บริษัท ศิลา นักปราชญ์ จำกัด

We believe that technology has the power to transform the classroom.

# MVVM in SwiftUI


## โดย ธิติ ธีระเธียร

Apple Certified Trainer (App Development with Swift)



**Issue:** January 2026


# The Massive View

| | |
|---|---|
| Name | : Jessica Anderson |
| Date of Birth | : 28 Aug 1979 |
| Gender | : Female |
| Weight | : 65.7 kg |
| Height | : 162.5 cm |
| Occupation | : Artist |
| Salary | : 32,000 BTH |
| Status of Membership | : Active |

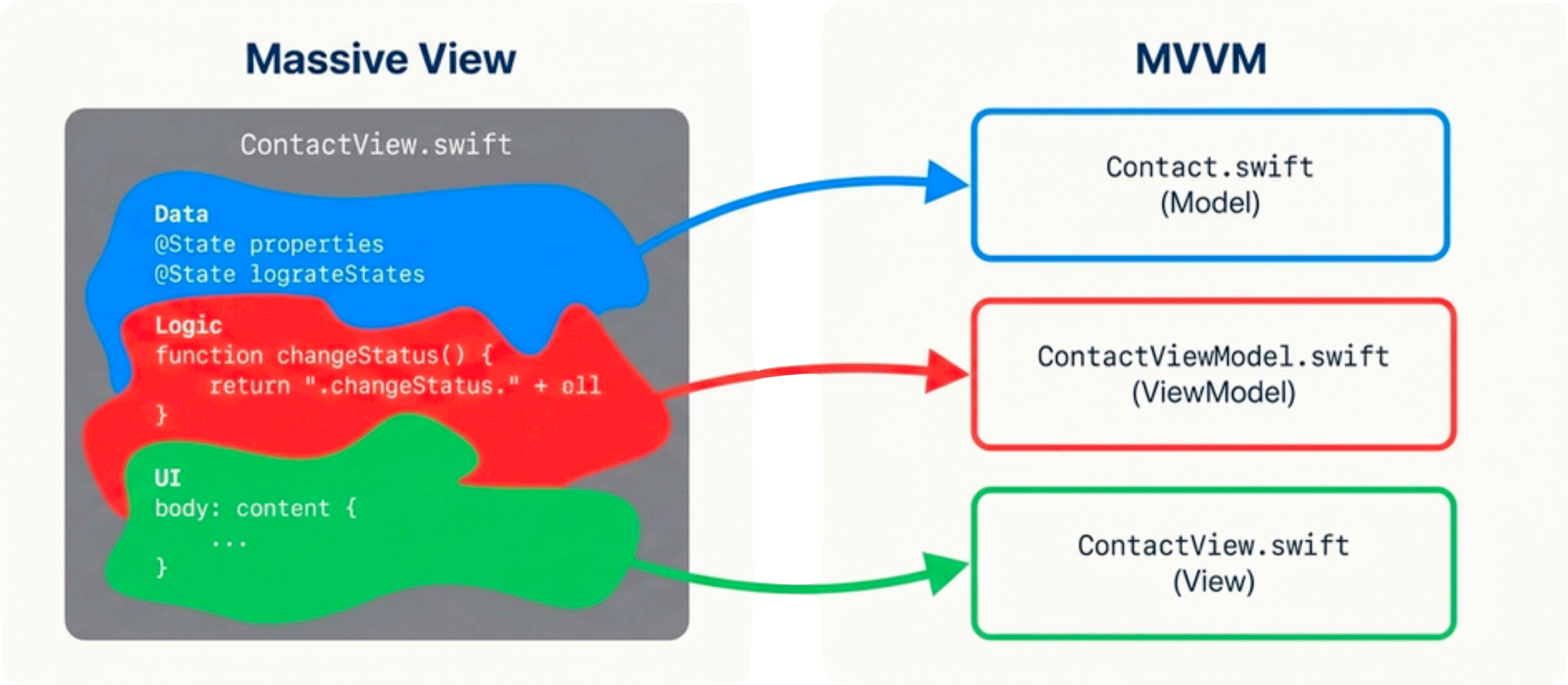
```swift
struct ContactView: View {

    // เก็บข้อมูล Contact ของสมาชิกไว้ใน View (ไม่แยก Model)
    @State private var name: String = "Jessica Anderson"
    @State private var dateOfBirth: String = "28 Aug 1979"
    @State private var gender: String = "Female"
    @State private var occupation: String = "Artist"
    @State private var salary: Int = 32000
    @State private var statusOfMembership: Bool = true

    var body: some View {
        VStack(alignment: .leading, spacing: 8) {
            // การแสดงผล UI
            Text("Name: \(name)")
            Text("Date of Birth: \(dateOfBirth)")
            Text("Gender: \(gender)")
            Text("Occupation: \(occupation)")
            Text("Salary: \(salary) THB")
            Text("Status: \(statusOfMembership ? "Active" : "Inactive")")
                .bold()
                .foregroundColor(statusOfMembership ? .green : .red)
                .padding(.top, 10)

            // ปุ่มที่มี Logic อยู่ใน View โดยตรง
            Button("Change Status") {
                changeStatus()          // UI ควบคุม Logic เอง
            }
            .buttonStyle(.borderedProminent)
            .padding(.top, 12)
        }
        .padding()
    }

    // Logic: การเปลี่ยนสถานะสมาชิก ถูกสร้างอยู่ใน View
    func changeStatus() {
        statusOfMembership.toggle()
    }
}
```

