David Granado Jordan

0. Why learning iOS Development

Class 1

David Granado Jordan

High labor demand

Class 1

David Granado Jordan

Programing languages for iOS Development

Class 1

David Granado Jordan

Swift or Objective C

Class 1

David Granado Jordan

What tools do I need to start working?

Class 1

- Mac
- iPhone (optional)
- Xcode 13
- Swift playground (optional)
- Appleid account

https://appleid.apple.com/

David Granado Jordan

Oficial documentation

Class 1

https://swift.org/documentation/

https://developer.apple.com/wwdc22/

https://developer.apple.com/design/human-interface-guidelines/

Extras:
https://www.raywenderlich.com/

David Granado Jordan

Class 1

David Granado Jordan

1. Swift

Class 1

David Granado Jordan

Code and Syntax

Class 1 & 2

David Granado Jordan

Class 1 & 2

David Granado Jordan

2. Xcode

Class 2

David Granado Jordan

Hello world! App

Class 2

David Granado Jordan

Class 2

David Granado Jordan

3. UI Elements

Class 3

BUTTONS

PICKERS

PAGE CONTROL

SEGMENTED CONTROLS

SWITCH

LABEL

PROGRESS

TEXT FIELD

TEXTVIEW

STACK VIEW

WEBKIT VIEW

David Granado Jordan

Class 3

David Granado Jordan

4. Communication with UI Elements

Class 3

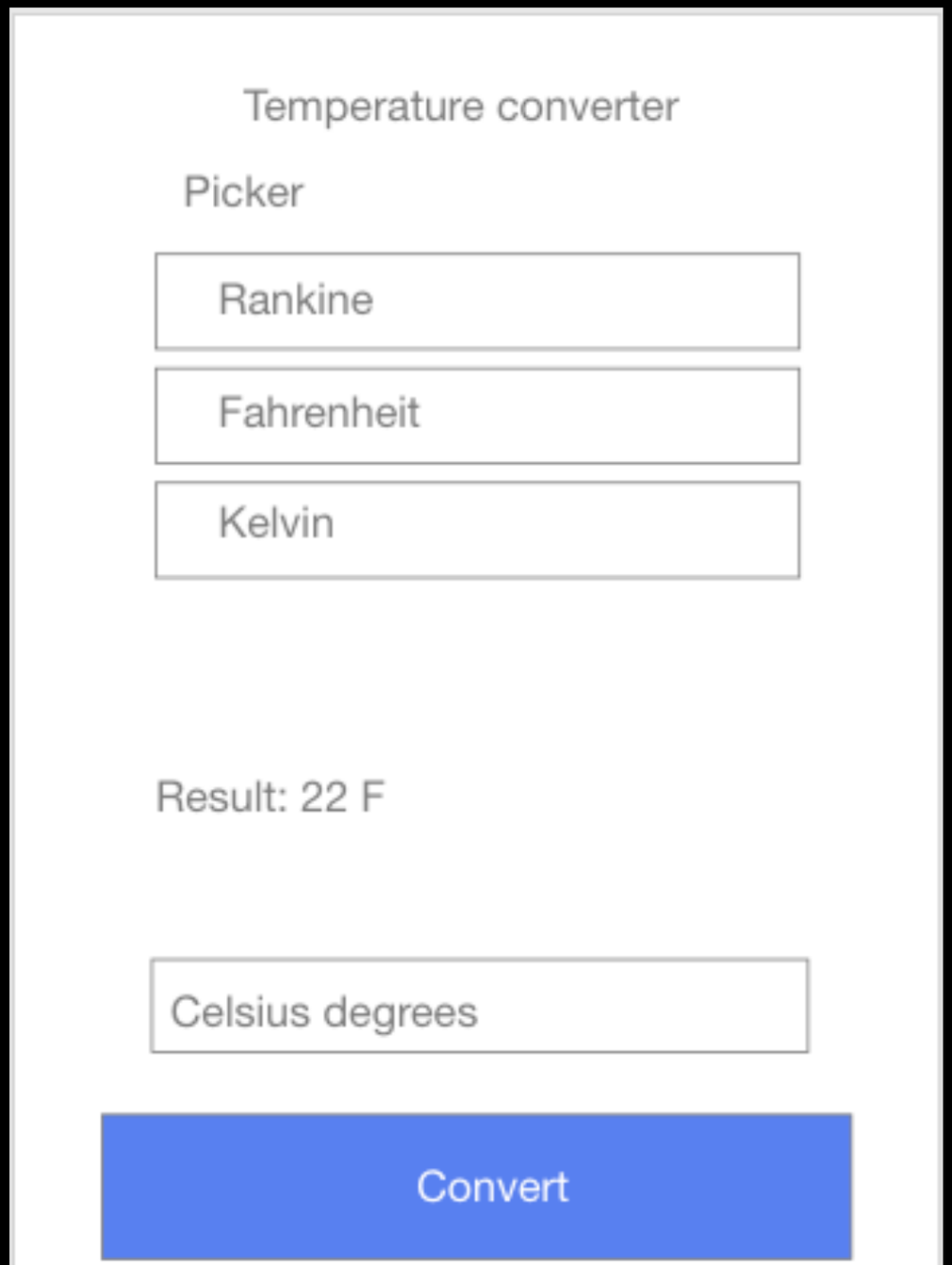
David Granado Jordan

IBoutlets, IBActions

Class 3

David Granado Jordan

Create an app “Temperature converter”



David Granado Jordan

Class 3

David Granado Jordan

5. Auto Layout and constraints

Class 4

When? How?

- Automatic
- Manual
- ScrollView

TASK: Apply auto layout to Temperature converter

David Granado Jordan

Scroll View

- 1. Add Scroll view

- 2. Add constraint from “Add New Constraint” section (all 0)

- 3. Go to Size inspector, disable “Content Layout Guides”

- 4. Add UIView inside of the Scroll View

- 5. Add constraint from “Add New Constraint” section (all 0)

- 6. Set Width of UIView (Add equal Width using Main View)

- 7. Set Height of UIView:

    - Option One: Fixed Height

    - Option Two: Height according to content (In this case you just need to add ui elements with specific heights and constraints)

TASK: Apply auto layout to Temperature converter

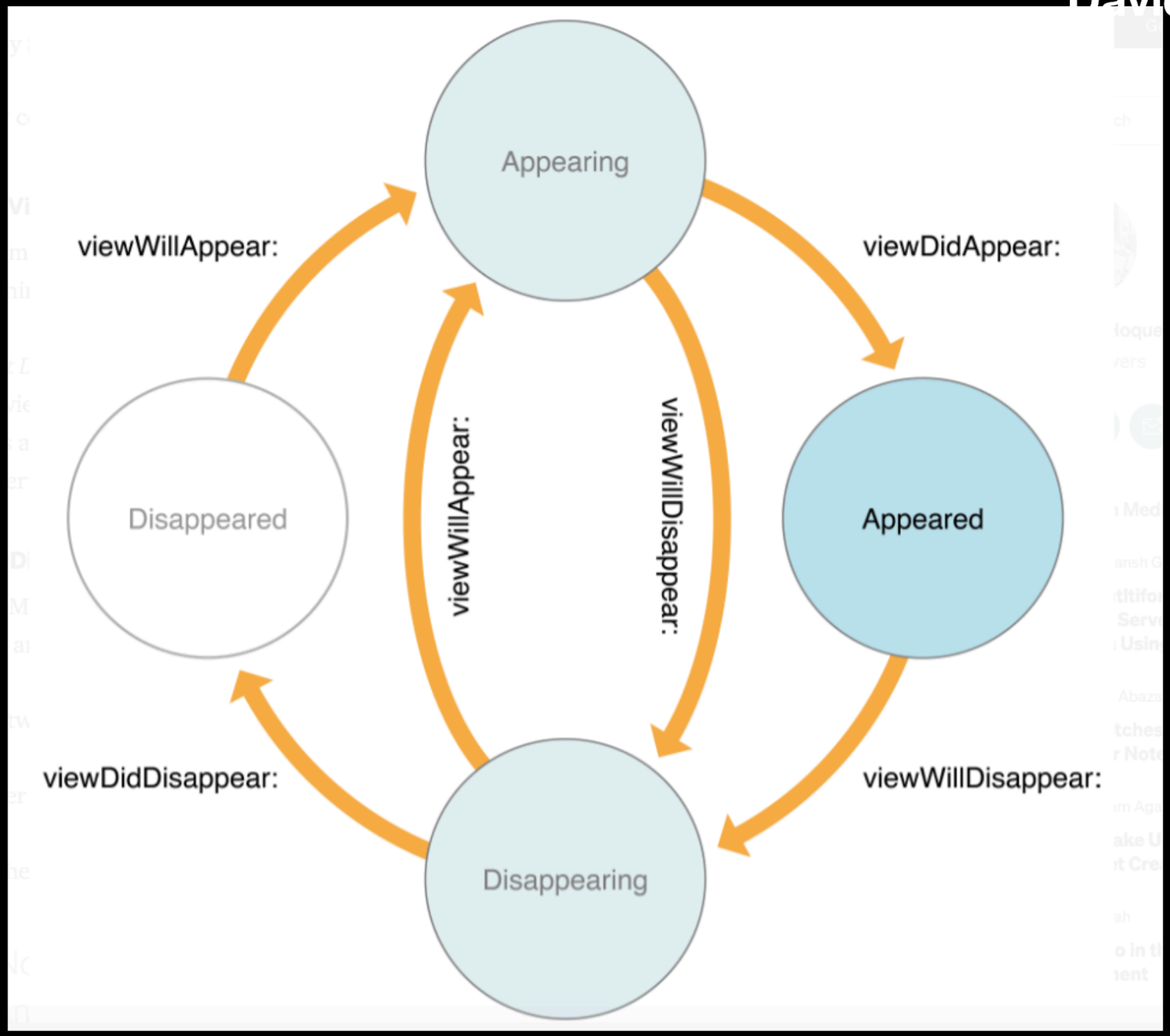
David Granado Jordan

Class 4

David Granado Jordan

6. View Controller Life Cycle

Class 5



viewDidLoad:

This Method is loaded once in view controller life cycle .Its Called When all the view are loaded .

Class 5

David Granado Jordan

Class 5

David Granado Jordan

7. Navigation between View Controllers

Class 5

David Granado Jordan

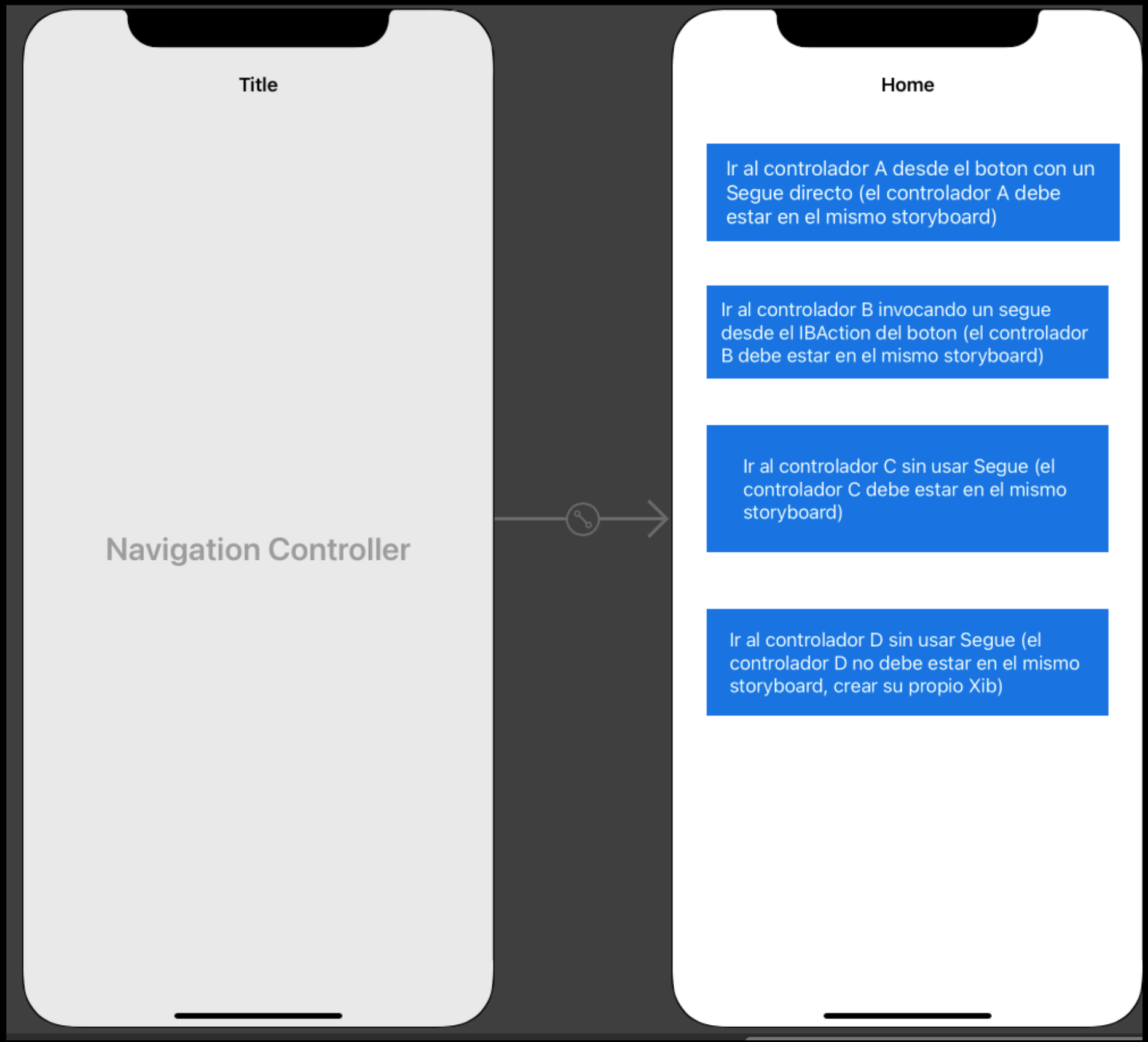
- Navigating using show method

- Navigating using segue

- Sending data between view controllers

David Granado Jordan

TASK:



David Granado Jordan

Class 5

David Granado Jordan

8. TableViews and CollectionViews

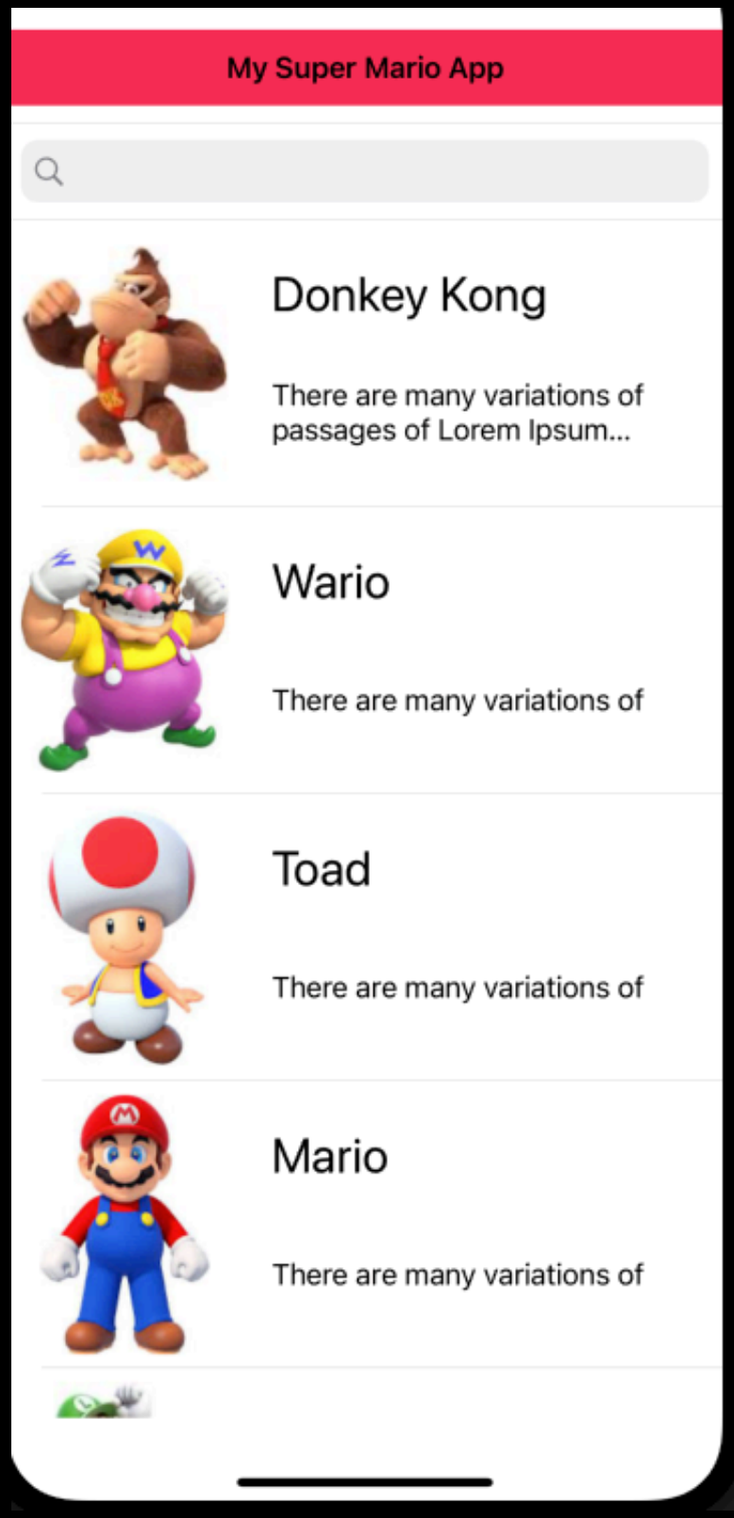
- Delegation Pattern

(Design pattern allows one object to send messages to another object when a specific event happens.)

