Crossing Boundaries

iOS Engineer's Journey in the Realm of **React Native**

Francesca Piccoli
Frontend Engineer at Musixmatch

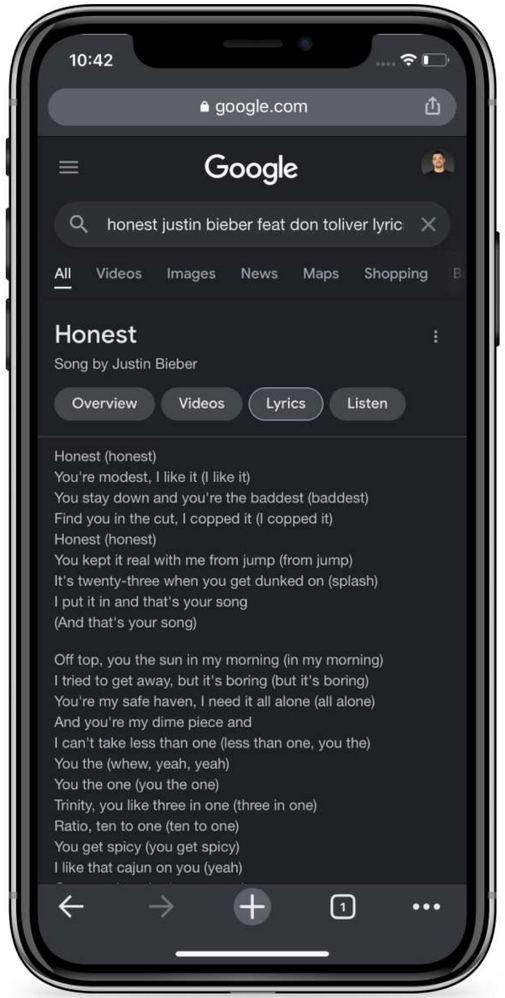
musixmatch

We are enhancing the global music experience

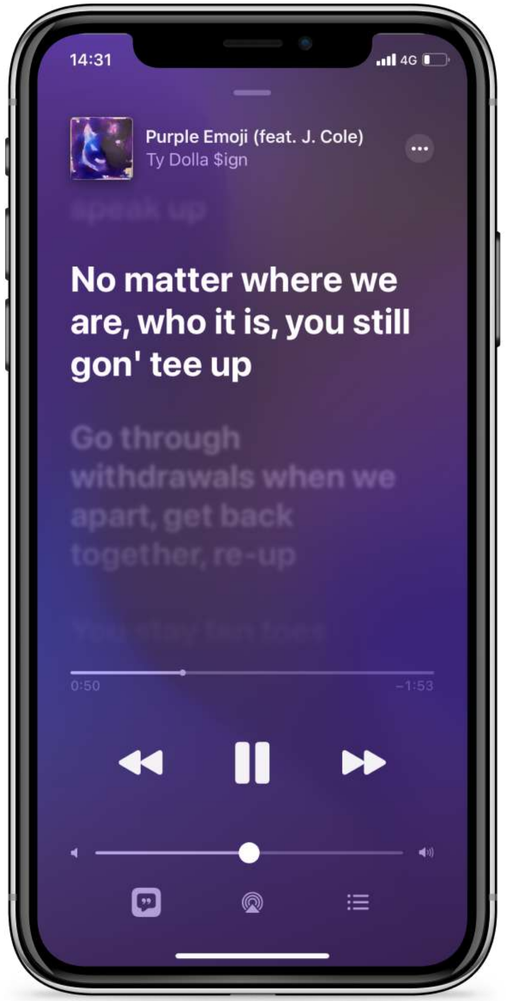
The world's largest living lyrics catalog

For over 12 years Musixmatch has passionately built a world-class tech platform and industry reputation