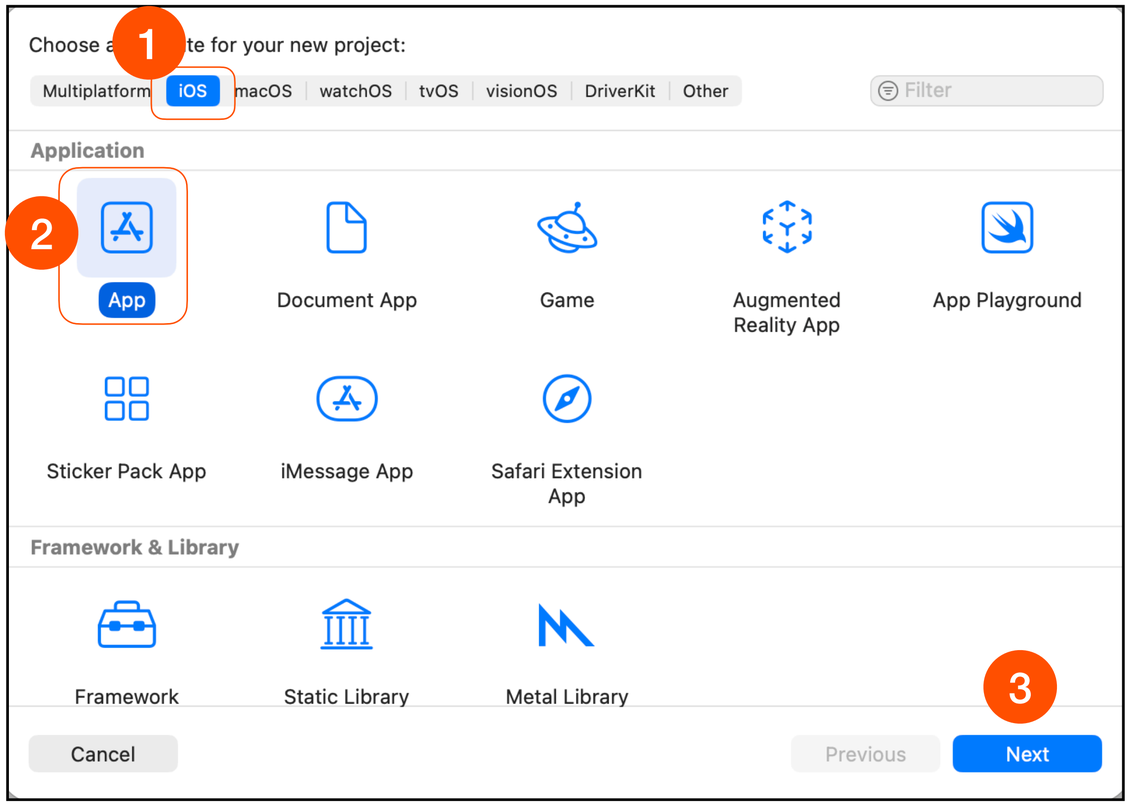
Data

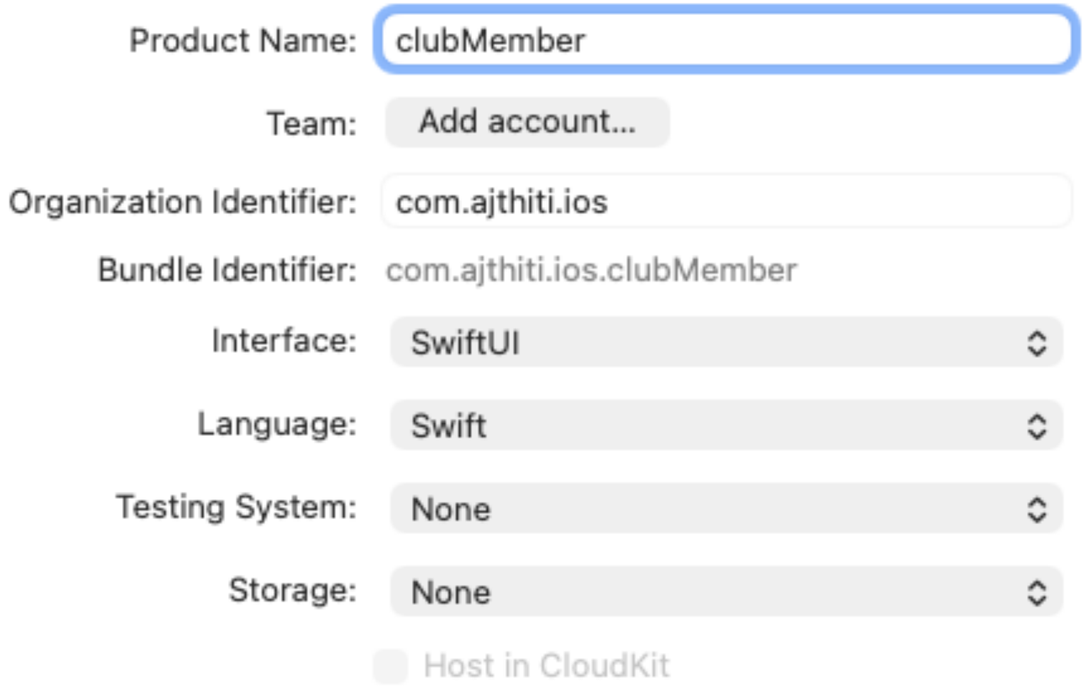
UI

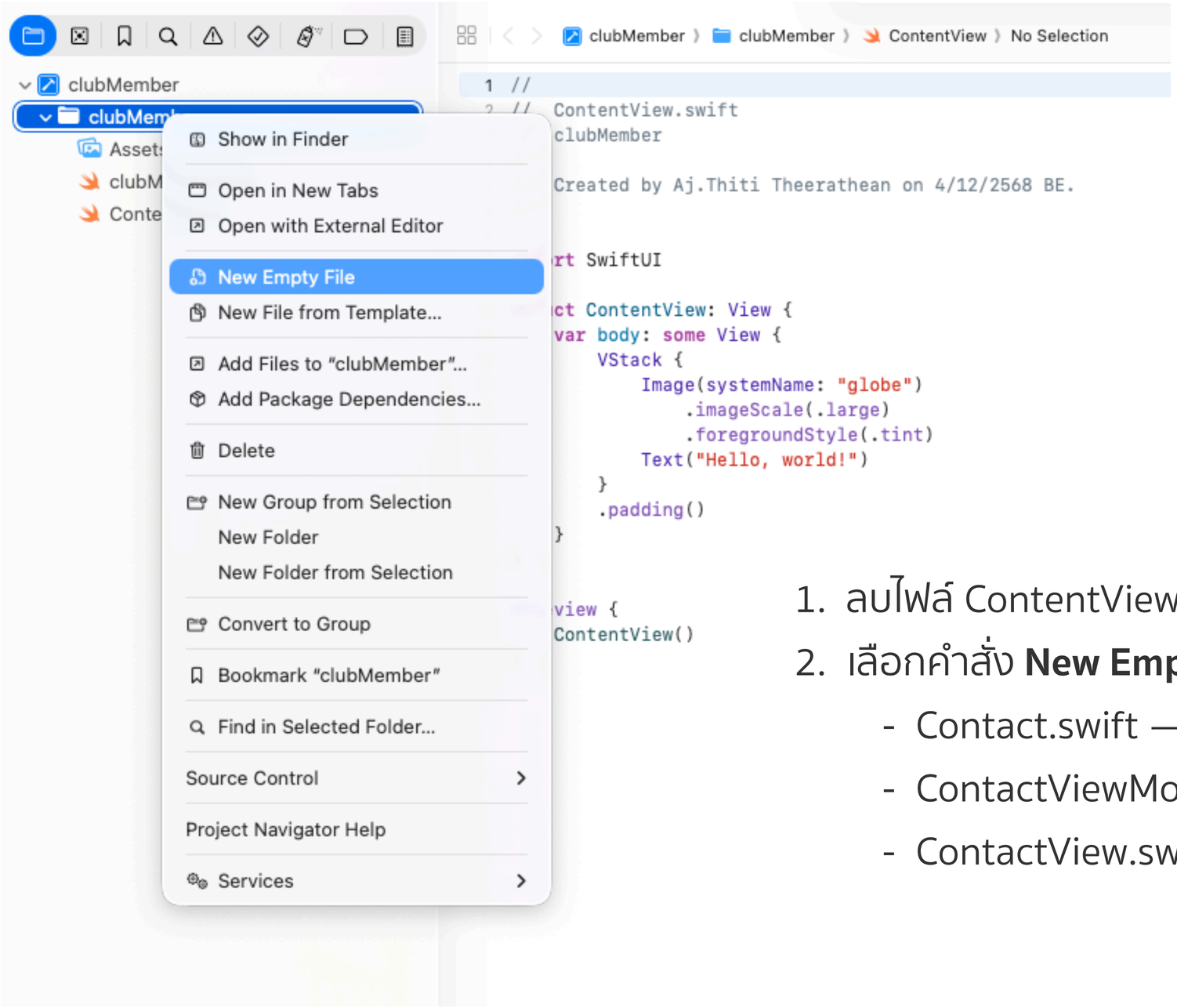
Logic

# Model–View–ViewModel (MVVM)




1
te for your new project:
Multiplatform
iOS
macOS
watchOS
tvOS
visionOS
DriverKit
Other
Filter
Application
2
App
Document App
Game
Augmented Reality App
App Playground
Sticker Pack App
iMessage App
Safari Extension App
Framework & Library
Framework
Static Library
Metal Library
3
Cancel
Previous
Next


Product Name:
clubMember
Team:
Add account...
Organization Identifier:
com.ajthiti.ios
Bundle Identifier:
com.ajthiti.ios.clubMember
Interface:
