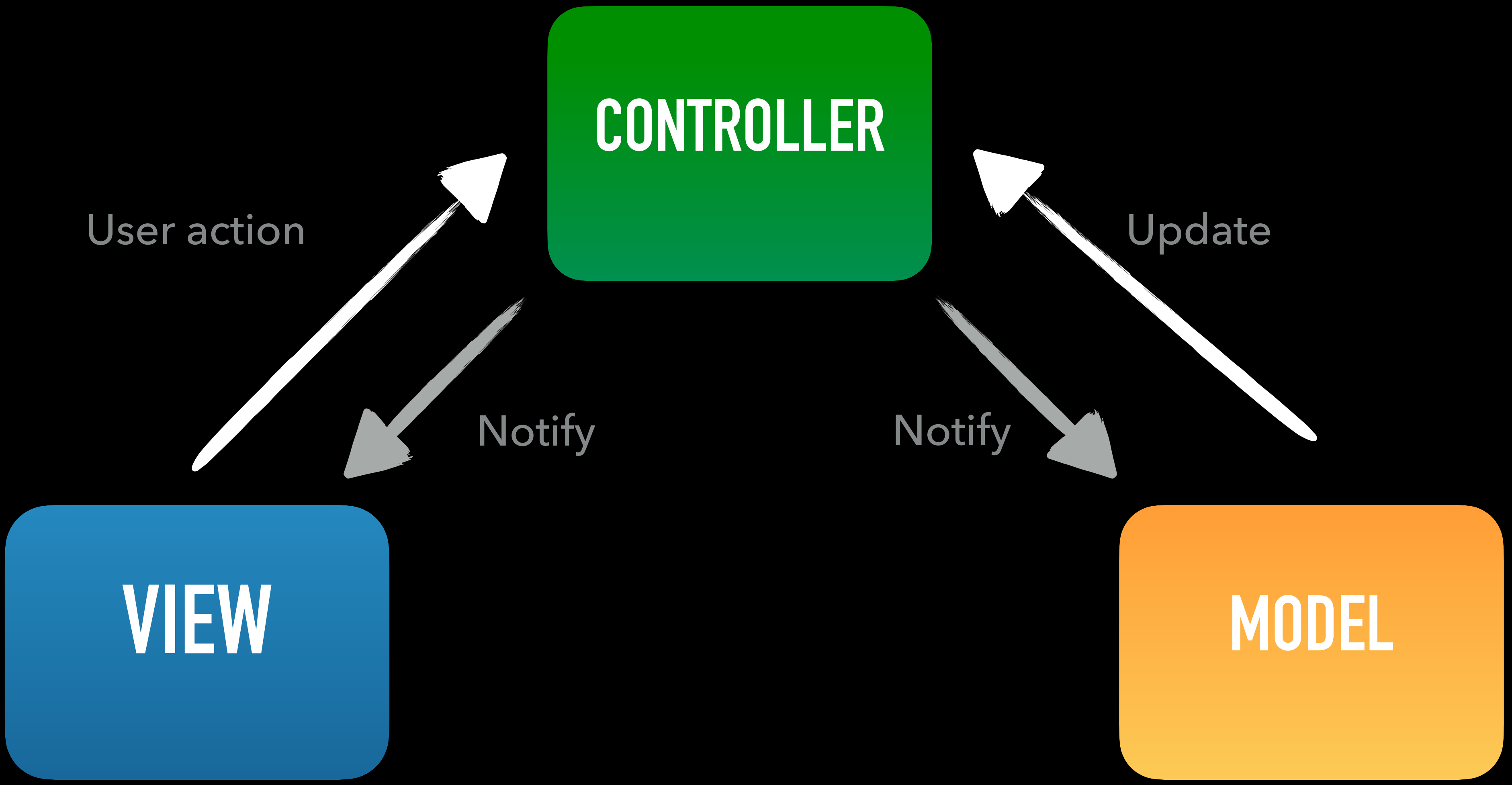
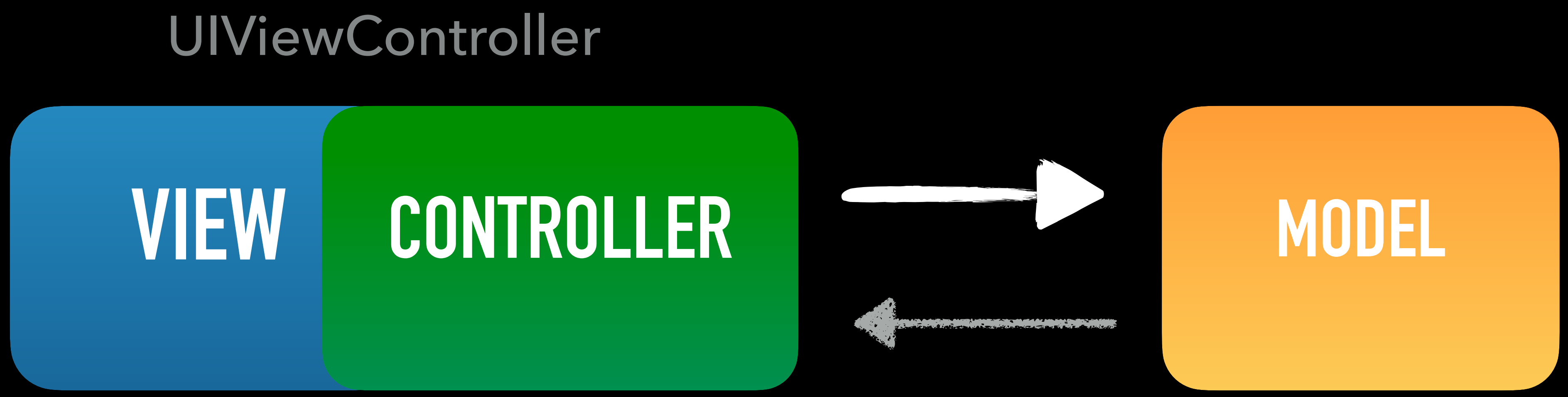
MVVM In Practice

Iqbal Ansyori (iOS Dev @Traveloka)