- TableViewCell
- CollectionViewCell

David Granado Jordan

Mario Kart App



Class 6

David Granado Jordan

Static Data

Mario Kart App

TASK: Create Mario Kart app using CollectionView

Class 6

David Granado Jordan

Class 6

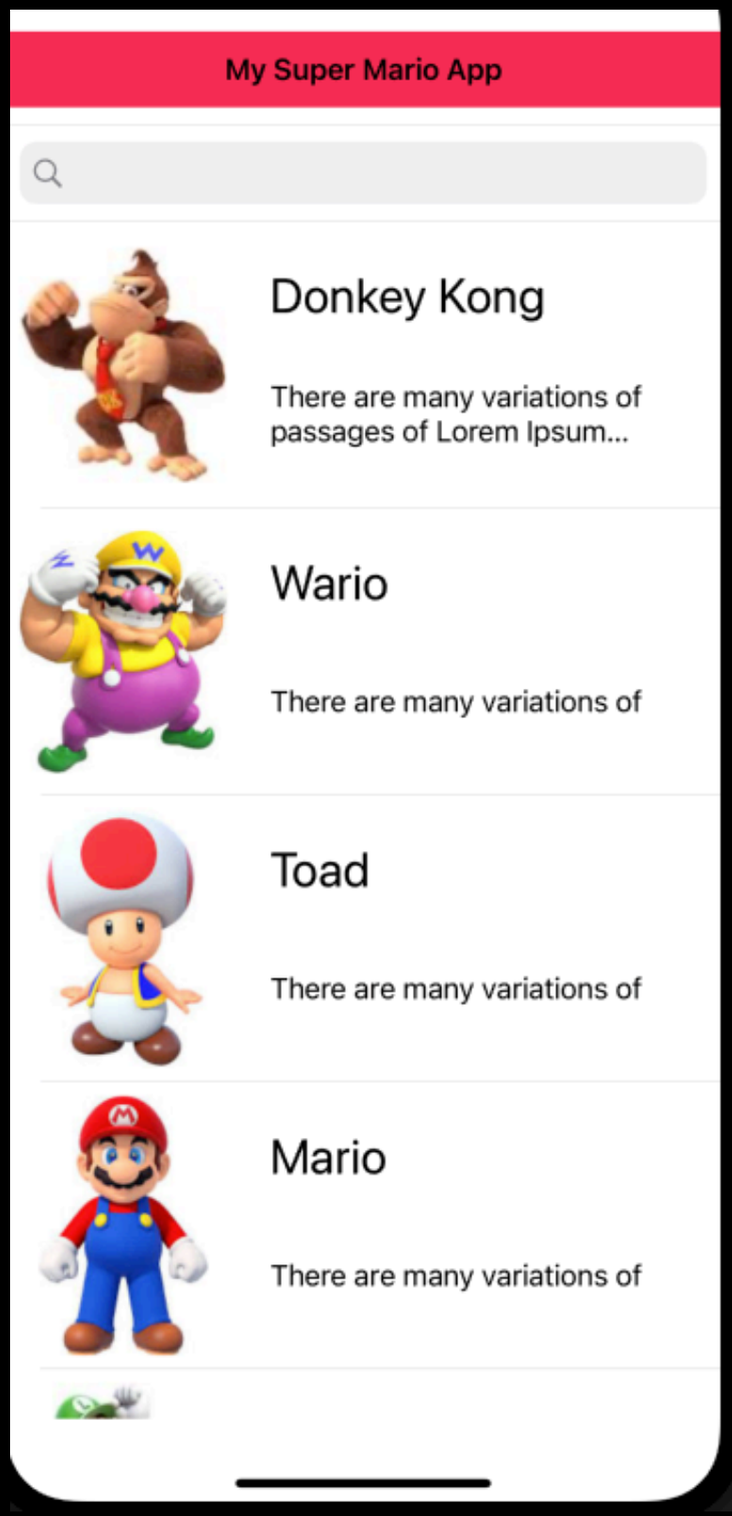
David Granado Jordan

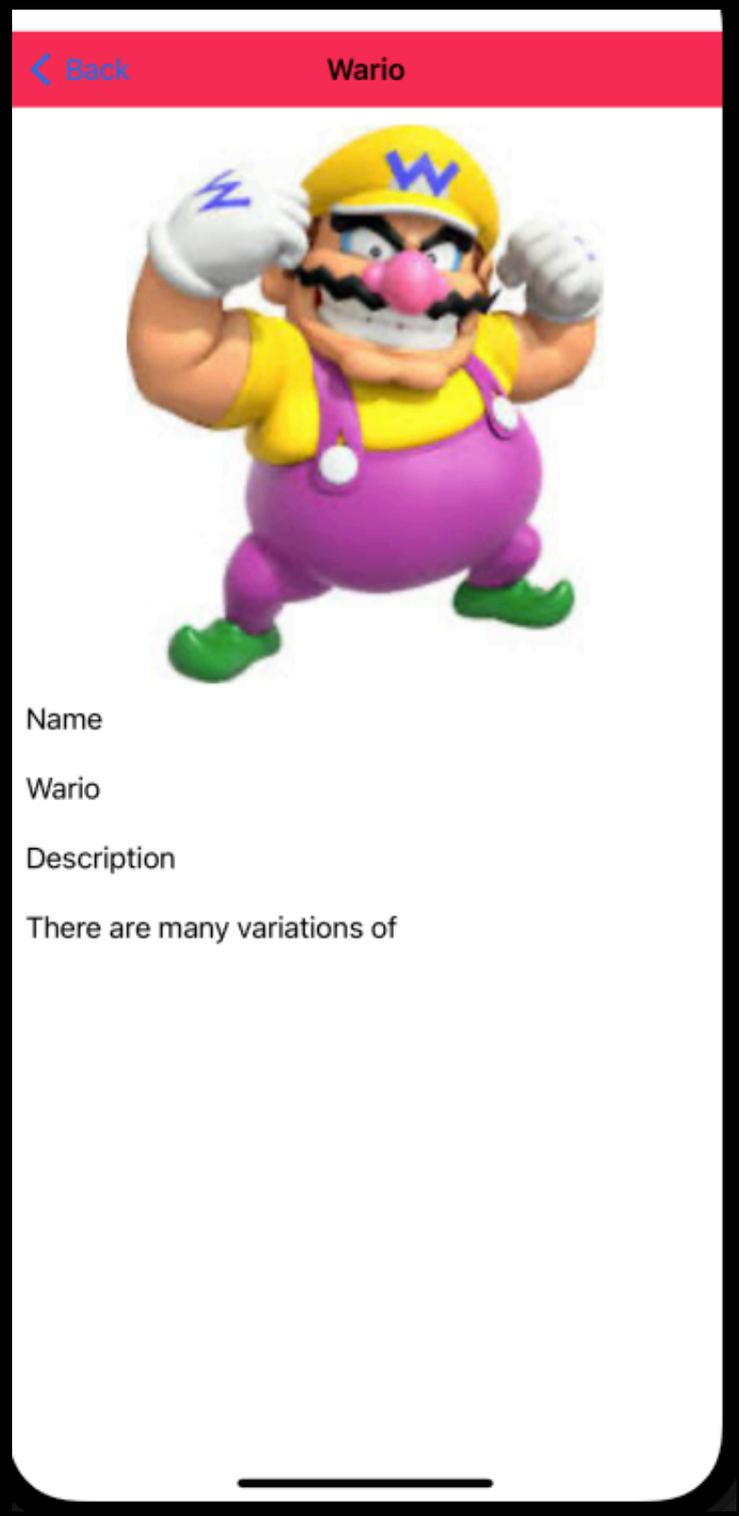
9. TableViews and CollectionViews (Go to detail Page)

- Navigation Controller

David Granado Jordan

Mario Kart App





Class 7

David Granado Jordan

Static Data

Mario Kart App

TASK:
Create Navigation in Mario Kart app using CollectionView

Class 7

David Granado Jordan

Class 7

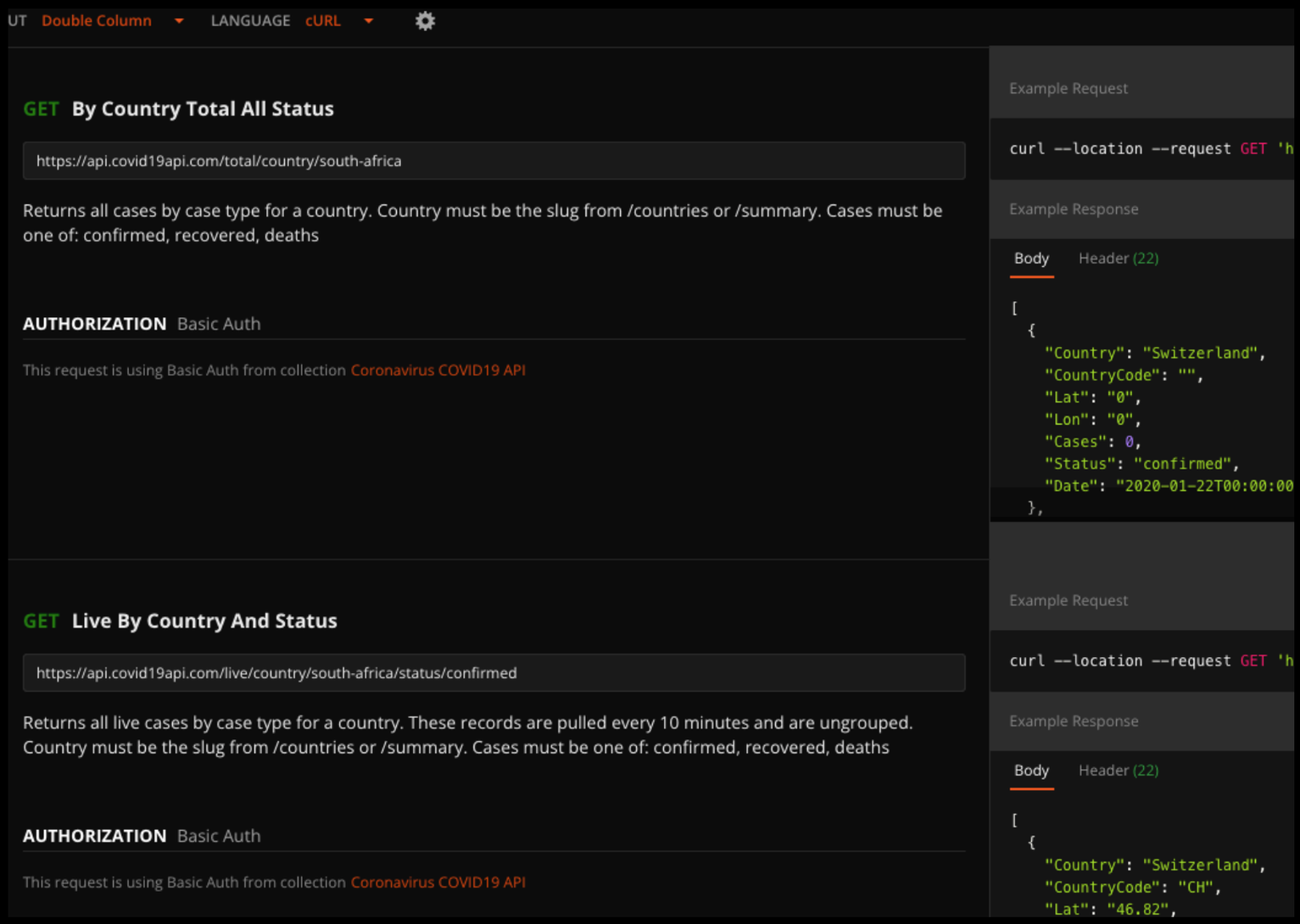
David Granado Jordan

10. TableViews and CollectionViews

- Create Covid status App
- Consume an API REST
- POD

David Granado Jordan

COVID 19. DOC:



https://documenter.getpostman.com/view/10808728/SzS8rjbc#6fbc46d6-0ddf-400b-a743-a149e9bba381

Class 8

TASK:

- Create a Shuffle cards app (doc): https://deckofcardsapi.com/

- Get deck id:

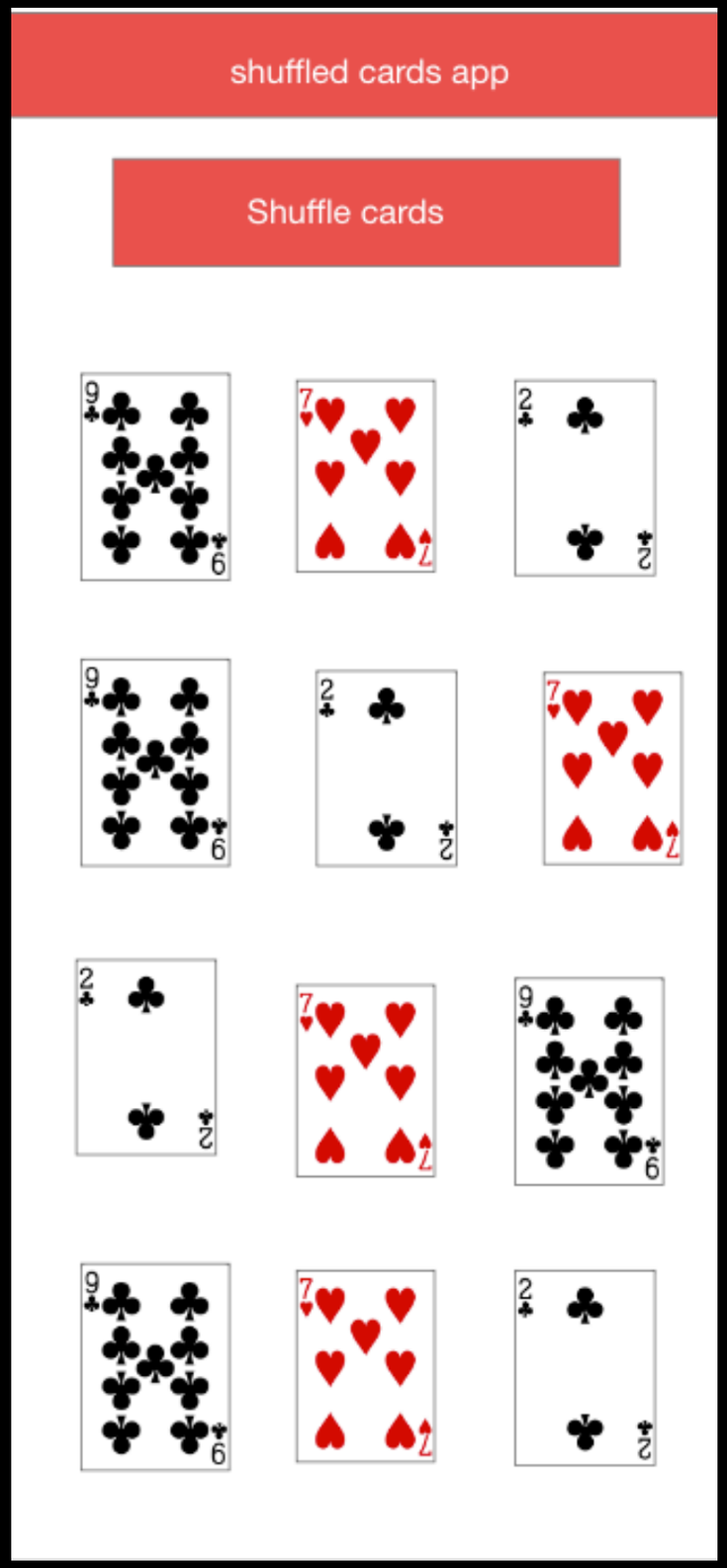
https://deckofcardsapi.com/api/deck/new/shuffle/?deck_count=1

