Struct ContentView: View

Variables

var issues: [Issue]

* Give us all data either by Tags or by issues
* Place issues into an array
* If there is a tag attached to the filter then search by tag first
* Else fetch all issues and place them into an array
  + Based on the minModificaiton Date
* Return allIssues sorted

Views

List

* showing results of issues either any issue
  + from a tag filter
  + or any issue based on a minModificationDate

the list changes the view to IssueRow

Functions

Func delete(\_ offsets: IndexSet)

* Deletes the issue from the list

Struct ContentView: View

@EnvironmentObject var dataController: DataController

* Calling in the Datacontroller to access coredata

var issues: [Issue]

let filter = dataController.selectedFilter ?? .all

* Select a filter if not default to .all
  + .all displays all issues

var allISsues: [Issue]

* Array of issues

If let tag = filter.tag {

allIssues = tag.issues?.allObjects as? [Issue] ?? []

* If there is a tag attached filter by the tag first

Else

Let request = Issue.fetchRequest()

Request.predicate NSPredicate(formate: “modificationDate ? %@”, filter.minModificationDate as NSDate)

* Requesting issues based on the date

allIssues =(try? dataController.container.viewContext.fetch(request)) ?? []

* If no tags then fetch all the issues if not return empty array

Return allIssues.sorted()

* Whatever issues were found return them sorted

Var body: View

List

ForEach(issues) { issue In

IssueRow(issue: issue)

.onDelete(preform: delete)

.navigationTitle(“Issues”

Func delete(\_ offsets: IndexSet)

For offset in offsets

Let item = tags[offset]

* tags[offset]
  + tags came form fetch request
  + offset is the item being passed into delete

dataController.delete(item)