# Q1. (2 Marks)

**Explain the ways that we can follow to save data on iOS, use diagram to support your answer.**

1. **File Data Storage**

• iOS enables saving data in files as a primary way as all operating systems. The techniques is simple.

•iOS, apps are isolated from each other and from the OS as well.

• Persistent storage is a mutable key-value store, meaning that data stored can be changed and retrieved at any time. Each value is associated with a key used to identify it when it is being written to or read from. Each installed app may have its own persistent storage with its own collection of fields that by default consists of a few standard directories: Documents, Library, and tmp. As a developer

• The Documents folder is backed up when the device is backed up. The tmp folder is not (temporary only).

• Persistent storage supports saving integers, strings and byte arrays. The maximum size of byte arrays and strings is currently set to 256 bytes.

1. **User Defaults**

• **NSUserDefaults** object is a simple way when you need to save settings and properties related to application or user data. For example, you could save a profile image set by the user or a default color scheme for the application. The objects will be saved in what is known as the iOS “defaults system”.

• User preferences are stored as key-value pairs. The key is a unique string that identifies a specific preference within a specific domain. .

• With **NSUserDefaults** you can save objects from the following class types: NSData, NSString, NSNumber, NSDate, NSArray, NSDictionary (Other data types can be archived and stored as an NSData object.)

1. **Core Data**

• Core Data is a data persistence powerful technique providing object-oriented storage.

• Core Data is based on the top of data store, which is SQLite database, makes working with objects easier in the database without worry about the underlying DB design and queries.

• Core Data framework stores data in files, either SQLite database (the default), or XML or binary data.

**The core data framework shown in the chart:**

◘ **Persistent Object Store** presents an interface, and all what you need to do is specify which file format to store the data in.

◘ **Persistent Store Coordinator** allows for having multiple data stores in the same app, and coordinates access to those stores.

◘ **Managed Object Model** is the description of the layout and structure of the data.

◘ **Managed Object Context**, allows to access the objects that are stored in the data file and keep track of multiple changes to the objects,and save the changes to the persistent store.

# Q2.(2 Marks)

**Discuss the functionality of the following files in iOS application:**

1. **AppDelegate.h and AppDelegate.m**

The App Delegate files defines instructions and manage issues related to the entire app and the life cycle of the app. typically, .h files are header files, while .m are implementation files.

• Header file (.h): .it contains the declaration for my classes. Here we write all of methods & attributes & instance variable I'm going to use in my program.

• Implementation file (.m) : write the actual code for those methods " you tell the class how to actually does its work".



1. **Main.storyboard**

• Used with iPhone5 and above, The Storyboard describes the views and app flow.

• The storyboard is used to design the interaction between multiple screens in your app as well as designing the layout of the individual screens.

1. **ViewController.h and ViewController.m**

• The view controller contains the code that controls the user interactions with the app.

• **AppDelegate** is used for the whole app, you can use it to manage the app life cycle. On the other hand, **ViewController** is used for a single view. you can use it to manage life cycle of a view. So, One app can have multiple views, but only one AppDelegate.

**Simply, iPhone file flow goes as following;**

1. The AppDelegate object is the starting point to the application.
2. the AppDelegate is responsible for creating a ViewController object when it notices that your app has launched.
3. The ViewController object is responsible for setting up the view as described in Main.storyboard and showing it on the screen to the user.

••••••••••