Step 1: Go to the main storyboard and from the object library we want to drag and drop a Map Kit View into the scene. Make sure your constraints are correct and we can move on.

Step 2: Now go to ViewController.swift and make sure to import UIKit:

```
import MapKit
```

Step 3: add the outlet to your main storyboard:

```
@IBOutlet private var mapView: MKMapView!
```

Make sure you link them like you would any other outlet

Step 4: Next we want to set our initial location and constrain the camera view a bit. Since we're doing locations on Grand Valley's campus we'll get the gps coordinates for the clock tower

```
// Set initial location to GVSU
let initialLocation = CLLocation(latitude: 42.96345, longitude: -85.88860)
```

Then we want to add this extension to ViewController.swift

```
private extension MKMapView {
  func centerToLocation(
    _ location: CLLocation,
    regionRadius: CLLocationDistance = 1000
  ) {
    let coordinateRegion = MKCoordinateRegion(
      center: location.coordinate,
      latitudinalMeters: regionRadius,
      longitudinalMeters: regionRadius)
    setRegion(coordinateRegion, animated: true)
  }
}
```

Not quite done with this part yet. Head back up to ViewDidLoad() and add in this bit of code:

```
mapView.centerToLocation(initialLocation)
```

This makes sure we zoom to the initial location once we start our app.

Step 5: We need to add some constraints to the viewable area of the camera:

```
    let GvCenter = CLLocation(latitude: 42.96345, longitude: -85.88860)
    let region = MKCoordinateRegion(
      center: oahuCenter.coordinate,
      latitudinalMeters: 50000,
      longitudinalMeters: 60000)
    mapView.setCameraBoundary(
      MKMapView.CameraBoundary(coordinateRegion: region),
      animated: true)

    let zoomRange = MKMapView.CameraZoomRange(maxCenterCoordinateDistance:
200000)
    mapView.setCameraZoomRange(zoomRange, animated: true)
```

That's a decent view area.

Step 6: Lets add some annotations to the map. In ViewController.swift the annotation format should look a bit like this. LocationName will be the subtitle that appears on the map when we tap on the pin. Lets have some fun with it.

```
let clocktower = Location(
  title: "Cook Carillon Clock Tower",
  locationName: "Lets be real. We're all kinda thinking about jumping right
now",
  discipline: "Sculpture",
  coordinate: CLLocationCoordinate2D(latitude: 42.96345, longitude: -
85.88860))
mapView.addAnnotation(clockTower)
```

Step 7: Time to configure the annotation view. This will determine the general structure of the popup bubble and should look a bit like this

```
extension ViewController: MKMapViewDelegate {
  // 1
  func mapView(
    _ mapView: MKMapView,
    viewFor annotation: MKAnnotation
  ) -> MKAnnotationView? {
    // 2
    guard let annotation = annotation as? Artwork else {
      return nil
    }
    // 3
    let identifier = "building"
    var view: MKMarkerAnnotationView
    // 4
    if let dequeuedView = mapView.dequeueReusableAnnotationView(
      withIdentifier: identifier) as? MKMarkerAnnotationView {
      dequeuedView.annotation = annotation
      view = dequeuedView
    } else {
      // 5
      view = MKMarkerAnnotationView(
        annotation: annotation,
        reuseIdentifier: identifier)
      view.canShowCallout = true
      view.calloutOffset = CGPoint(x: -5, y: 5)
      view.rightCalloutAccessoryView = UIButton(type: .detailDisclosure)
    }
    return view
  }
}
```

Make sure you also set your mapView delegate in ViewController.swift, under ViewDidload()

```
mapView.delegate = self
```

A couple more steps to go and then we will have ourselves a basic maps app.

Step 8: We need to handle the callout when we open the app. Head over to Location.swift and add the following

```
import Contacts


var mapItem: MKMapItem? {
  guard let location = locationName else {
    return nil
  }

  let addressDict = [CNPostalAddressStreetKey: location]
  let placemark = MKPlacemark(
    coordinate: coordinate,
    addressDictionary: addressDict)
  let mapItem = MKMapItem(placemark: placemark)
  mapItem.name = title
  return mapItem
}
```

Step 9: Now back to ViewController.swift

```
func mapView(
  _ mapView: MKMapView,
  annotationView view: MKAnnotationView,
  calloutAccessoryControlTapped control: UIControl
) {
  guard let building = view.annotation as? Artwork else {
    return
  }

  let launchOptions = [
    MKLaunchOptionsDirectionsModeKey: MKLaunchOptionsDirectionsModeDriving
  ]
  building.mapItem?.openInMaps(launchOptions: launchOptions)
}
```

That's it! You're done. Pat yourself on the back and move on with your life.