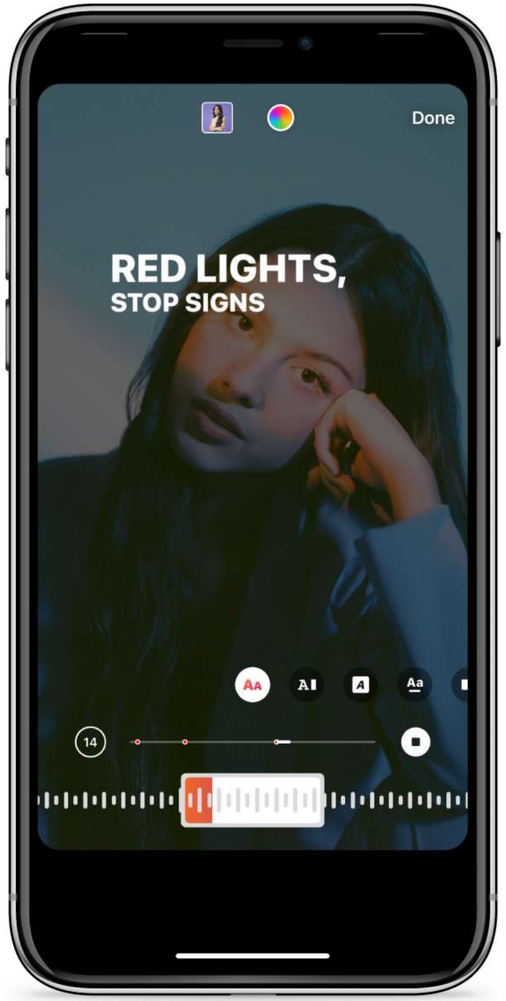
Google



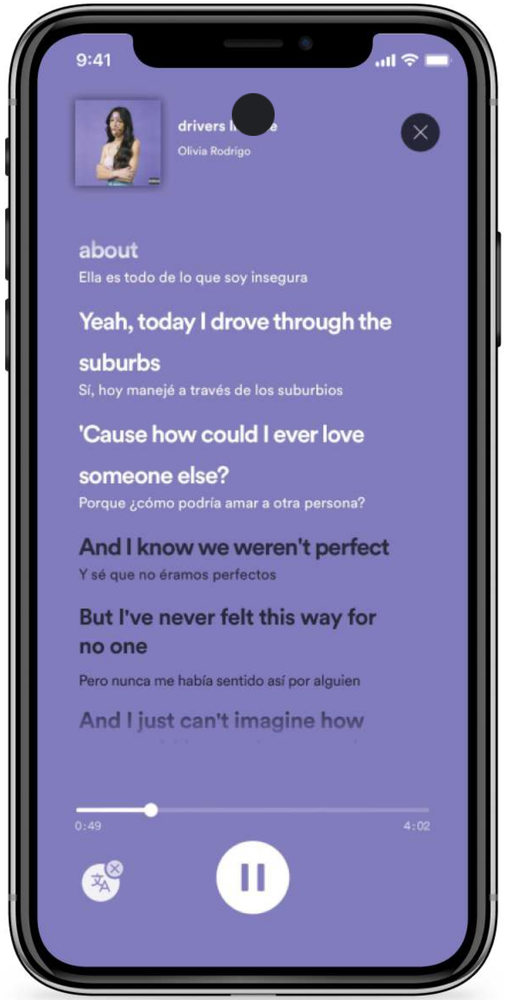
Apple Music



Instagram



Spotify



2022

I started in the mobile team working on the Musixmatch native iOS application

2023

I started learning React Native to create cross platform products

iOS Native vs React Native

DIFFERENCES & SIMILARITIES

- Fundamentals
- Performance
- Debug & test
- Community
- Maintenance
- Showcases


Imagine a world
Me
My React Native Dev colleague
LIGHTYEAR
Where you write code once