SwiftUI
Language:
Swift
Testing System:
None
Storage:
None
Host in CloudKit




1. ลบไฟล์ ContentView.swift ออกจาก Project
2. เลือกคำสั่ง **New Empty File** เพื่อสร้างไฟล์ใหม่ จำนวน 3 ไฟล์ ดังนี้
   - Contact.swift —> Model
   - ContactViewModel.swift —> ViewModel
   - ContactView.swift —> View

```swift
import Foundation

struct Contact {
    let name: String
    let dateOfBirth: String
    let gender: String
    var weight: Double
    var height: Double
    var occupation: String
    var salary: Int
    var statusOfMembership: Bool
}
```

```swift
import SwiftUI
import Combine

class ContactViewModel: ObservableObject {

    @Published var clubMember = Contact(
        name: "Jessica Anderson",
        dateOfBirth: "28 Aug 1979",
        gender: "Female",
        weight: 65.7,
        height: 162.5,
        occupation: "Artist",
        salary: 32000,
        statusOfMembership: true
    )

    func changeStatus() {
        clubMember.statusOfMembership.toggle()
    }
}
```

```swift
@StateObject private var viewModel = ContactViewModel()
```

# การใช้ Combine เพื่อการอัปเดต UI แบบ Reactive

**ObservableObject**

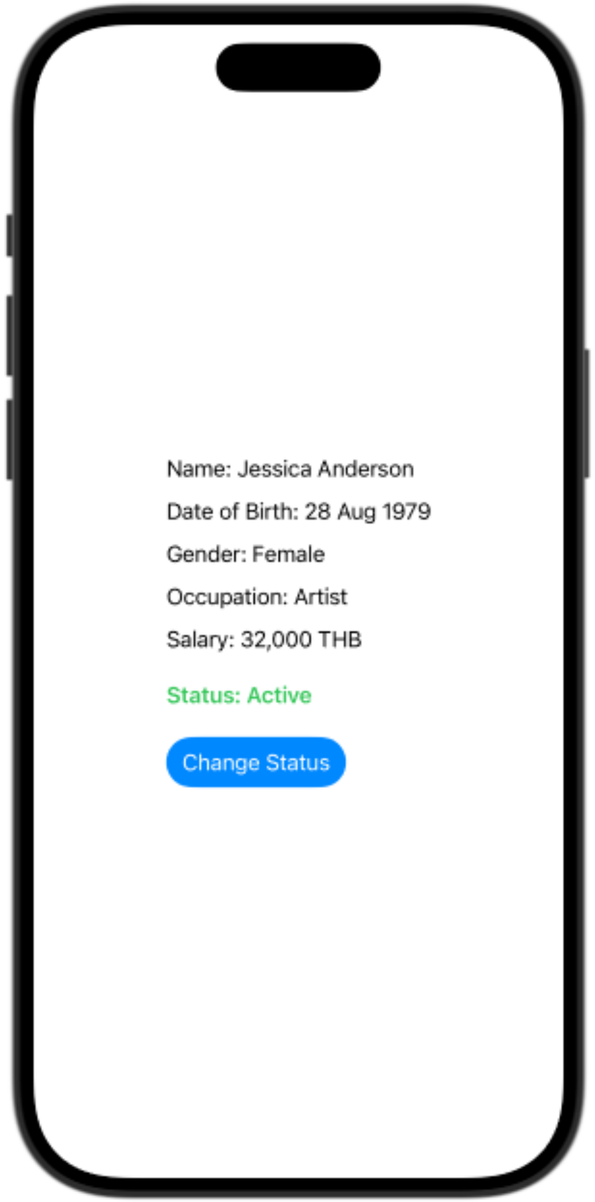
เป็น Protocol ที่ทำให้ ViewModel สามารถส่งสัญญาณเพื่อแจ้งการเปลี่ยนแปลงข้อมูล

