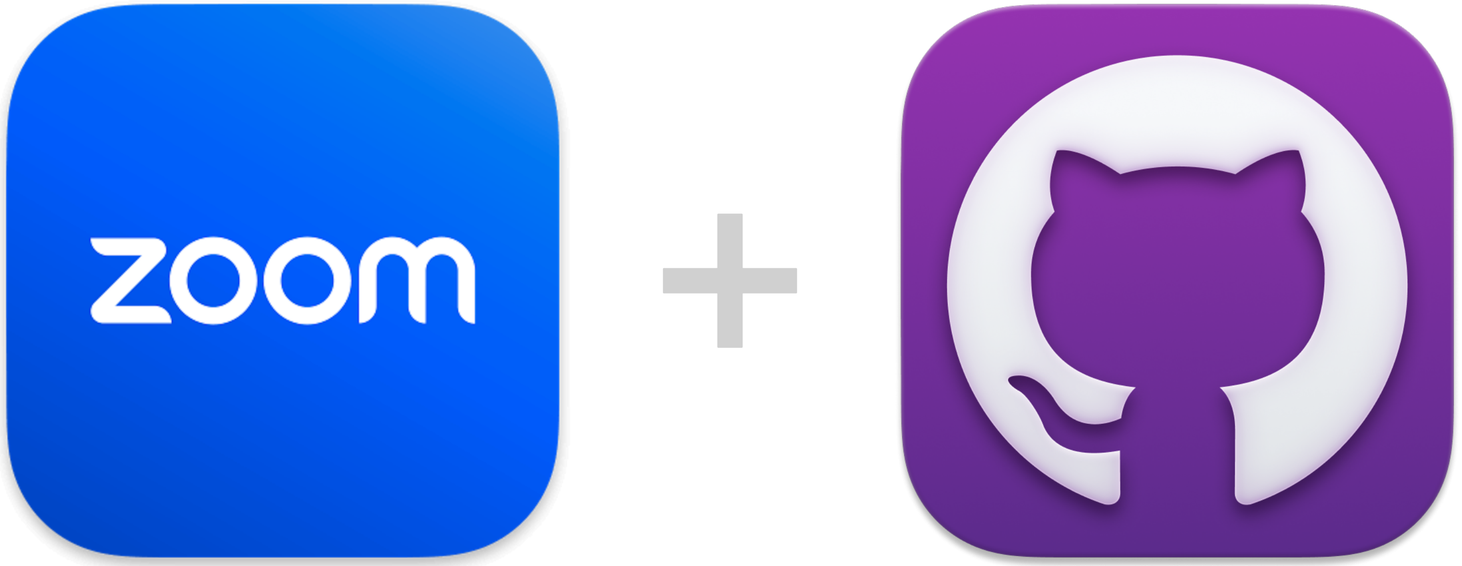
GitHub Labels

Tagging pull requests with a “SwiftUI” label on GitHub made it easier for the Pod to track and guide these changes.



Code Review Sessions & Zoom

Weekly meetings where we come together to review the same pull requests as a group.



