

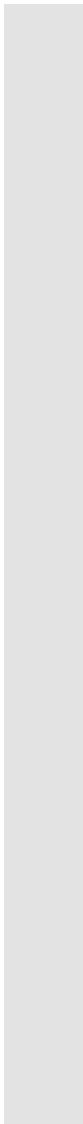
MedicalStatsApp

醫療數據資料iOS程式設計

使用語言:Swift

# 目錄

- 製作動機
- 系統規劃
- 系統雛型
- 製作流程
- 實機操作
- 製作心得
- 未來展望



# 製作動機

## 製作動機

- 家裡長輩需要於每天進行量測體溫、血氧、血壓，但一直以來都是使用紙本來登記，但因為要一直準備紙且紙會爛掉，所以就打算製作一個App，假設有狀況的時候也可以馬上將資料快速的調出來使用，也可以方便做相關的資料管理與監控。



▲ 圖為使用之設備



# 系統規劃

# 系統規劃

- 製作系統規劃的方式為使用Gherkin描述語法的Cucumber來撰寫，將整個系統的架構寫在.feature檔中，Scenario Outline寫頁面裡的主要動作。

```
Feature:Medical data Statistics

Scenario Outline:ContentView page
    Given An email <email>
    And an password <pw>
    When Click to the LOGIN
    Then login successful
    And Enter the ListView

Scenario Outline:ListView page
    Given Messages in to the TextField
    When Click to the Add
    Then Input message to Firebase FireStore database
    When Click to the Edit
    Then Can delete database
    When Click to the list
    Then Can update any database

    When click to the Logout
    Then Logout this system
    And Cannot CRUD the database
```



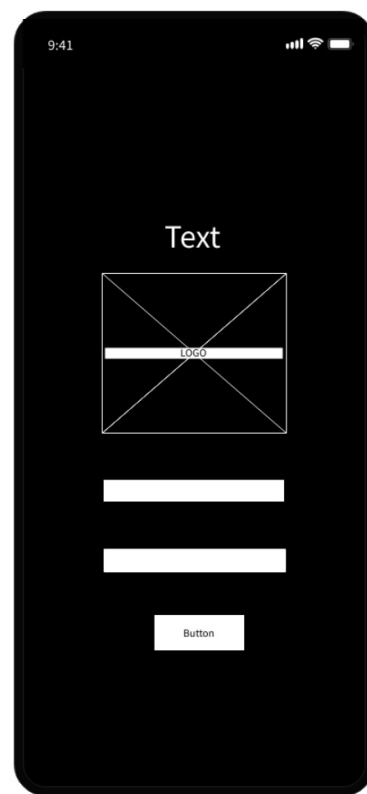
# 系統雛型

# 系統雛型

- 系統雛型利用Wireframe的Workflow去拉線，將整個專案做個demo，並照此雛型圖去製作此系統

ContentView

---





# 系統雛型

- 主要進行CRUD的頁面，此頁面將照系統規劃的內容進行製作

## ListView

---





# 製作流程

# 製作流程

- 先建立好主頁面的內容以及將一些controller製作結構

```
struct SignInView: View {
    @State public var email:String = ""
    @State public var pw:String = ""

    // @State var AuthenticationFAIL:Bool = false
    // | @State var AuthenticationSuccess:Bool = false

    @EnvironmentObject var viewModel: AppviewModel

    var body: some View {
        ZStack{
            VStack{
                Spacer()
                WelcomeText()
                UIImage()
                UserTextFieldContent(email: $email)
                UserSecureFieldContent(pw: $pw)
                /*
                if AuthenticationFAIL{
                    Text("Something was wrong.Please try
                        again.").offset(y:-10).foregroundColor(.red)
                }
                */

                Button(action: {
                    guard !email.isEmpty, !pw.isEmpty else {return}
                    viewModel.signIn(email: email, pw: pw)
                }) {
                    ButtonContent()
                }.buttonStyle(GradientBackGroundStyle()).padding()
                Spacer()
            }.padding()
        }
    }
}
```

# 製作流程

- 再建立一個類別來做驗證登入

```
class AppviewModel: ObservableObject{

    let auth = Auth.auth()
    @Published var signedIn = false

    var isSignedIn: Bool{
        return auth.currentUser != nil
    }
    func signIn(email: String,pw: String) {
        auth.signIn(withEmail: email, password: pw){
            [weak self]result, error in guard result != nil, error ==
                nil else{
                    return
                }
            DispatchQueue.main.async {
                self?.signedIn = true
            }
        }
    }
    func signOut() {
        try? auth.signOut()
        self.signedIn = false
    }
}
```

# 製作流程

- 之後就建立Database 並開啟驗證的Rule

## Cloud Firestore

資料規則索引用量

編輯規則監控規則開發及測試

★ 昨天 • 3:19 上午

○ 6月 17, 2021 • 3:16 下午

○ 6月 17, 2021 • 3:16 下午

○ 6月 17, 2021 • 11:39 上午

```
1 rules_version = '2';
2 service cloud.firestore {
3   match /databases/{database}/documents {
4     match /{document=**} {
5       allow read, write: if request.auth != null;
6     }
7   }
8 }
```

# 製作流程

- 並建立這Database的結構

```
import Foundation
import Firebase
import FirebaseFirestoreSwift

struct Message : Identifiable, Codable {

    @DocumentID var id : String?
    var msg : String
    var date : Timestamp

    enum CodingKeys : String, CodingKey {
        case id
        case msg = "message"
        case date
    }
}
```

