



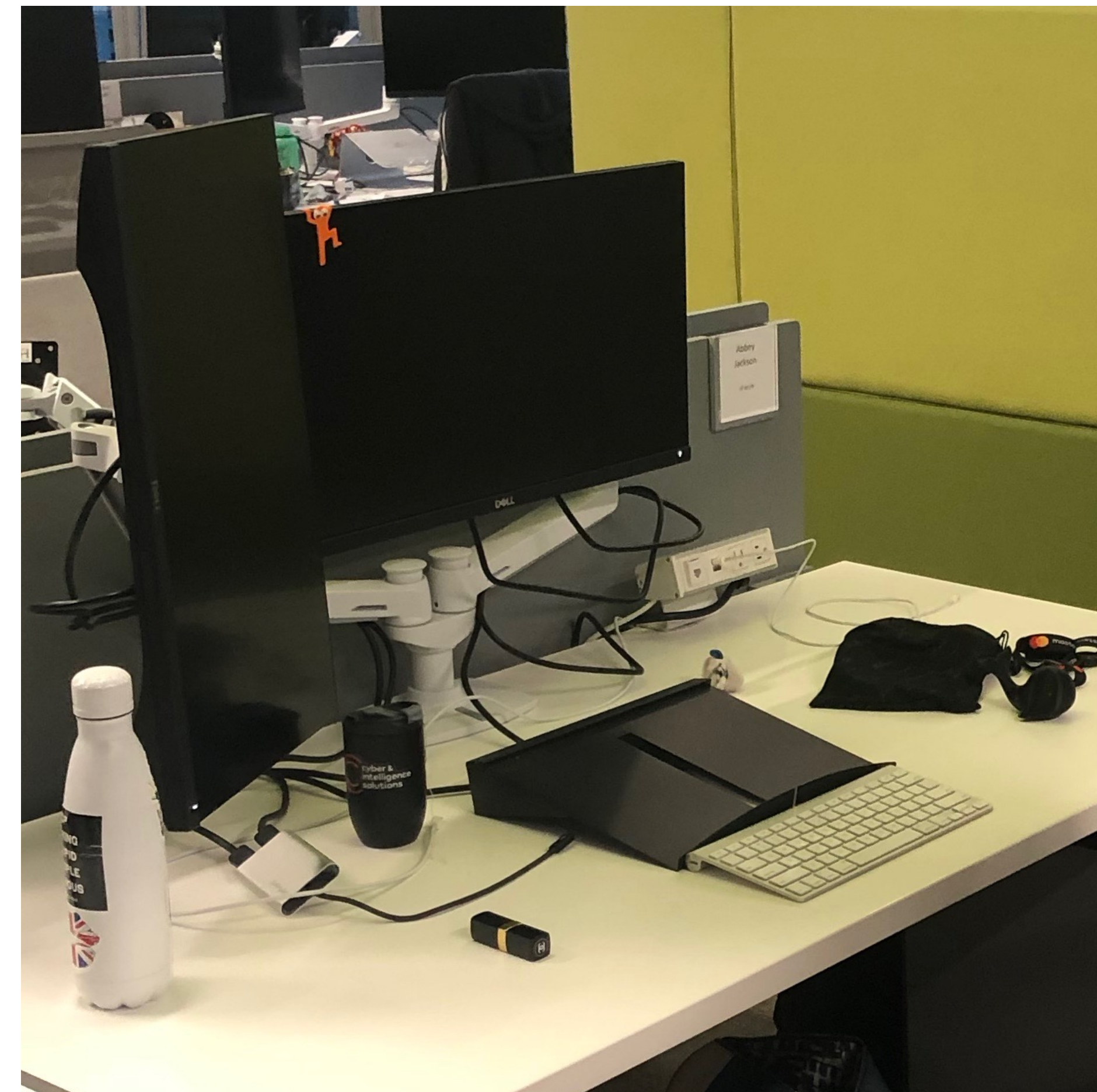
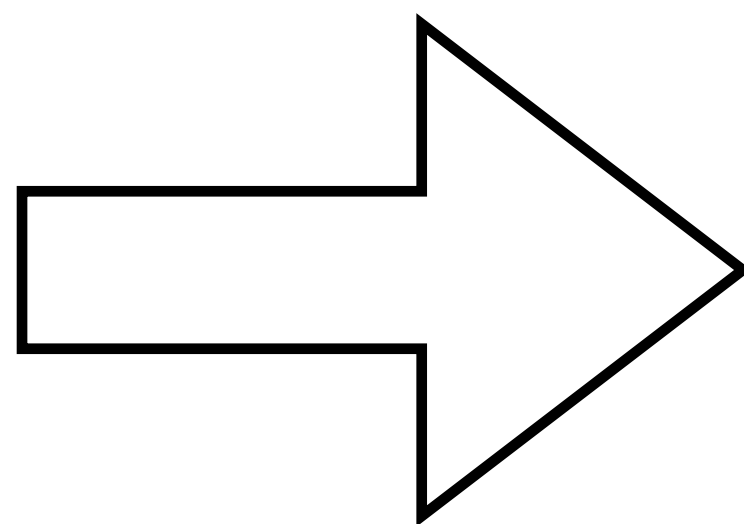
**my name is**