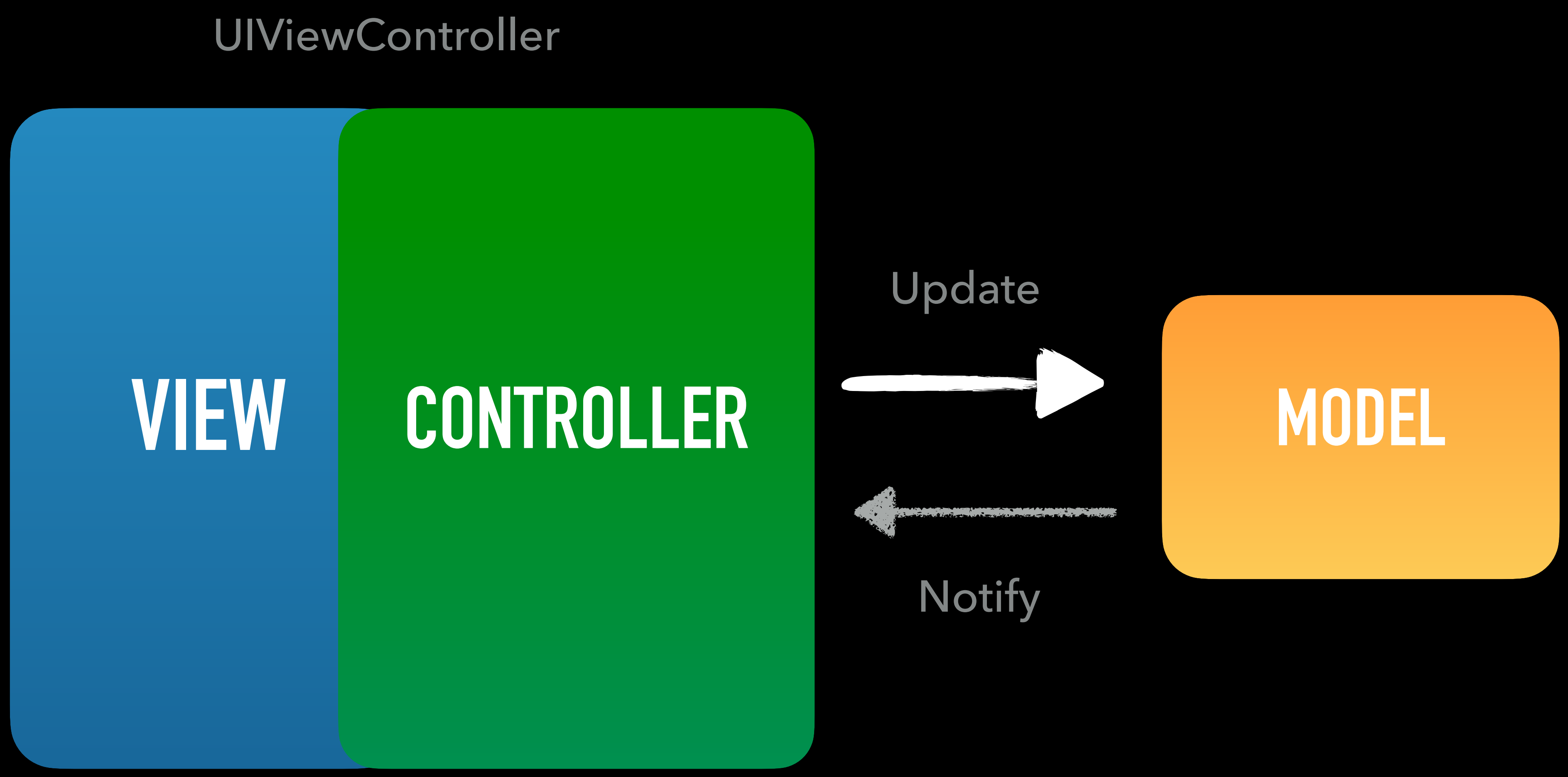


Apple’s MVC

https://developer.apple.com/library/archive/documentation/General/Conceptual/DevPedia-CocoaCore/MVC.html



Real Apple's MVC



MVC

Massive View Controller

VIEW CONTROLLER

Everything except model

Layout & display logic

Data transformation

Network fetcher

Screen state

UI Behaviour

Navigation

...