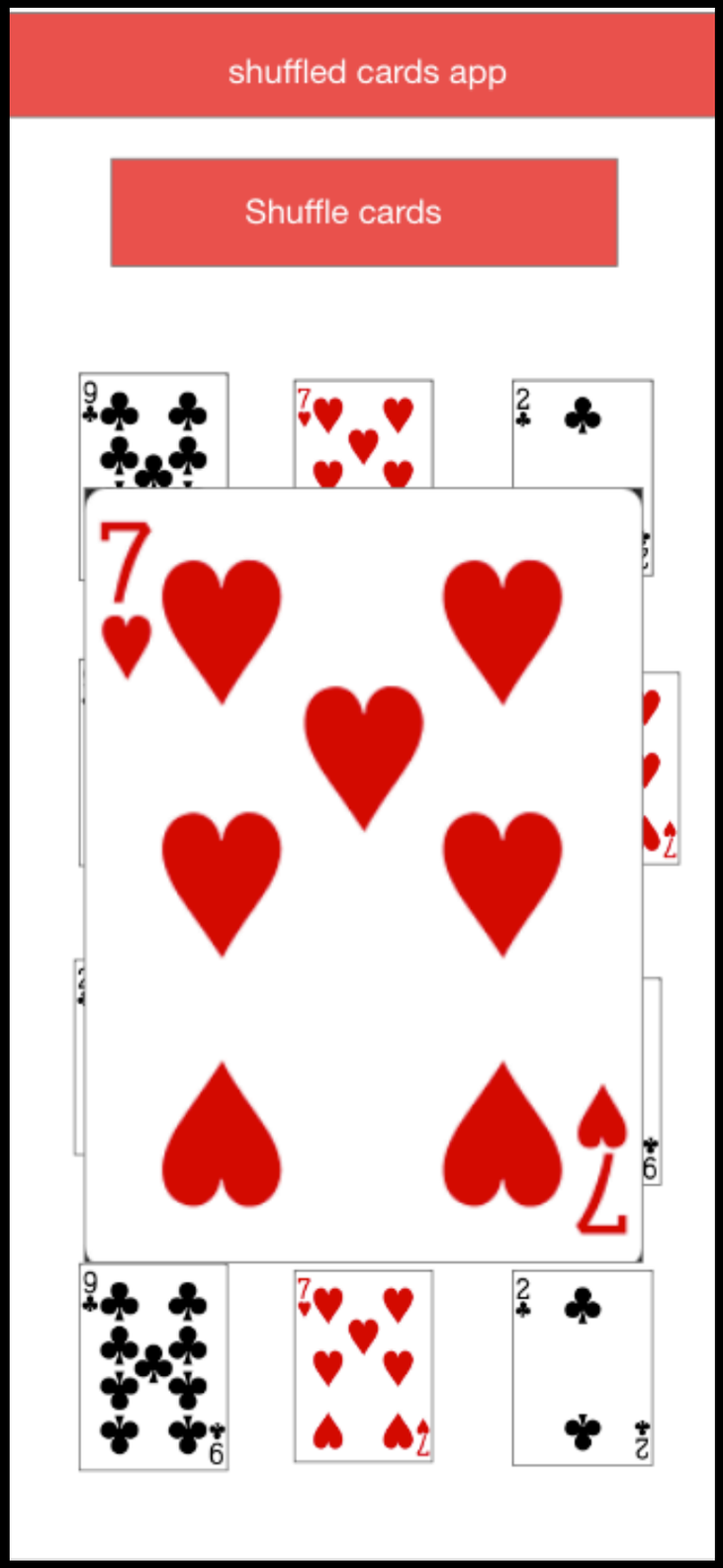
- Get shuffled cards by deck id:

https://deckofcardsapi.com/api/deck/cg5zlfjnyu50/draw/?count=52

David Granado Jordan

Shuffle cards app





Class 9

David Granado Jordan

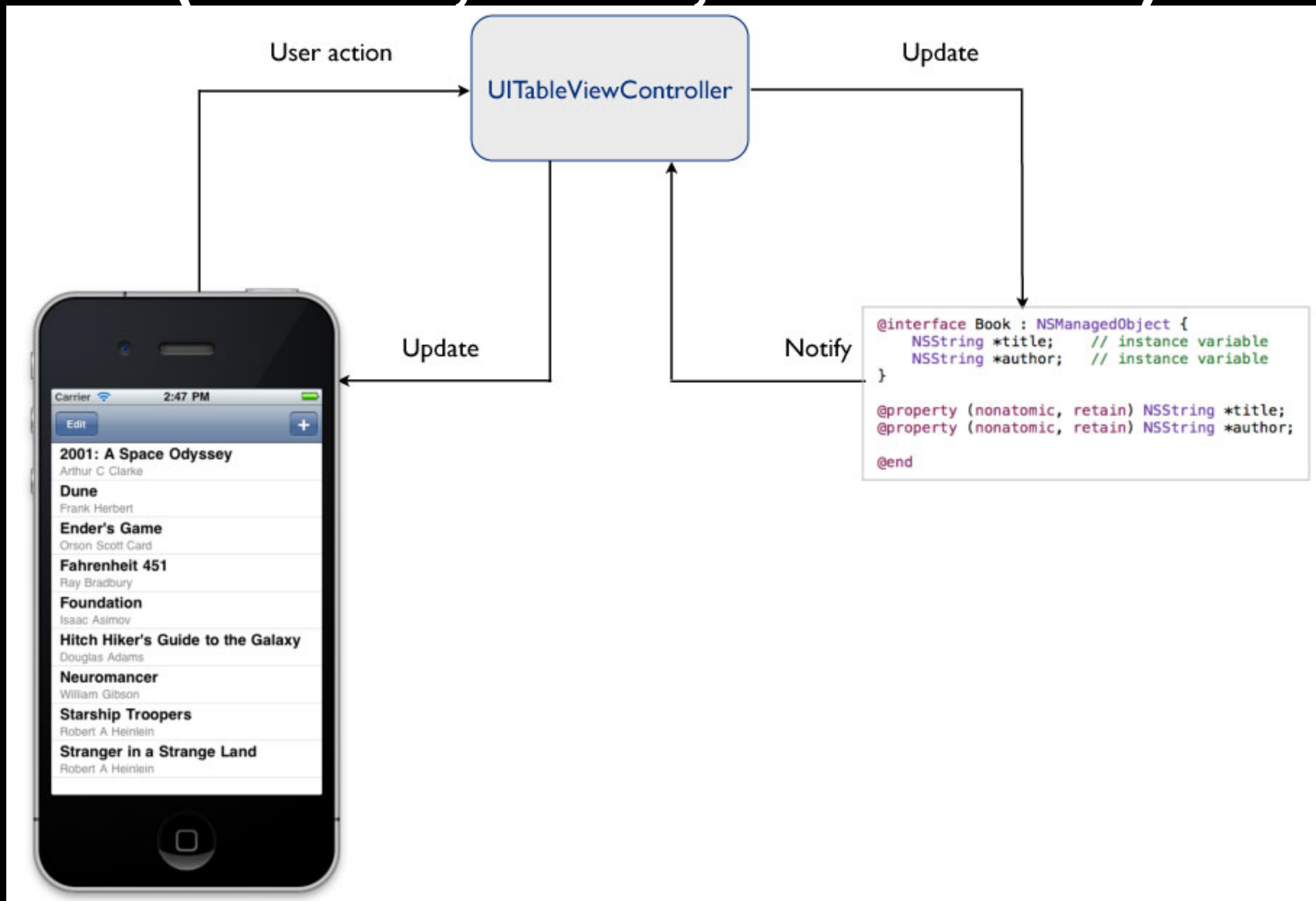
Class 9

David Granado Jordan

Class 9

David Granado Jordan

12. MVC (Model, View, Controller)



David Granado Jordan

Make Covid App using MVC

- Implement Decodable protocol
- Network Manager

Class 10

12. DEBUGGING

David Granado Jordan

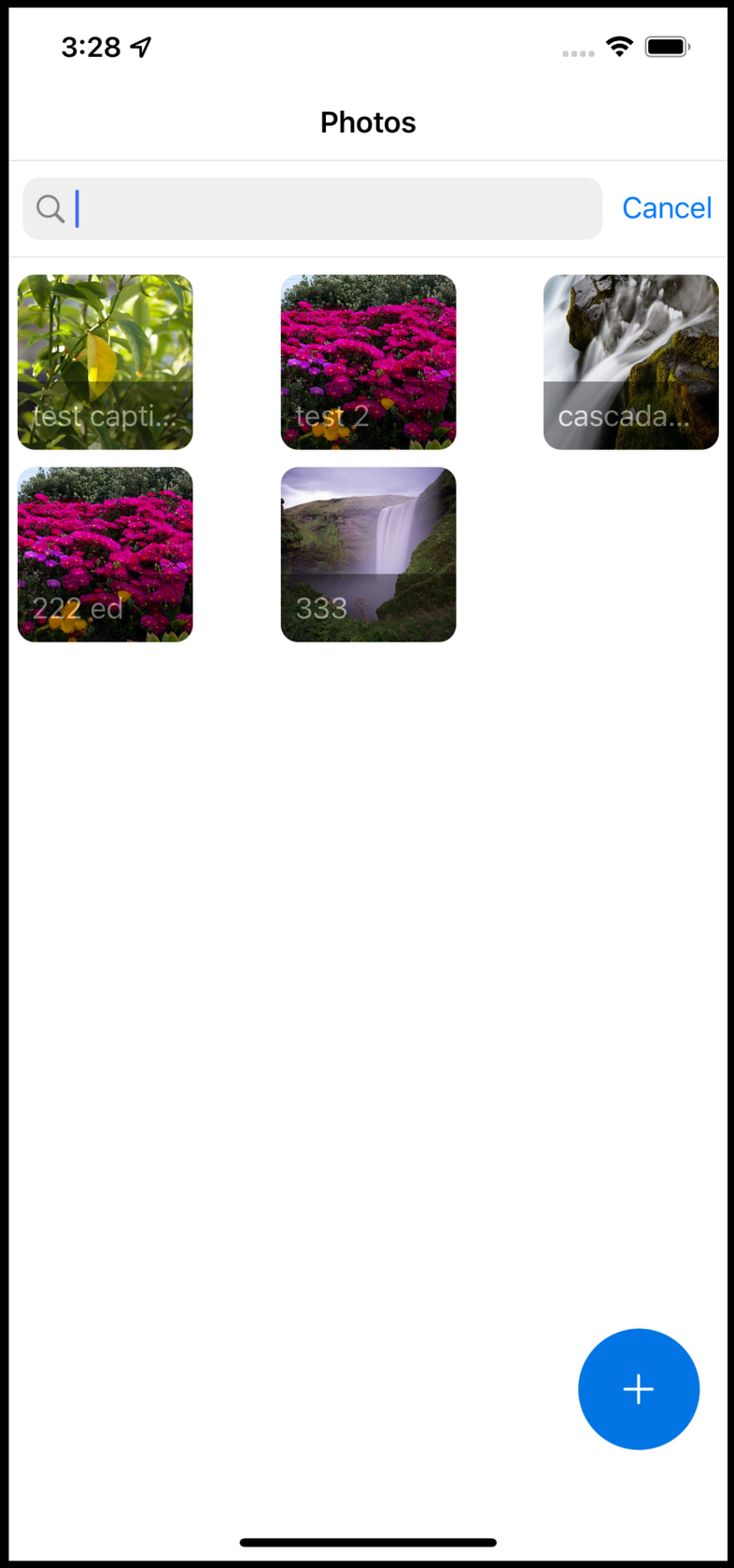
- **Debugging** is a process of searching and removing *bugs* from the code
- The process of debugging might be not easy and sometimes becomes very tricky
- Writing clean, well-documented code that conforms coding conventions greatly simplifies debugging process

David Granado Jordan

Class 10

David Granado Jordan

13. Access to PhotoLibrary and Camera



Class 11 & 12

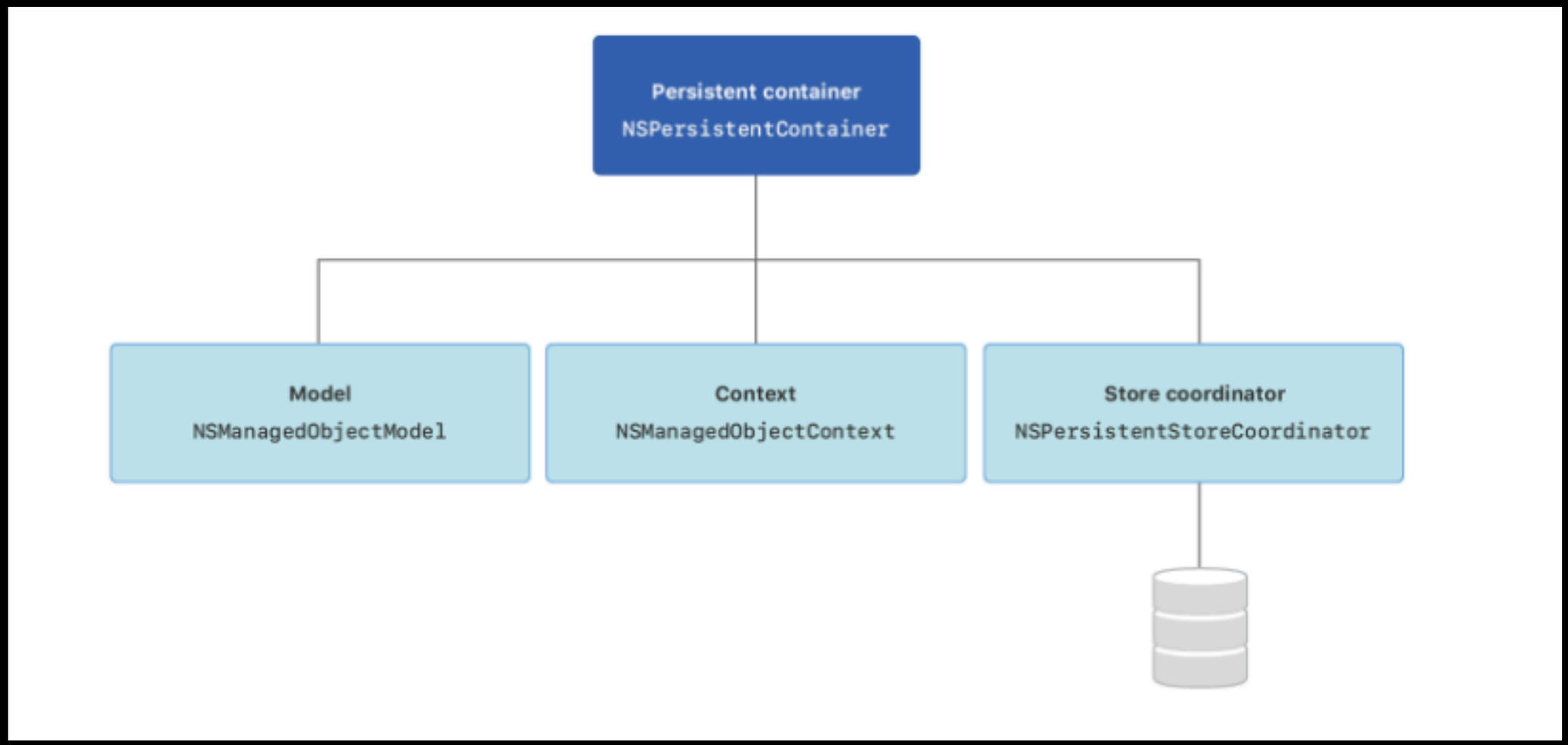
David Granado Jordan

13. Image viewer app

- UIImagePickerControllerDelegate
- UIImagePickerController
- CollectionViews
- Core Data

David Granado Jordan

13. Core Data



Class 11 & 12

David Granado Jordan

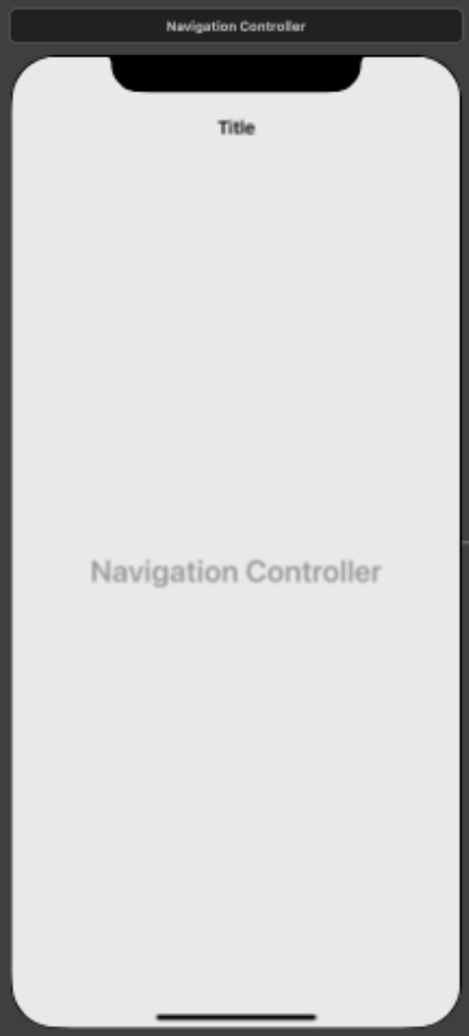
Class 11 & 12

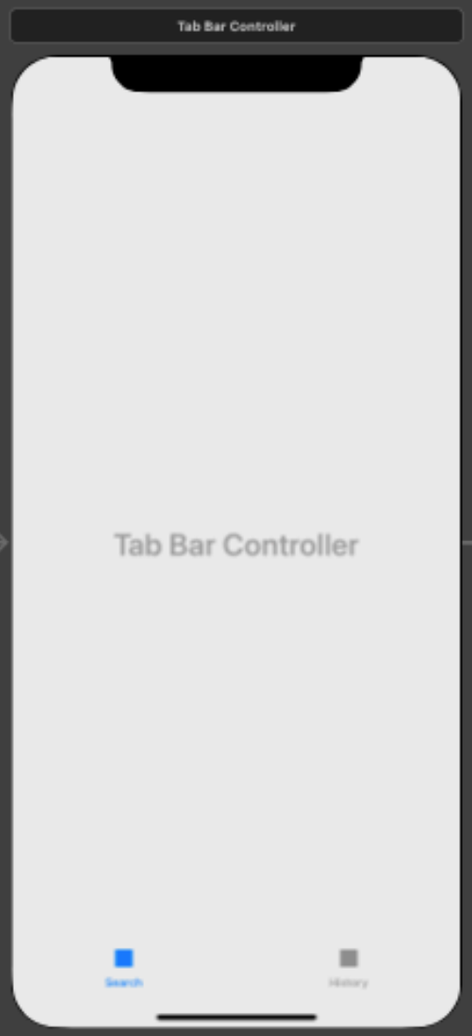
13.1. TASK: Weather app

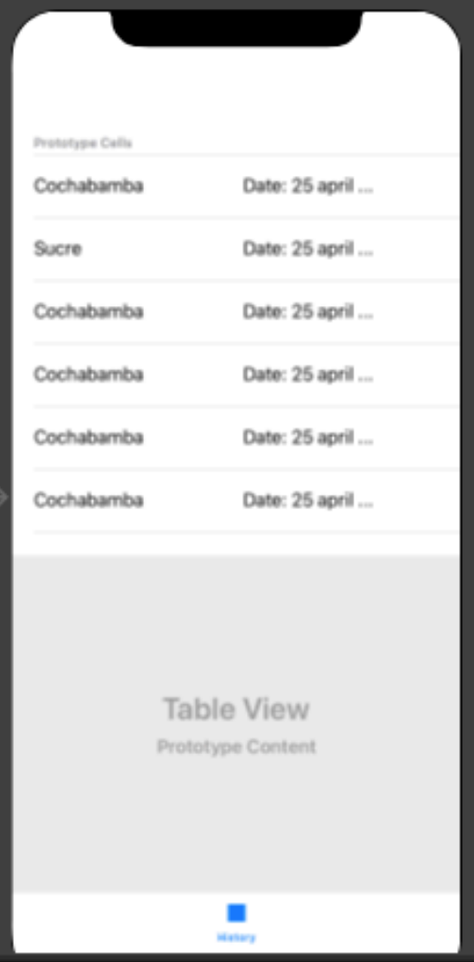
Get weather by city:
https://openweathermap.org/data/2.5/weather?q=cochabamba&appid=439d4b804bc8187953eb36d2a8c26a02&units=metric