**Abbey  
Jackson**





**2015**



**2019**

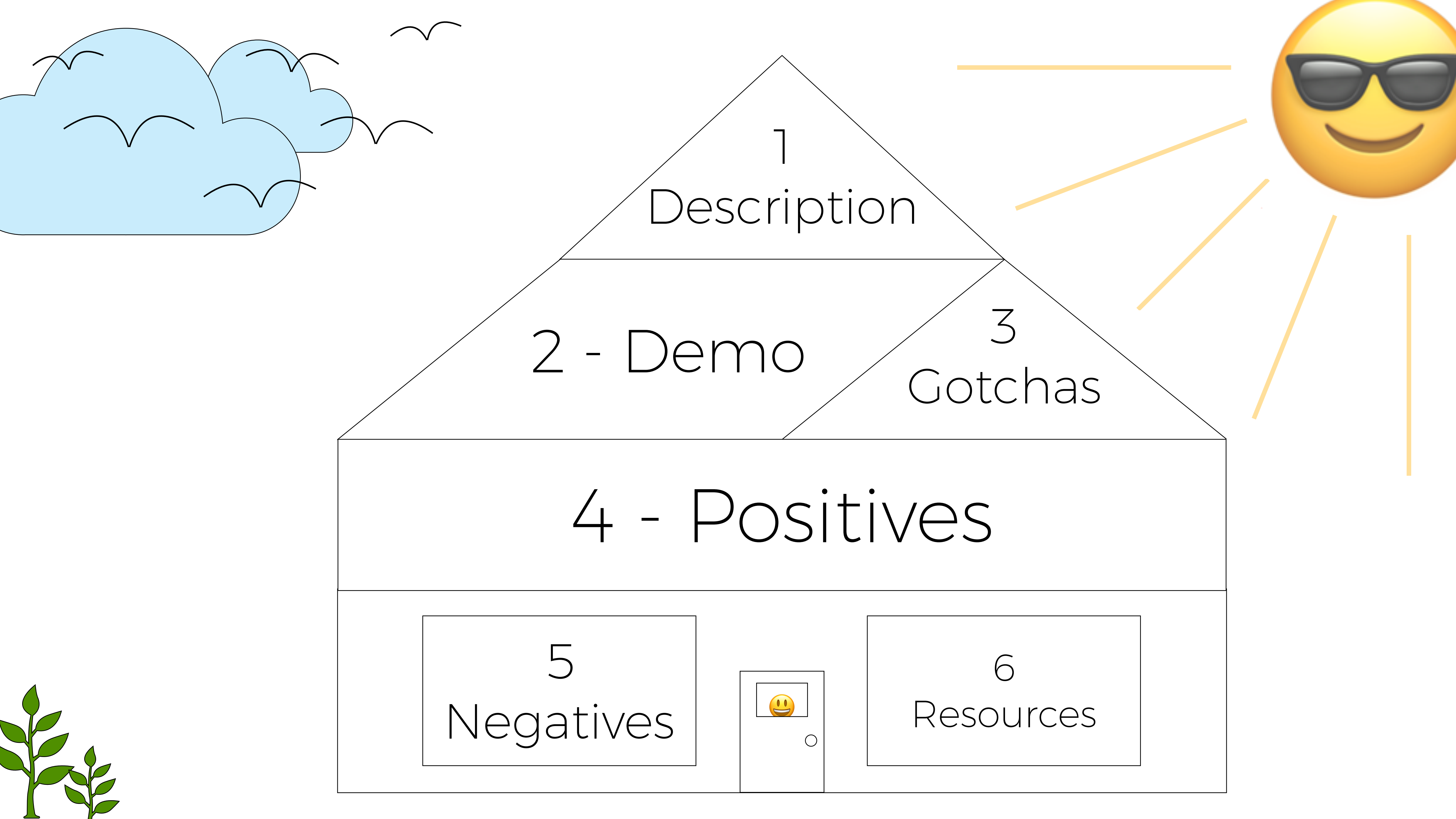


**MODULARIZE  
ALL THE THINGS!**



 @earthabbey

 @abbeyjackson





# **What is Modular App Development?**







**Networking**

**User Data**

**User Interface**

**Authentication**

**Profile Info**

Profile

Request

**Location**

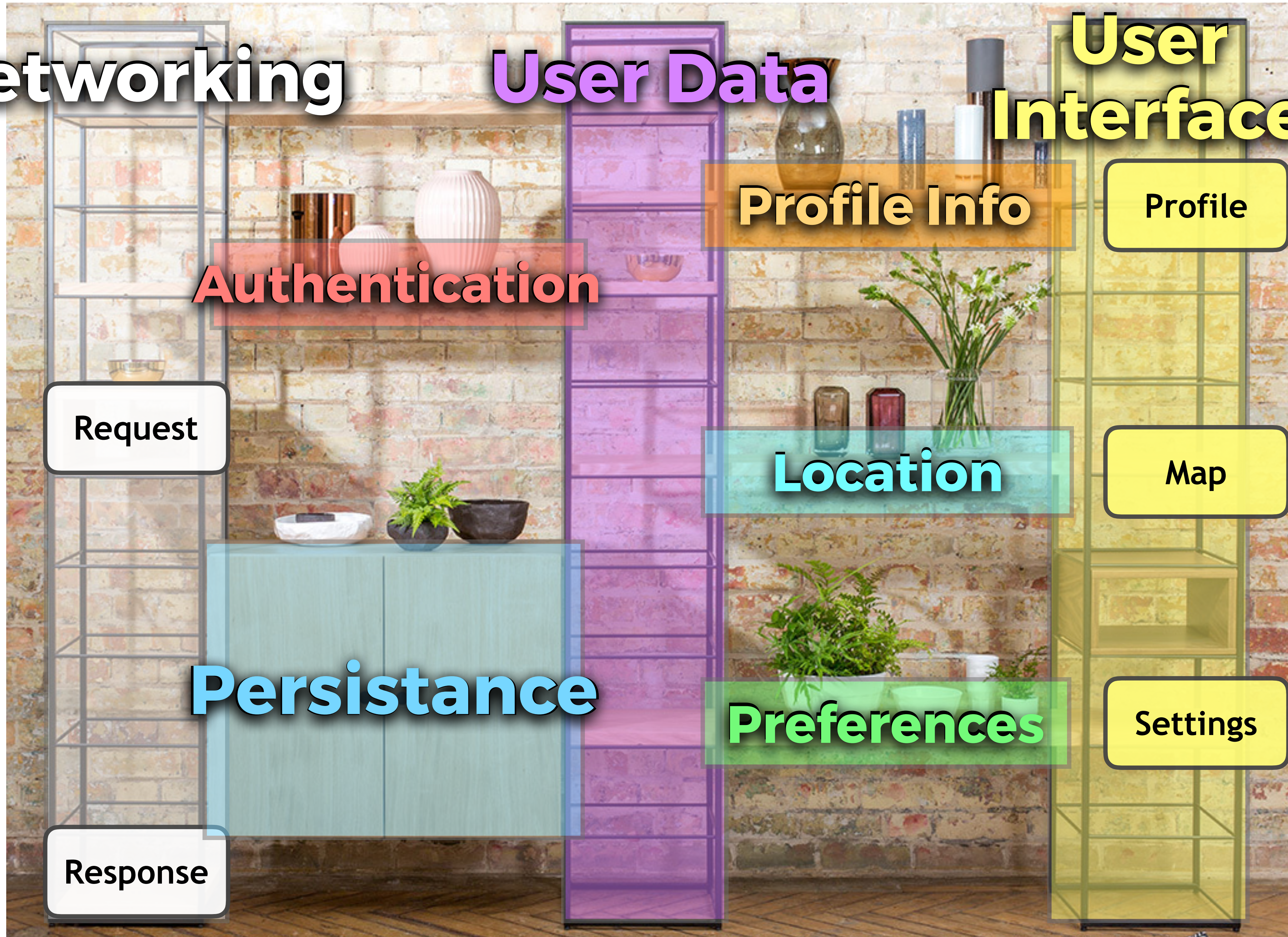
Map

**Persistance**

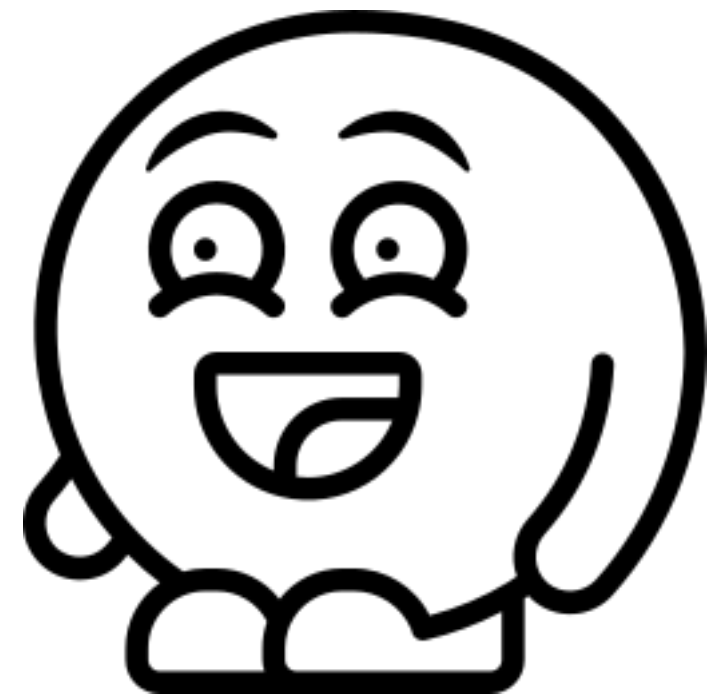
**Preferences**

Settings

Response







# **How to modularize your projects**



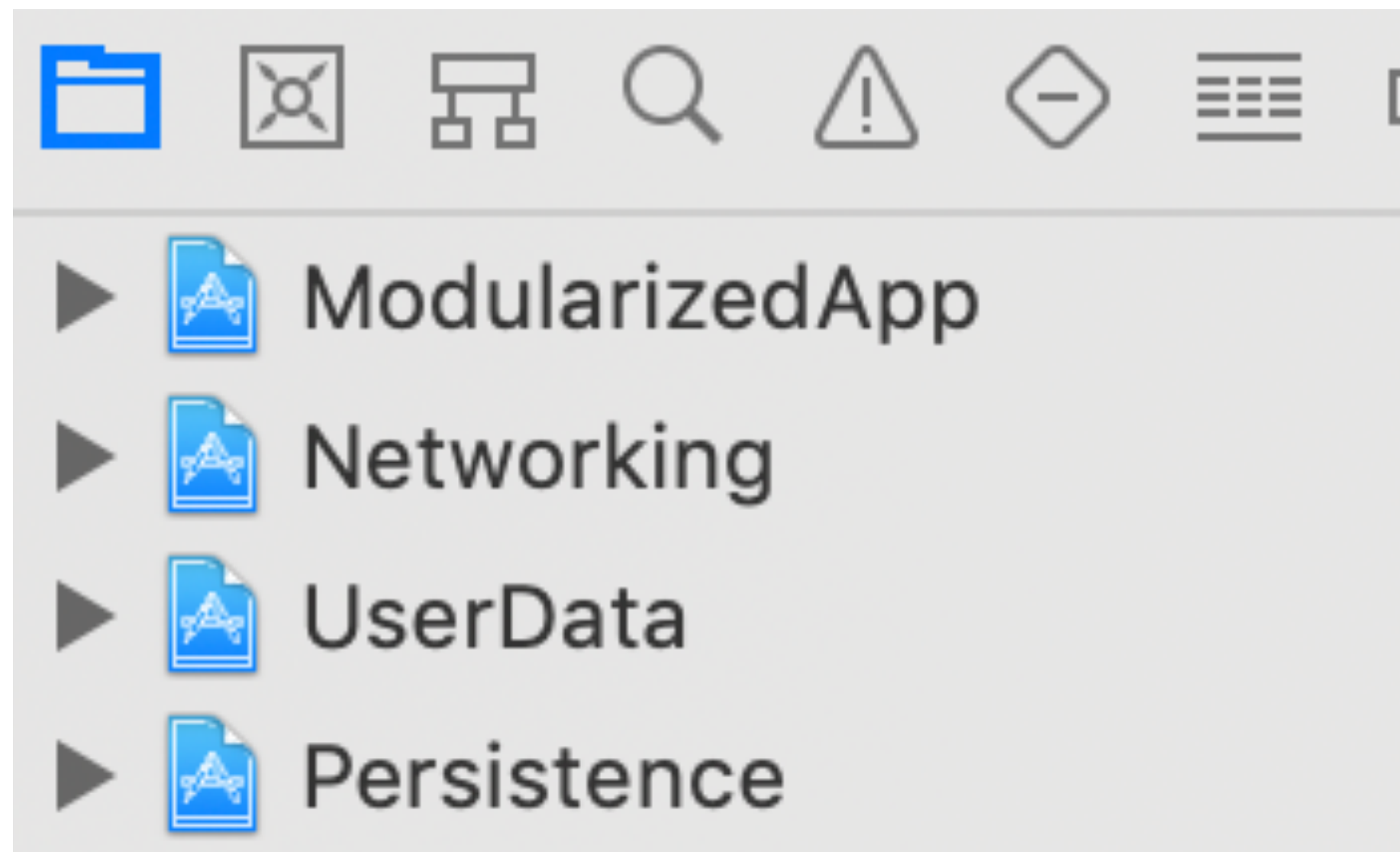