MODEL

Data Representation

JSON Parser

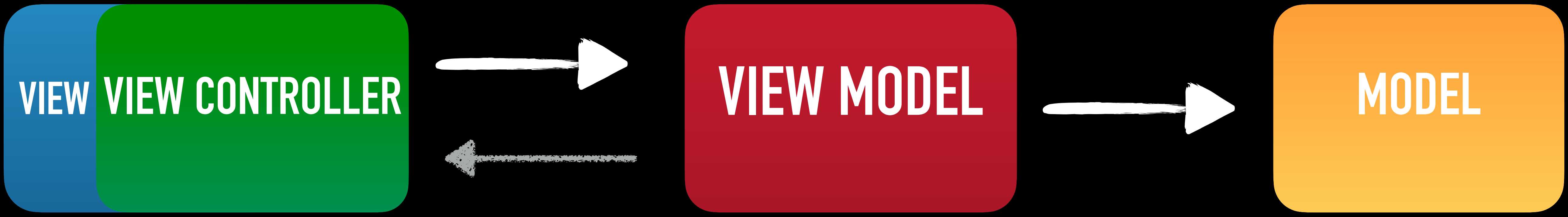
MVC ✗

High complexity UIViewController

Poor reusability

Poor testability

UH, HELLO, ANYONE?
HELP?!
imgflip.com



https://www.objc.io/issues/13-architecture/mvvm/

VIEW CONTROLLER

Layout & display logic

UI Behaviour

Navigation

Presentation layer

VIEW MODEL

Data transformation

Screen state

Business logic layer

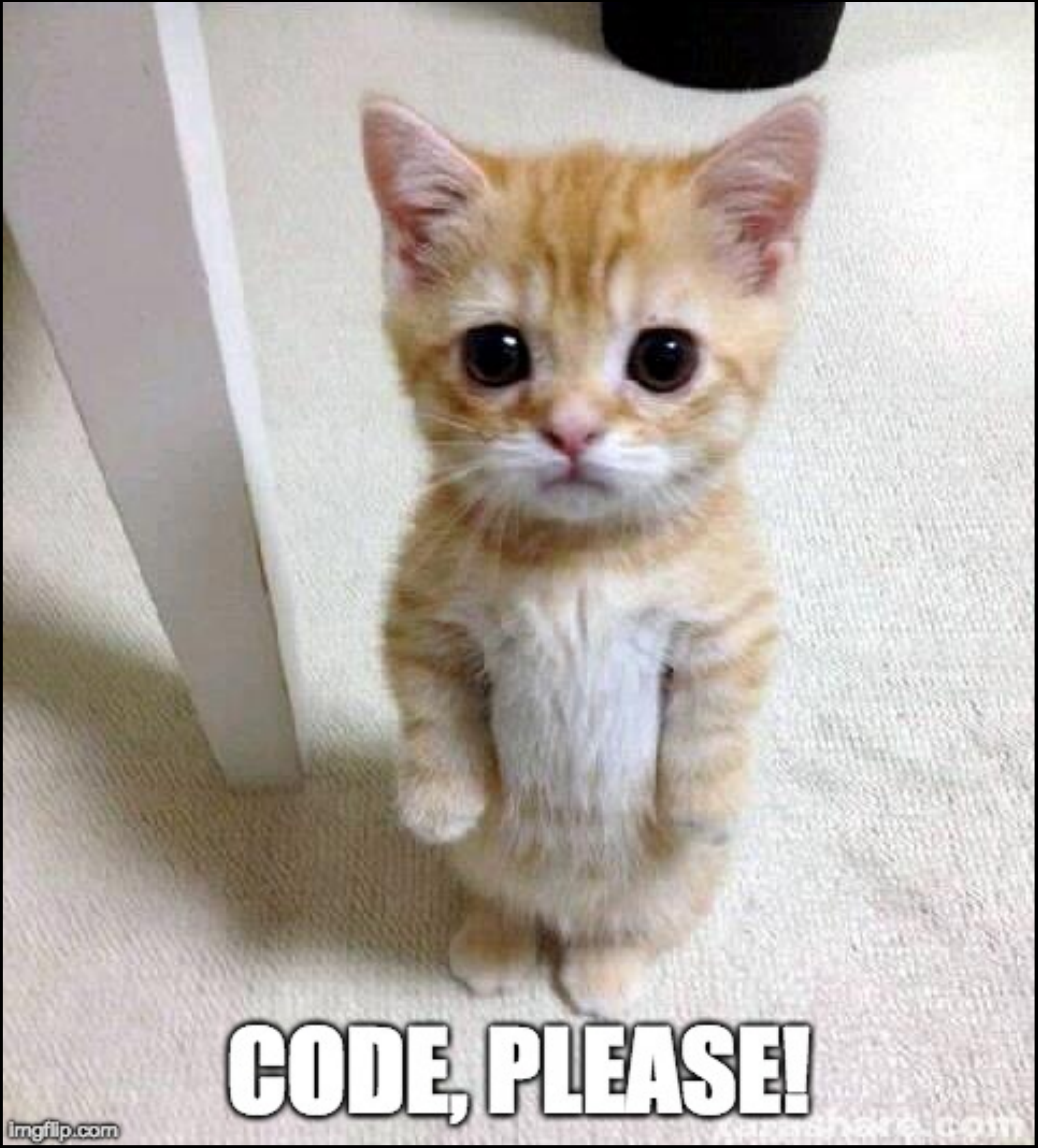
MODEL

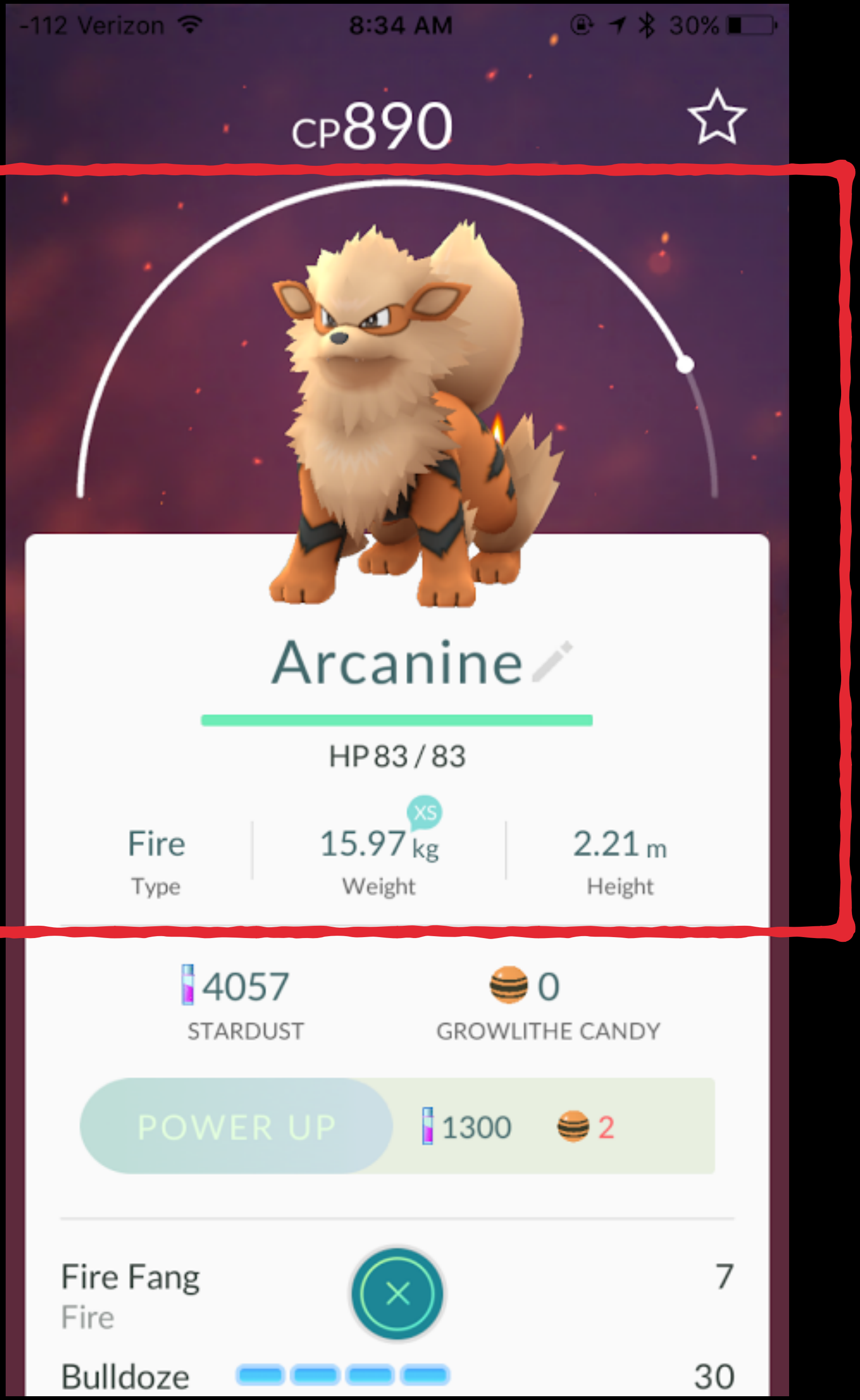
Entities

Persistence

Network layer

Data layer


CODE, PLEASE!
imgflip.com



-112 Verizon 8:34 AM 30%

CP890

Arcanine

HP 83 / 83

Fire	15.97 kg	2.21 m
Type	Weight	Height

XS

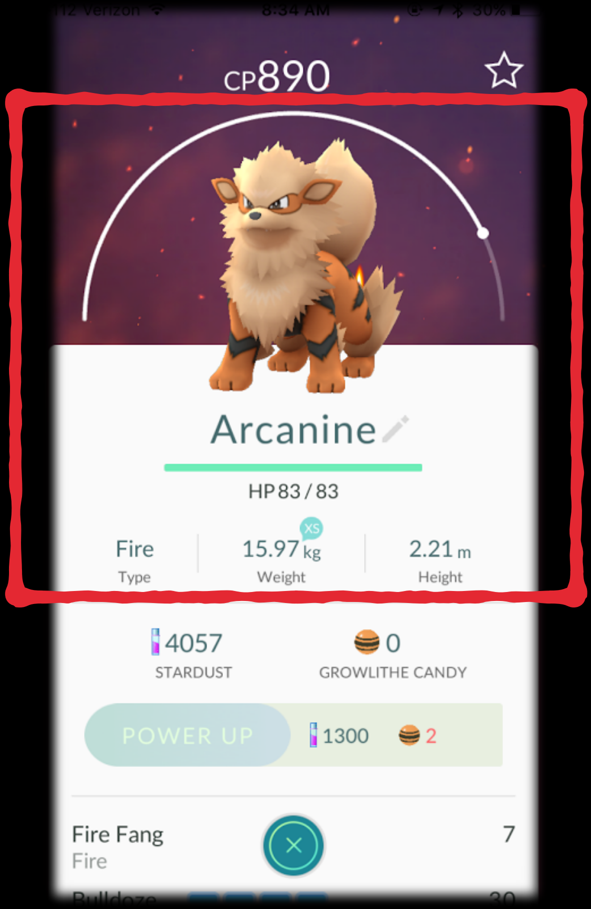
4057	0
STARDUST	GROWLITHE CANDY

POWER UP 1300 2

Fire Fang		7
Fire		
Bulldoze		30

MODEL

```
struct Pokemon {
    // 137584931
    let id: Int64

    // URL of image
    let image: PokemonImage

    // "Arcanine"
    let name: String

    // Provide `PokemonType` with `String
    let type: PokemonType

    // 15.79
    let wight: Float

    // 2.21
    let height: Float
}
```



CP890

Arcanine

HP 83 / 83

Fire
Type

15.97 kg
Weight

2.21 m
Height

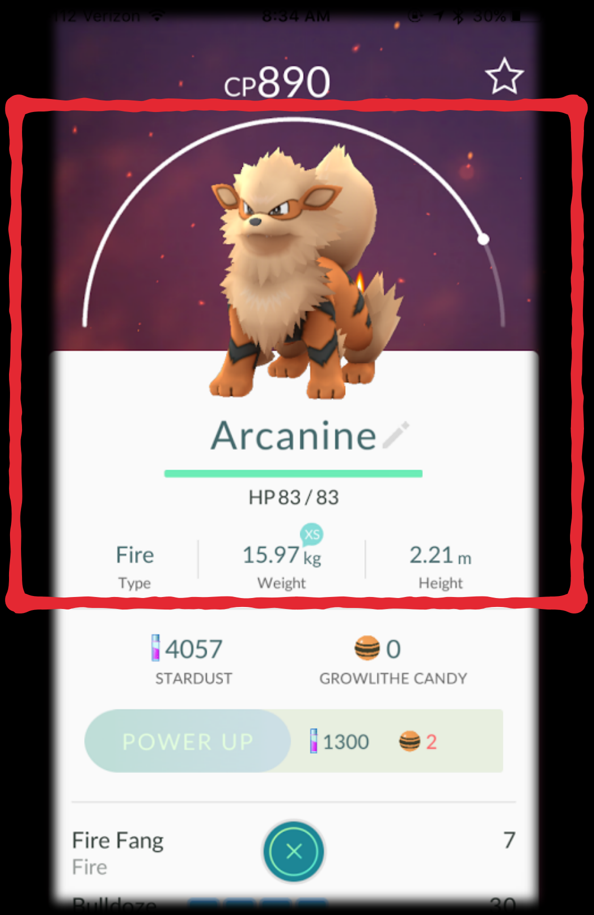
4057
STARDUST

0
GROWLITHE CANDY

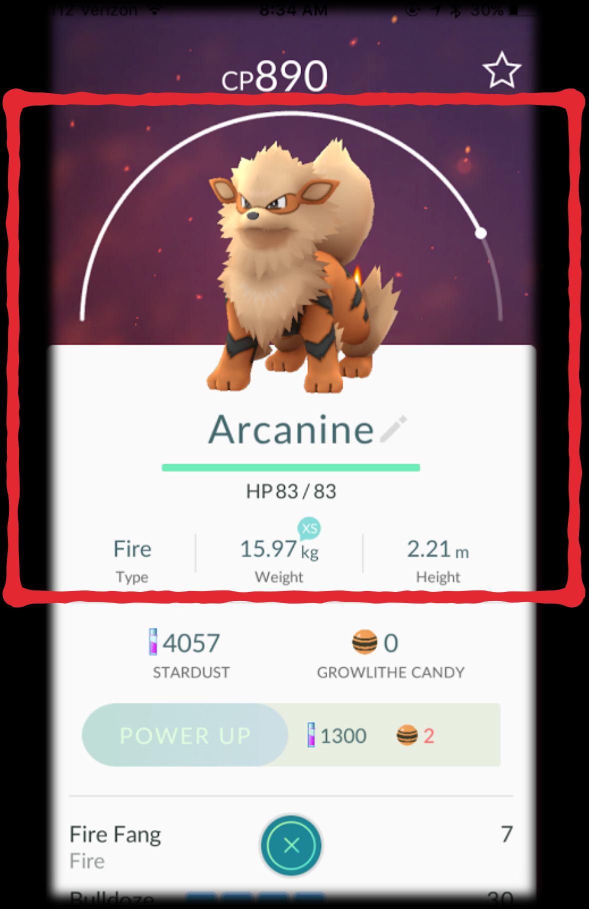
Fire Fang
Fire
7

MODEL

```
enum PokemonType: String {
    case fire = "fire"
    case water = "water"
    case rock = "rock"
    case dragon = "dragon"

    var description: String {
        return NSLocalizedString(self.rawValue, comment: "")
    }
}

enum PokemonImage {
    case url(url: String)
    case placeholder(name: String, hexColor: UInt64)
}
```



