9: Campus Walk II

Due: Noon October 29

Description

In this assignment you will modify your Campus Walk app from last week to include more functionality. This assignment will also require you to implement the required features with your own design. You will be adding support for walking directions using overlays and step by step directions.

You will also be adding in editing support for modifying your building's info page. If you did not include a page with info about each building in your app from last week, you will need to add it in for this week.

Required Features

This list is simply the required features of the app. You need to implement all of them into your own robust design.

1. Your app should include the option to get directions from one location to another. This could be between two buildings, from user location to a building, or from a building to the user's current location. The selection process of the source and destination should be an intuitive UI that is simple for the user.
2. Your app should show the overlay for the path selected by the user. There should be a way for removing the overlay.
3. Your app needs to show the step-by-step directions for the walking route. This information is provided in the response to a directions request. There should be two options for viewing the direction steps. The user should be able to look through each step one at a time appearing over the map or view all of the steps at once in a list. There are many ways to accomplish these two viewing options.
4. You should show the ETA for a walking route. The time should be nicely formatted using a DateFormatter.
5. Your app needs to show info about each building upon selecting the building. The building info should include at least the building's name and photo. Feel free to add other details about the building (i.e., year built). If no photo is available, please display an image that says "No Photo Available".
6. The user should be able to modify the photo for a building. They should have the option to take a new photo (if camera is available) or select an existing photo from their Photos album. This will require use of a `UIImagePickerController` and its delegate. If a new image is taken/selected, show this on the building info page instead of what was previously shown. When the user leaves the building info page and comes back, the new image should persist (while the app is running -- if you kill the app, the new image does not have to persist).
7. The user should easily be able to navigate on the map to their current location.

Hints

1. You may find it easier to present the image picker controller programmatically rather than in IB. Be sure to follow the recommended presentation styles as described in the `UIImagePickerController`'s documentation. The user should not be able to pick a source option that is not available.
2. To compute the ETA you can use information in the `MKRoute` object.

3. You do not need to make a separate request to compute the ETA of a route.

Testing

1. Your project should build without errors or warnings. Be sure to address any Auto Layout errors or warnings.
2. Test your app on both iPhone and iPad simulators using both orientations. Ensure that your app does not crash.
3. You can add photos to the simulator's photo app by viewing an image in Safari on the simulator and then using the long press gesture on the image followed by selecting "Save Image."
4. Your project should have a clean and pleasant user interface. User interface elements should be arranged logically, be aligned nicely; fonts should be appropriately sized; colors should be chosen to enhance the app's appearance.

Troubleshooting

Make effective use of the debugger. Set breakpoints and examine values.

Submission

Your submission should be pushed on the master branch. Be sure to verify that your project builds and remove all cruft and compiler warnings.