Get icon:
https://openweathermap.org/img/wn/02n@2x.png

David Granado Jordan

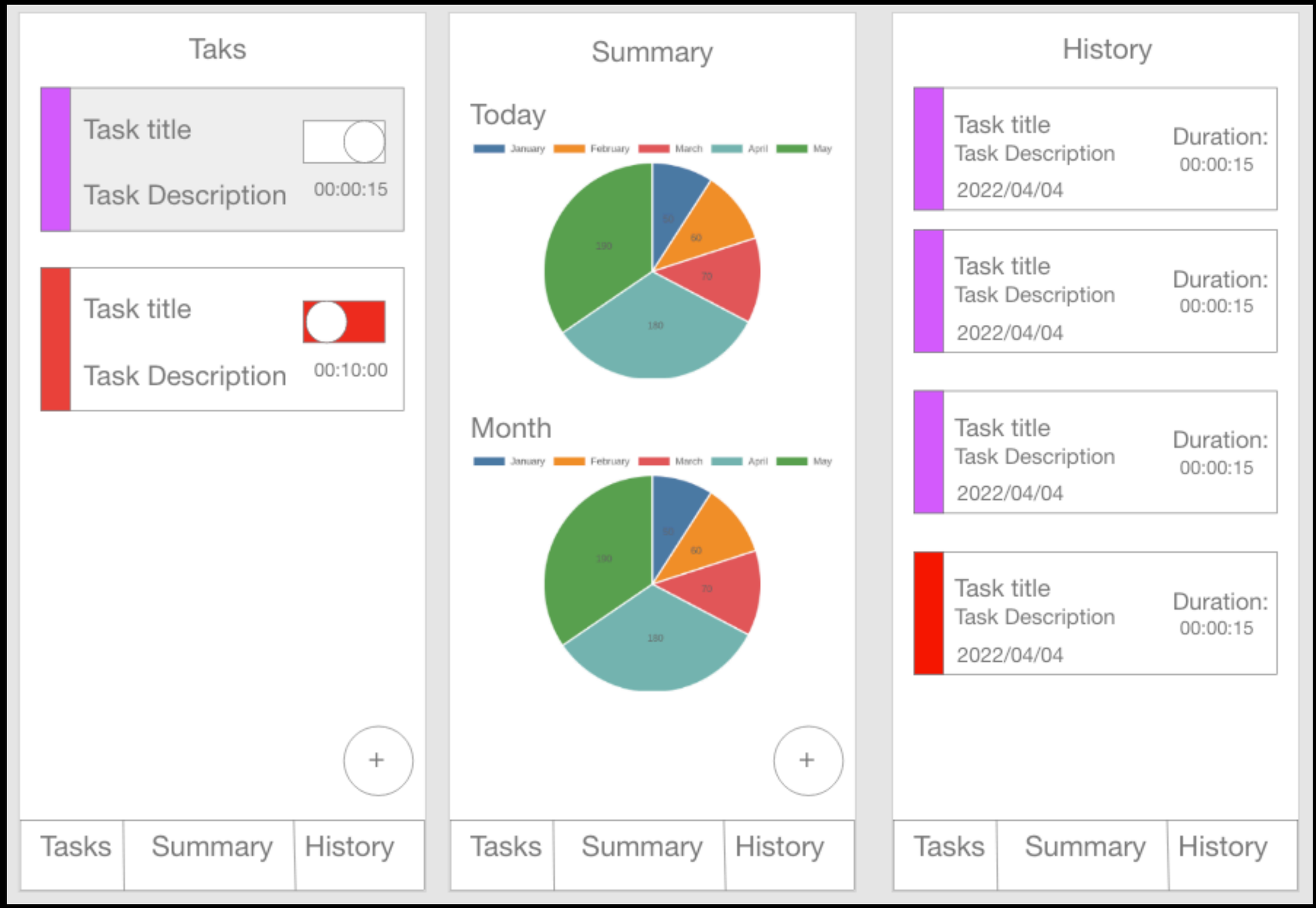






David Granado Jordan

14. TimeTracking app



David Granado Jordan

14. TimeTracking app

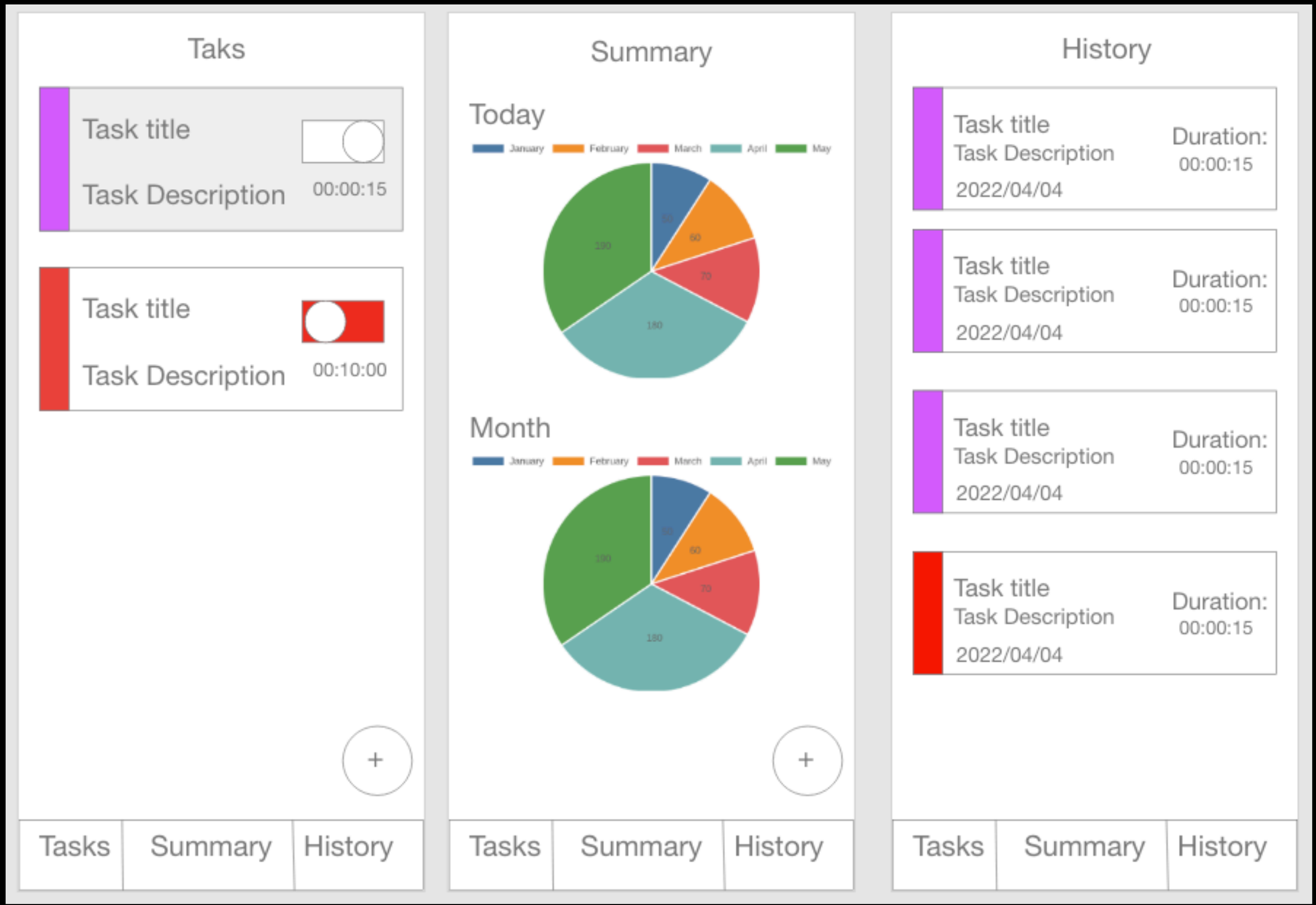
- Tab bar Controller
- Local Push notifications
- TableViews
- Core data
- Api Rest
- POD

David Granado Jordan

Class 13 & 14

David Granado Jordan

14. TimeTracking app (Push notifications)



Class 15

14. TimeTracking app

- Add Local Push notifications
- NotificationCenter
- Remote push notifications

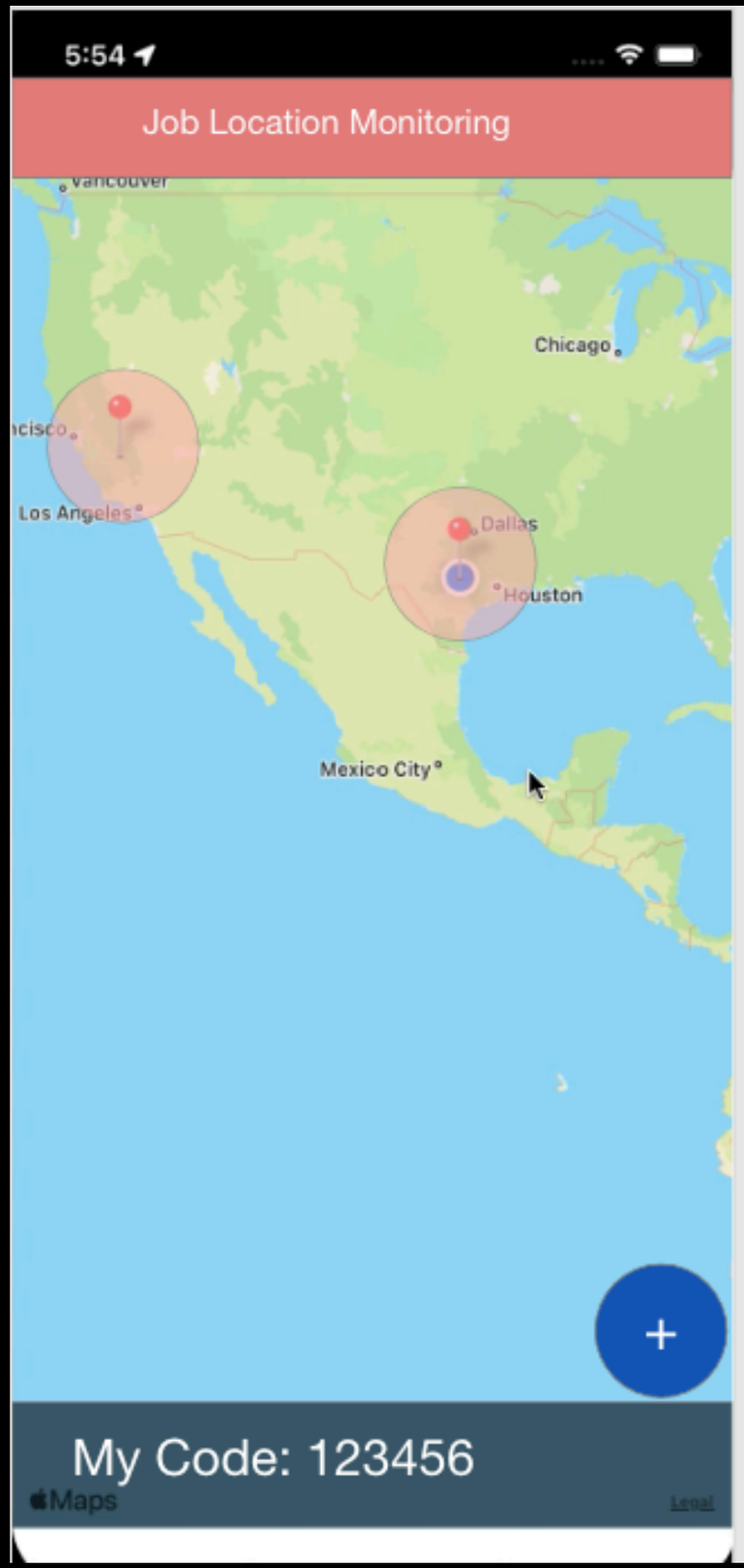
TASK: Schedule push notification if the user created a task and did not create a time punch more than 8 hours ago

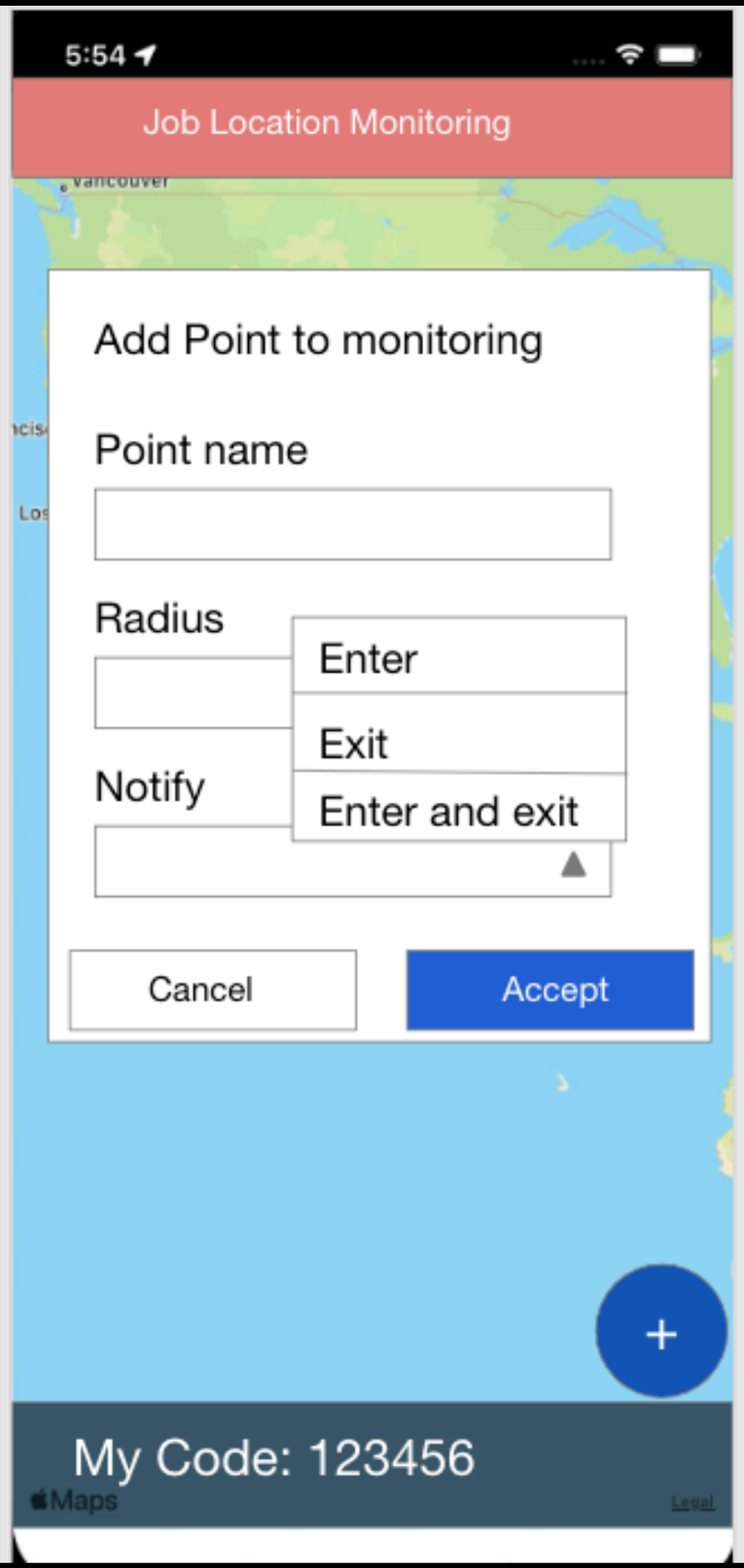
David Granado Jordan

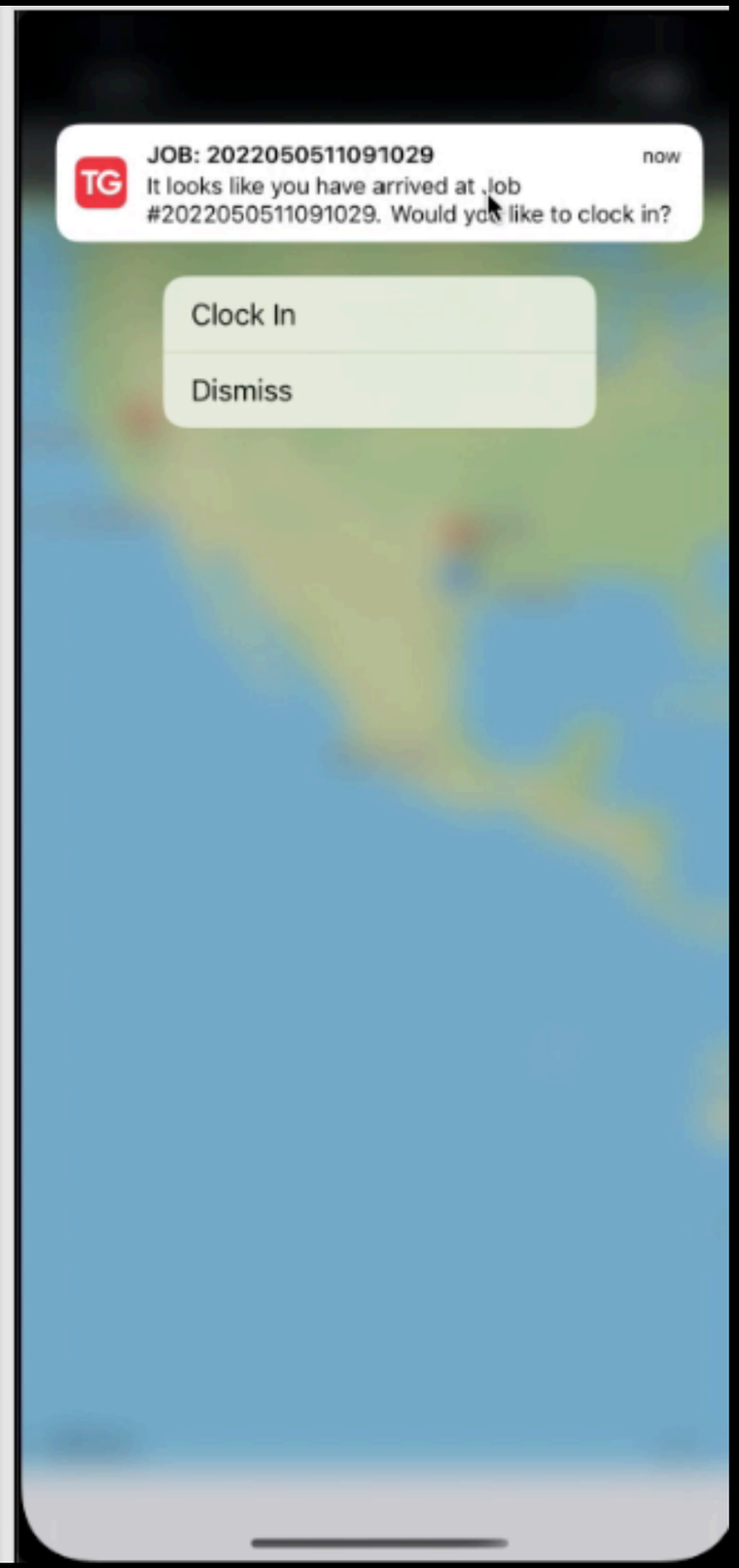
Class 15

David Granado Jordan

15. Job Location monitoring







David Granado Jordan

15. Job Location monitoring

- MKMapView
- Info.plist
- Ask for permissions
- Location
- GEOFENFING
- Push notifications
- POD
    - REALM
- Notification Center