MODEL


CP890

Arcanine

HP 83 / 83

Fire	15.97 kg	2.21 m
Type	Weight	Height

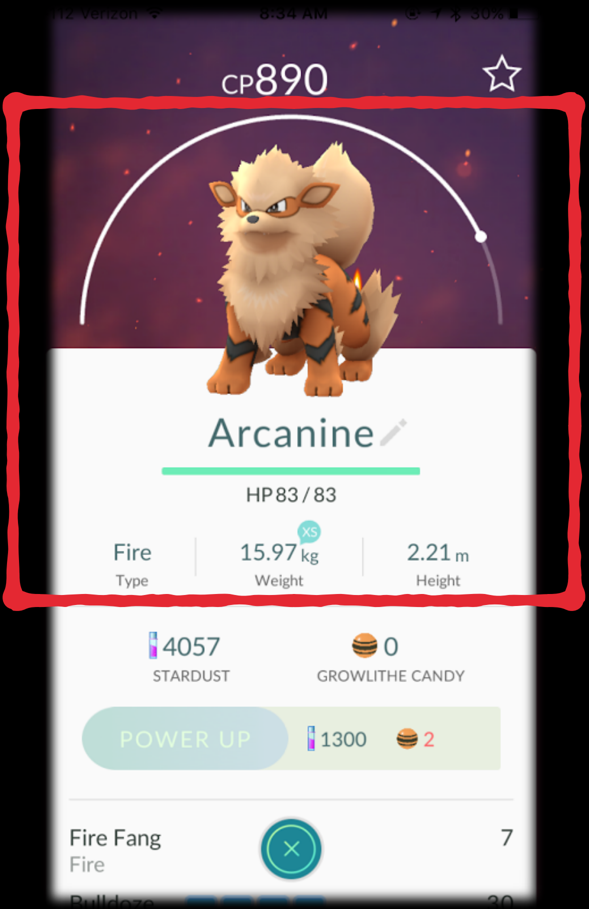
4057 STARDUST

0 GROWLITHE CANDY

```
enum Result<T> {
    case success(value: T)
    case error
}

protocol PokemonNetworkModel {
    func changeName(with name: String, completionHandler: @escaping (Result<String>) ->())
}
```

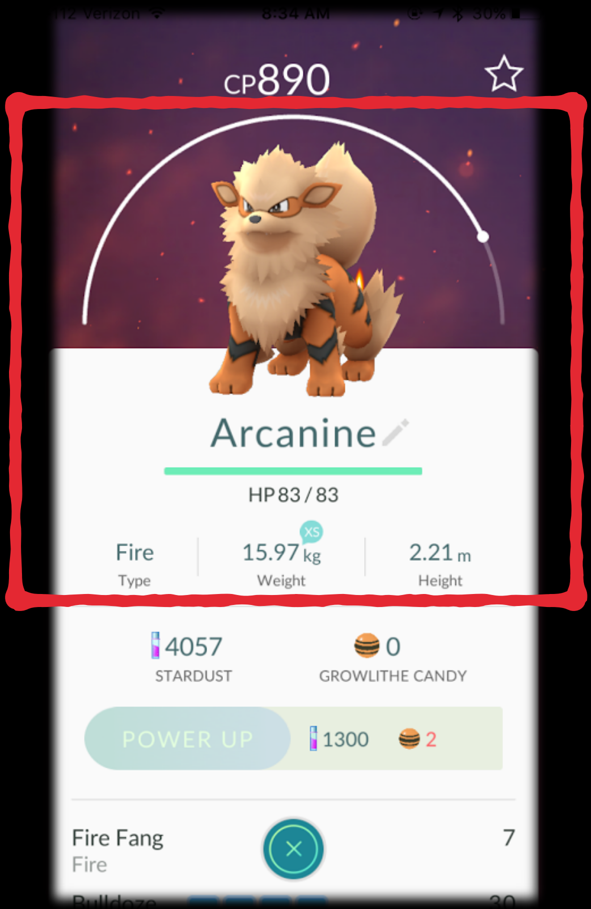
VIEW MODEL

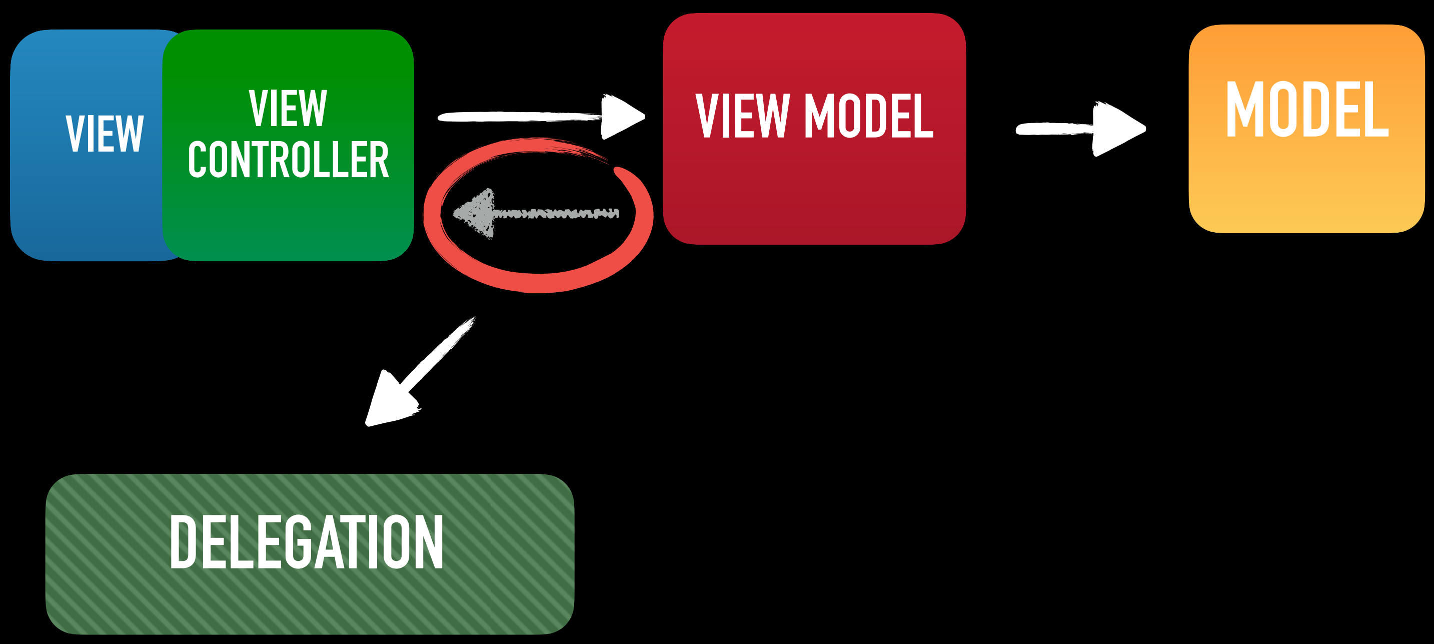
```
struct PokemonProfileViewModel {

    private let pokemon: Pokemon

    // Provide UIImage based on `PokemonType`
    let backgroundImage: UIImage

    // Provide UIColor based on weight or height perhaps
    let lineColor: UIColor

    // "15.79kg"
    let weight: String

    // "2.21m"
    let height: String

    init(pokemon: Pokemon) {
        // Code here
        //
        //
        //
        //
```

