

View Controllers:

```
BaseViewControllerProtocol: protocol
    -----
    - onClickRetryAction()
    - onClickCloseAction()
```

```
BaseViewController: class
    -----
    - var isLoadingScreenShowing
    - var isErrorScreenShowing
    - let loadingScreenView: UIView
        - let view
        - let activityIndicator
        - let loadingLabel
    - let errorLabel: UILabel
        - let errorLabel
    - var retryButton: UIButton
        - let retryButton
    - var errorScreenView: UIView
        - let view
        - let errorImageView
        - let closeButton
        - let symbolConfig
    -----
    - viewDidLoad()
    - viewWillAppear(_ animated: Bool)
    - showLoadingScreen()
    - showError( error:String, withRetryButton isShowingRetry:Bool = false )
        - showContent()
        - onClickCloseError()
        - onClickRetry()
        - hideErrorScreen()
        - hideLoadingScreen()
```

```
HomePageViewController: class
    -----
    - var cvMealList: UICollectionView!
        - let viewModel
    -----
    - viewDidLoad()
    - fetchDataFromServer()
    - onClickRetry()
    - debug()
```

```
HomePageMealCollectionViewCell: class
    -----
    - var lblName: UILabel!
    - var imgItem: UIImageView!
    -----
    - awakeFromNib()
    - configure(with meal:Meal)
```

```
DetailViewController: class
    -----
    - var lblName: UILabel!
    - var tblIngredients: UITableView!
    - var lblInstructions: UILabel!
    - var isInstructionsExpanded
    - var scrollView: UIScrollView!
    - let mealDetail: MealDetail? required @objc
    -----
    - viewDidLoad()
    - viewDidLoadLayoutSubviews()
        - showMore()
        - checkLabelHeight()
    - reloadData()
    - init?(mealDetail: MealDetail)
    - init?(coder: NSCoder)
```

```
DetailViewController: extension
    -----
    -----
    - tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int
    - tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell
```

View Modal

```
ViewModel: class
-----
- var originalMealList
- var displayMealList
- var currentDetailObject: MealDetail?
- let dependency: ViewModelDependency
- let networkHandler: NetworkHandler
-----
- getMealList(completion: @escaping ([Meal], Error?) -> Void)
- getMealDetails(completion: @escaping (MealDetail?, Error?) -> Void)
- getImageFor(url: String, completion: @escaping ((UIImage) -> Void))
- init(withDependency: ViewModelDependency, andNetworkHandler: NetworkHandler =
NetworkHandler())
```

```
ViewModelDependency: protocol
-----
-----
- getMealList(completion: @escaping ([Meal], Error?) -> Void)
- getMealDetails(completion: @escaping (MealDetail?, Error?) -> Void)
- getImageFor(url: String, completion: @escaping ((UIImage) -> Void))
```

```
ViewModelDependencyClass: class
-----
-----
- getMealList(completion: @escaping ([Meal], Error?) -> Void)
- getMealDetails(completion: @escaping (MealDetail?, Error?) -> Void)
- getImageFor(url: String, completion: @escaping ((UIImage) -> Void))
```

Modal Classes

```
Meal: class
-----
- let name: String
- let imageURL: String
- let id: String
- var image: UIImage
- var isImageDownloaded
-----
- == (lhs: Meal, rhs: Meal) -> Bool
- updateMealWithImage(image: UIImage)
- init(name: String, imageURL: String, id: String, image: UIImage?)
```

```
MealDetail: class
-----
- let name: String
- let instructions: String
- var ingredients
- var image: UIImage
-----
- == (lhs: MealDetail, rhs: MealDetail) -> Bool
- init(name: String, instructions: String, ingredients: [Ingredient], image:
  UIImage? = nil)
```

```
Ingredient: class
-----
- let name: String
- let quantity: String
- let displayName: String
-----
- init(name: String, quantity: String)
```

Interface Object

```
MealDTO: struct
-----
- var meals: T
-----
```

```
MealObjectFromServer: struct
-----
- let name: String
- let imageURL: String
- let id: String
-----
- getInterfaceObject () -> Meal
```

```
MealDetailsObjectFromServer: struct
-----
- let id: String
- let name: String
- let instructions: String
- let ingredient1: String?
- let ingredient2: String?
- let ingredient3: String?
- let ingredient4: String?
- let ingredient5: String?
- let ingredient6: String?
- let ingredient7: String?
- let ingredient8: String?
- let ingredient9: String?
- let ingredient10: String?
- let ingredient11: String?
- let ingredient12: String?
- let ingredient13: String?
- let ingredient14: String?
- let ingredient15: String?
- let ingredient16: String?
- let ingredient17: String?
- let ingredient18: String?
- let ingredient19: String?
- let ingredient20: String?
- let measurement1: String?
- let measurement2: String?
- let measurement3: String?
- let measurement4: String?
- let measurement5: String?
- let measurement6: String?
- let measurement7: String?
- let measurement8: String?
- let measurement9: String?
- let measurement10: String?
- let measurement11: String?
- let measurement12: String?
- let measurement13: String?
- let measurement14: String?
- let measurement15: String?
- let measurement16: String?
- let measurement17: String?
- let measurement18: String?
- let measurement19: String?
- let measurement20: String?
-----
- getInterfaceObject () -> MealDetail
```


server response

Network:

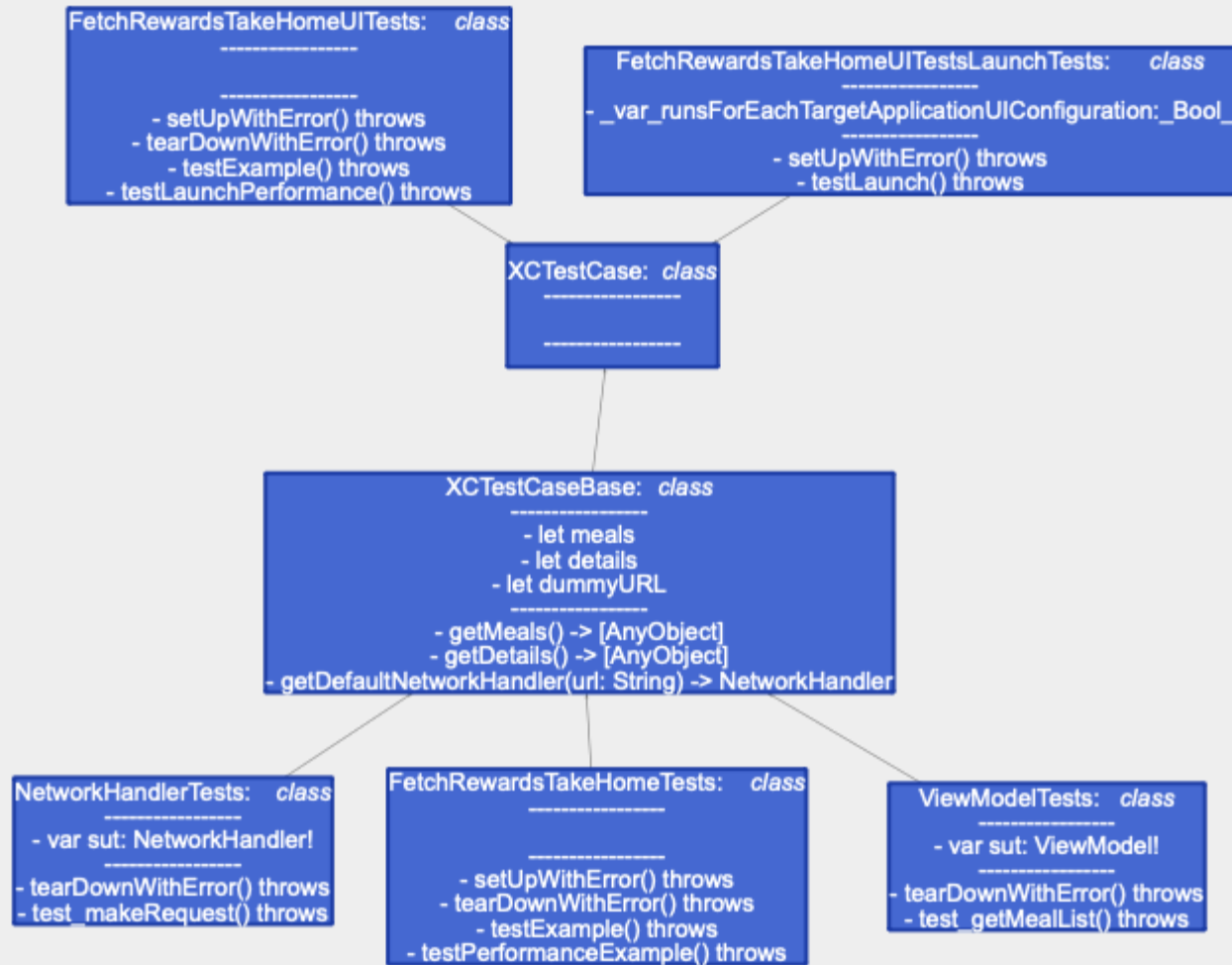
```
NetworkHandler: struct
-----
- var session: URLSession
-----
- makeAPICall<T:Codable>(with url: URL, completion: @escaping ((T?, Error?)
-> Void))
- makeAPICall<T:Codable>(with url: String, completion: @escaping ((T?, Error?)
-> Void))
- init(session: URLSession = URLSession.shared)
```

```
URLProtocolMock: class
-----
- _var_testData_
-----
- _canInit(with_request: URLRequest) -> Bool
- _canonicalRequest(for_request: URLRequest) -> URLRequest
- startLoading()
- stopLoading()
```

```
URLProtocol: class
-----
-----
```



XCTestCase



Architecture