**@Published**

เป็น Property Wrapper ที่ทำหน้าที่ประกาศให้ทราบทันทีเมื่อค่าหรือสถานะของตัวแปรมีการเปลี่ยนแปลง

**@StateObject**

เป็น Property Wrapper ที่ทำให้ View ติดตามการเปลี่ยนแปลงจาก ObservableObject

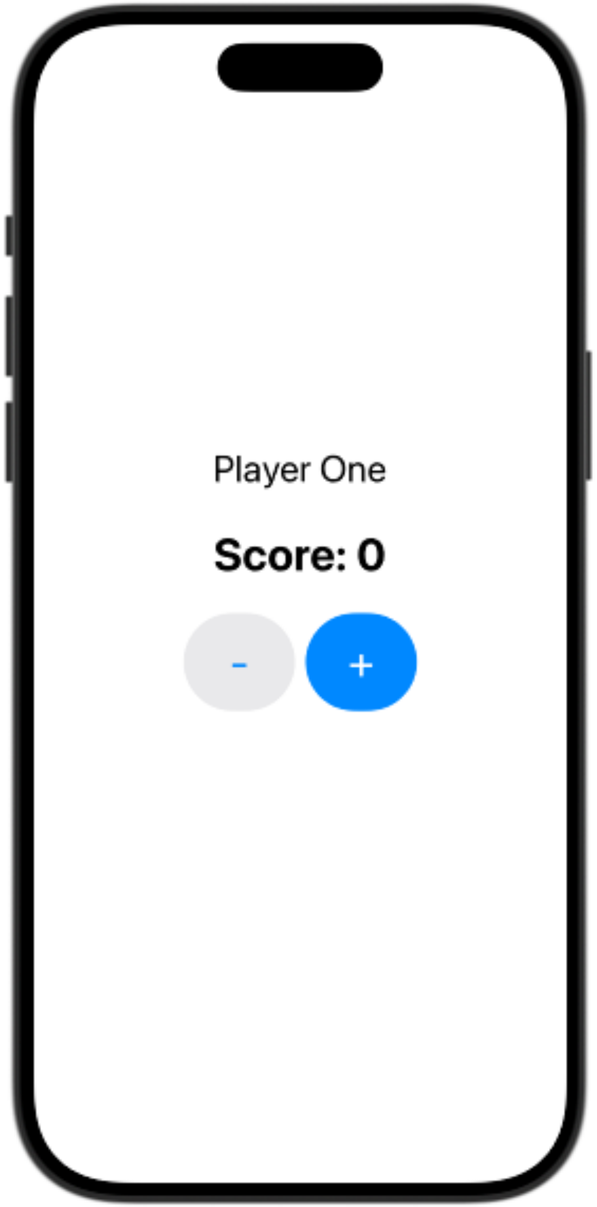


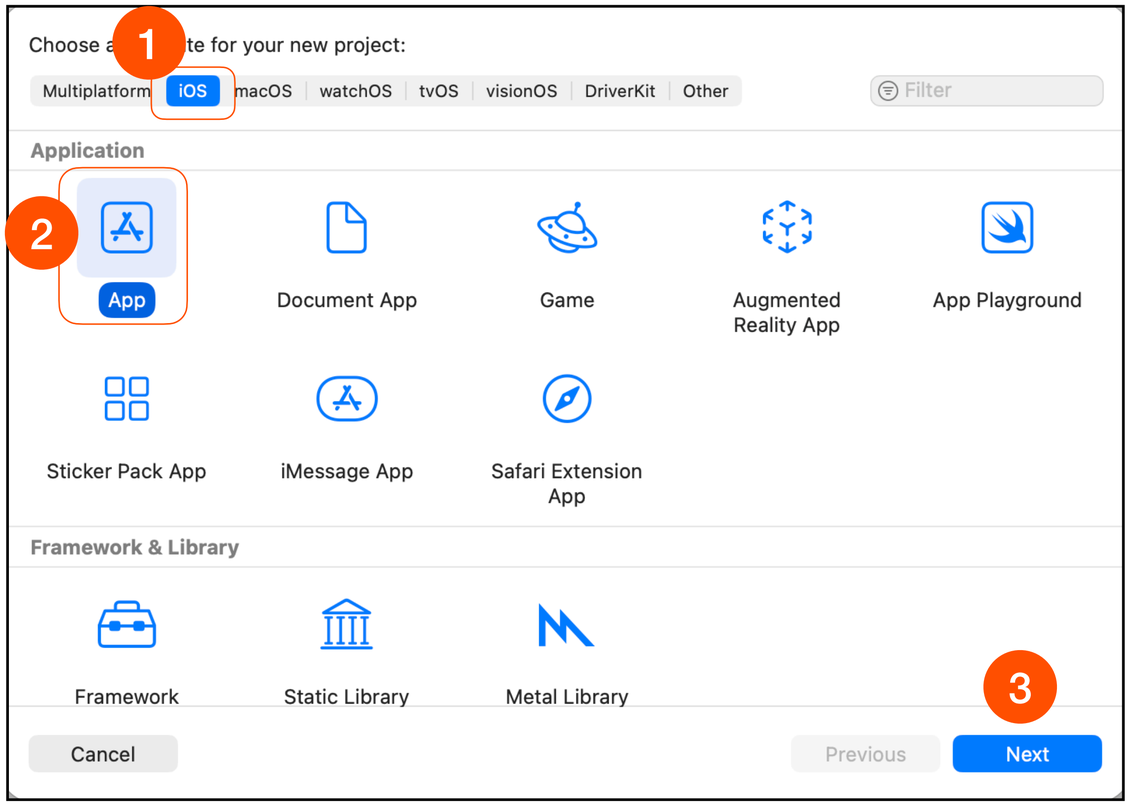

```swift
import SwiftUI

struct ContactView: View {

    @StateObject private var viewModel = ContactViewModel()

    var body: some View {
       VStack(alignment: .leading, spacing: 8) {
            Text("Name: \(viewModel.clubMember.name)")
            Text("Date of Birth: \(viewModel.clubMember.dateOfBirth)")
            Text("Gender: \(viewModel.clubMember.gender)")
