College of San Mateo
Official Course Outline

1. **COURSE ID:** CIS 137 **TITLE:** iOS/Swift Programming
   Units: 4.0 units **Hours/Semester:** 48.0-54.0 Lecture hours; 48.0-54.0 Lab hours; and 96.0-108.0 Homework hours
   Method of Grading: Grade Option (Letter Grade or Pass/No Pass)
   Recommended Preparation:
   CIS 254

2. **COURSE DESIGNATION:**
   Degree Credit
   Transfer credit: CSU; UC
   AA/AS Degree Requirements:
   CSM - GENERAL EDUCATION REQUIREMENTS: E2b. Communication and Analytical Thinking

3. **COURSE DESCRIPTIONS:**
   Catalog Description:
   Introduction to the architecture, API and techniques used to create robust, high-performance apps for iOS mobile devices (iPhone, iPad and wearable) with the Swift programming language. An overview of the most common tools and techniques for designing and creating iOS mobile apps. Other topics include object-oriented programming, user interface design, Storyboards, SwiftUI, MVC design pattern, UIKit, multimedia, debugging, sensors, and user events. Storage strategies for persistent information are introduced, including the Core Data framework and the use of SQLite database features. Intended for students with previous programming experience.

4. **STUDENT LEARNING OUTCOME(S) (SLO'S):**
   Upon successful completion of this course, a student will meet the following outcomes:
   1. Explain the iOS mobile operating system architecture.
   2. Install and use appropriate tools for iOS development, including IDE, frameworks, and device simulator.
   3. Build user interfaces with SwiftUI, Storyboard and UI components.
   4. Employ the UIKit framework to create custom view controllers.
   5. Store application data on the mobile device, in internal or external storage locations.
   6. Create an advanced mobile app employing sensors, gestures, camera, GPS, maps, geolocation and other features.

5. **SPECIFIC INSTRUCTIONAL OBJECTIVES:**
   Upon successful completion of this course, a student will be able to:
   1. Explain the iOS mobile operating system architecture.
   2. Install and use appropriate tools for iOS development, including IDE, frameworks, and device simulator.
   3. Build user interfaces with SwiftUI, Storyboard and UI components.
   4. Employ the UIKit framework to create custom view controllers.
   5. Store application data on the mobile device, in internal or external storage locations.
   6. Create an advanced mobile app employing sensors, gestures, camera, GPS, maps, geolocation and other features.

6. **COURSE CONTENT:**
   Lecture Content:
   1. Introduction to Swift
      A. Swift Playgrounds
      B. The UI
      C. Variable types
      D. Arrays and dictionaries
      E. Any and AnyObject type
      F. Control flow and code execution
      G. Functions
      H. Closures
      I. Objects and Classes
      J. Struct and enum types

K. Tuples
L. Inheritance
M. Protocol

2. Introduction to Xcode and iOS
   A. Installing Xcode
   B. Xcode features
   C. Intro to the iOS SDK
   D. iOS Human Interface Guidelines
   E. Create a new project
   F. Project templates
   G. Simulator for iPhone, iPad, Apple Watch
   H. Interface Builder
3. Views and View Controllers
   A. Model-View-Controller (MVC) design
   B. Design views with Storyboards and segues
   C. Connect views to their view controllers with outlets
   D. Create and configure segues that allow users to transition to and from different scenes
   E. Define actions to respond to control events
   F. Use auto layout to create flexible and robust interfaces
   G. Localization
   H. Add images, gestures, and animations
   I. UIKit framework
   J. SwiftUI
4. Container View Controllers
   A. Generic view controller
   B. Cocoa Touch APIs
   C. Custom-designed container view controllers for split views, table views, tab views, custom views, and collections
   D. Create a data source to fill in a table view, and allow users to insert and remove data in a table
   E. Create hierarchical apps using navigation controllers
   F. Use tab bar controllers to create multi-mode apps
   G. Build a flexible UI that can run correctly on different models of the iPhone and iPad
   H. Web services
   I. Networking
5. Performance and Debugging
   A. Avoid and handle memory warnings
   B. Use Instruments to fix memory leaks and profile code
   C. Use the debugging tools built into Xcode
   D. Troubleshoot common errors and warnings
   E. Breakpoints
   F. Navigation
   G. Console
6. Data Storage
   A. Files
   B. Core Data framework
      a. Use modeling tools to model persistent app data
      b. Create, read (fetch), update, and delete persistent entities
      c. Manage model relationships
   C. SQLite database
   D. iCloud
7. Building and Deploying Apps
   A. Prepare an app for distribution
   B. Configure project settings
      a. Identity section
      b. Deployment Info section
   C. Code signing
   D. Application icons and launch images
   E. Link frameworks and libraries
   F. App Store submission

Lab Content:

1. Storyboards
2. User Interface Design
3. Create Views and Container View Controllers
4. SwiftUI
5. Resources
6. Animation and Graphics
7. Computation
8. Handle User Events
9. Media and Camera
10. GPS, Location and Sensors
11. Connectivity
12. Text and Input
13. Use Core Data API
14. SQLite Database
15. Files
16. Testing and debugging
17. App Deployment

7. **REPRESENTATIVE METHODS OF INSTRUCTION:**
   Typical methods of instruction may include:
   A. Lecture
   B. Lab
   C. Activity
   D. Directed Study
   E. Discussion
   F. Observation and Demonstration
   G. Other (Specify): • Student reading of textbooks and supplemental course materials • Individual and team programming projects • Review of subject matter videos

8. **REPRESENTATIVE ASSIGNMENTS**
   Representative assignments in this course may include, but are not limited to the following:
   Writing Assignments:
   Students will complete and submit exercises and programming assignments on a weekly or biweekly basis.
   Reading Assignments:
   Students will read assigned chapters in the textbook and supplemental handouts.

9. **REPRESENTATIVE METHODS OF EVALUATION**
   Representative methods of evaluation may include:
   A. Class Work
   B. Exams/Tests
   C. Group Projects
   D. Homework
   E. Lab Activities
   F. Projects
   G. Quizzes
   H. Written examination

10. **REPRESENTATIVE TEXT(S):**
    Possible textbooks include:
    A. Keur, C., Hillegass, A. *iOS Programming*, 7th ed. Big Nerd Ranch Guides, 2020
    B. Sahar, A. *iOS 14 Programming for Beginners*, 5th ed. Packt, 2020
    C. Gray, A., Manning, J. et al. *Head First Swift*, 1st ed. O'Reilly, 2021
    D. Smyth, N. *SwiftUI Essentials - iOS Edition*, 1st ed. Payload Media, 2019

Origination Date: November 2020
Curriculum Committee Approval Date: November 2020
Effective Term: Fall 2021
Course Originator: Melissa Green