React Native

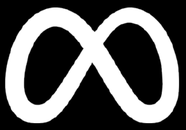
🛠️👨‍💻

Open source framework

Javascript

🍎 + 🤖

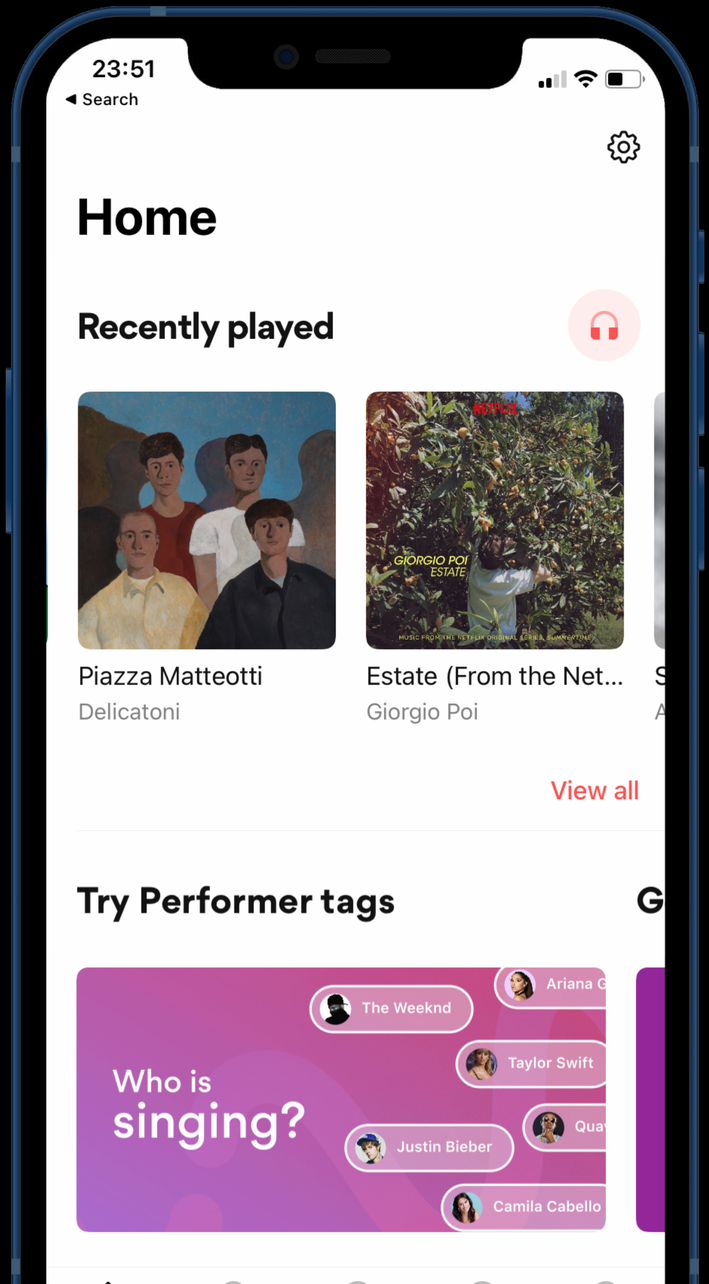
Cross-platform

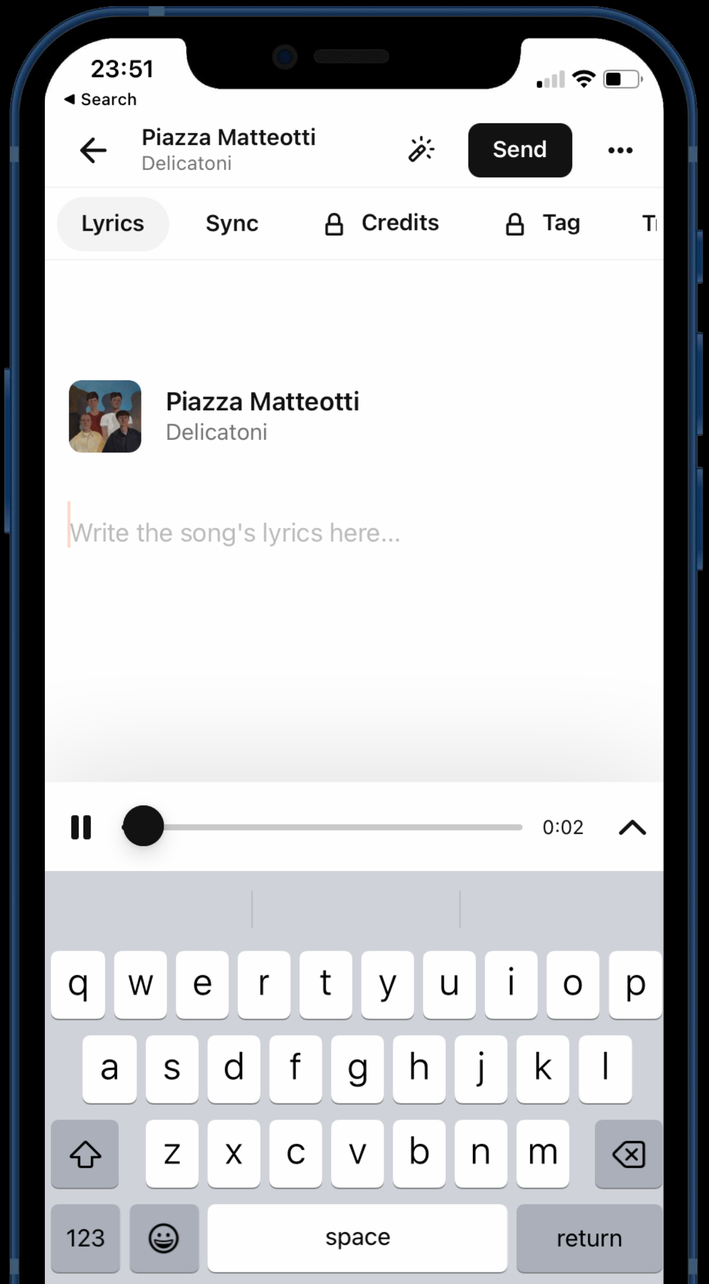


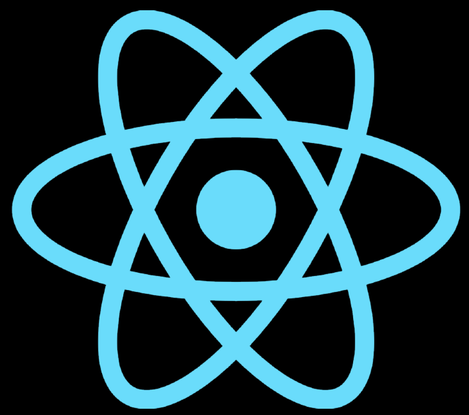
Provided by Meta

React Native

Directly invokes native UI components



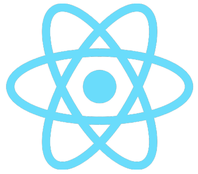




Comparison


01 WINS
99
SCORPION
SUB-ZERO
ROUND 1

Both  SwiftUI and

 React Native use

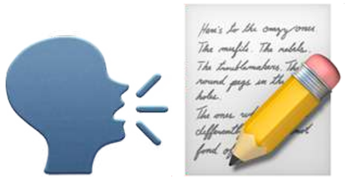
declarative and

reactive programming

Declarative

Interface development

The developer describes the components

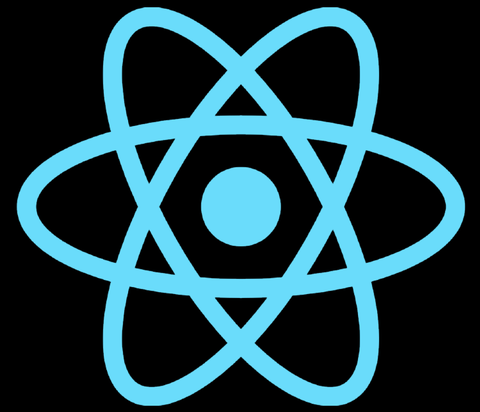


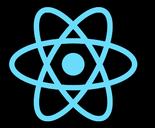
Reactive

Handle data flows and propagation of changes

Work asynchronously in a more responsive manner

Examples





```swift
import SwiftUI

struct LoginPage: View {

    @State private var username: String = ""
    @State private var password: String = ""

    var body: some View {
            VStack {
                Image(systemName: "lock")
                    .resizable()
                    .aspectRatio(contentMode: .fit)
                    .frame(width: 50, height: 50)
                    .padding(.bottom)

                TextField("Username", text: $username)
                    .padding()
                    .background(Color.gray.opacity(0.2))
                    .cornerRadius(8)
                    .padding(.horizontal)

                SecureField("Password", text: $password)
                    .padding()
                    .background(Color.gray.opacity(0.2))
                    .cornerRadius(8)
                    .padding(.horizontal)

                Button {
                    if username.isEmpty || password.isEmpty {
                        // Handle empty fields
                    } else {
                        // Perform login validation
                    }
                } label: {
                    Text("Login")
                        .padding()
                        .foregroundColor(.white)
```

```jsx
import React, { useState } from 'react';
import {
  View,
  Text,
  TextInput,
  StyleSheet,
  Pressable,
} from 'react-native';
import Icon from '@expo/vector-icons/Ionic

const LoginPage = () => {

  const [username, setUsername] = useState
  const [password, setPassword] = useState

  const handleLogin = () => {
    if (username.trim() === '' || password
      // Handle empty fields
    } else {
      // Perform login validation
    }
  };

  return (
    <View style={styles.container}>
      <Icon name="lock-closed-outline" sty
      <TextInput
        style={styles.input}
        placeholder="Username"
        onChangeText={(text) => setUsernam
      />
      <TextInput
        style={styles.input}
        placeholder="Password"
        secureTextEntry={true}
        onChangeText={(text) => setPasswor
      />
      <Pressable style={styles.button} onPress={handleLogin}>
        <Text style={styles.buttonText}>Login</Text>
      </Pressable>
    </View>
  );
```

```jsx
const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
    // Other styles below...
    padding: 20,
  },
  logo: {
    marginBottom: 20,
  },
  input: {
    width: '100%',
    padding: 15,
    marginBottom: 15,
    backgroundColor: 'rgba(128, 128, 128, 0.2)',
    borderRadius: 8,
  },
  button: {
    width: '100%',
    padding: 15,
    backgroundColor: 'rgb(0, 122, 255)',
    borderRadius: 8,
  },
  buttonText: {
    color: 'white',
    textAlign: 'center',
  },
});
```

Handling state

@State

```
@State private var username: String = ""
@State private var password: String = ""
```

useState

```
const [username, setUsername] = useState('');
const [password, setPassword] = useState('');
```

Core components

VStack
Image

View
Icon

```swift
var body: some View {
        VStack {
            Image(systemName: "lock")
                .resizable()
                .aspectRatio(contentMode: .fit)
                .frame(width: 50, height: 50)
                .padding(.bottom)
```

```jsx
return (
    <View style={styles.container}>
      <Icon name="lock-closed-outline" style={styles.logo} size={50}/>
  ...
```