            Text("Occupation: \(viewModel.clubMember.occupation)")
            Text("Salary: \(viewModel.clubMember.salary) THB")
            Text("Status: \(viewModel.clubMember.statusOfMembership ? "Active" : "Inactive")")
                .bold()
                .foregroundColor(viewModel.clubMember.statusOfMembership ? .green : .red)
                .padding(.top, 10)

            Button("Change Status") {
                viewModel.changeStatus()
            }
            .buttonStyle(.borderedProminent)
            .padding(.top, 12)
        }
        .padding()
    }
}

#Preview {
    ContactView()
}
```

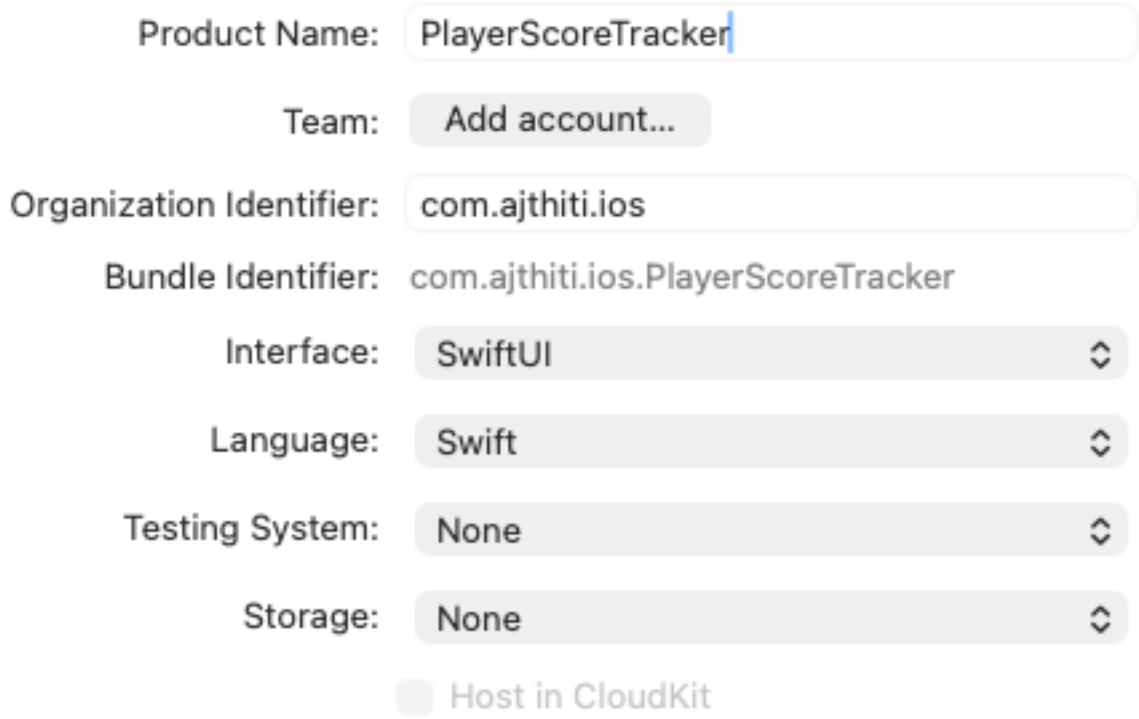
# การสร้างแอป Player Score Tracker ด้วย MVVM