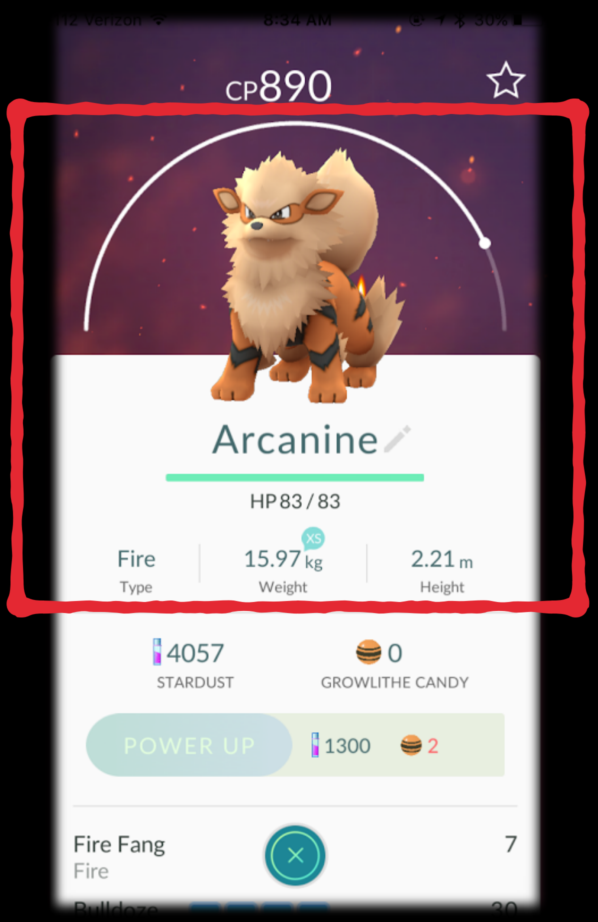


VIEW VIEW CONTROLLER

```
final class PokemonProfileViewController: UIViewController {

    @IBOutlet private weak var backgroundImageView: UIImageView!
    @IBOutlet private weak var nameLabel: UILabel!
    @IBOutlet private weak var lineView: UIView!
    @IBOutlet private weak var typeLabel: UILabel!
    @IBOutlet private weak var heightLabel: UILabel!

    var viewModel: PokemonProfileViewModel? {
        didSet {
            guard let viewModel = viewModel else {
                return
            }

            configureView(with: viewModel)
        }
    }


    func configureView(with viewModel: PokemonProfileViewModel) {
        backgroundImageView.image = viewModel.backgroundImage
        nameLabel.text = viewModel.pokemon.name
        lineView.backgroundColor = viewModel.lineColor
        typeLabel.text = viewModel.pokemon.type.description
        heightLabel.text = viewModel.height
    }
}
```

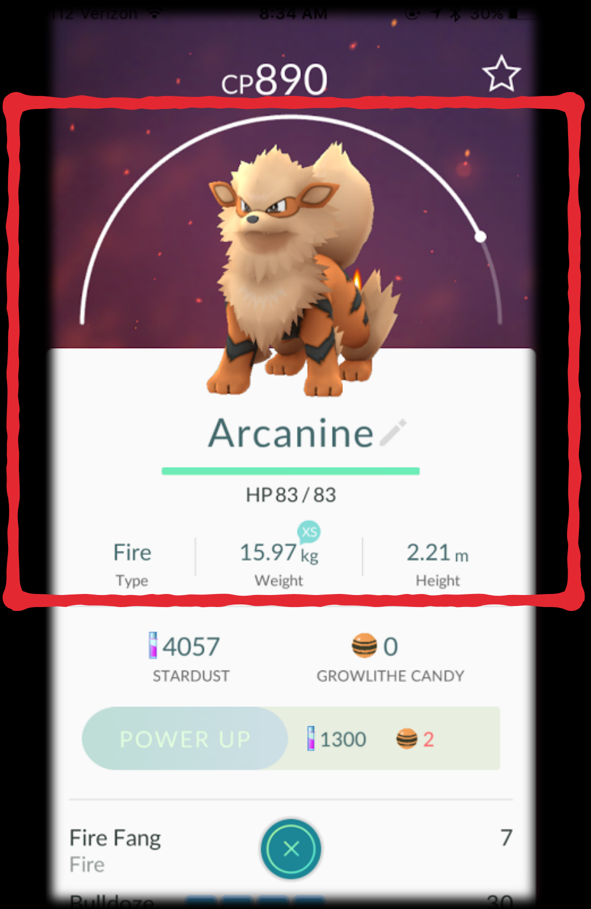



VIEW
VIEW CONTROLLER
VIEW MODEL
MODEL
DELEGATION


CP890
Arcanine
HP 83 / 83
Fire
Type
15.97 kg
Weight
2.21 m
Height
4057
STARDUST
0
GROWLITHE CANDY
POWER UP
1300
2
Fire Fang
Fire
7

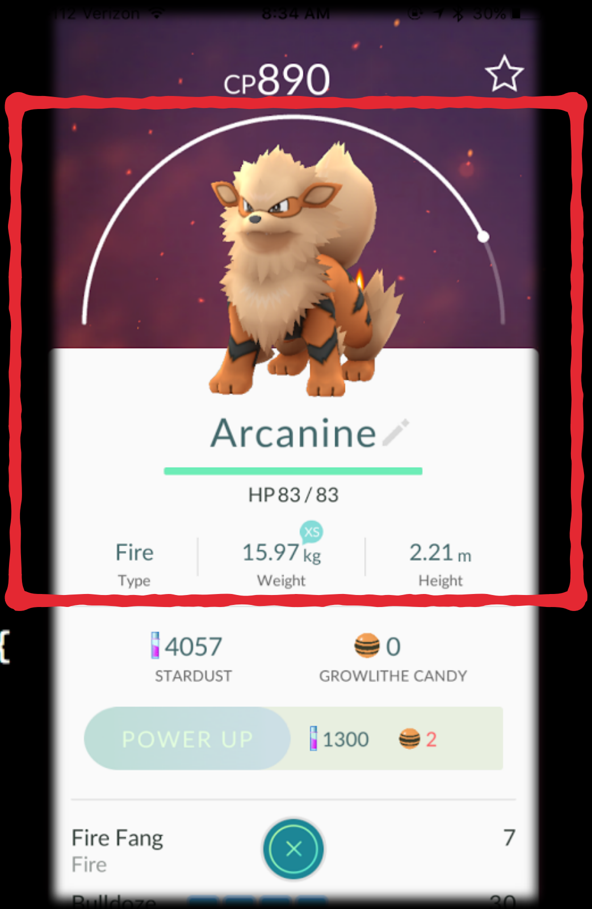
DELEGATION

```swift
protocol PokemonProfileViewModelDelegate: class {
    func viewModel(_ viewModel: PokemonProfileViewModel, didChangeName name: String)
}

// PokemonProfileViewModel

    func changeName(with name: String) {
        networkModel.changeName(with: name) { [weak self] (result: Result) in

            guard let `self` = self else {
                return
            }

            switch result {

            case .success(let name):
                self.delegate?.viewModel(self, didChangeName: name)

            case .error: break
                // Handle error here
            }
        }
    }
```