Text inputs

TextField
SecureField

```
TextField("Username", text: $username)
    .padding()
    .background(Color.gray.opacity(0.2))
    .cornerRadius(8)
    .padding(.horizontal)

SecureField("Password", text: $password)
    .padding()
    .background(Color.gray.opacity(0.2))
    .cornerRadius(8)
    .padding(.horizontal)
```

TextInput
TextInput + secureTextEntry

```
<TextInput
    style={styles.input}
    placeholder="Username"
    onChangeText={(text) => setUsername(text)}
/>
<TextInput
    style={styles.input}
    placeholder="Password"
    secureTextEntry={true}
    onChangeText={(text) => setPassword(text)}
/>
```

Buttons

Button
Action
Text

```swift
Button {
    if username.isEmpty || password.isEmpty {
        // Handle empty fields
    } else {
        // Perform login validation
    }
} label: {
    Text("Login")
        .padding()
        .foregroundColor(.white)
        .frame(maxWidth: .infinity)
        .background(Color.blue)
        .cornerRadius(8)
}
.padding()
```

Pressable
Function
Text

```jsx
<Pressable style={styles.button} onPress={handleLogin}>
    <Text style={styles.buttonText}>Login</Text>
</Pressable>
```

```jsx
const handleLogin = () => {
    if (username.trim() === '' || password.trim() === '') {
      // Handle empty fields
    } else {
      // Perform login validation
    }
};
```

Styling

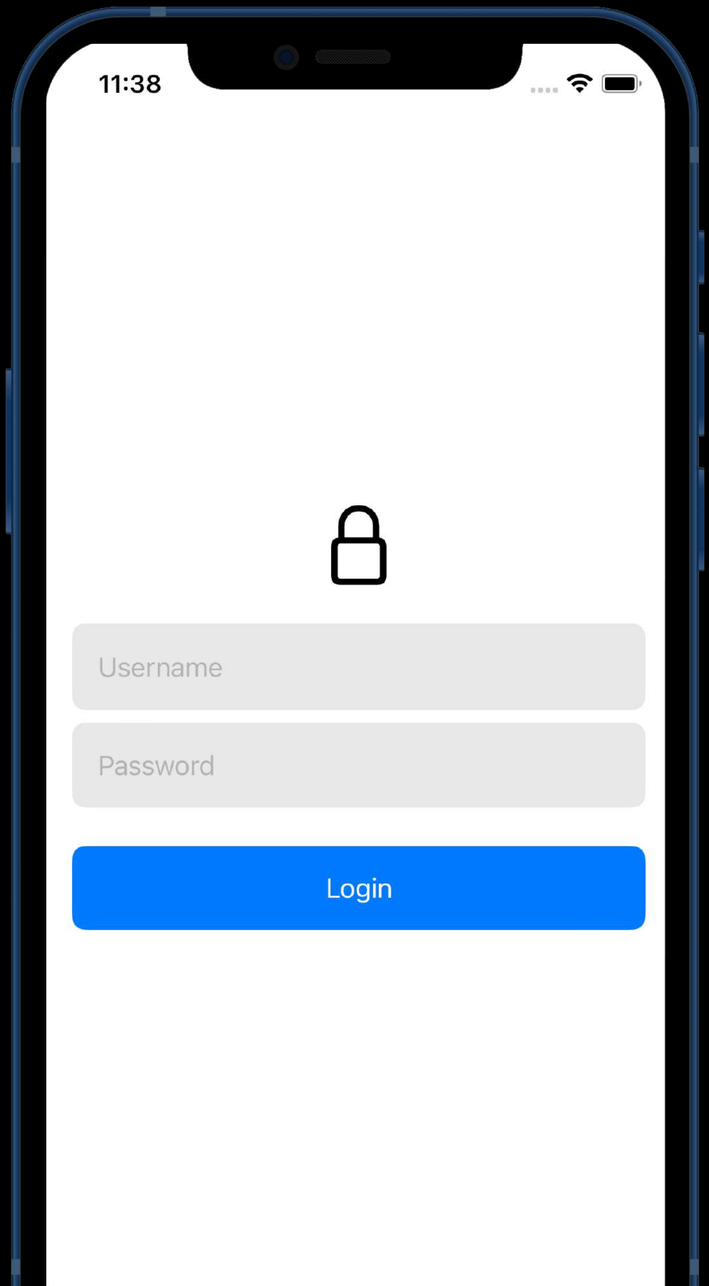
Modifiers

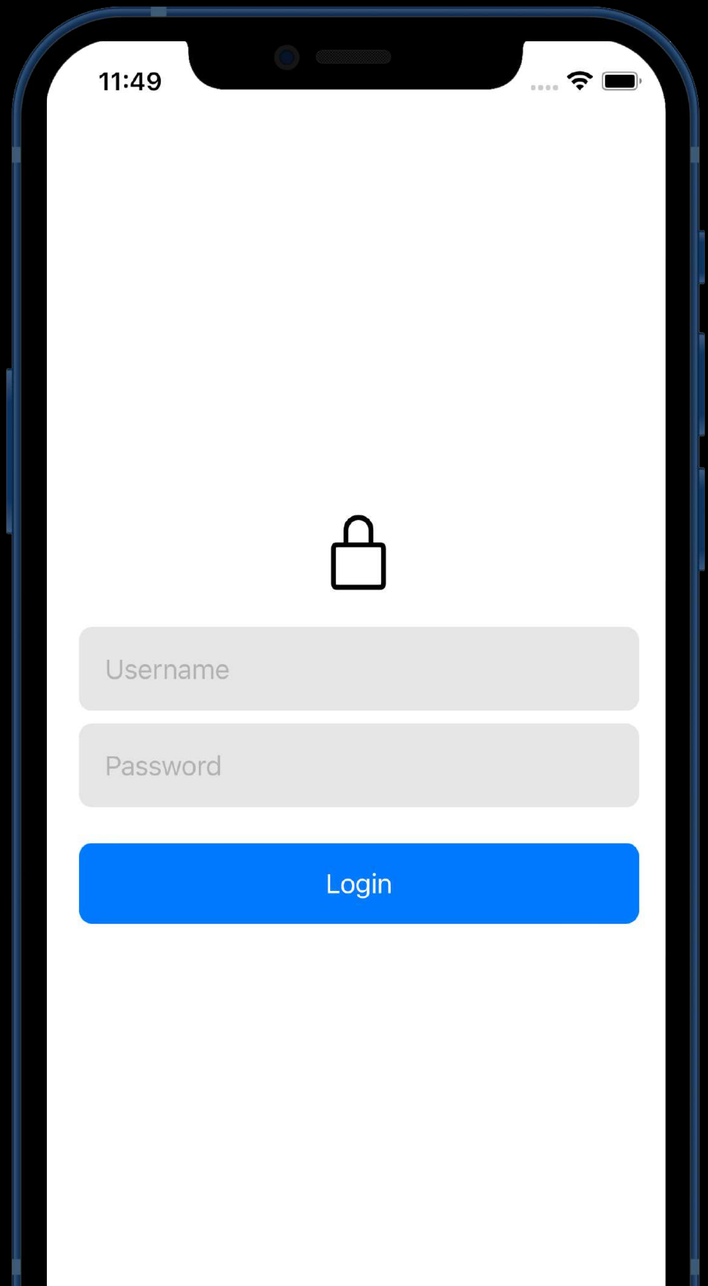
```
.resizable()
.aspectRatio(contentMode: .fit)
.frame(width: 50, height: 50)
.padding(.bottom)
```