**Dynamic or Static?**



# Set Up Multiple...





There are two different approaches you can take

... projects?



... targets?

## TARGETS

-  ModularizedApp
-  Persistence
-  Networking
-  UserData



## ... projects?

Can open separately

Files can only belong  
to one project

Visual separation in  
filetree is automatic

Build settings are not  
related to parent app



## ... targets?

Can not open separately

Files can belong to  
multiple targets

Visual separation has to  
be maintained manually

Build settings can be  
affected by inheritance



# Demo - A working project

1 Set up your workspace

2 Link your frameworks

3 Use your frameworks

4 Add a new framework

5 Use that framework



# Example: Passing Data

## (protocol method)

```
public protocol DataObjectProtocol {
    var objectId: String {get}
    var objectData: Data? {get}
}

public struct DataObject: DataObjectProtocol { ... }

public class Database {
    public func store(_ object: DataObject) {
        objects[object.objectId] = object
    }
    public func get(_ objectId: String) -> DataObjectProtocol? {
        return objects[objectId]
    }
}
```

Persistence



# Example: Passing Data

## (protocol method)

Modularized App

```
import Persistence
import UserData

extension User: DataObjectProtocol {
    public var objectId: String { ... }
    public var objectData: Data? { ... }
}

let user = User( ... )
let userDataObject = DataObject(objectId: user.objectId,
                                objectData: user.objectData)

let database = Database( ... )
database.store(userDataObject)
```



