GEMRAIN CONSULTING

iOS APPLICATION DEVELOPMENT – INTERMEDIATE

iOS-101

Duration: 3 days; Instructor-led | Virtual Instructor-led

WHAT WILL YOU LEARN

This is an intermediate iOS Development course.
During this course, you will create a reasonably complex iOS program. In the process, it covers storyboards, table view, table view controller, editing table view, navigation controllers, camera, auto layout, gesture recognisers, push view controllers and modal view controllers. You will also learn how to use unit tests and debugging tools, and how to upload your app to the App Store.

Programs you will write:
Food Tracker - A simple meal tracking app. This app shows a list of meals, including a meal name, rating, and photo. A user can add a new meal and remove or edit an existing meal. To add a new meal or edit an existing one, users navigate to a different screen where they can specify a name, rating, and photo for a particular meal.

METHODOLOGY

This program will be conducted with interactive lectures, PowerPoint presentations, discussions, and practical exercises.

OBJECTIVES

Upon completion of this program, participants should be able to:

- Create a reasonably complex iOS application, using storyboards, table view, table view controller, navigation controllers, camera, auto layout, gesture recognisers, push view controllers and modal view controllers.
- Test an application using unit tests and debugging tools.
- Save application data so it persists between runs.
- Upload the app to the App Store.

PRE-REQUISITES

Knowledge of a programming language is advantageous.

SYSTEM REQUIREMENTS

- A Mac computer running macOS 10.15 Catalina or 11.0 Big Sur
- Xcode 12.x
- Swift 5.x

REFERENCES

- https://docs.swift.org/swift-book/GuidedTour/GuidedTour.html
- https://developer.apple.com/library/archive/referencelibrary/GettingStarted/DevelopiOSAppsSwift/index.html
- https://itunes.apple.com/us/book/app-development-with-swift/id1219117996?mt=11

COURSE CONTENT

Module 1: Getting Started

- Get the tools.
- Your first program
- Simple values
- Explicit type declaration
- Conversion
- Including values in strings
- Arrays and Dictionaries
- Initializer syntax
- Control flow
- Optionals
- ?? operator
- Switches
- for-in
- while
- Loop indexes
- Functions
- Argument labels
- Tuples
- Variable number of arguments
- Nested functions
- Functions as return values.
- Functions as arguments
- Closures
- Writing closures more concisely
- Class
- Instance
- Initializers
- Subclass
- Property getters and setters
- willSet and didSet
- Working with optional values
- Enumerations
- Raw values

- init?(rawValue:)
- Case values
- Reference
- Values associated with case.
- Structures
- Protocols
- Adopting a protocol (class)
- Adopting a protocol (struct)
- Extension
- Protocol type
- Error handling
- Throw and Throws
- Do-Catch
- Multiple catch blocks
- try?
- Defer

Module 2: Build a basic UI

- Create a project in Xcode.
- Identify the function of key files that are created with an Xcode project template.
- Open and switch between files in a project
- Run an app in Simulator.
- Add, move, and resize UI elements in a storyboard.
- Edit the attributes of UI elements in a storyboard using the Attributes inspector.
- View and rearrange UI elements using the outline view.
- Preview a storyboard UI using the Preview assistant editor.
- Lay out a UI that automatically adapts to the user's device size using Auto Layout

Module 3: Connect the UI to code.

- Explain the relationship between a scene in a storyboard and the underlying view controller.
- Create outlet and action connections between UI elements in a storyboard and source code.
- Process user input from a text field and display the result in the UI.
- Make a class conform to a protocol.
- Understand the delegation pattern.
- Follow the target-action pattern when designing app architecture.

Module 4: Work with View Controllers

- Understand the view controller life cycle and when its callbacks occur, such as viewDidLoad, viewWillAppear and viewDidAppear.
- Pass data between view controllers
- Dismiss a view controller.
- Use gesture recognizers as an additional level of generating events.
- Anticipate object behaviour based on the UIView/UIControl class hierarchy.
- Use the asset catalogue to add image assets to a project.

Day 2

Module 5: Implement a Custom Control

- Create and associate custom source code files with elements in a storyboard.
- Define a custom class.
- Implement an initializer on a custom class.
- Use UIView as a container.
- Understand how to display views programmatically.

Module 6: Define your data model.

- Create a data model.
- Write failable initializers on a custom class.
- Demonstrate a conceptual understanding of the difference between failable and nonfailable initializers.
- Test a data model by writing and running unit tests.

Module 7: Create a table view.

- Understand the key components of a table view.
- Create and design a custom table view cell.
- Understand the roles of table view delegates and data sources.
- Use an array to store and work with data.
- Display dynamic data in a table view.

Module 8: Implement Navigation

- Embed an existing view controller within a navigation controller in a storyboard.
- Create segues between view controllers.
- Edit the attributes of a segue in a storyboard using the Attributes inspector.
- Pass data between view controllers using the prepareForSegue(_:sender:) method
- Perform an unwind segue.
- Use stack views to create robust, flexible layouts.

Day 3

Module 9: Implement Edit and Delete behaviour.

- Differentiate between push and modal navigation.
- Dismiss view controllers based on their presentation style.
- Understand when to use different type cast operators for down casting.

- Leverage optional binding to check for complex conditions.
- Use segues identifiers to determine which segue is occurring.

Module 10: Persist Data
- Create a structure.
- Understand the difference between static properties and instance properties.
- Use the NSCoding protocol to read and write data.

Module 11: Uploading your app to the App Store
- Getting an Apple Developer Account
- Exploring the Apple Developer Account
- Submitting your app to the App Store
- Internal and external testing