StyleSheet

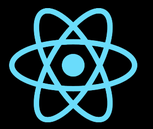
```
const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
    padding: 20,
  },
```

Login page Results





```swift
import SwiftUI

struct CardView: View {
    var imageName: String
    var title: String
    var description: String

    var body: some View {
        ZStack {
            Color.white

            VStack(alignment: .leading) {
                Image(imageName)
                    .resizable()
                    .aspectRatio(contentMode: .fill)
                    .frame(height: 150)
                    .clipped()

                Text(title)
                    .font(.title)
                    .fontWeight(.bold)
                    .padding()

                Text(description)
                    .font(.body)
                    .padding(.horizontal)

                Spacer()
            }
            .clipped()
        }
        .frame(width: 300, height: 400)
        .cornerRadius(30)
        .shadow(color: Color.gray.opacity(0.4), radius: 5, x: 0, y: 2)
    }
}
```



```jsx
import React from 'react';
import { View, Text, Image, StyleSheet, ImageSourcePropType } from 'react-
native';

const CardView = ({ image, title, description }: Props) => {
  return (
    <View style={styles.container}>
    <View style={styles.card}>
      <Image
        source={image}
        style={styles.image}
        resizeMode="cover"
      />
      <Text style={styles.title}>{title}</Text>
      <Text style={styles.description}>{description}</Text>
    </View>
    </View>
  );
};

export default CardView;

type Props = {
    image: ImageSourcePropType;
    title: string;
    description: string;
}

const styles = StyleSheet.create({
  card: {
    width: 300,
    height: 400,
    ...
```

Properties

View's initialisers

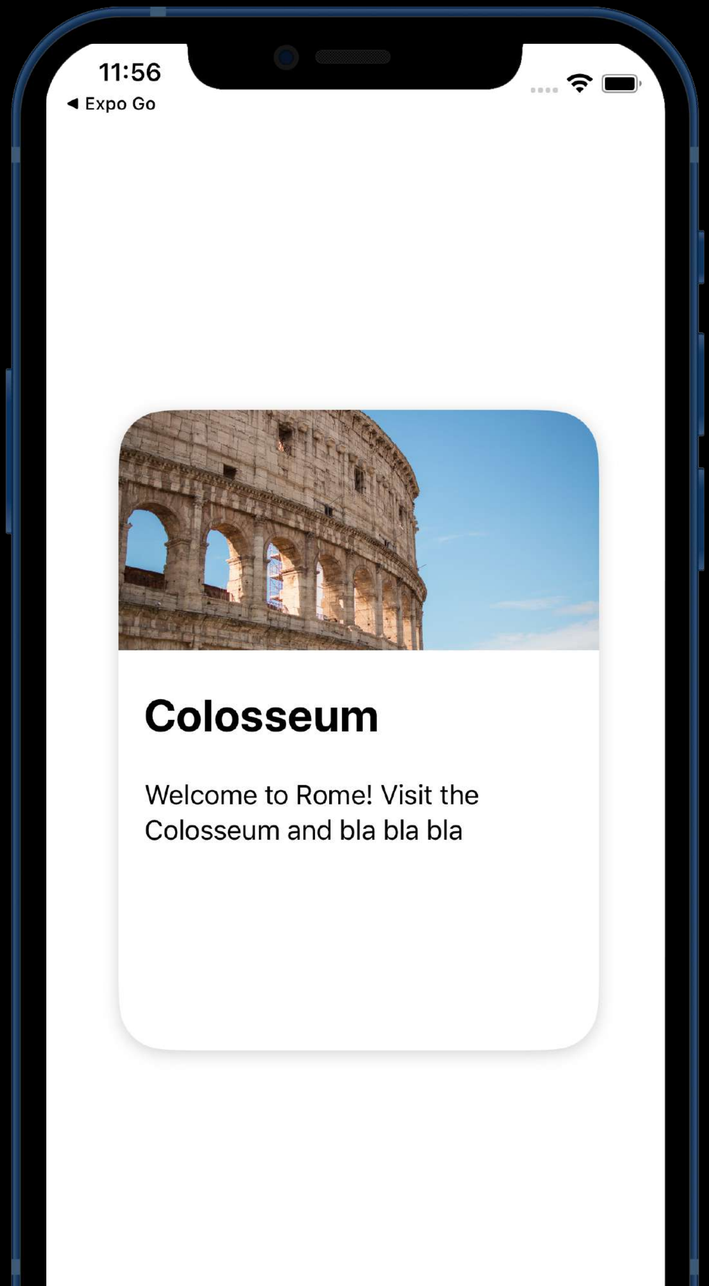
```
var imageName: String
var title: String
var description: String
```