DELEGATION

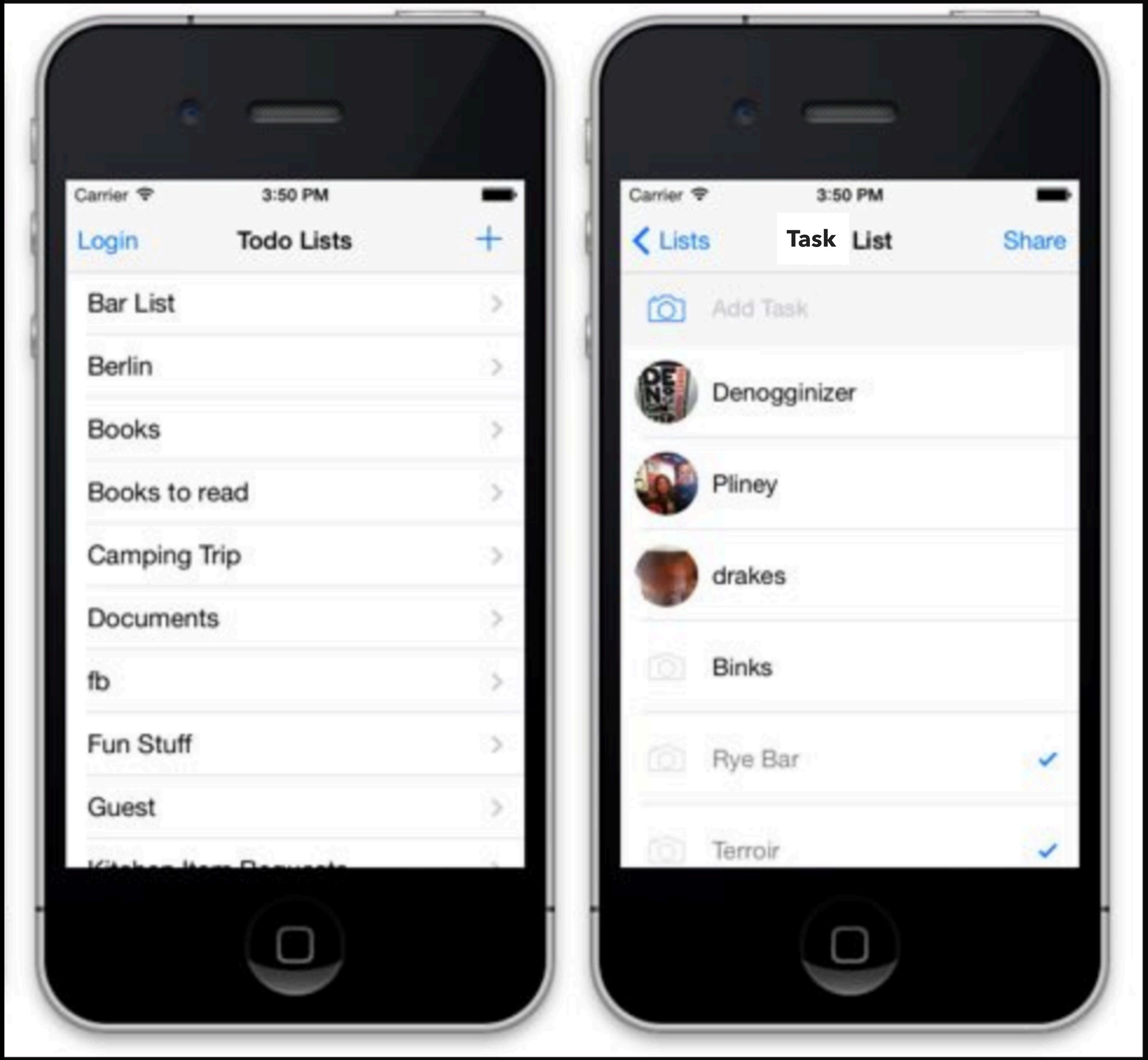
```
extension PokemonProfileViewController: PokemonProfileViewModelDelegate {
    func viewModel(_ viewModel: PokemonProfileViewModel, didChangeName name: String) {
        nameLabel.text = name
    }
}
```



WHAT CAN WE DO BETTER

WHAT CAN WE DO BETTER

PROTOCOL

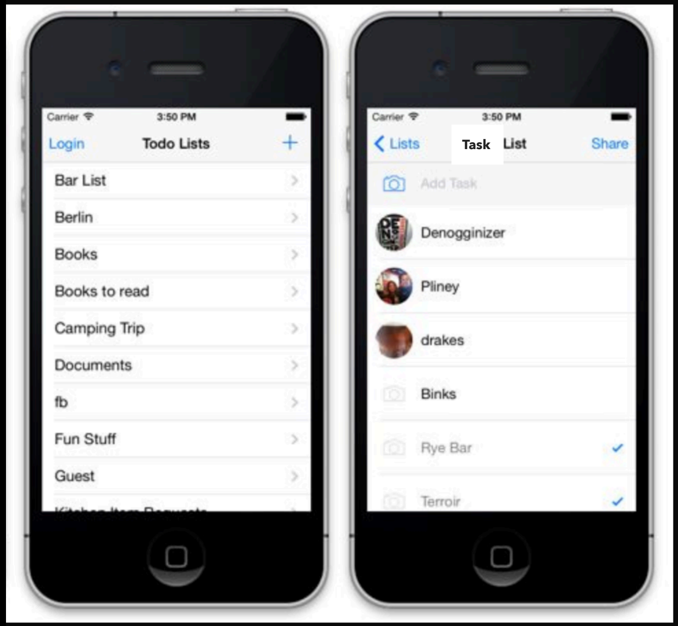


Bar List
Berlin
Books
Books to read
Camping Trip
Documents
fb
Fun Stuff
Guest

Denogginizer
Pliney
drakes
Binks
Rye Bar ✓
Terroir ✓

WHAT CAN WE DO BETTER

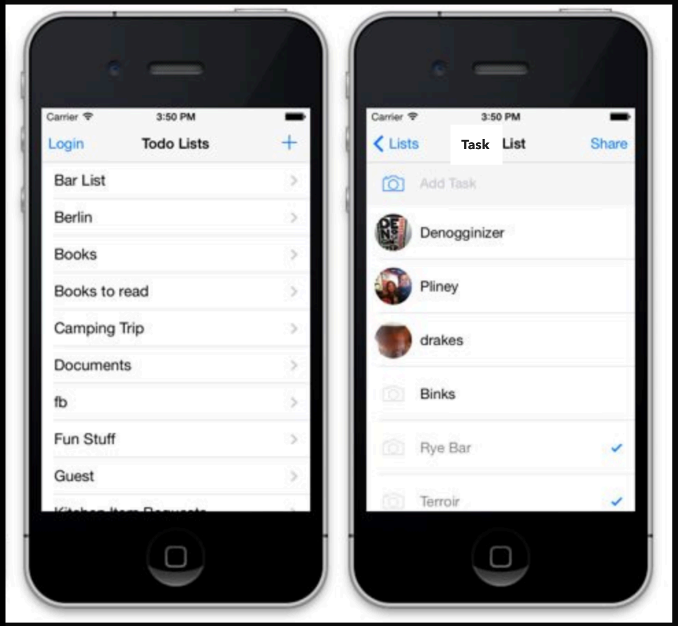
PROTOCOL



```
final class TodoListCellViewModel {
    var text: String?
    var textColor: UIColor?
    var font: UIFont?
}

final class TaskListCellViewModel {
    var text: String?
    var textColor: UIColor?
    var font: UIFont?

    var image: UIImage?
}
```

WHAT CAN WE DO BETTER

PROTOCOL

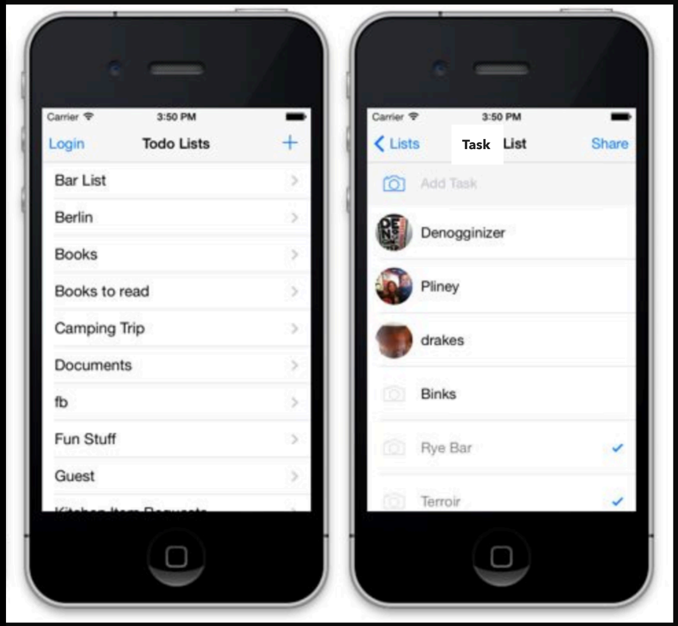


```swift
final class TaskListCellViewModel {
    var text: String?
    var textColor: UIColor?
    var font: UIFont?

    var image: UIImage?

    var isSelected: Bool = true
    var onSelected: (() -> ())?
}
```

WHAT CAN WE DO BETTER

PROTOCOL

```
final class TodoListCellViewModel {
    var text: String?
    var textColor: UIColor?
    var font: UIFont?
}

final class TaskListCellViewModel {
    var text: String?
    var textColor: UIColor?
    var font: UIFont?

    var image: UIImage?
}
```