ออกแบบ "ระบบติดตามคะแนนผู้เล่นอย่างง่าย" ประกอบด้วย

- การแสดงชื่อและคะแนนของผู้เรียน (เริ่มต้นเป็น 0)
- ปุ่มเพื่อเพิ่มคะแนน
- ปุ่มเพื่อลดคะแนน
- เงื่อนไข: คะแนนต้องไม่ติดลบ




1
Choose a te for your new project:
Multiplatform
iOS
macOS
watchOS
tvOS
visionOS
DriverKit
Other
Filter
Application
2
App
Document App
Game
Augmented Reality App
App Playground
Sticker Pack App
iMessage App
Safari Extension App
Framework & Library
Framework
Static Library
Metal Library
3
Cancel
Previous
Next


Product Name: PlayerScoreTracker
Team: Add account...
Organization Identifier: com.ajthiti.ios
Bundle Identifier: com.ajthiti.ios.PlayerScoreTracker
Interface: SwiftUI
Language: Swift
Testing System: None
Storage: None
Host in CloudKit

## Model(Player.swift)

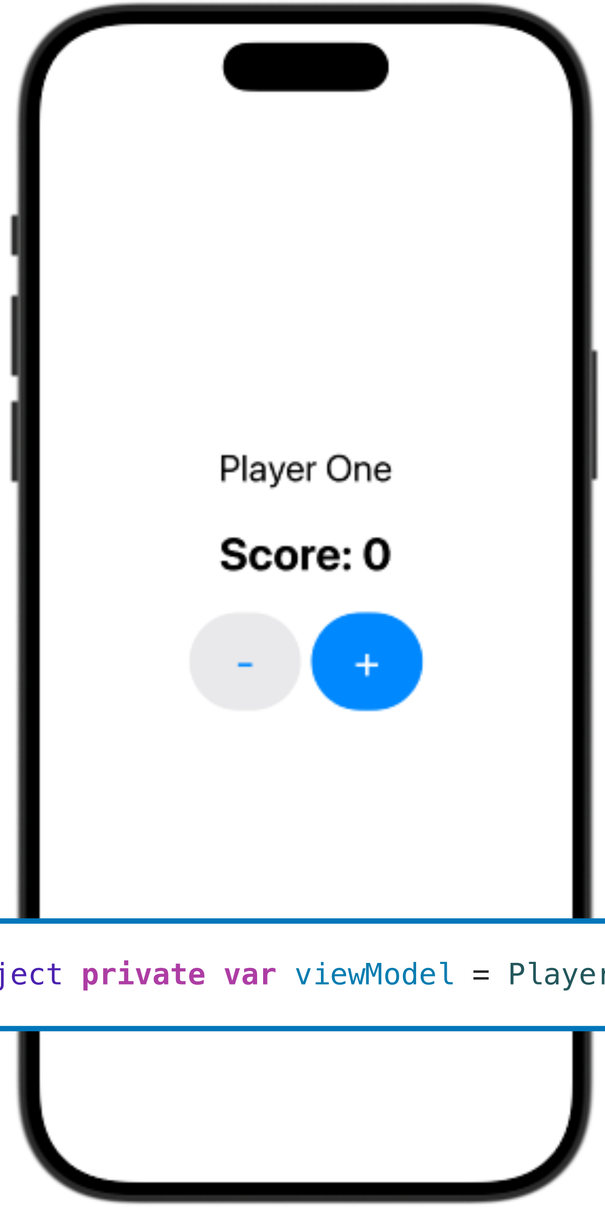
```swift
struct Player {
    var name: String
    var score: Int
}
```

## ViewModel(PlayerViewModel.swift)

```swift
class PlayerViewModel: ObservableObject {
    @Published var player = Player(name: "Player One", score: 0)

    func increaseScore() { player.score += 1 }
    func decreaseScore() {
        if player.score > 0 { player.score -= 1 }
    }
}
```