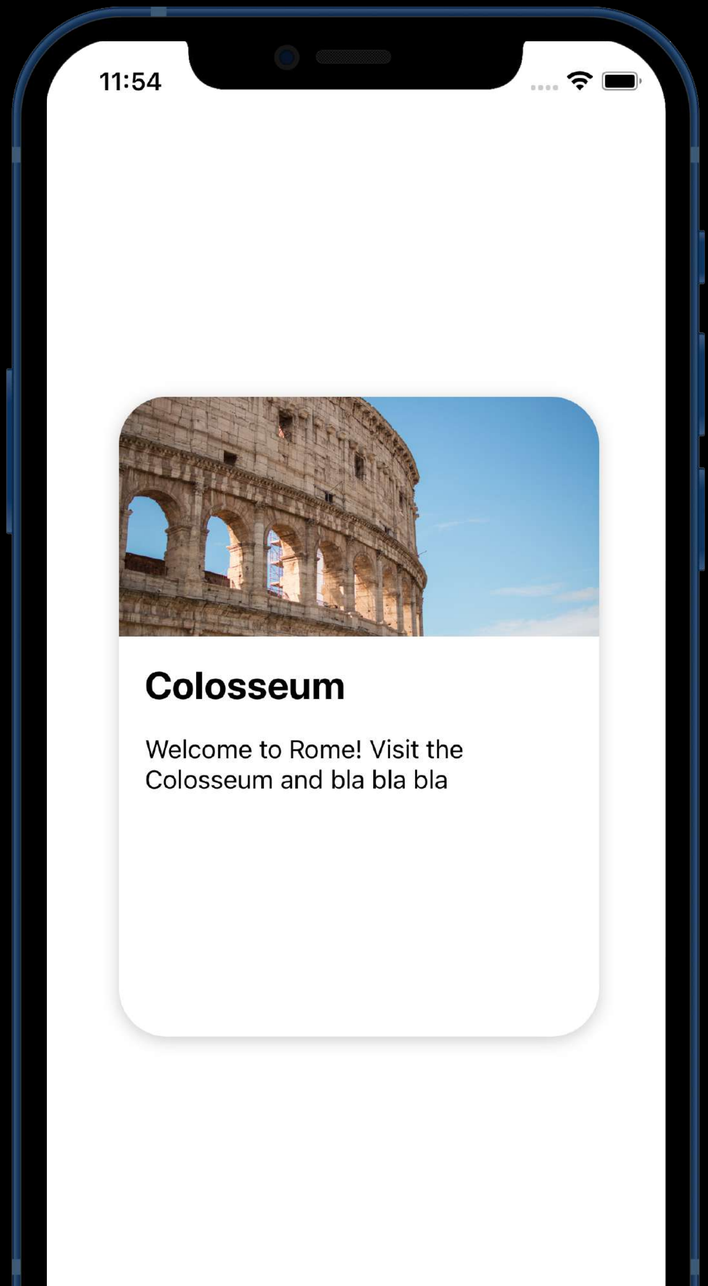

Props

```
const CardView = ({ image, title, description }: Props) => {
    ...
```

```
type Props = {
    image: ImageSourcePropType;
    title: string;
    description: string;
}
```

CardView Results





View lifecycle & side effects

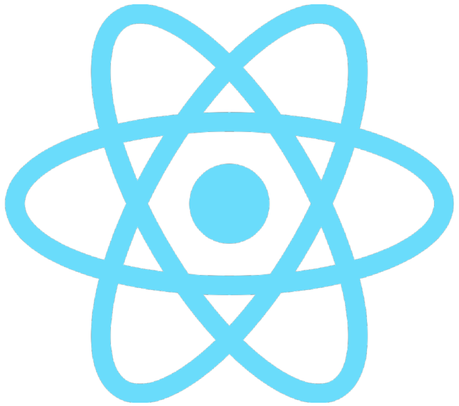
SwiftUI uses **.onAppear** and **.onDisappear** modifiers

```swift
import SwiftUI

struct ContentView: View {
    var body: some View {
        VStack{
            // View
        }.onAppear {
            // Code to run when the view appears
            print("View appeared")
        }
        .onDisappear {
            // Code to run when the view disappears
            print("View disappeared")
        }
    }
}
```

React Native uses **hooks** like useEffect

```jsx
import React, { useEffect } from 'react';
import { View, Text } from 'react-native';

const ContentView = () => {
  useEffect(() => {
    // Code to run when the component mounts (view appears)
    console.log('View appeared');

    return () => {
      // Code to run when the component unmounts (view disappears)
      console.log('View disappeared');
    };
  }, []);

  return (
    <View>
      {/* Add your content here */}
    </View>
  );
};

export default ContentView;
```

Performance & Efficiency




COME ON IN...
WORLD'S
BEST
BOSS
IT'S PERFORMANCE REVIEW TIME!

Native development creates products with a better performance

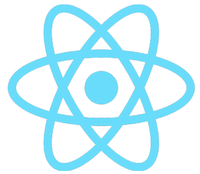
Why  native performs better?

No **JavaScript** layer that runs in between as React Native

Access to latest features

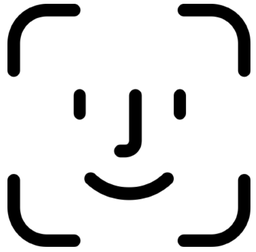
Optimised debugging

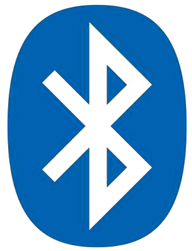
Smaller app size

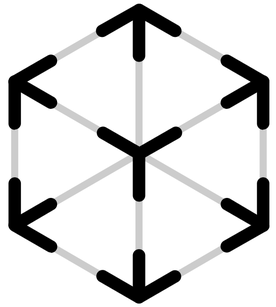


React Native doesn't have access to some native APIs

Such as...







Pay

- Core Bluetooth
- AR features
- Audio Processing
- Custom Push Notifications
- HealthKit
- Face ID
- Apple Pay
- SiriKit
- Native calendar

AND MORE...

So what?

