single View ⇒ check Core Data
// Main.storyboard
MapKit View

// CoreData ⇒ entities → places
Attribute → longitude: Double
latitude: Double
subtitle: String
title: String

// ViewController.swift

```
import UIKit
import MapKit
import CoreLocation
import CoreData

class ViewController: UIViewController, MKMapViewDelegate, CLLocationManagerDelegate{
    @IBOutlet weak var mapView: MKMapView!
    var locationManager = CLLocationManager()
    var chosenLatitude = Double()
    var chosenLongitude = Double()
    var selectedTitle = ""
    var selectedTitleID: UUID?
    var annotationTitle = ""
    var annotationSubTitle = ""
    var annotationLatitude = Double()
    var annotationLongitude = Double()
    override func viewDidLoad(){
        super.viewDidLoad()
        mapView.delegate = self                                   // go to plist "+" Location when in use
        locationManager.delegate = self
        locationManager.desiredAccuracy = kCLLocationAccuracyBest
        locationManager.requestWhenInUseAuthorization()
        locationManager.startUpdatingLocation()

        let gestureRecognizer = UILongPressGestureRecognizer(target: self, action: #selector(chooseLocation(gestureRecognizer:)))
        gestureRecognizer.minimumPressDuration = 3
        mapView.addGestureRecognizer(gestureRecognizer)

        if selectedTitle != ""{
            let appDelegate = UIApplication.shared.delegate as! AppDelegate
            let context = appDelegate.persistentContainer.viewContext
            let request = NSFetchRequest<NSFetchRequestResult>(entityName: "places")
            let idString = selectedTitleID!.uuidString
            request.predicate = NSPredicate(format: "id = %@", idString)
            request.returnsObjectsAsFaults = false
            do{
                let results = try context.fetch(fetchRequest)
                if results.count > 0{
                    for result in sesults as! [NSManagedObject]{
                        if let title = result.value(forKey: "title") as? String{
                            annotationTitle = title
```

↓ similar thing for subtitle, latitude, longitude (do it as nested if functions

```
                            let annotation = MKPointAnnotation()
                            annotation.title = annotationTitle
                            annotation.subtitle = annotationSubtitle
                            let coordinate = CLLocationCoordinate2D(latitude: annotationLatitude,
```

```
                                longitude: annotationLongitude)
                annotation.coordinate = coordinate
                mapView.addAnnotation(annotation)
                nameText.text = annotationTitle
                nameText.text = annotationSubtitle

                locationManager.stopUpdatingLocation()
                let span = MKCoordinateSpan(latitudeDelta: 0.05, longitudeDelta: 0.05)
                let region = MKCoordinateRegion(center: coordinate, span: span)
                mapView.setRegion(region, animated: true)

            }
        }
    }
    } catch {
        print("error")
    }
} else {

}
}

@objc func chooseLocation(gestureRecognizer: UILongPressGestureRecognizer){
    if gestureRecognizer.state == .began {
        let touchedPoint = gestureRecognizer.location(in: self.mapView)
        let touchedCoordinates = self.mapView.convert(touchedPoint, toCoordinateFrom: self.mapView)
        chosenLatitude = touchedCoordinates.latitude
        chosenLongitude = touchedCoordinates.longitude
        let annotation = MKPointAnnotation()
        annotation.coordinate = touchedCoordinates
        annotation.title = "New Annotation"
        annotation.subtitle = "Travel Book"
        self.mapView.addAnnotation(annotation)
    }
}

func locationManager(_ manager: CLLocationManager, didUpdateLocations locations: [CLLocation]){
    if selectedTitle == "" {
        let location = CLLocation(coordinate2D(latitude: locations[0].coordinate.latitude,
                                               longitude: locations[0].coordinate.longitude)
        let span = MKCoordinateSpan(latitudeDelta: 0.05, longitudeDelta: 0.05)
        let region = MKCoordinateRegion(center: location, span: span)
        mapView.setRegion(Region, animated: true)
    }
}

func mapView(_ mapView: MKMapView, viewFor annotation: MKAnnotation) -> MKAnnotationView? {
    if annotation is MKUserLocation {
        return nil
    }
```

```
        let reuseId = "myAnnotation"
        var pinView = mapView.dequeueReusableAnnotationView(withIdentifier: reuseId) as? MKPinAnnotationView
        if pinView == nil{
            pinView = MKPinAnnotationView(annotation: annotation, reuseIdentifier: reuseId)
            pinView?.canShowCallout = true
            pinView?.tintColor = UIColor.black

            let button = UIButton(type: UIButton.ButtonType.detailDisclosure)
            pinView?.rightCalloutAccessoryView = button
        } else{
            pinView?.annotation = annotation
        }
        return pinView
    }

func mapView(_ mapView: MKMapView, annotationView view: MKAnnotationView, calloutAccessoryControlTapped control: UIControl){
    if selectedTitle != ""{      // we have a chosen latitude and chosen longitude
        var requestLocation = CLLocation(latitude: annotationLatitude, longitude: annotationLongitude)
        CLGeocoder().reverseGeocodeLocation(requestLocation){(placemarks, error) in
            if let placemark = placemarks{
                if placemark.count > 0{
                    let newPlacemark = MKPlacemark(placemark: placemark[0])
                    let item = MKMapItem(placemark: newPlacemark)
                    item.name = self.annotationTitle
                    let launchOptions = [MKLaunchOptionsDirectionsModeKey: MKLaunchOptionsDirectionsModeDriving]
                    item.openInMaps(launchOptions: launchOptions)
                }
            }
        }
    }
}

// CoreData => Attribute => entities -> places
//                          id: UUID
//                          longitude: Double
//                          latitude: Double
//                          subtitle: String
//                          title: String

@IBAction func saveButtonClicked(_ sender: Any){
    let appDelegate = UIApplication.shared.delegate as! AppDelegate
    let context = appDelegate.persistentContainer.viewContext

    let newPlace = NSEntityDescription.insertNewObject(forEntityName: "Places", into: context)

    newPlace.setValue(nameText.text, forKey: "title")
    newPlace.setValue(commentText.text, forKey: "subtitle")
    newPlace.setValue(chosenLatitude, forKey: "latitude")
    newPlace.setValue(chosenLongitude, forKey: "longitude")
    newPlace.setValue(UUID(), forKey: "id")
    do{
        try context.save()
    }catch{
        print("error")
    }

    NotificationCenter.default.post(name: NSNotification.Name("newPlace"), object: nil)
    navigationController?.popViewController(animated: true)
}
```