What Worked Well

01 Code Reviews

02 Getting Quick Wins

03 Balancing Deadlines

04 Improving Test Coverage

05 Accessibility From The Ground Up

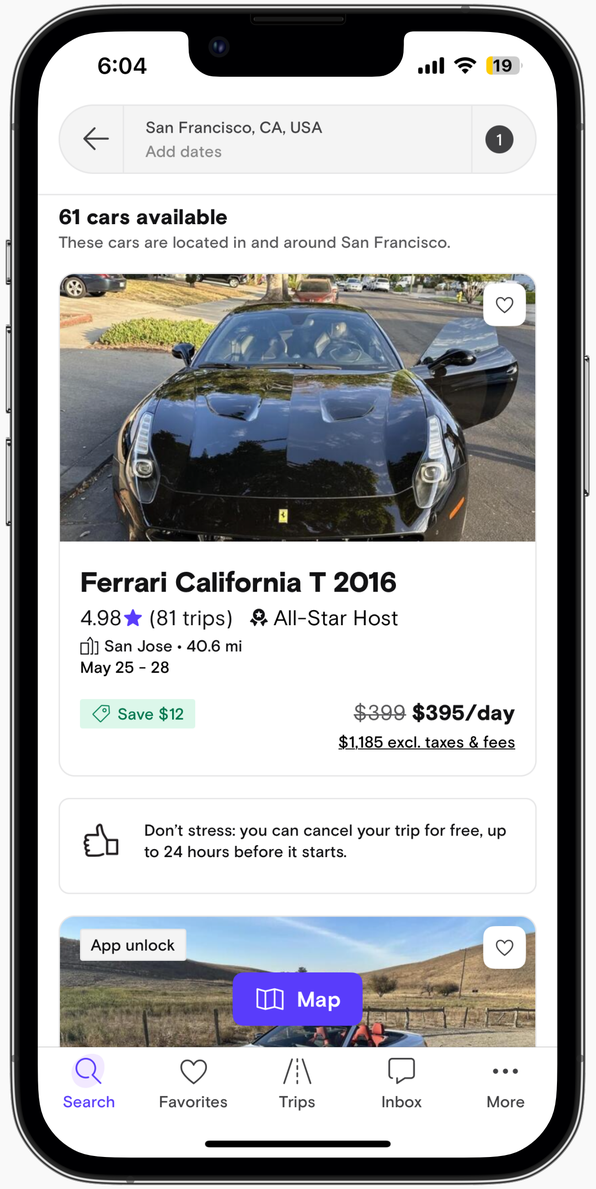
06 Flexible Onboarding

Design System

Custom UI Components

Custom UI Components

Custom Assets



Custom UI Components

Custom Buttons

Custom Typography

Custom Icons

Getting Quick Wins

Design Systems

Having a set of UI components and helpers ready in advance helped the team focus on learning declarative programming and data flow in SwiftUI, without worrying about styling, animations, or creating custom components.

Resources

Transitioning to a new technology can be overwhelming. To support our team, we set up detailed documentation, office hours, mentorship, code review sessions, and recordings. This helped developers overcome roadblocks and get their first win quickly.

What Worked Well

01 Code Reviews

02 Getting Quick Wins

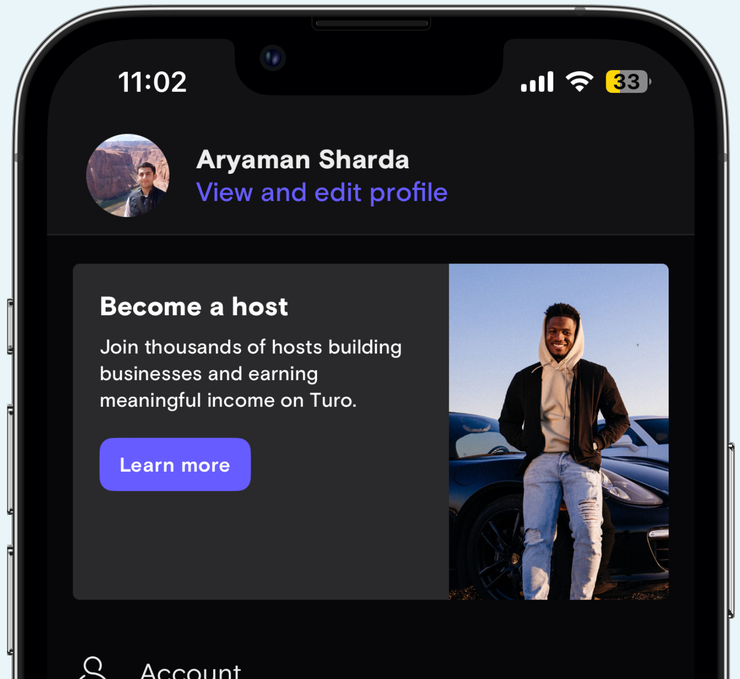
03 Balancing Deadlines

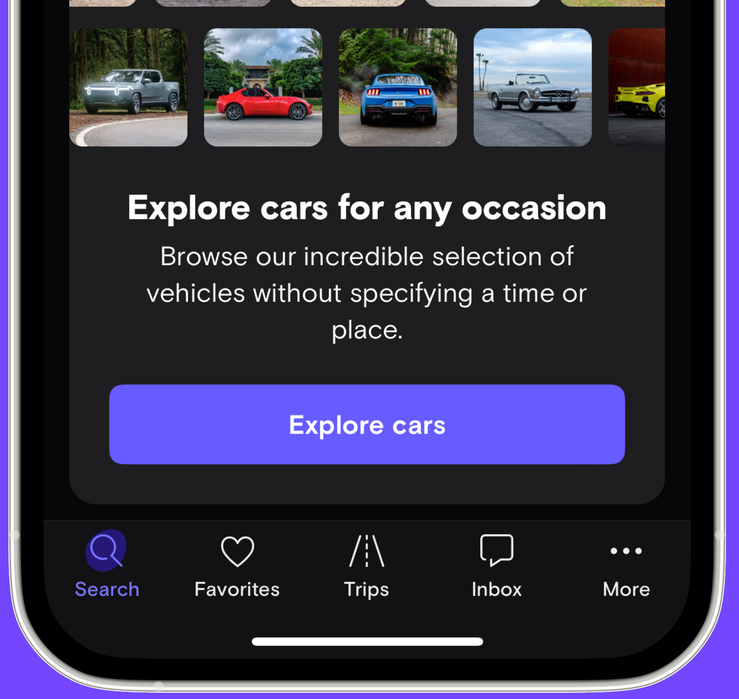
04 Improving Test Coverage

05 Accessibility From The Ground Up

06 Flexible Onboarding

Aligning Teams & Managing Deadlines





Transitioning to SwiftUI was a team effort involving designers, product teams, and managers to find the right time to make the switch.