Use third-party or community-contributed libraries


Write custom native modules on your own


Native dev
React dev
PM

Development speed & code sharing

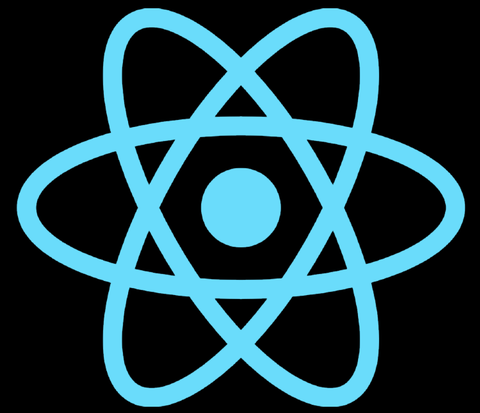
Rapid prototyping (Preview)

Limited code sharing

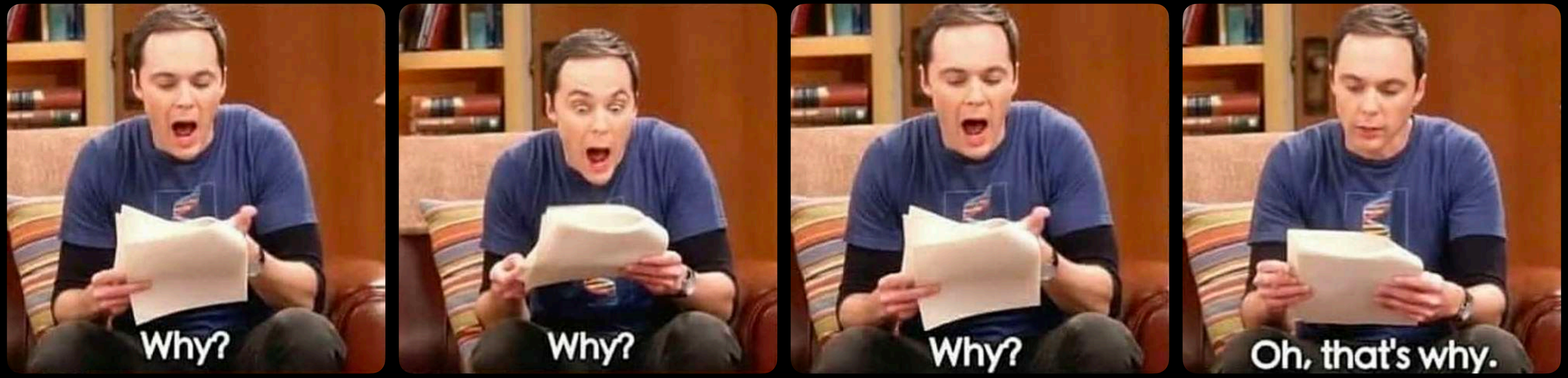
Rapid Development

High code reusability

Testing & Debugging



Developers everyday



Debugging

Simulator
LLDB debugger and breakpoints

Simulator
Debug view Hierarchy
Debug navigator

Xcode
Live previews

Simulator
React Developer Tools

```
console.log("Help me!")
```

Testing

UNIT TEST

XCTest
Native framework

UI TEST

XCTest
Native framework

SNAPSHOT TEST

External Libraries

UNIT TEST

Jest
Javascript framework

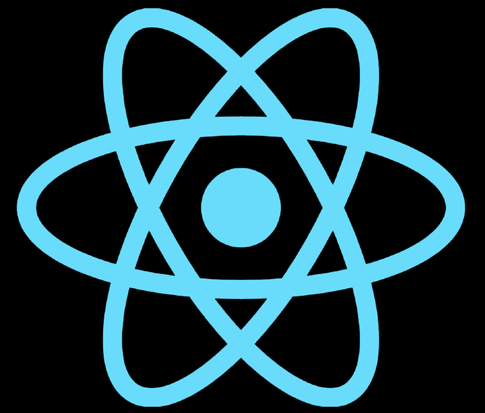
COMPONENT TEST

External Libraries
i.e, react-test-renderer

SNAPSHOT TEST

External Libraries
i.e, react-test-renderer

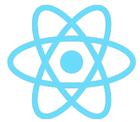
Community & Ecosystem



Community & Ecosystem

Small but passionate and specialised community

Product related language



Large, various, decentralised community

Less “authoritative”