// ListViewController

```
import UIKit
class ListViewController: UIViewController, UITableViewDelegate, UITableViewDataSource{
    @IBOutlet weak var tableView: UITableView!
    var titleArray = [String]()
    var idArray = [UUID]()
    var chosenTitle = ""
    var choseTitleId : UUID?

    override func viewDidLoad(){
        super.viewDidLoad()
        navigationController?.navigationBar.topItem?.rightBarButtonItem = UIBarButtonItem(barButtonSystemItem:
            UIBarButtonItem.SystemItem.add, target: self, action: #selector(addButtonClicked))
        tableView.delegate = self
        tableView.dataSource = self
        getData()
    }

    override func viewWillAppear(_ animated: Bool){
        NotificationCenter.default.addObserver(self, selector: #selector(getData), name: NSNotification.Name("newPlace"), object: nil)
    }

@objc func getData(){
    let appDelegate = UIApplication.shared.delegate as! AppDelegate
    let context = appDelegate.persistentContainer.viewContext

    let request = NSFetchRequest<NSFetchRequestResult>(entityName: "places")
    request.returnsObjectsAsFaults = false
    do{
        let results = try context.fetch(request)
        if results.count > 0 {
            self.titleArray.removeAll(keepingCapacity: false)
            self.idArray.removeAll(keepingCapacity: false)
            for result in results as [NSManagedObject]{
                if let title = result.value(forKey: "title") as? String{
                    self.titleArray.append(title)
                }
                if let id = result.value(forKey: "id") as? UUID{
                    self.idArray.append(id)
                }
                tableView.reloadData()
            }
        }
    }
}
```

```
@objc func addButtonClicked(){
    chosenTitle = ""
    performSegue(withIdentifier: "toViewController", sender: nil)
}
func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
    return titleArray.count
}

func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
    let cell = UITableViewCell()
    cell.textLabel?.text = titleArray[indexPath.row]
    return cell
}

func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath){
    chosenTitle = titleArray[indexPath.row]
    chosenTitleId = idArray[indexPath.row]
    performSegue(withIdentifier: "toViewController", sender: nil)
}

override func prepare(for segue: UIStoryboardSegue, sender: Any?){
    if segue.identifier == "toViewController"{
        let destinationVC = segue.destination as! ViewController
        destinationVC.selectedTitle = chosenTitle
        destinationVC.selectedTitleID = chosenTitleID
    }
}
```