What Worked Well

01 Code Reviews

02 Getting Quick Wins

03 Balancing Deadlines

04 Improving Test Coverage

05 Accessibility From The Ground Up

06 Flexible Onboarding

What Worked Well

01 Code Reviews

02 Getting Quick Wins

03 Balancing Deadlines

04 Improving Test Coverage

05 **Accessibility From The Ground Up**

06 Flexible Onboarding

What Worked Well

01 Code Reviews

02 Getting Quick Wins

03 Balancing Deadlines

04 Improving Test Coverage

05 Accessibility From The Ground Up

06 Flexible Onboarding

05

What Didn’t Work

01

Managing SwiftUI Skill Levels

Challenge

Managing the team’s different SwiftUI experience was difficult - some were pros, while others were just starting.

Those new to SwiftUI helped us identify when solutions were getting too complicated (reflecting how the larger team might react), but slowed down the Pod's start as we had to ensure everyone had the basics down

01

Managing SwiftUI Skill Levels

What We’d Do Differently

Looking back, starting with experienced SwiftUI developers would have let us dive into key conversations immediately.

We could then use onboarding to see if the approach was intuitive and easy to use, and adjust it if needed.

02

Context Switching

Challenge

Developers had multiple responsibilities like code reviews, interviews, on-call shifts, and mentoring, making it tough to focus on SwiftUI.

This constant context switching slowed down the Pod's progress.

Without a dedicated build tooling team, Pod members were stretched thin, balancing SwiftUI work with their day-to-day tasks.

02

Context Switching

What We’d Do Differently

We could have structured Pods to give developers longer focus periods.

Negotiating one day per sprint for SwiftUI or rotating developers out of product work for a week at a time would have been helpful.

This would have reduced context switching and allowed for more efficient progress on SwiftUI initiatives.

03

Discoverability

Challenge

As everyone started creating SwiftUI modifiers and custom views, keeping track of what was already in the codebase became difficult. People often recreated the same modifiers simply because they didn’t know they already existed.

This is bound to happen in large codebases, but with SwiftUI's frequent need for new modifiers and views, we felt this pain more often.

03

Discoverability

What We'd Do Differently

We started mentioning new general-purpose modifiers and components in our weekly iOS meetings.

We created a new module for these reusable components and updated its documentation to include a list of everything it contained, the use cases, and references to where they're being used. We also organized them by use case (e.g. ScrollView) rather than by type (e.g. ViewModifier).

These strategies helped, but it's still a work in progress.

04

SwiftUI & iOS Versions

Challenge

Supporting iOS 15 meant we sometimes hit limits with SwiftUI APIs like presentationDetents(), Grid, or MultiDatePicker. Though rare, these issues required us to work with designers and product teams to find alternative UI designs.

Ensuring consistent design across platforms meant changes in iOS could impact Android and Web teams, creating ripple effects beyond iOS.

04

SwiftUI & iOS Versions

What We’d Do Differently

If you're supporting older iOS versions, set aside time to identify styling limitations and communicate them to your designers for more compatible designs.

Converting UIKit to SwiftUI helped us uncover these issues early. When we hit a SwiftUI limitation, we could fall back on the existing UIKit implementation. Though workarounds were usually easy, it's important to keep this on your radar.

06

Wrapping Up

Takeaways

01 **Learning & Skill Building**

02 Conversions

03 Long Onboarding

04 Good Communication

05 Focus Time

06 Investing In Developer Tooling

Takeaways

01 Learning & Skill Building

02 **Conversions**

03 Long Onboarding

04 Good Communication

05 Focus Time

06 Investing In Developer Tooling

Takeaways

01 Learning & Skill Building

02 Conversions

03 Long Onboarding

04 Good Communication

05 Focus Time

06 Investing In Developer Tooling

Takeaways

01 Learning & Skill Building

02 Conversions

03 Long Onboarding

04 Good Communication

05 Focus Time

06 Investing In Developer Tooling

Takeaways

01 Learning & Skill Building

02 Conversions

03 Long Onboarding

04 Good Communication

05 Focus Time

06 Investing In Developer Tooling

Takeaways

01 Learning & Skill Building

02 Conversions

03 Long Onboarding

04 Good Communication

05 Focus Time

06 **Investing In Developer Tooling**

@aryamansharda

Thanks SwiftCraft!

Blog

digitalbunker.dev

Newsletter

indie.watch

Books

tinyurl.com/as-books