# Example: Passing Data

Protocols

Delegates

Reactive Programming

Property Observers





**Where to  
look to avoid  
moments like this**





# Not Embedding the Binary






## ▼ Embedded Binaries

 Persistence.framework ...in build/Debug-iphoneos

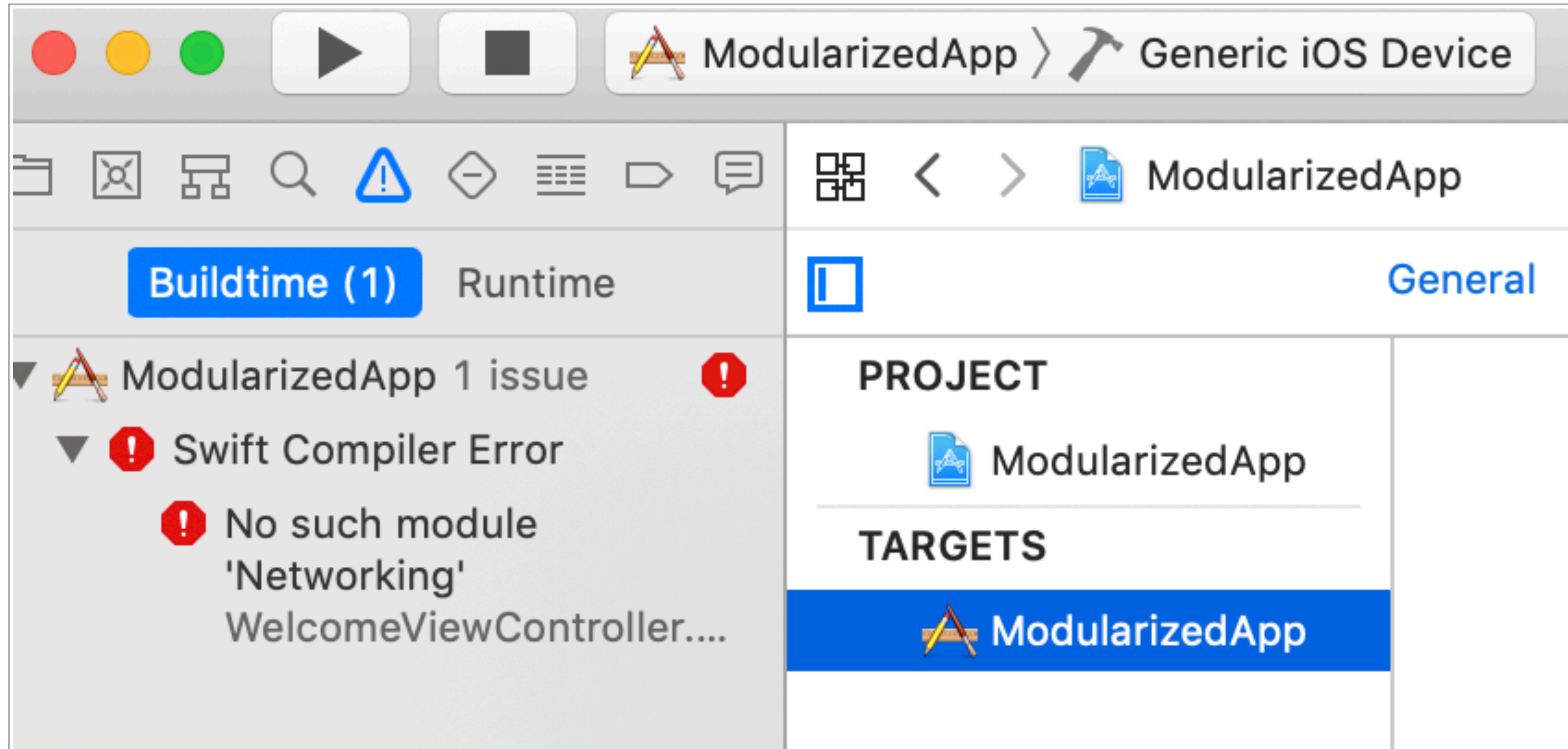
 UserData.framework ...in build/Debug-iphoneos



## ▼ Linked Frameworks and Libraries

Name	Status
 Persistence.framework	Required ⌵
 Networking.framework	Required ⌵
 UserData.framework	Required ⌵
+ -	







Not Embedding the Binary

Not Linking in Target Dependencies



▼ **Target Dependencies (2 items)**



Persistence (Persistence)



UserData (UserData)



```
private func getUserResponse() ->
    UserResponseData? {

    let profileDictionary: [String: Any] = [
        "firstName": "Abbey" as AnyObject,
        "lastName": "Bob" as AnyObject,
        "email": "billy@bob.com" as AnyObject
    ]

    let locationDictionary: [String: Any] = [
        "address": "123 Smith Street" as AnyObject,
        "city": "Toronto" as AnyObject,
```