David Granado Jordan

Class 15 & 16 & 17

16. USER DEFAULTS

David Granado Jordan

UserDefaults in Swift is a class that provides an interface to save key-value pairs in a user's stored database.

```swift
let defaults = UserDefaults.standard
defaults.set(25, forKey: "Age")
defaults.set(true, forKey: "UseTouchID")
defaults.set(CGFloat.pi, forKey: "Pi")

defaults.set("Paul Hudson", forKey: "Name")
defaults.set(Date.now, forKey: "LastRun")
```

```swift
let age = defaults.integer(forKey: "Age")
let useTouchID = defaults.bool(forKey: "UseTouchID")
let pi = defaults.double(forKey: "Pi")
```

```swift
let array = ["Hello", "World"]
defaults.set(array, forKey: "SavedArray")

let dict = ["Name": "Paul", "Country": "UK"]
defaults.set(dict, forKey: "SavedDict")
```

```swift
let savedArray = defaults.object(forKey: "SavedArray") as? [String] ?? [String]()
```

David Granado Jordan

16. USER DEFAULTS

Custom objects (Save object):

```swift
import Foundation

struct Note: Codable {

    // MARK: - Properties

    let id: Int

    // MARK: -

    let title: String
    let body: String

}

// Create Note
let note = Note(id: 1, title: "My Note", body: "Lorem ipsum dolor sit amet, consectetur adipiscing elit.")

do {
    // Create JSON Encoder
    let encoder = JSONEncoder()

    // Encode Note
    let data = try encoder.encode(note)

    // Write/Set Data
    UserDefaults.standard.set(data, forKey: "note")

} catch {
    print("Unable to Encode Note (\(error))")
}
```

David Granado Jordan

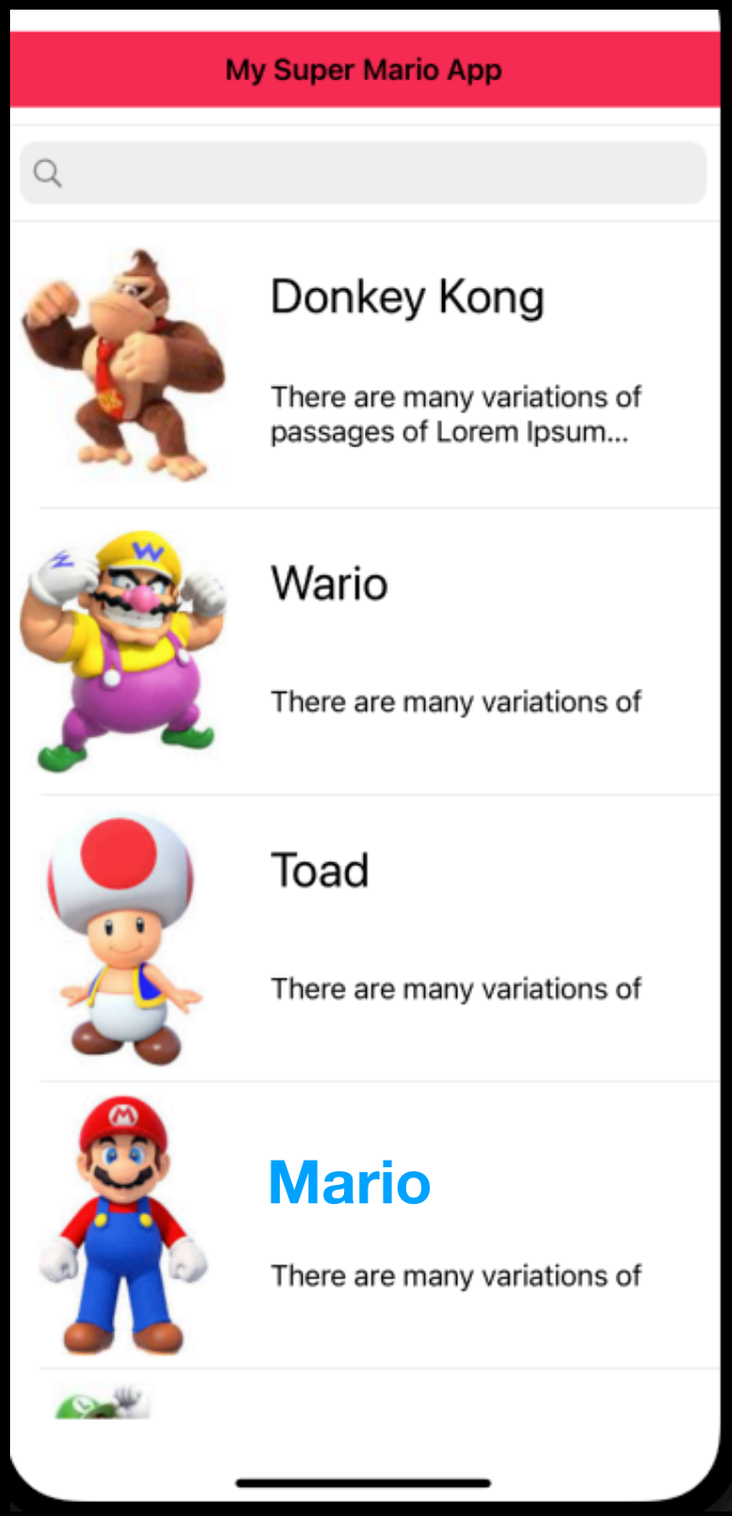
16. USER DEFAULTS

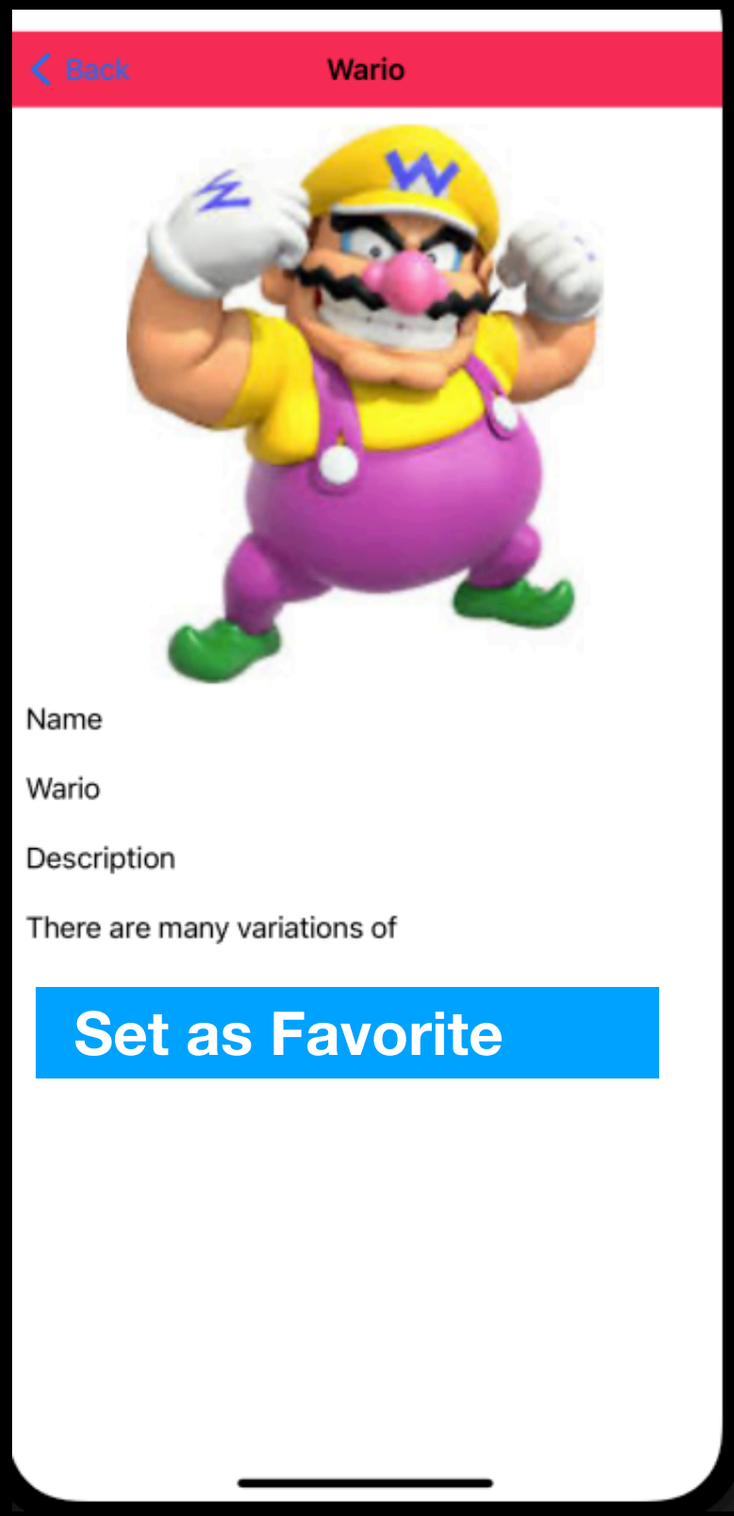
Custom objects (Get object):

```swift
import Foundation

// Read/Get Data
if let data = UserDefaults.standard.data(forKey: "note") {
    do {
        // Create JSON Decoder
        let decoder = JSONDecoder()

        // Decode Note
        let note = try decoder.decode(Note.self, from: data)

    } catch {
        print("Unable to Decode Note (\(error))")
    }
}
```

David Granado Jordan

Mario Kart App





Add option to set a character as favorite.

Save this data using User defaults.

David Granado Jordan

Class 18

David Granado Jordan

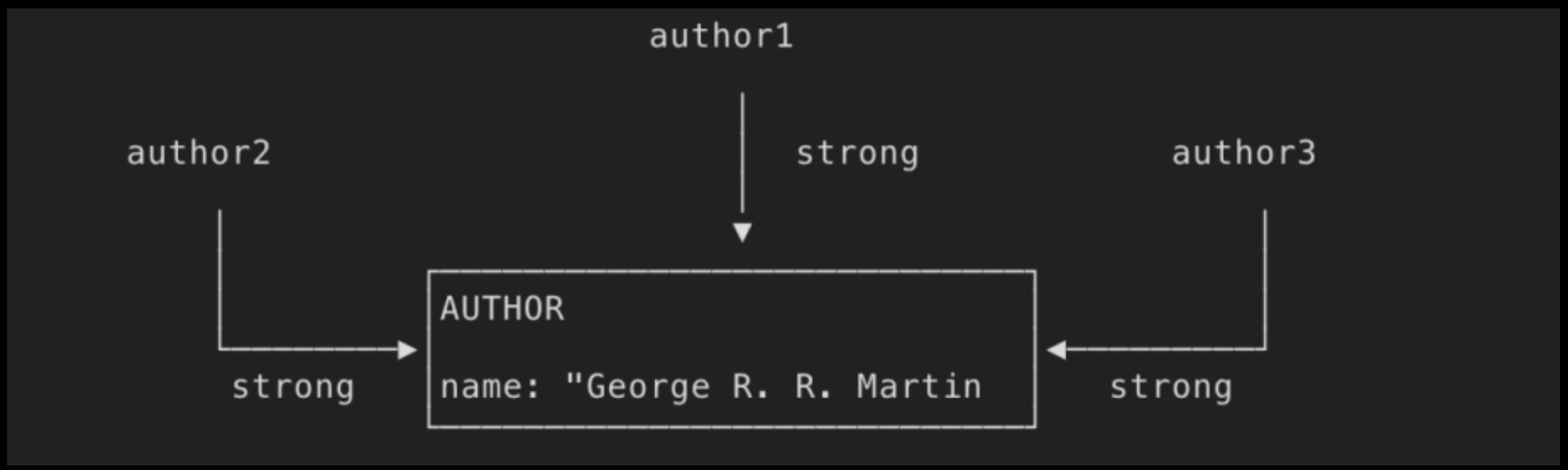
Automatic Reference Counting

https://docs.swift.org/swift-book/LanguageGuide/AutomaticReferenceCounting.html

Strong Reference

David Granado Jordan

Problem:



Strong Reference

David Granado Jordan

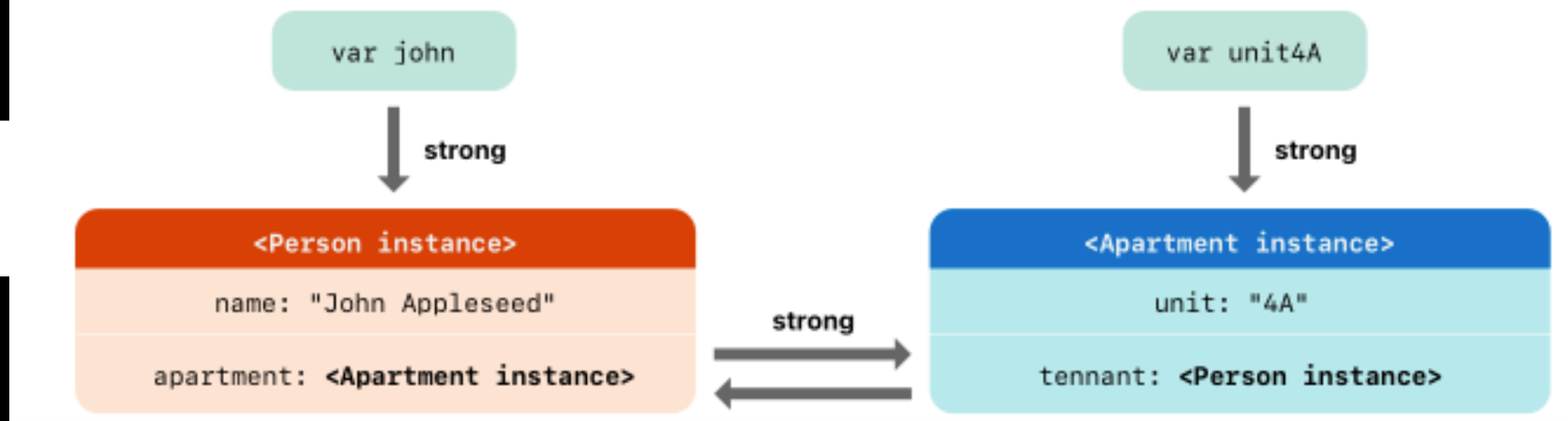
```
class Person {
    let name: String
    init(name: String) { self.name = name }
    var apartment: Apartment?
    deinit { print("\(name) is being deinitialized") }
}

class Apartment {
    let unit: String
    init(unit: String) { self.unit = unit }
    var tenant: Person?
    deinit { print("Apartment \(unit) is being deinitialized") }
}
```