## View(PlayerView.swift)

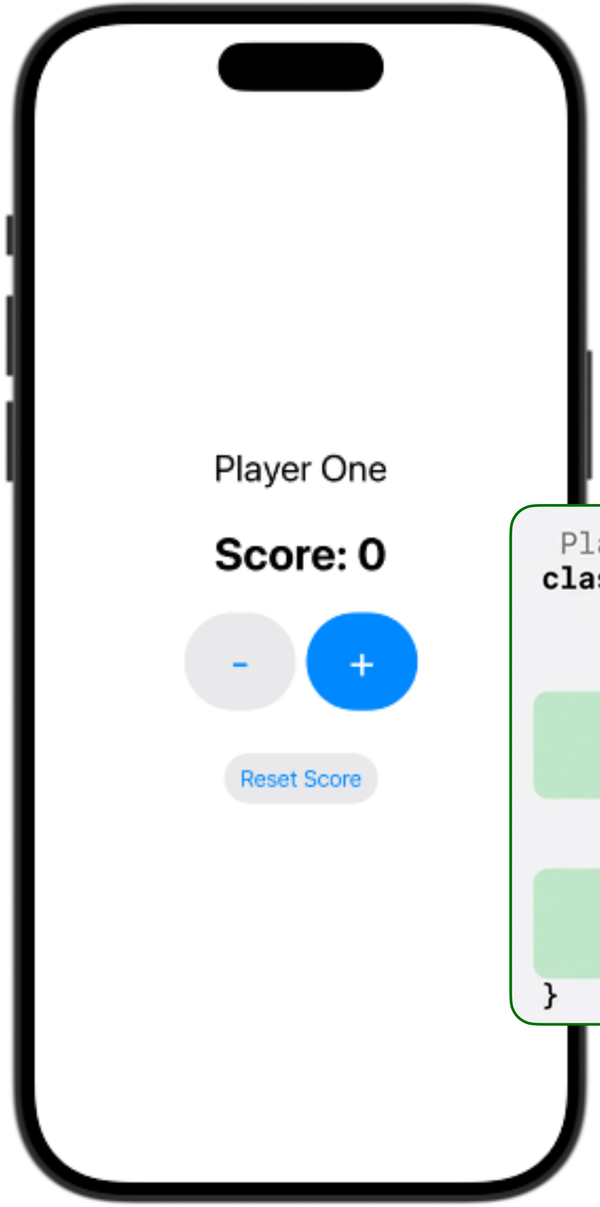
```swift
struct PlayerView: View {
    @StateObject private var viewModel = PlayerViewModel()
    var body: some View { /* UI Code calling viewModel functions */ }
}
```




```
@StateObject private var viewModel = PlayerViewModel()
```

```
VStack(spacing: 20) {

    Text(viewModel.player.name)
        .font(.title)

    Text("Score: \(viewModel.player.score)")
        .font(.largeTitle)
        .bold()

    HStack {
        Button(action: {
            viewModel.decreaseScore()
        }) {
            Text("-")
                .font(.largeTitle)
                .frame(width: 60, height: 60)
        }
        .buttonStyle(.bordered)

        Button(action: {
            viewModel.increaseScore()
        }) {
            Text("+")
                .font(.largeTitle)
                .frame(width: 60, height: 60)
        }
        .buttonStyle(.borderedProminent)
    }
}
.padding()
```

# การขยายฟีเจอร์ของแอป




- เพิ่มกฎว่า "ห้ามให้คะแนนเกิน 10"
- เพิ่มปุ่ม Reset Score

PlayerViewModel.swift

```swift
class PlayerViewModel: ObservableObject {
    @Published var player = Player(...)

    func increaseScore() {
        if player.score < 10 { // << เพิ่มกฎใหม่ที่นี่
            player.score += 1
        }
    }
    func decreaseScore() { ... }
    func resetScore() { // << เพิ่มฟังก์ชันใหม่
        player.score = 0
    }
}
```

```swift
Button("Reset") { viewModel.resetScore() }
```

# คำถามเพื่อการสะท้อนผลการเรียนรู้จากการทำกิจกรรม

1. นักเรียนสามารถอธิบายได้หรือไม่ว่า เหตุใดการรวมทุกอย่างไว้ใน View เดียวจึงทำให้โค้ดกลายเป็น Massive View และขัดกับหลัก Separation of Concerns (SoC)
2. การแยกโครงสร้างเป็น Model–View–ViewModel ช่วยให้ระบบซอฟต์แวร์จัดการง่ายขึ้นอย่างไรบ้าง? ยกตัวอย่างจากโค้ด Contact หรือ Player Score Tracker ที่สร้างขึ้น
3. เมื่อเพิ่มฟีเจอร์ใหม่ เช่น ปุ่ม Reset Score หรือ การตรวจสอบกฎ “ห้ามให้คะแนนเกิน 10” เราจะต้องจัดการฟีเจอร์เหล่านี้ในส่วนใดของ MVVM และ เพราะเหตุใดจึงเลือกปรับปรุงคำสั่งในส่วนนั้น ?
4. Reactive Programming คืออะไร และใน Combine Framework บทบาทของ `ObservableObject`, `@Published`, และ `@StateObject` ช่วยทำให้ SwiftUI ทำงานแบบ Reactive ได้อย่างไร ?
5. นักเรียนคิดว่าแนวคิด MVVM และ SoC เกี่ยวข้องกับแนวคิดทางวิศวกรรมซอฟต์แวร์อื่น ๆ เช่น Agile หรือ Clean Architecture อย่างไรบ้าง?