12:42

< Back

Billy Bob

123 Smith



Not Embedding the Binary

Not Linking in Target Dependencies

Using Common Module Name

Basic

Customized

All

Combined

Levels



Q name

## ▼ Packaging

Setting



Networking

▶ **Product Module Name**

**Networking**

**Product Name**

**Networking**

Strings File Output Encoding

`$(PRODUCT_NAME:c99extidentifier)`

## ▼ Search Paths

Setting



Basic

Customized

All

Combined

Levels



Q name

▼ Packaging

Setting



Networking

▶ Product Module Name

Networking

Product Name

Networkinga

Strings File Output Encoding

\$(APP\_PREFIX)|\$(PRODUCT\_NAME:c99extid

▼ Search Paths

Not Embedding the Binary

Not Linking in Target Dependencies



Using Common Module Name










Test Target Greyed Out




▶

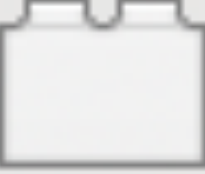

■

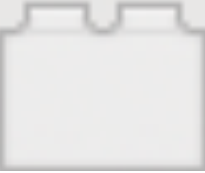
 ModularizedApp > 







< >

▶  ModularizedAppTests 1 test 



▼  PersistenceTests 1 test










1 


2 

▶


■


 Persistence >  if







< >

▶  ModularizedAppTests 1 test

▼  PersistenceTests 1 test

1 

2 



Persistence >



iPhone XR

▶  **Build**  
1 target

▶  **Run**  
Debug

▶  **Test**  
Debug

▶  **Profile**  
Release

▶  **Analyze**  
Debug

▶  **Archive**  
Release

Info







Build Configur

De



Debug Proc







Tests




 <b>Build</b> 1 target	<a href="#">Info</a>
 <b>Run</b> Debug	Build Configur
 <b>Test</b> Debug	Deb
 <b>Profile</b> Release	
 <b>Analyze</b> Debug	
 <b>Archive</b> Release	

Tests

D  ModularizedApp >  iPhone XR

 <b>Build</b> 2 targets	<a href="#">Info</a>
 <b>Run</b> Debug	Build Configur
 <b>Test</b> Debug	Debi
 <b>Profile</b> Release	Debug Proce
 <b>Analyze</b> Debug	
 <b>Archive</b> Release	

Tests

 ModularizedAppTests

Not Embedding the Binary

Not Linking in Target Dependencies

Using Common Module Name

Test Target Greyed Out

Violating Proper Unit Testing Principles



Persistence

GeneralResource TagsInfoBuild SettingsBuild PhasesBuild

OBJECT

Persistence

Targets

Persistence

PersistenceTests

Filter

▼ Target Dependencies (1 item)

Persistence (Persistence)

+

—

► Compile Sources (1 item)

x

▼ Link Binary With Libraries (2 items)

x

Name	Status
<div>Networking.framework</div> Req... <div>⌵</div>	
<div>Persistence.framework</div> Req... <div>⌵</div>	
<div>+</div> Drag to reorder framew...	

Manual > Pe...ce > Pe...sts > PersistenceTests.swift > M testT

1

//

2

// PersistenceTests.swift

3

// PersistenceTests

4

//

5

// Created by Abbey Jackson on 2019-08-12.

6

// Copyright © 2019 Abbey Jackson. All rights reserved.

7

//

8

9

import XCTest

10

@testable import Persistence

11

@testable import Networking

12

13

✓ class PersistenceTests: XCTestCase {

14

15

✓ func testTheInternalsOfNetworking() {

16

let userResponse1 = Networking().getUserResponse1()

17

let userResponse2 = Networking().getUserResponse2()

18

XCTAssertEqual(userResponse1!.dataDictionary.count,

19

userResponse2!.dataDictionary.count)