Bar List
Berlin
Books
Books to read
Camping Trip
Documents
fb
Fun Stuff
Guest

Denogginizer
Pliney
drakes
Binks
Rye Bar ✓
Terroir ✓

WHAT CAN WE DO BETTER

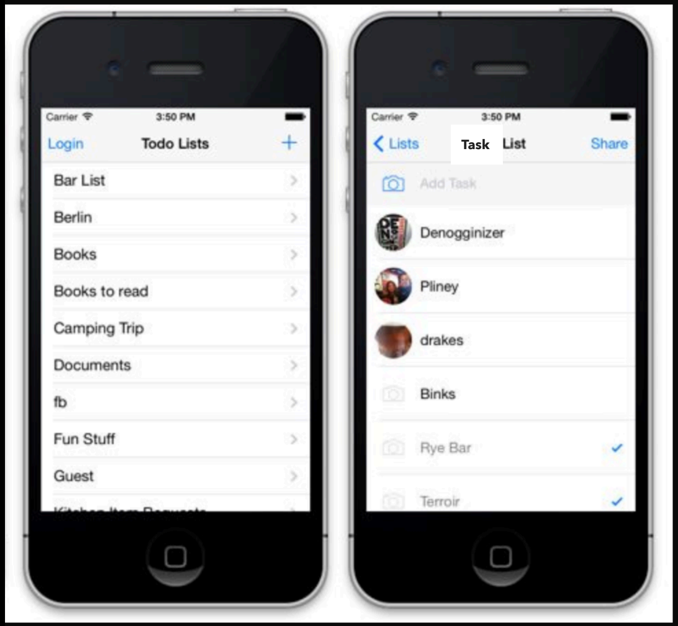
PROTOCOL

```
protocol TextPresentable {
    var text: String { get }
    var textColor: UIColor { get }
    var font: UIFont { get }
}

extension TextPresentable {
    var text: String {
        return ""
    }

    var textColor: UIColor {
        return UIColor.black
    }

    var font: UIFont {
        return UIFont.systemFont(ofSize: 12)
    }
}
```

WHAT CAN WE DO BETTER

PROTOCOL

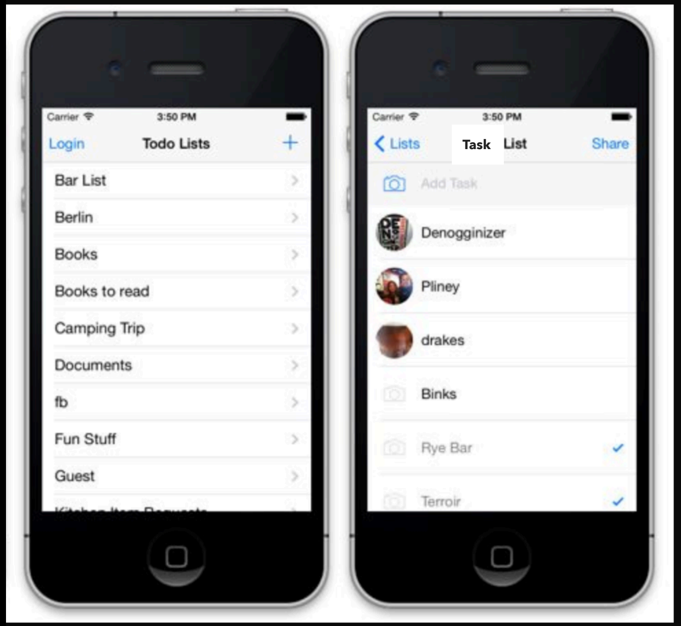
```
protocol ImagePresentable {
    var image: UIImage? { get }
}


extension ImagePresentable {
    var image: UIImage? {
        return UIImage(named: "placeholder")
    }
}
```



WHAT CAN WE DO BETTER

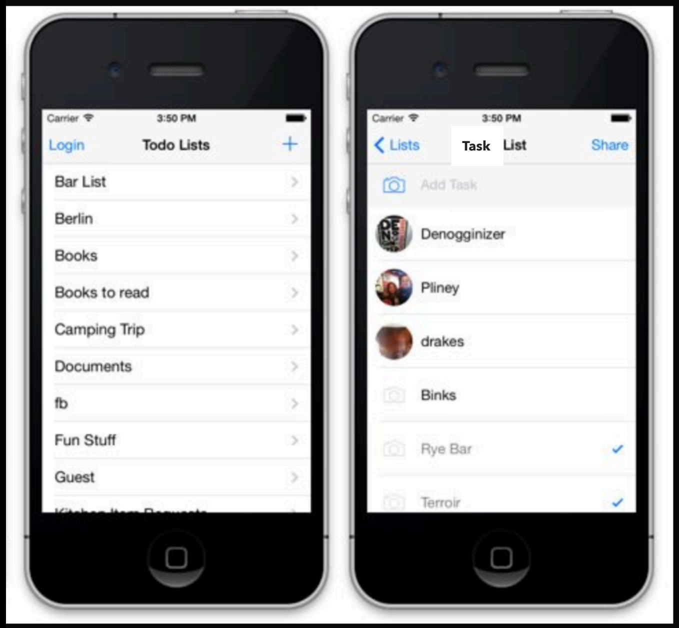
PROTOCOL

```
protocol Selectable {
    var isSelected: Bool { get }
    var onSelected: (() -> ())? { get }
}

extension Selectable {
    var isSeleted: Bool {
        return false
    }

    var onSelected: (() -> ())? {
        return nil
    }
}
```



WHAT CAN WE DO BETTER

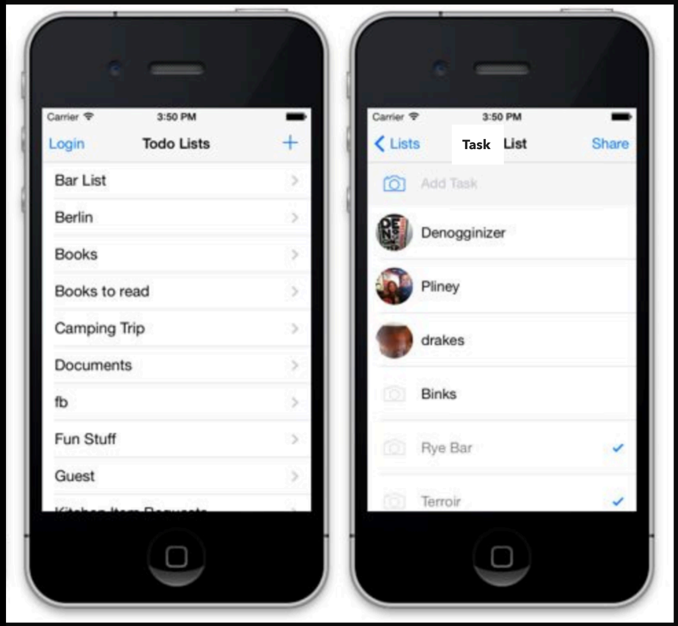
PROTOCOL



```swift
final class TodoListCellViewModel: TextPresentable {
    var text: String {
        return "List"
    }

    // init here
}

final class TaskListCellViewModel: TextPresentable, ImagePresentable, Selectable {

    var image: UIImage? {
        return UIImage(named: "list-placeholder")
    }

    // init here
}
```

