Individual/Partner Project #1

CS275 Fall 2021
30 points
Deliverable #1: due Saturday, Oct. 16th, 11:59 pm
Deliverable #2: due Saturday, Nov. 6th, 11:59 pm

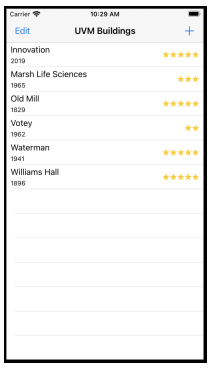
1 Overview-Detail Interface

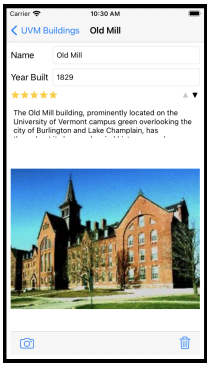
Build an app similar to the LootLogger app, using your choice of data. You may work individually or with a partner. Graduate students must work individually.

1.1 Description of UI

Here are the elements that your interface should have:

- a `UITableView`
- a `UINavigationController` to manage the transitions between views
- an overview-detail interface, with a `UIStackView` on the detail interface
- the ability to add, update, and delete entries in the list, with a user alert when the user wants to delete a row
- + and Edit bar buttons in the `UINavigationBar` on the overview view
- a delete button (with a trash can icon) in a `UIToolBar` on the detail view
- persistence, so that the data of the app is saved between invocations of the app





I show an image in my detail view; only grad students need to have this feature (see below). Undergraduates should change the left bar button of the toolbar to the Camera system item, but they should disable it in Interface Builder.

1.2 Behavior

When the delete button on the detail view is touched, the current item should be removed from the list, and the detail view should be replaced by the overview view. First though, put up an alert (just as when the user tries to delete a row from the overview view).

Alert controllers are described on pp. 300-301. Use `.alert` as the preferred style for the alert controller.

Use a navigation controller to control the navigation from the overview view to the detail view (Ch. 12).

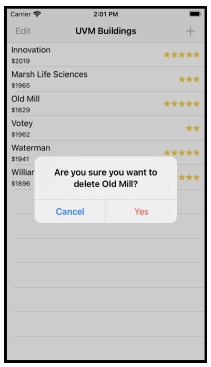
Use a vertical stack view on the Detail view, with at least one horizontal stack view embedded in the vertical stack view (as described on pp. 232-233). All of your data should be editable (the use should be able to change any part of the model data using the Detail view).

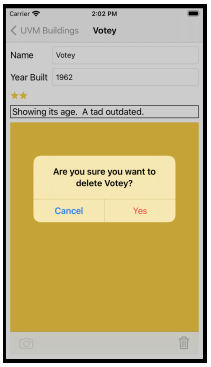
Put a toolbar at the bottom of the detail view (see pp. 294-295). Put a `UIBarButtonItem` with a "Trash" System Item on the right-hand side of the toolbar. To do this, you first add a `UIBarButtonItem` to the toolbar. Then you add a Flexible Space Bar Button Item in the middle of the toolbar, between the two buttons. (You can disable the left bar button if you'd like.)

This delete button should first prompt the user "Are you sure you want to delete this item?" If the user says yes, then delete the item and return to the previous view, by making this call:

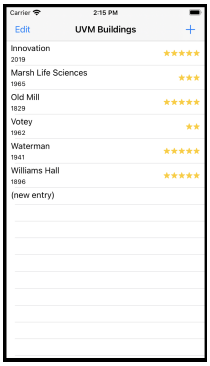
```
self.navigationController!.popViewController(animated: true)
```

Here's my detail view, showing the alert:





The + button in the overview view should create a new row, which the user can then edit by selecting on it and bringing it up in the detail view. For example, here's what I have after pressing the + button:



These features and mechanisms are covered in Ch. 9-14 in BNRG.

1.3 Layouts and constraints

Constraints—and UI design in general—are tricky. You might end up with a few warning messages about missing constraints or messages like this: "Trailing constraint is missing, which may cause overlapping with other views." But you should not have any errors from your storyboard. But the screens in your UI should be correct (no missing elements), and they should look good.

Do not worry about making different layouts for different orientations—portrait orientation is fine.

But, and this is important, do create your interface so that it will look and function correctly on an iPhone 8.

1.4 Model

You should have two Swift files for your model: a *Model*.swift, and a *ModelStore*.swift (with the name of your model, e.g. Building, or Beer, or Pokemon). You can hardcode a few entries in the initializer of your model data. But, the app should put the hard-coded entries into the ModelStore only if it doesn't ready any model objects from the persistent store during app initialization.

2 Deliverables

2.1 Deliverable #1

By Saturday, Oct. 16th, at 11:59: a description of the data you will use; a rough sketch of your overview and detail views; and, if you are working with a partner, the name of your partner.

2.2 Deliverable #2

By Saturday, Nov. 6th at 11:59 pm: the completed project.

3 Evaluation

I'll check that you've fulfilled all of the functional requirements and that your app doesn't crash or behave unpredictably. In addition, I'll reserve a portion of the grade for the "wow factor": how attractive is your interface? Did you try implementing something that we haven't covered in class? Did you go above and beyond just the bar minimum `UILabel` and `UITextField`?

4 Graduate Students

Students taking the course for graduate credit, and undergraduates who want a little extra credit: implement an image store, as described in Ch. 15, and let the user either take a photo or use an existing photo from the device's photos library. Persist the images, as described in Ch. 15.

5 What to Submit

If you have your project in github, then submit a link to your project. Otherwise, you can zip up your project, starting at the top level (so there should be a folder *ProjectName*, and that folder should have two subfolders: *ProjectName* and *ProjectName.xcodeproj*).