19

}

20

21

}

Not Embedding the Binary

Not Linking in Target Dependencies

Using Common Module Name

Test Target Greyed Out

Violating Proper Unit Testing Principles

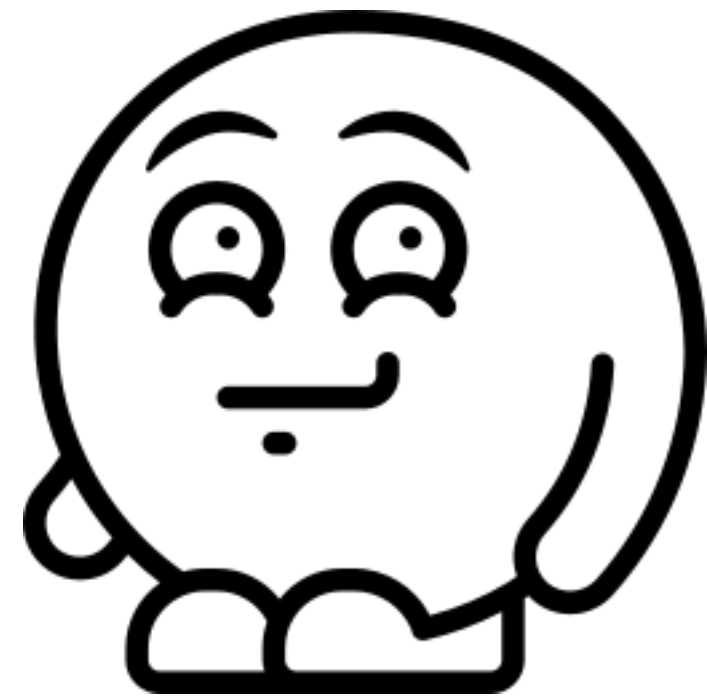
Inconsistency Between Targets



iOS Deployment Target

Swift Version (if < 5.0)

Active Architectures



**Why might  
you want to  
modularize?**



# Wide-Spread Benefits

Modularizing your app can affect more than just your developers

**Business**



**Product**



**Development**



Lower Tooling  
Costs

Less Developer  
Resourcing

Faster  
Releases



Improved  
Communication

Happier  
Clients



# Wide-Spread Benefits

Modularizing your app can affect more than just your developers

**Business**



**Product**



**Development**



Better  
Reviews

Improved  
Communication

Less Bugs

Faster  
Fixes

Less Support  
Resourcing

Fewer  
Regressions

Happier  
Clients



# Wide-Spread Benefits

Modularizing your app can affect more than just your developers

**Business**



**Product**



**Development**





Increased  
Code Integrity

Better  
Sense of  
Ownership

Uncoupled  
Dependencies

Easier  
Debugging

Pipeline  
Improvements

Enforce  
Best Practices

Improved  
Testing

Faster  
Buildtimes

Better  
Metrics



**ALL THE THINGS!**



# Wide-Spread Benefits

Modularizing your app can affect more than just your developers

**Business**



**Product**



**Development**







**When is  
modularization  
not so great?**

Handling Assets

Where to put stuff

Separate Project Settings

Sharing UI

**MODULARIZE  
ALL THE THINGS!**





# MODULARIZE ALL THE THINGS!



## Sample Code

[Modularized Workspace](https://abgl.ca/mod-basic) ([abgl.ca/mod-basic](https://abgl.ca/mod-basic))

Please read the README.md that comes with the sample code as it explains some additional things.

The sample code contains a workspace with an app linked to modules. It also demonstrates passing data and additionally demonstrates the testing issue I mentioned to watch out for.

## Recordings

[All Available On My Website](https://abgl.ca/talks) ([abgl.ca/talks](https://abgl.ca/talks))

 @earthabbey

 @abbeyjackson

MODULARIZE  
ALL THE THINGS!



@earthabbey



@abbeyjackson

## Further Reading

It's always a good idea to learn from those who have come before you, even if everything is working out great and you feel confident about your set up!

- [Modularising iOS apps into powerful reusable kits](#), from *Zomato* (abgl.ca/mod-1)
- [Modular iOS Part 1: Strangling the Monolith](#), from *Sam Dods* (abgl.ca/mod-2)
  - [Modular iOS Part 2: Splitting a Workspace Into Modules](#), from *Sam Dods* (abgl.ca/mod-3)
  - [Modular iOS Part 3: Configuration & Testing of Modules](#), from *Sam Dods* (abgl.ca/mod-4)
  - [Modular iOS Part 4: Sharing Configuration Between Modules](#), from *Sam Dods* (abgl.ca/mod-5)
- [Our iOS modularization story](#), from *wehkamp* (abgl.ca/mod-6)
- [Using CocoaPods to Modularize a Big iOS App](#), from *Hubspot* (abgl.ca/mod-7)
  - [Update: How We're Building iOS Apps Today](#), from *Hubspot* (abgl.ca/mod-8)
- [iOS at Scale: Modularization of the Wayfair App](#), from *Wayfair* (abgl.ca/mod-9)
- [Building a Composable, Flexible, and Functional Layout in our iOS App](#), from *Wayfair* (abgl.ca/mod-10)
- [How to create bridges between frameworks in an iOS app](#), from Rayane Kurrimboccus (abgl.ca/mod-11)
- And [this SO post](#) which shows the kinds of confusing troubles you can get into (abgl.ca/mod-12)