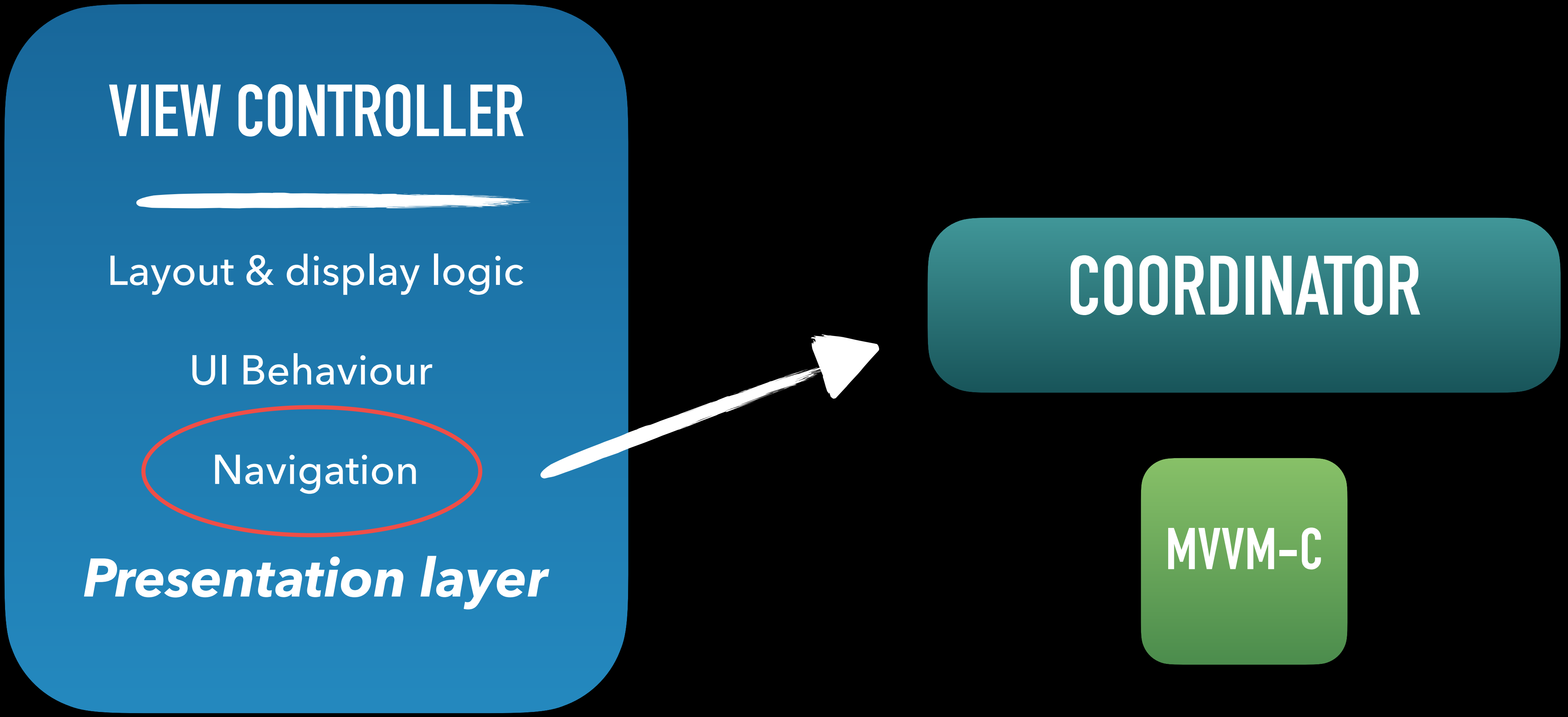
WHAT CAN WE DO BETTER

PROTOCOL



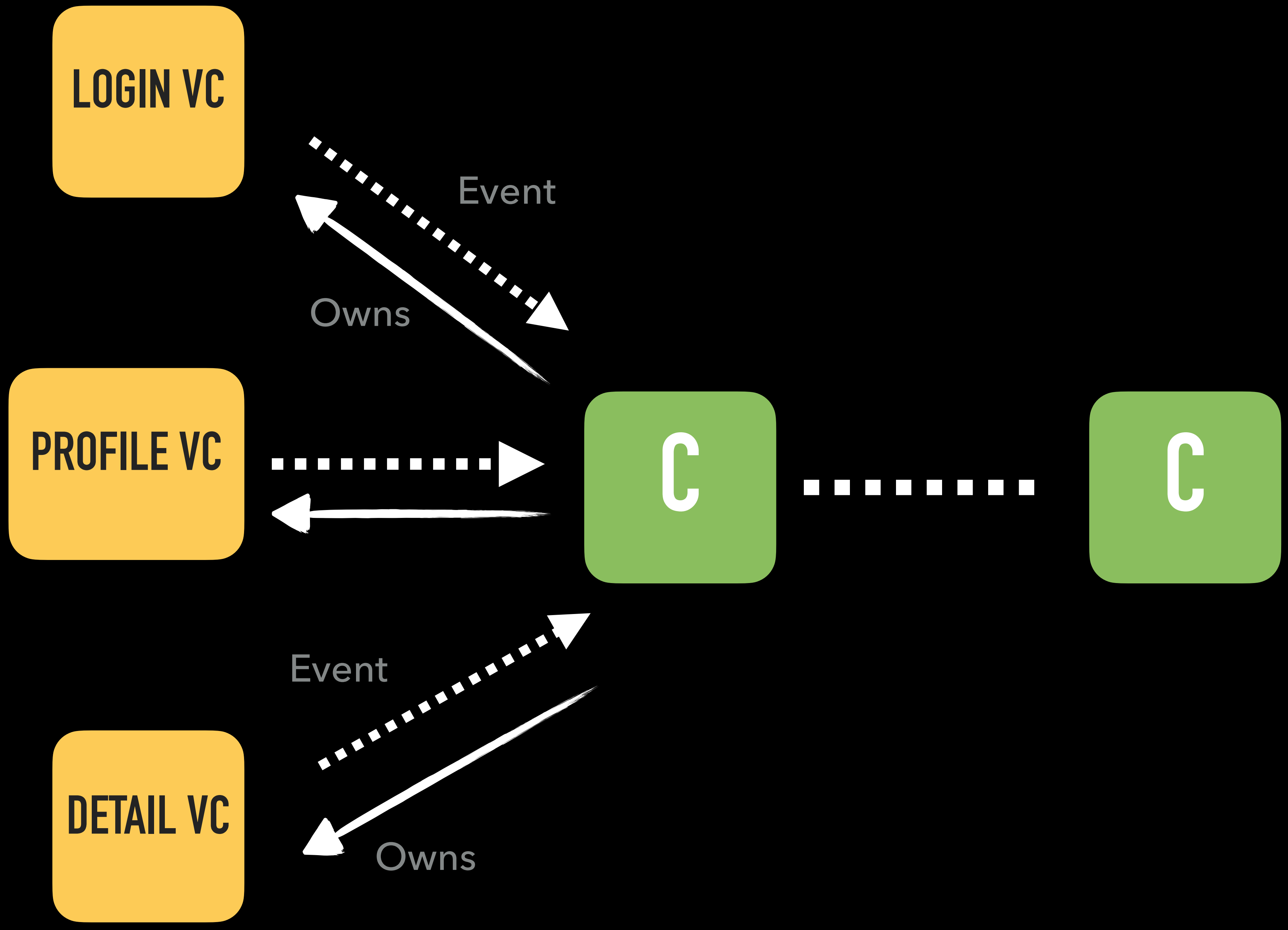
```
 class TodoListCell<T: TextPresentable>: UITableViewCell {

     var viewModel: T?

     func configure(with viewModel: T?) {
         // Configure view here
     }
 }

class TasListCell<T>: UITableViewCell where T: TextPresentable, T: ImagePresentable, T: Selectable {

    var viewModel: T?

    func configure(with viewModel: T?) {
        // Configure view here
    }

}
```

WHAT CAN WE DO BETTER



WHAT CAN WE DO BETTER

COORDINATOR



WHAT CAN WE DO BETTER

COORDINATOR

```
final class TodoListViewController: UIViewController {

    var onBackTapped:(() -> ())?

    var onListSelected:((List) -> ())?

}
```

WHAT CAN WE DO BETTER

COORDINATOR

```
final class TodoListCoordinator {

    private var navigationContoller: UINavigationController?

    init(navigationController: UINavigationController) {
        self.navigationContoller = navigationController
    }

    func pushTodoListScreen() {

        let todoListViewController = TodoListViewController()

        todoListViewController.onBackTapped = { [weak self] in
            self?.navigationContoller?.popViewController(animated: true)
        }

        todoListViewController.onListSelected = { [weak self] (list: List) in
            self?.pushTaskViewController(with: list)
        }

        self.navigationContoller?.pushViewController(todoListViewController, animated: true)
    }
```

WHAT CAN WE DO BETTER

COORDINATOR

```
func pushTaskViewController(with list: List) {
    let viewModel = TaskViewModel(list: list)
    let viewController = TaskViewController(viewModel: viewModel)

    // Handle navigation event here

    self.navigationContoller?.pushViewController(viewController, animated: true)
}
```

WHAT CAN WE DO BETTER

COORDINATOR

```
final class TodoListViewController: UIViewController {

    enum Event {
        case back
        case list(list: List)
    }

    var onEvent: ((Event) -> ())?
}
```

WHAT CAN WE DO BETTER

COORDINATOR

```swift
final class TodoListCoordinator {

    private var navigationContoller: UINavigationController?

    init(navigationController: UINavigationController) {
        self.navigationContoller = navigationController
    }

    func pushTodoListScreen() {

        let todoListViewController = TodoListViewController()

        todoListViewController.onEvent = { [weak self] (event: TodoListViewController.Event) in
            switch event {
            case .back:
                self?.navigationContoller?.popViewController(animated: true)
            case .list(let list):
                self?.pushTaskViewController(with: list)
            }
        }

        self.navigationContoller?.pushViewController(todoListViewController, animated: true)
    }
```

WHAT CAN WE DO BETTER

MVVM OFFERS BETTER REUSABILITY & TESTABILITY

PROTOCOL

MVVM-C

RXSWIFT

http://bit.ly/mvvminpractice

Q & A