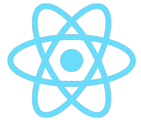
Lots of devs come from web

Maintenance & Upgrades

Retro compatible

Bigger core and ecosystem

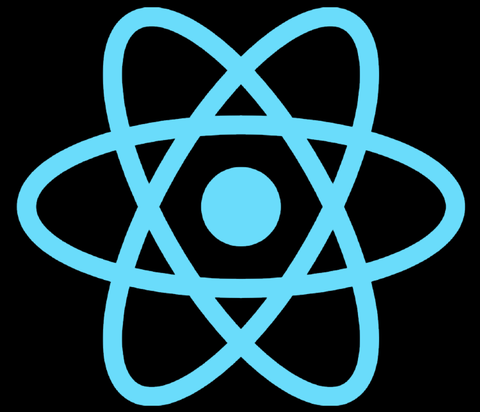
Possible issues with external libraries

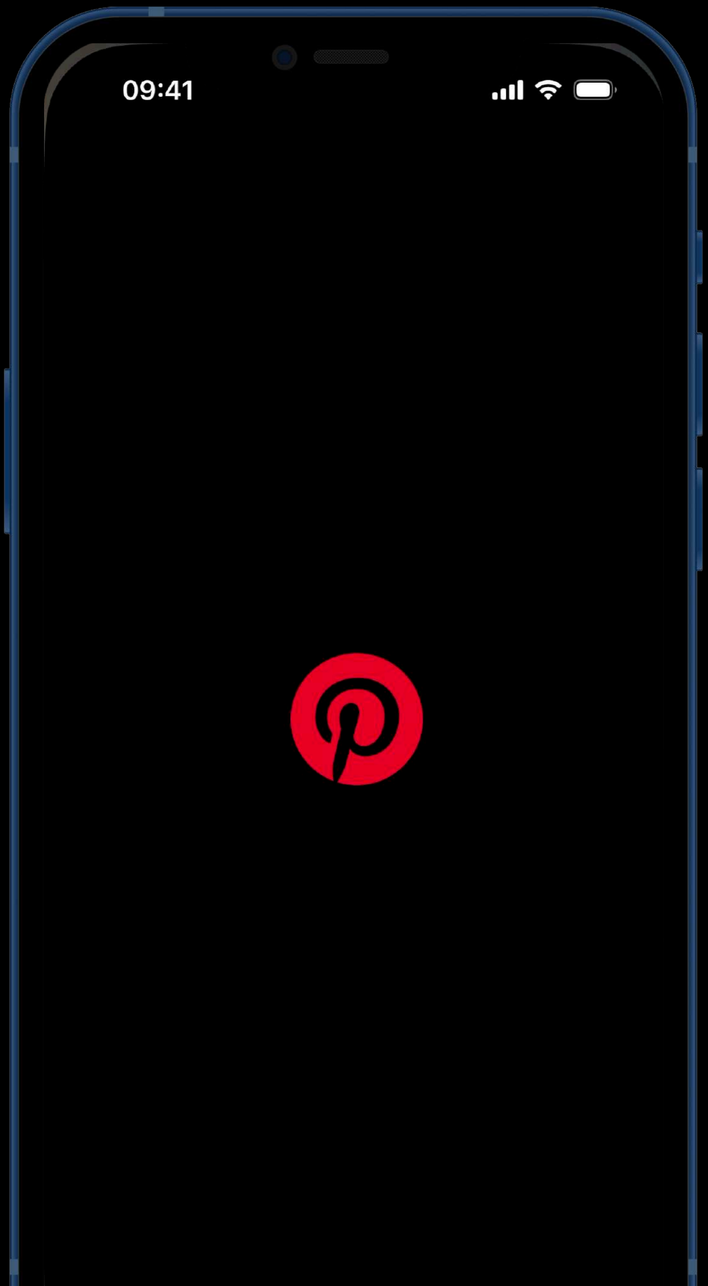


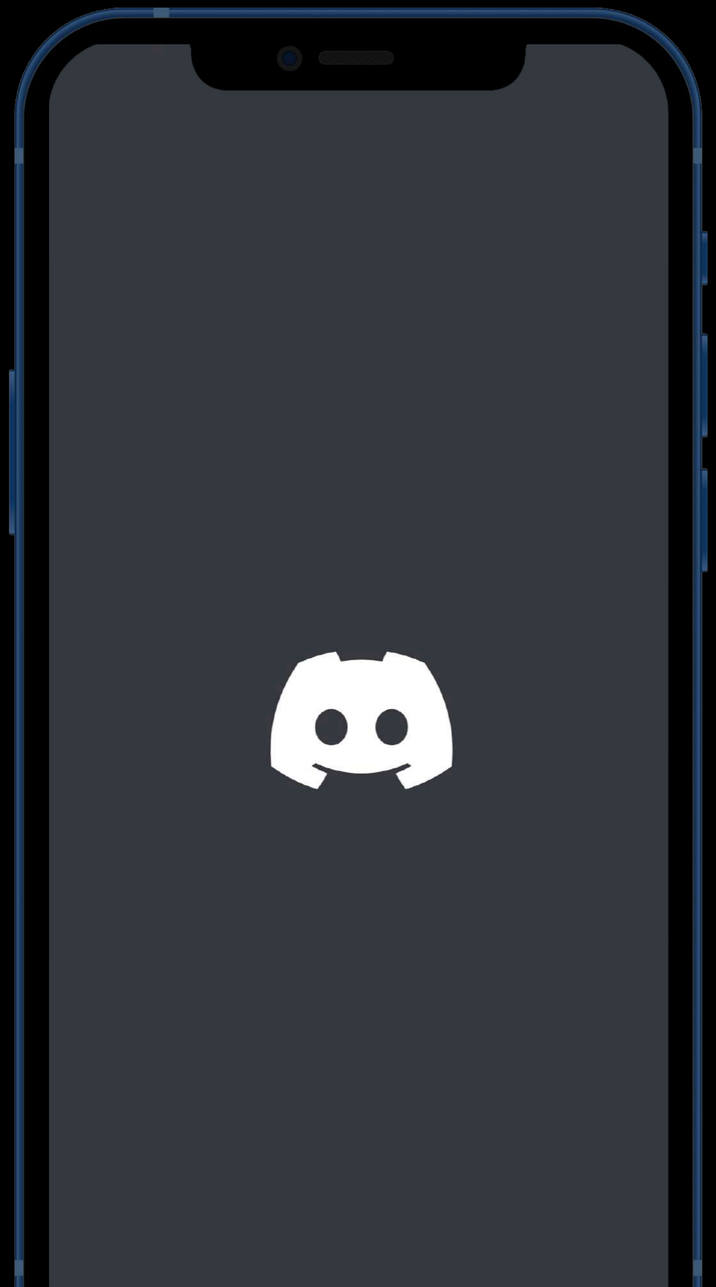
React Native is based a lot on dependencies

The core is small

Showcases




09:41



Create

Browse

Categories | Live Channels

Overwatch 2
30,4K viewers
FPS | Shooter | Action

Dead by Daylight
7,7K viewers
Strategy | Indie Game | Action | Horror

World of Warcraft
32,2K viewers
RPG | Adventure Game | MMO | Action

Apex Legends
20,1K viewers
FPS | Shooter | Action

Just Chatting
173,4K viewers
IRL

League of Legends
78,(
RP ... Action

Sort and Filter

09:41

Request to book

Tiny home

Unique and Secluded AirShip with Breathtaking Highland Views

★ 4.94 (362) · Superhost

Your trip

Dates Edit

27 Jun – 2 Jul 2024

Guests Edit

1 guest

Travel insurance

Add travel insurance for € 86.68

Get reimbursed if you need to cancel due to illness, flight delays and more.

Add

What's covered

Choose how to pay

09:41

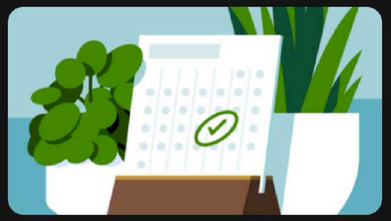
Where to? Now

Suggestions

See all

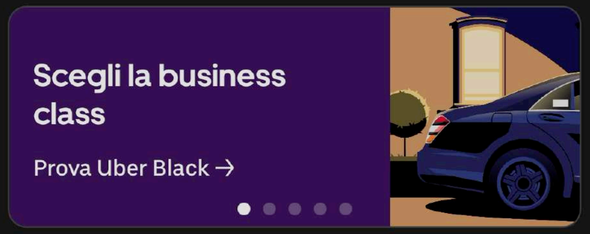
Ride | 2 Wheels | Reserve | Travel

Ways to plan with Uber



Reserve and relax →
Book up to 90 days in advance

High-end SUVs ·
Get a luxury ride fo



Ways to save with Uber


09:41
coinbase

Learning I’ve gotten to

Share my knowledge being a native developer, with a profound understanding of the iOS core

And experience working in a larger team

Q&A

Let’s connect

Francesca Piccoli
Frontend Engineer at Musixmatch
iOS Development, React Native