Strong Reference

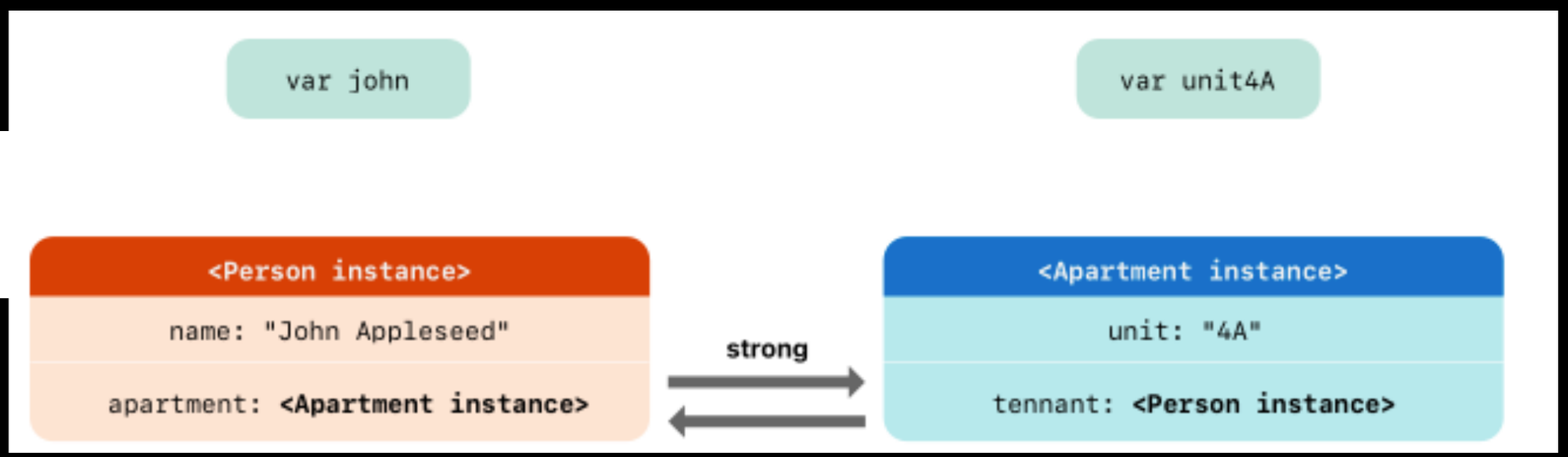
David Granado Jordan

```
1  john = Person(name: "John Appleseed")
2  unit4A = Apartment(unit: "4A")
```


```
1  john!.apartment = unit4A
2  unit4A!.tenant = john
```

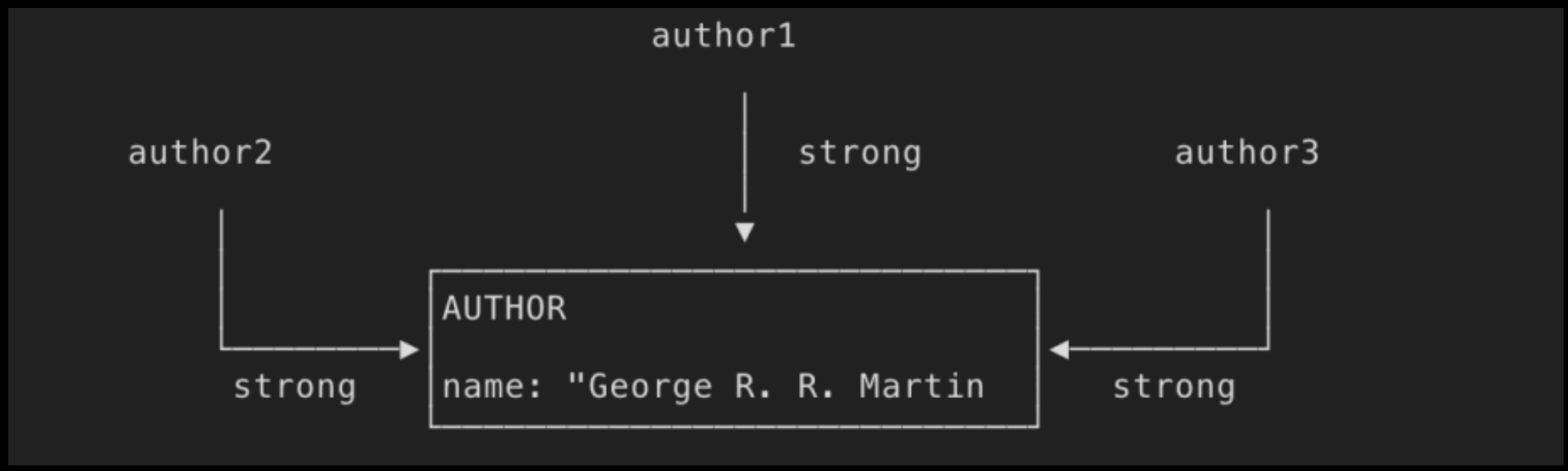



```
1  john = nil
2  unit4A = nil
```



David Granado Jordan

Strong Reference Problem:



Solution:

```
// It is Calling to deinit
john?.apartment = nil
unit4A?.tenant = nil
```

Weak Reference

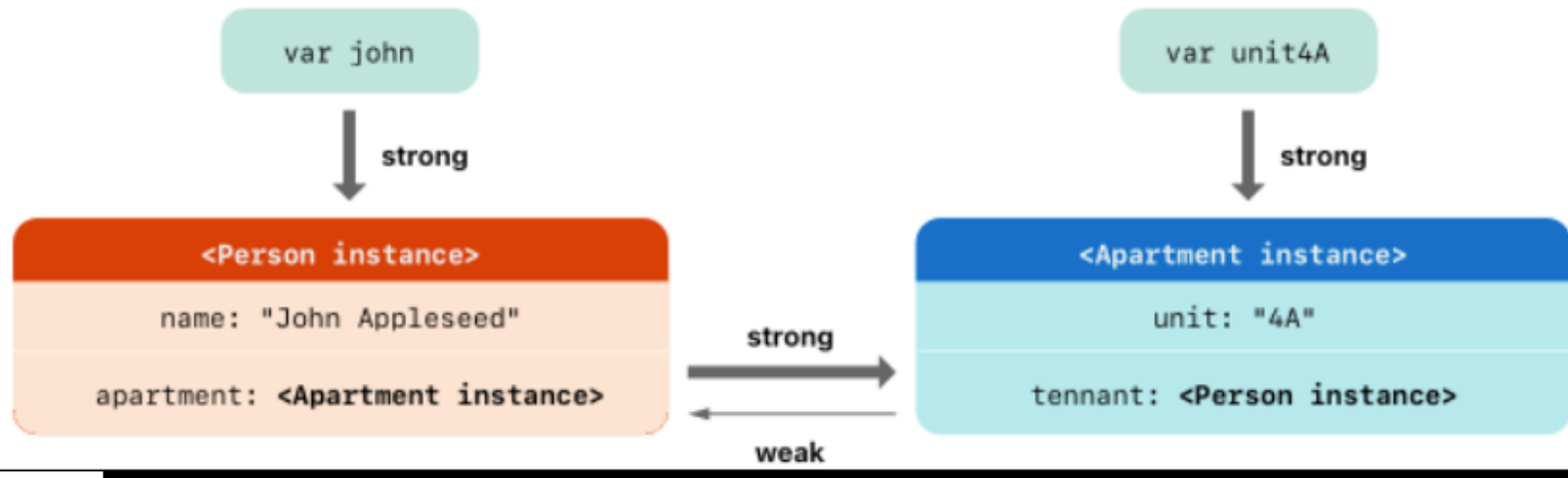
David Granado Jordan

```
class Person {
    let name: String
    init(name: String) { self.name = name }
    var apartment: Apartment?
    deinit { print("\(name) is being deinitialized") }
}

class Apartment {
    let unit: String
    init(unit: String) { self.unit = unit }
    weak var tenant: Person?
    deinit { print("Apartment \(unit) is being deinitialized") }
}
```

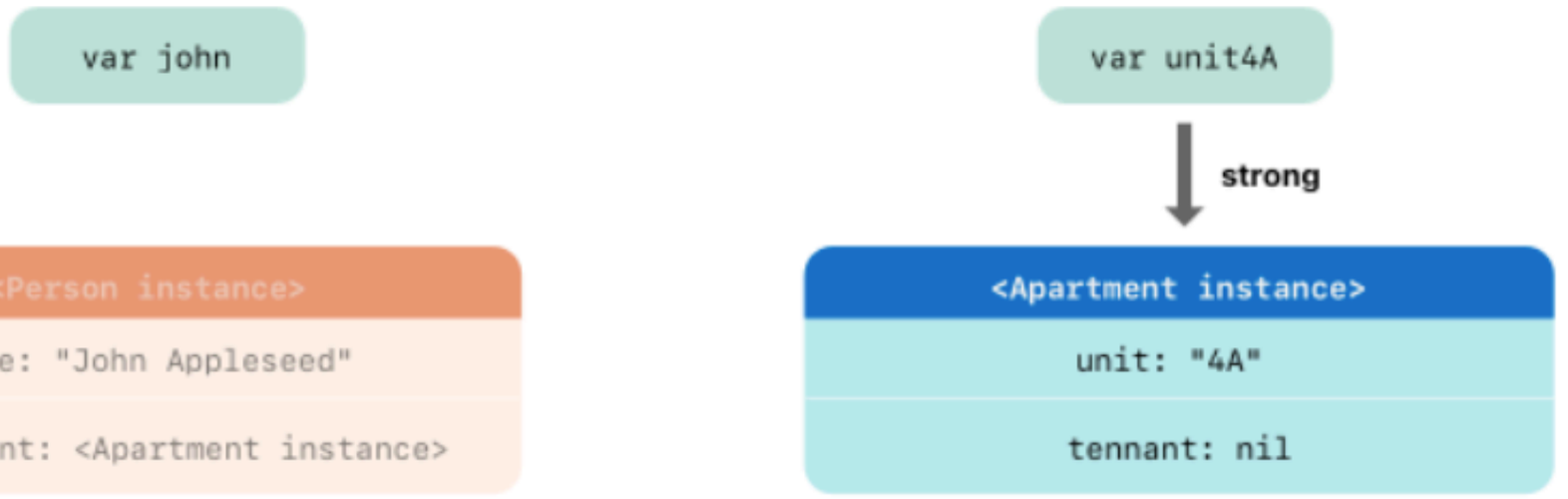
Weak Reference

David Granado Jordan

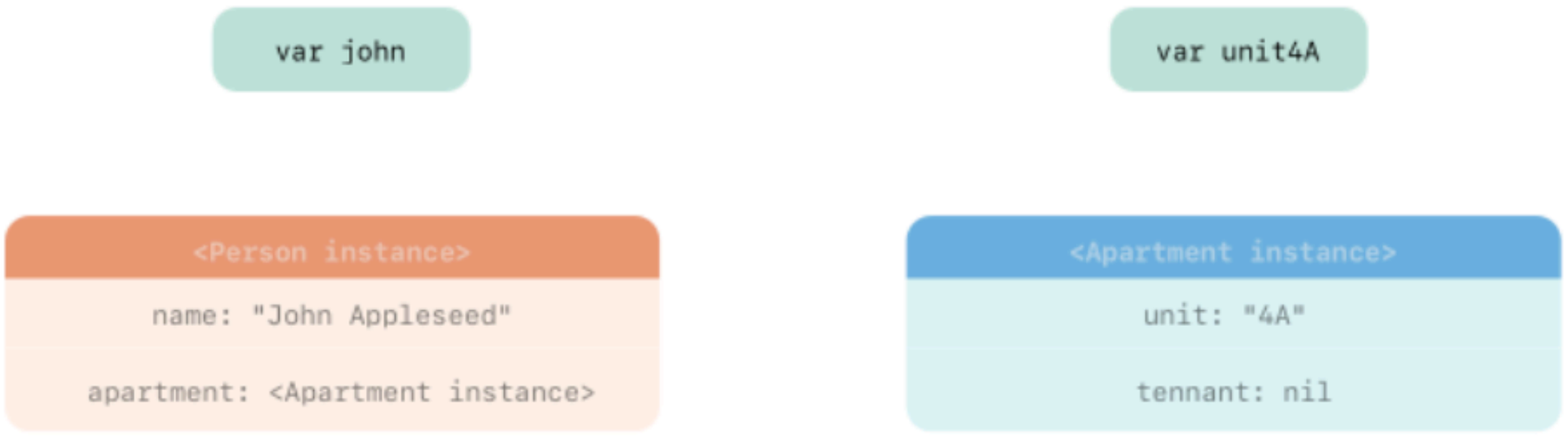
```swift
var john: Person?
var unit4A: Apartment?

john = Person(name: "John Appleseed")
unit4A = Apartment(unit: "4A")

john!.apartment = unit4A
unit4A!.tenant = john
```




```swift
john = nil
// Prints "John Applesee
```




```swift
unit4A = nil
// Prints "Apartment 4A
```



David Granado Jordan

Class 18

David Granado Jordan

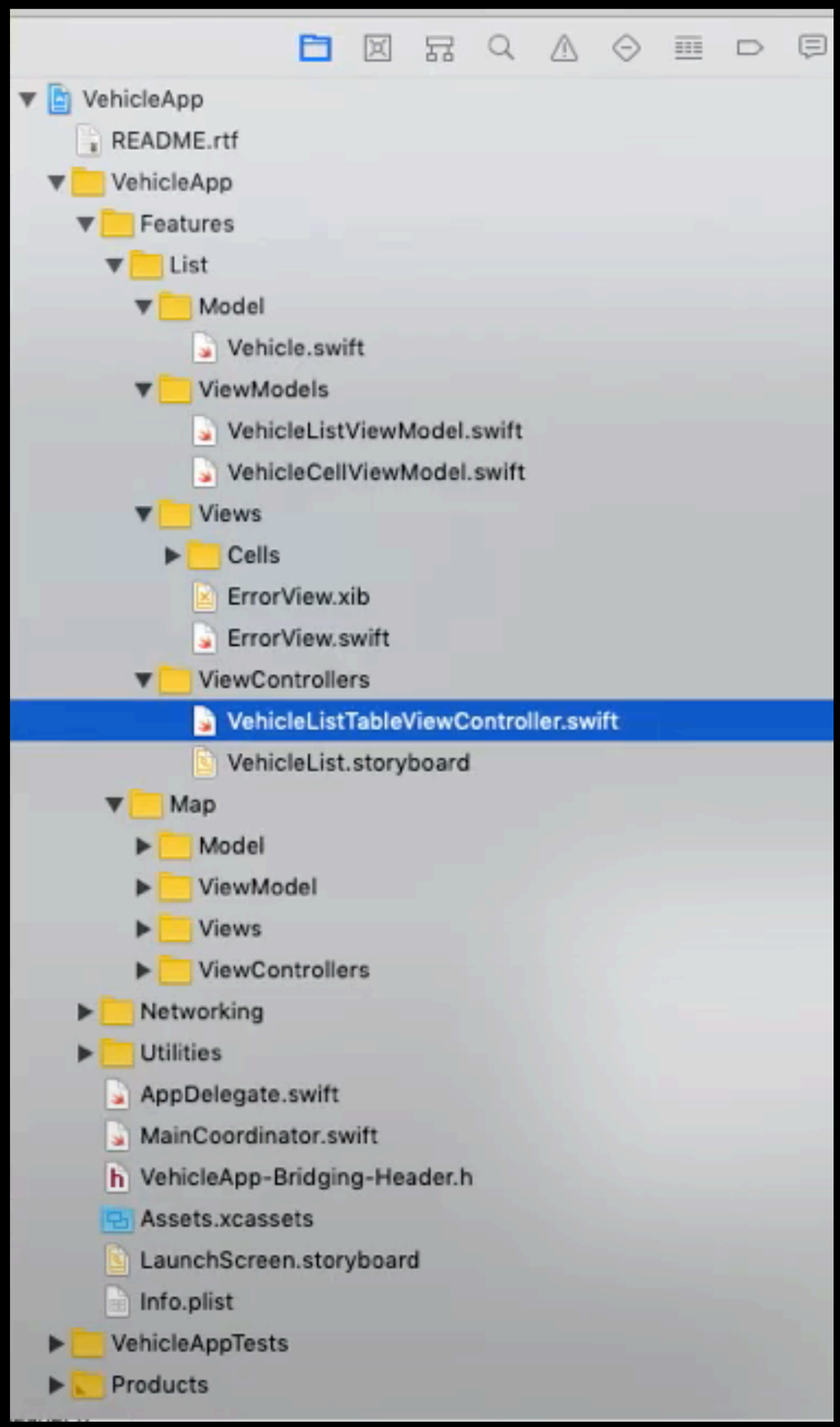
17. MVVM

MVC

MVVM

David Granado Jordan

17. MVVM



Class 18

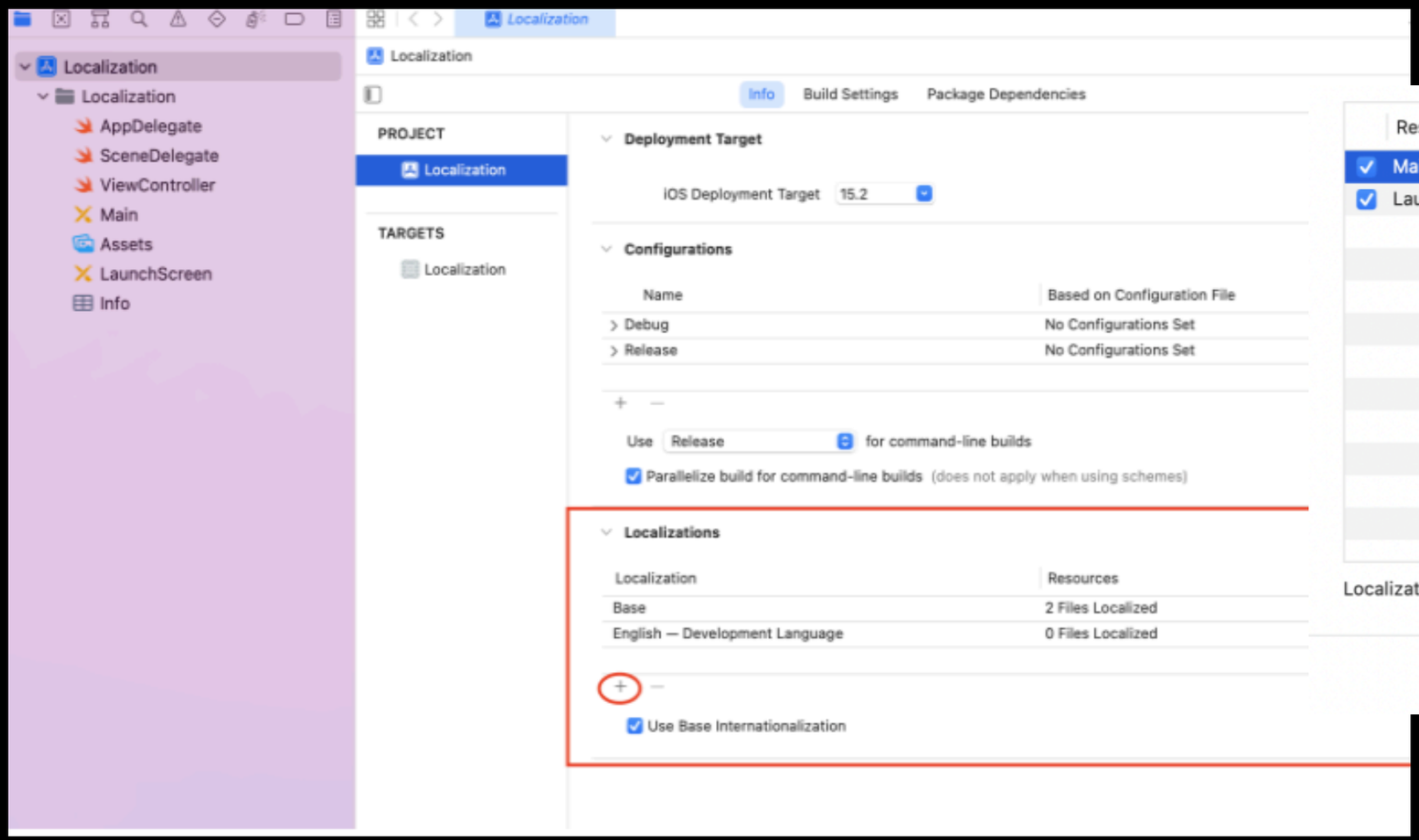
David Granado Jordan

Class 18

David Granado Jordan

Support multiple languages

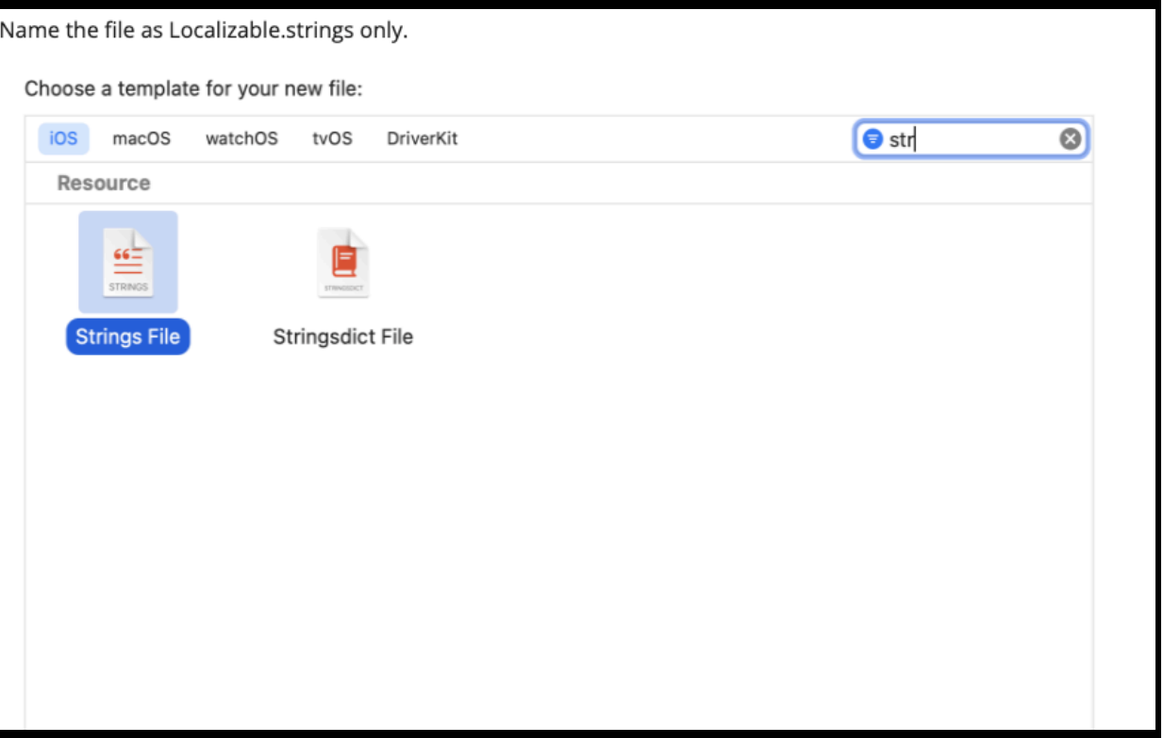
Add Localizable.strings

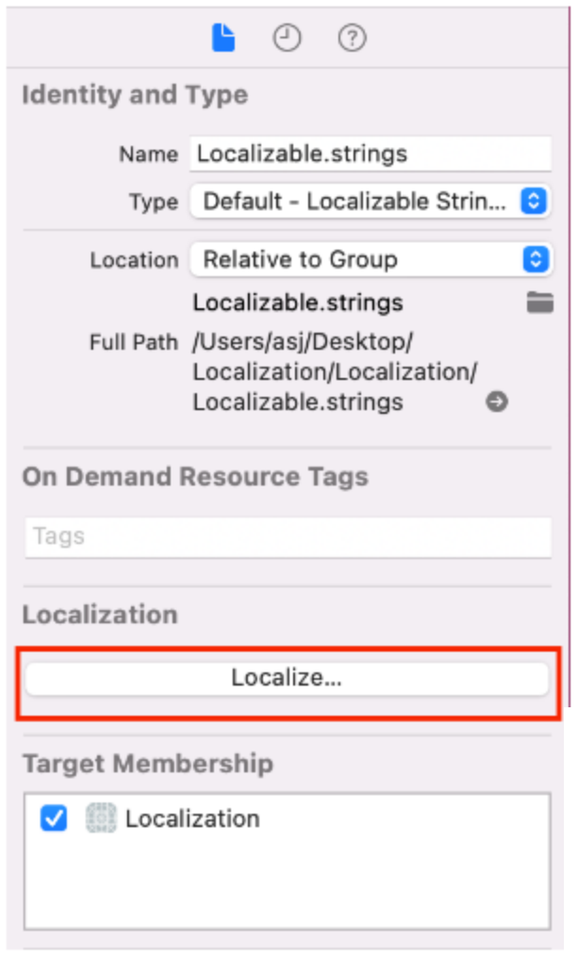


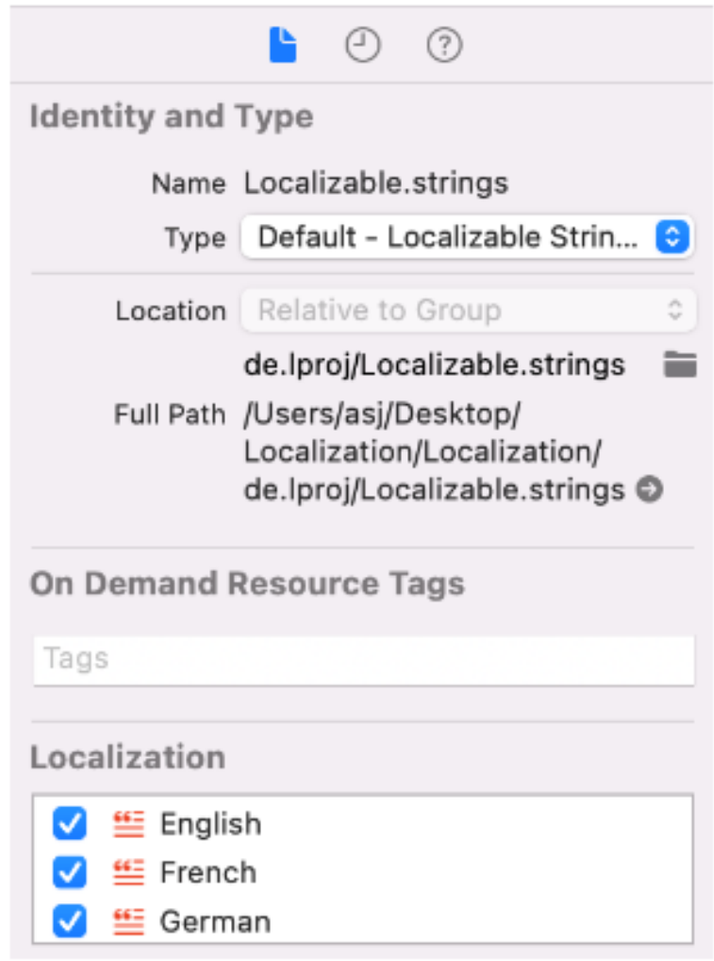
David Granado Jordan

Support multiple languages

Add Localizable.strings







Select all Localization Languages

Class 18

Support multiple languages

Create Helper managers

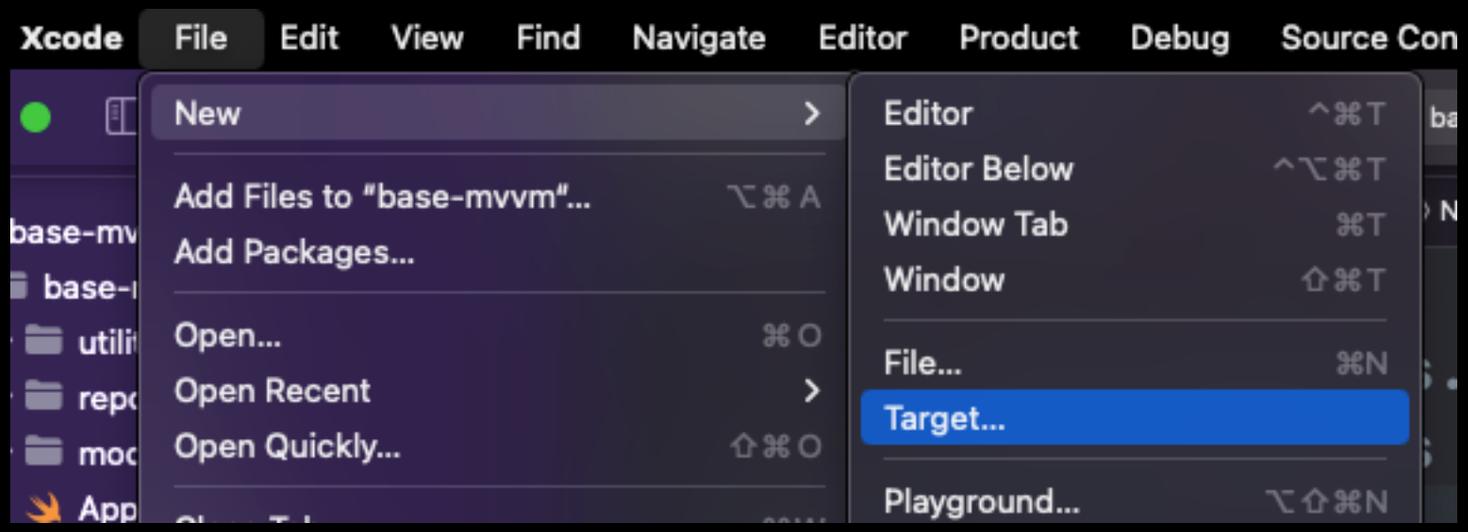
```
import Foundation

extension String {
    static func localizeString(localizedString: String) -> String {
        return NSLocalizedString(localizedString, comment: "")
    }

    static func emailAlreadyExists(localizedString: String, forEmail
email: String) -> String {
        String.localizedStringWithFormat(
            NSLocalizedString(localizedString,
                              comment: "Alert message when adding an
existing email."),
            email
        )
    }
}

label.text = String.localizeString(localizedString: "info")
        label.text = String.emailAlreadyExists(localizedString:
"email_already_exists", forEmail: "test@test.com")
```

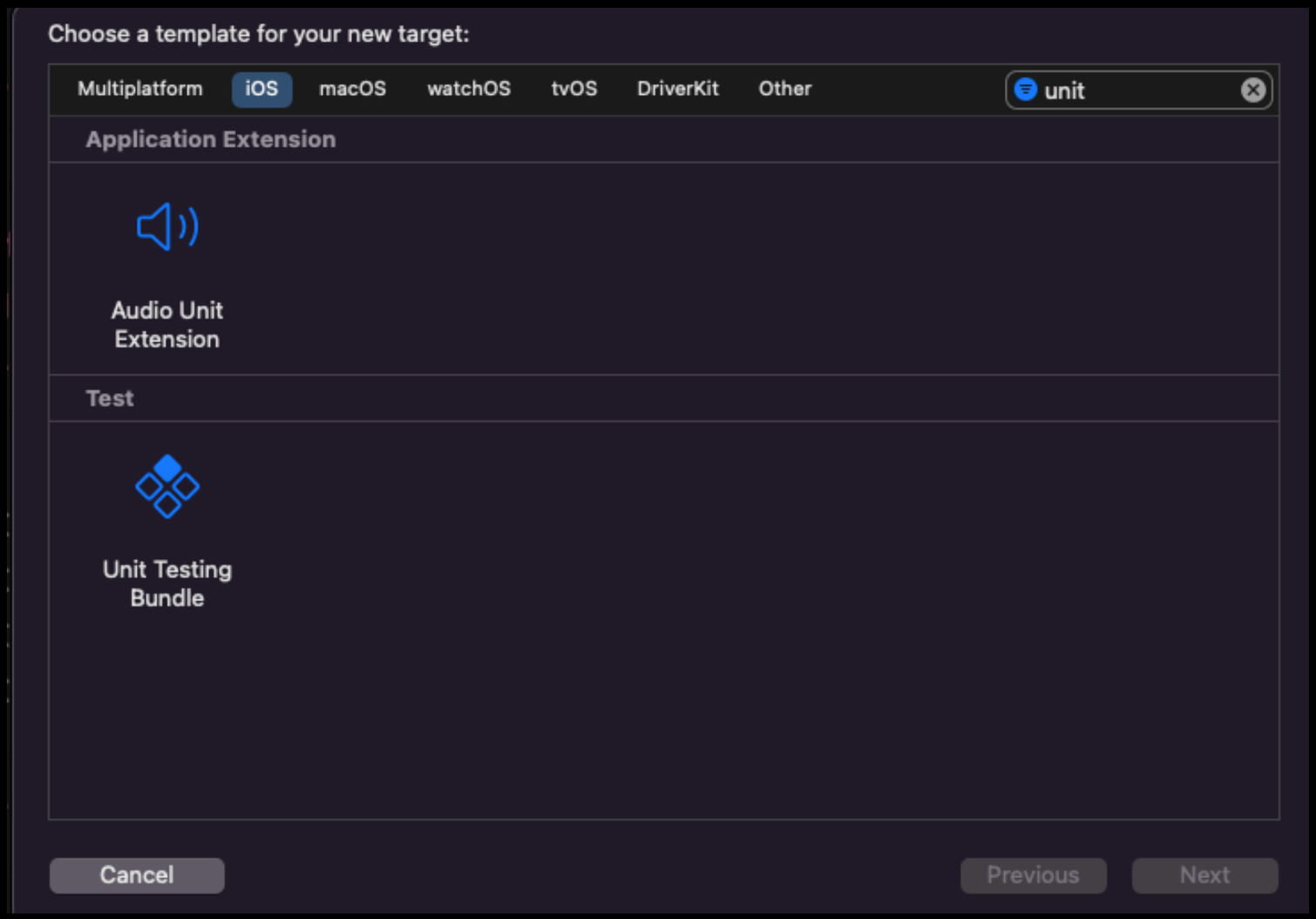
David Granado Jordan

Class 18

David Granado Jordan

Unit testing





David Granado Jordan

Unit testing

```swift
import XCTest
@testable import base_mvvm

class base_mvvmTests: XCTestCase {
    // System under test (SUT)
    var authFirebaseManager: AuthFirebaseManager!

    override func setUpWithError() throws {
        // Put setup code here. This method is called before the invocation of each test method in the class.
        authFirebaseManager = AuthFirebaseManager()
    }

    override func tearDownWithError() throws {
        // Put teardown code here. This method is called after the invocation of each test method in the class.
        authFirebaseManager = nil
    }

    func testInvalidLogin() throws {
        // given
        let invalidEmail = "123"
        let invalidPassword = "123"

        let expectation = self.expectation(description: "InvalidLogin")
        var error: Error?

        // when
        authFirebaseManager.login(email: invalidEmail, password: invalidPassword) { result in
            switch result {
            case .success(_):
                print("Success")
            case .failure(let err):
                error = err
            }
            expectation.fulfill()

        }

        wait(for: [expectation], timeout: 10)

        // then
        XCTAssertNotNil(error)
    }
```

Class 18

David Granado Jordan

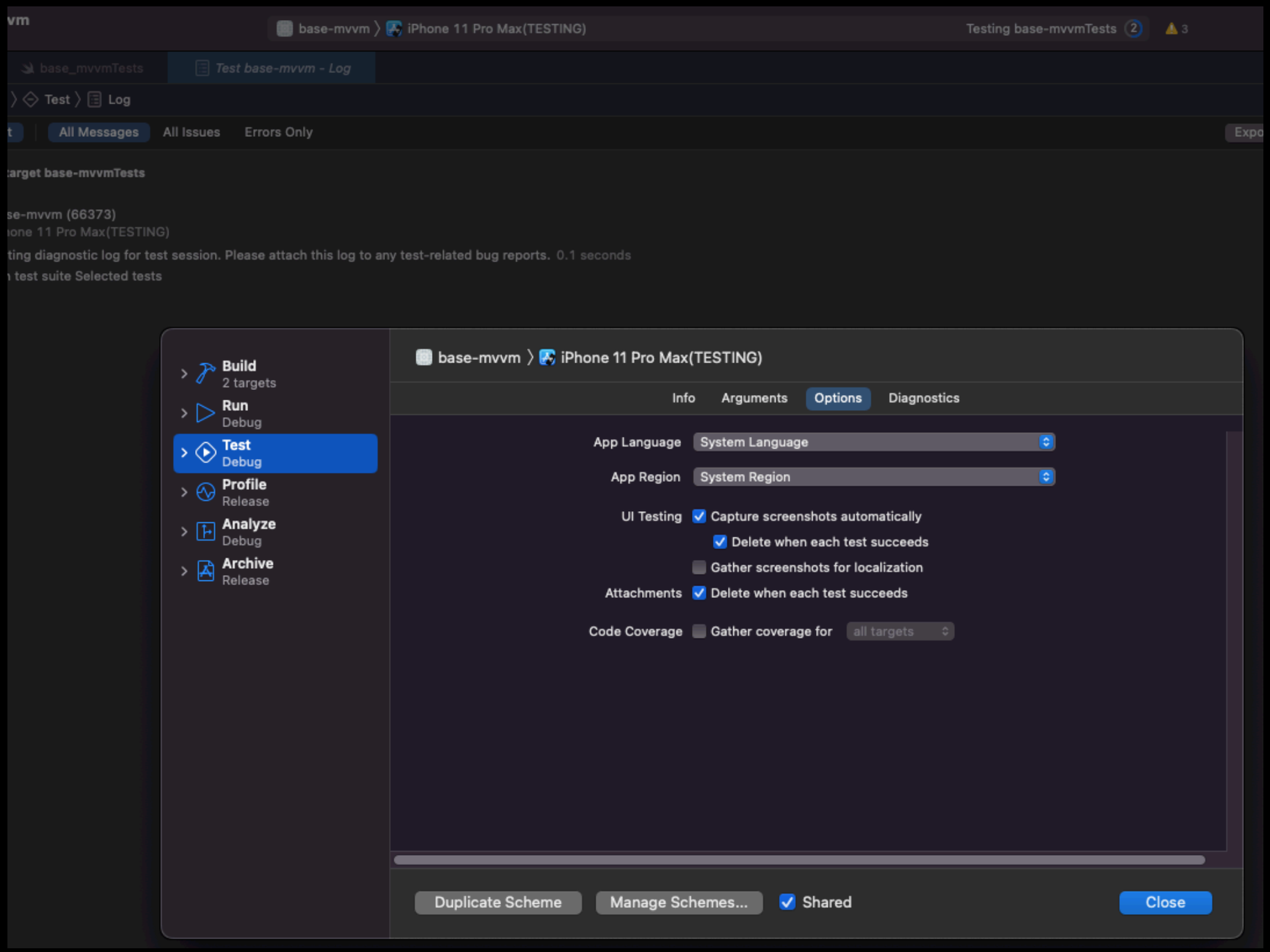
Unit testing

```
    func testValidLogin() throws {
        // given
        // MARK: - Make sure there is already a user created with this credentials
        let validEmail = "test1@test.com"
        let validPassword = "test123"

        let expectation = self.expectation(description: "ValidLogin")
        var error: Error?
        var user: User?
        // when
        authFirebaseManager.login(email: validEmail, password: validPassword) { result in
            switch result {
            case .success(let u):
                user = u
            case .failure(let err):
                error = err
            }
            expectation.fulfill()
        }

        wait(for: [expectation], timeout: 10)

        // Then
        // Test Valid user
        XCTAssertNotNil(user)
        // Test No errors
        XCTAssertNil(error)
    }

    func testPerformanceExample() throws {
        // This is an example of a performance test case.
        measure {
            // Put the code you want to measure the time of here.
            for index in 0...5000 {
            }
            XCTAssertEqual(true, true)
        }
    }

}
```

David Granado Jordan

Unit testing

Code coverage



David Granado Jordan

Class 18

Offline mode

1 - Add status property to Models.

```
 lastSyncTS: Date
 syncStatus: Int
```

```
Status:
SYNC_STATUS_NOT_SYNCED | SYNC_STATUS_PROCESSING |  SYNC_STATUS_COMPLETE |
SYNC_STATUS_FAILED
```

2 - If there is internet:

- Perform the request and change the object status to SYNC_STATUS_PROCESSING.

- After the request was completed change the object status to SYNC_STATUS_COMPLETE or SYNC_STATUS_FAILED according to the response.

- Update the lastSyncTS to current Date.

3 - If there is not internet:

- Just save the object in the Database and change the object status to SYNC_STATUS_NOT_SYNCED.

4 - Implement Manager to listen Internet Connection (ReachabilitySwift).:

- When the internet go back get all objects with: SYNC_STATUS_NOT_SYNCED, SYNC_STATUS_PROCESSING, SYNC_STATUS_FAILED.

- Perform the request and change the object status to SYNC_STATUS_PROCESSING.

David Granado Jordan

Class 18

David Granado Jordan

Swift UI

SwiftUI provides views, controls, and layout structures for declaring your app's user interface. The framework provides event handlers for delivering taps, gestures, and other types of input to your app, and tools to manage the flow of data from your app's models down to the views and controls that users will see and interact with.

https://developer.apple.com/xcode/swiftui/

https://developer.apple.com/tutorials/swiftui/creating-and-combining-views

https://developer.apple.com/documentation/swiftui/

David Granado Jordan

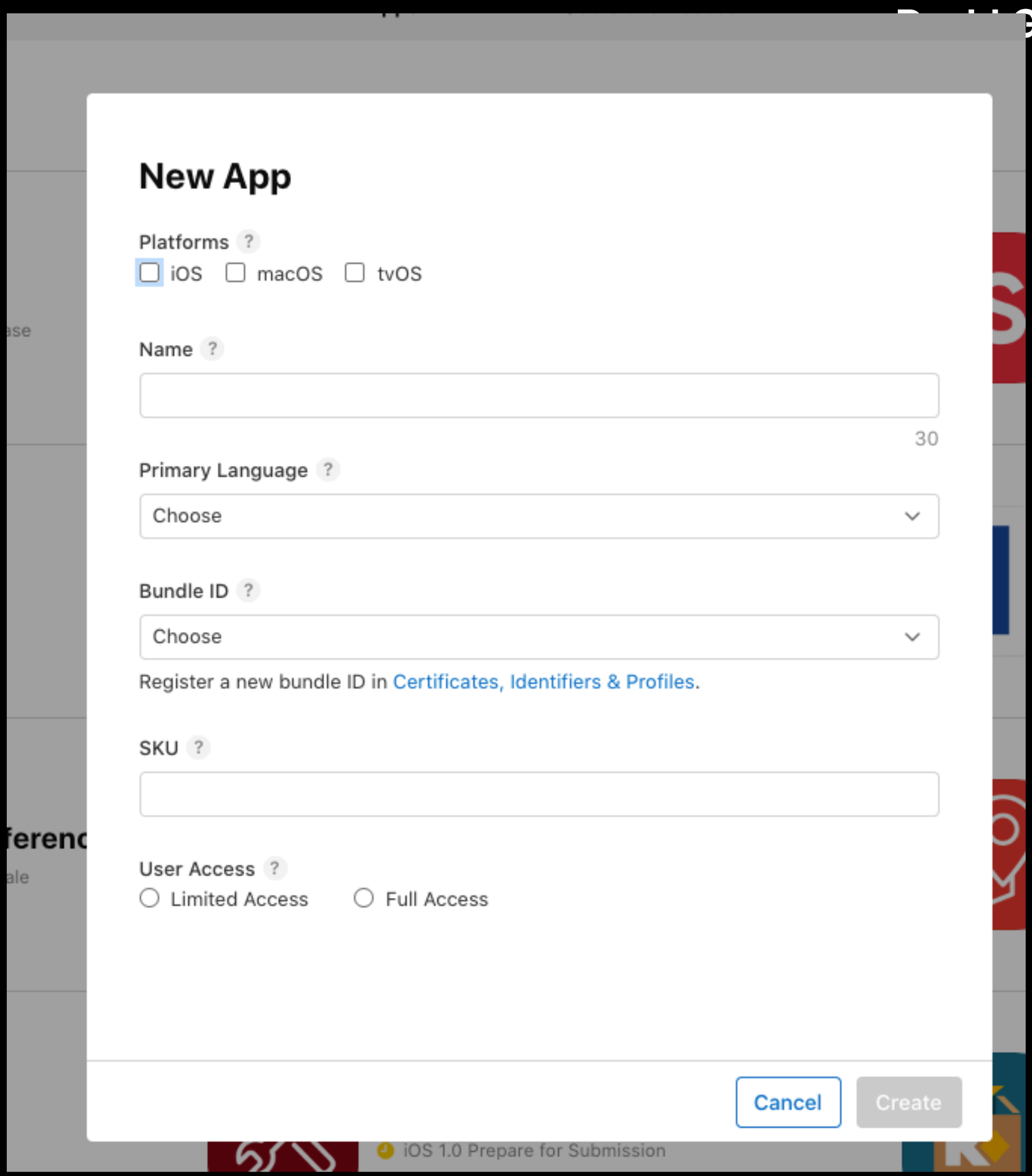
Class 18

- Testflight

TestFlight makes it easy to invite users to test your apps and App Clips and collect valuable feedback before releasing your apps on the App Store. You can invite up to 10,000 testers using just their email address or by sharing a public link.

- App Store Connect

As an Apple Developer Program member, you can easily upload, submit, and manage your apps on the App Store with App Store Connect on the web, iPhone, and iPad. This suite of tools also lets you view sales reports, access app analytics, invite users to test your apps with TestFlight, and much more.


New App
Platforms
iOS
macOS
tvOS
Name
30
Primary Language
Choose
Bundle ID
Choose
Register a new bundle ID in Certificates, Identifiers & Profiles.
SKU
User Access
Limited Access
Full Access
Cancel
Create
iOS 1.0 Prepare for Submission

App Store Services TestFlight Xcode Cloud

iOS App

1.0.11 Ready for Sale

macOS App

1.0 Prepare for Submission

Add tvOS App

General

App Information

Pricing and Availability

App Privacy

Ratings and Reviews

Version Information

App Review

iOS App 1.0.11

Version Information

The product page for this app version will be published on the App Store with the assets and metadata below.

App Previews and Screensh

iPhone 6.5" Display

New Version

Store Version Number ?

64

Cancel Create

App Store | Services | TestFlight | Xcode Cloud

iOS App

- 22.1.0 Pending Developer R...
- 22.0.4 Ready for Sale

Add macOS App

Add tvOS App

General

- App Information
- Pricing and Availability
- App Privacy
- Ratings and Reviews
- Version History
- App Review

Features

- In-App Purchases
- Subscriptions
- App Store Promotions
- Custom Product Pages
- In-App Events
- Product Page Optimization

iOS App 22.1.0

Save | Release This Version

You can only edit some information while your version is pending developer release. To edit all information, cancel this release.

Version Information

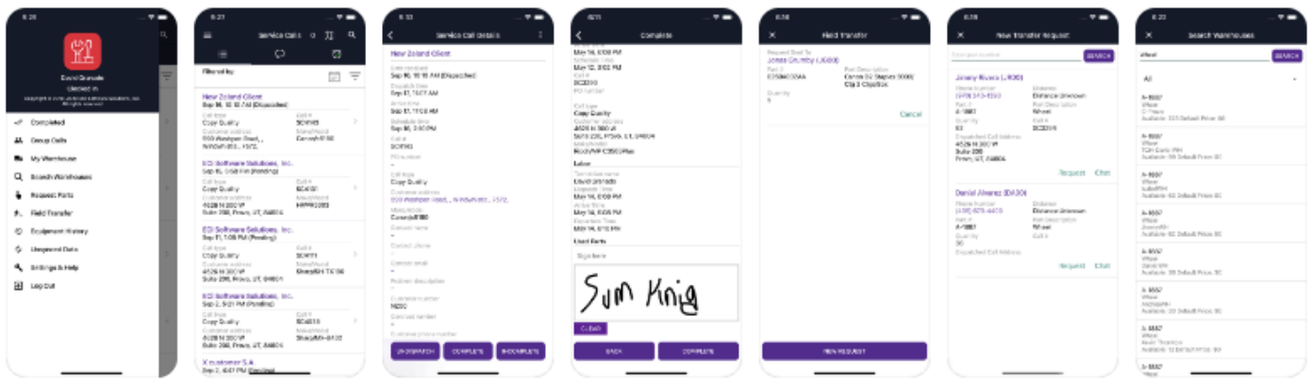
English (U.S.)

The product page for this app version will be published on the App Store with the assets and metadata below.

App Previews and Screenshots

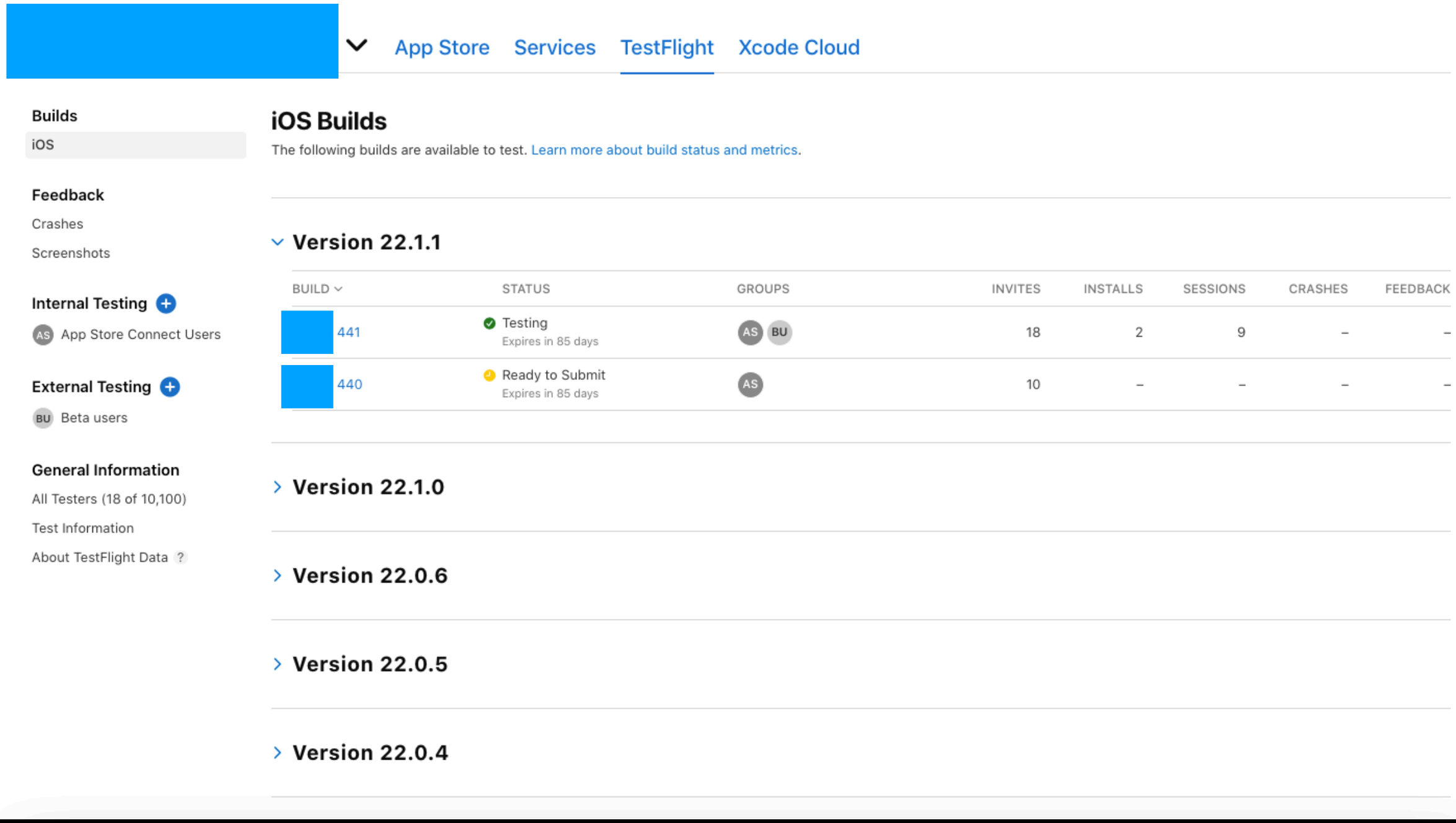
View All Sizes in Media Manager

iPhone 6.5" Display | iPhone 5.5" Display | iPad Pro (3rd Gen) 12.9" Display | iPad Pro (2nd Gen) 12.9" Display



Promotional Text Edit

David Granado Jordan



David Granado Jordan

App Store | Services | TestFlight | Xcode Cloud

< iOS Builds

22.1.1 (441)

Expire Build | Save

Test Information | Build Metadata

English (U.S.)

Test Details

What to Test ?

Let your testers know what you would like them to test in this build. This information will be available to testers in all groups who have access to this build.

Items addressed: FSD02-419 with new gaps style.

3,953

Groups (2)

Add groups to this build. All testers in these groups will have access to this build.

GROUP NAME	GROUP TYPE	TESTERS
Beta users	External	10

Individual Testers (0)

Add an individual tester to this specific build.

No testers have been added for this build.

David Granado Jordan

Class 18

David Granado Jordan

Coordinator Pattern

Class 18

David Granado Jordan

Class 18