# 製作流程

- 之後建立一個類別將ListView的資料進行CRUD

```
import Foundation
import SwiftUI
import Firebase
import FirebaseFirestoreSwift

class ListViewModel: ObservableObject {

    @Published var messages : [Message] = []

    let ref = Firestore.firestore()

    func getAllMessages() { ... }

    func addMessage(message: Message, completion: @escaping (Bool) -> ()) { ... }

    func deleteMessage(docId: Int) { ... }

    func updateMessage(message : String, docId: String, completion: @escaping (Bool)->()) { ... }
}
```



# 實際操作



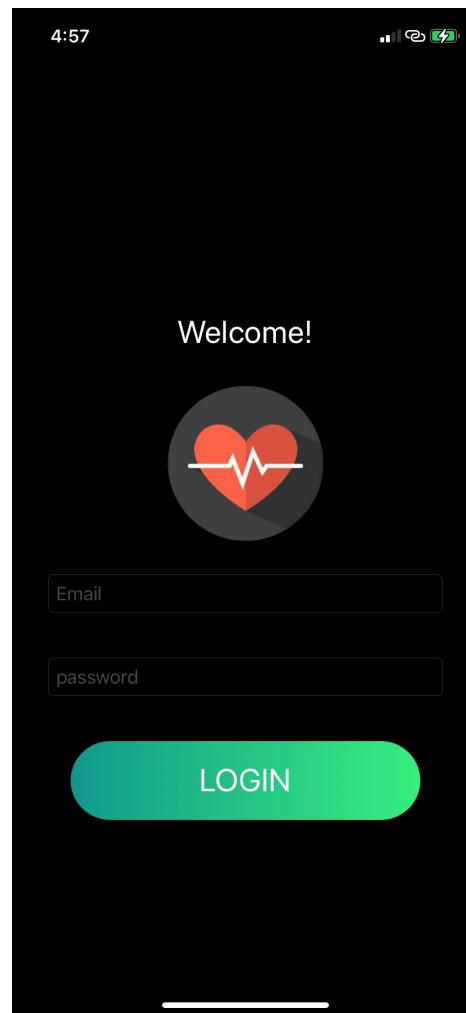
# 實際操作

- 主畫面的Icon



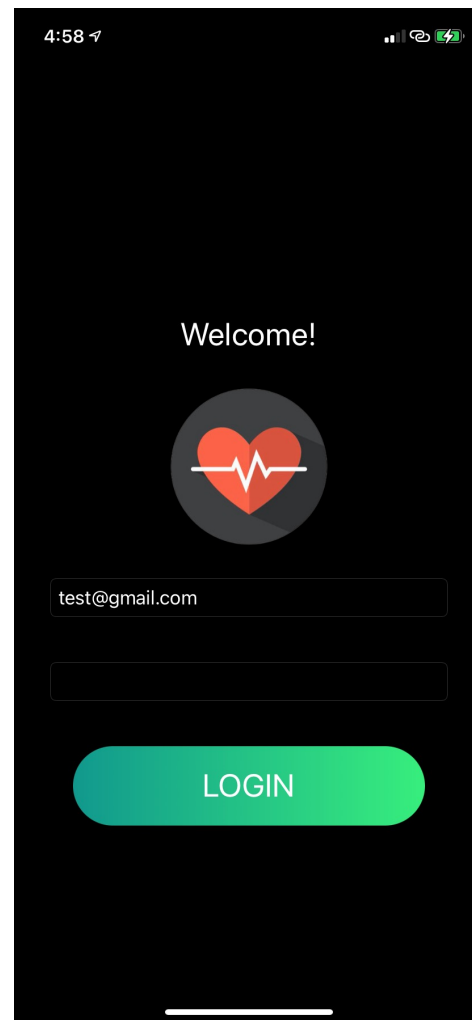
# 實際操作

- 進入主登入頁面



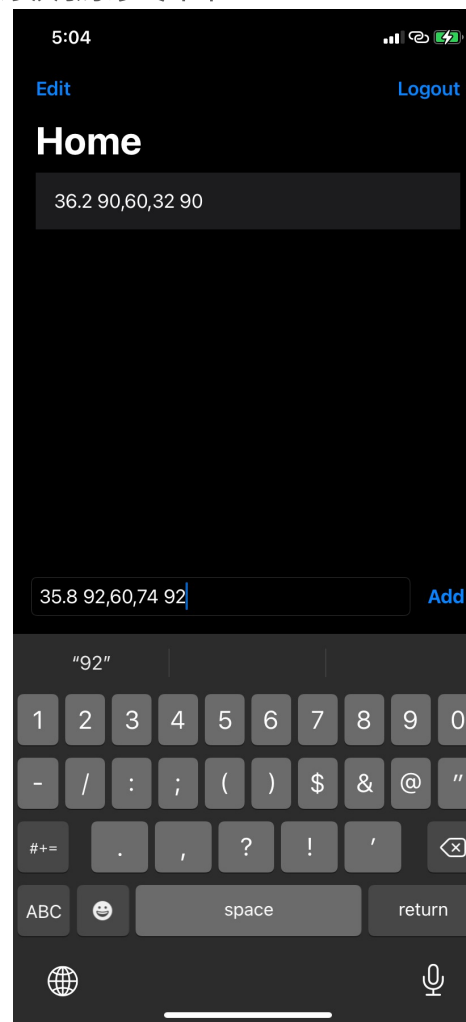
# 實際操作

- 進行帳號登入



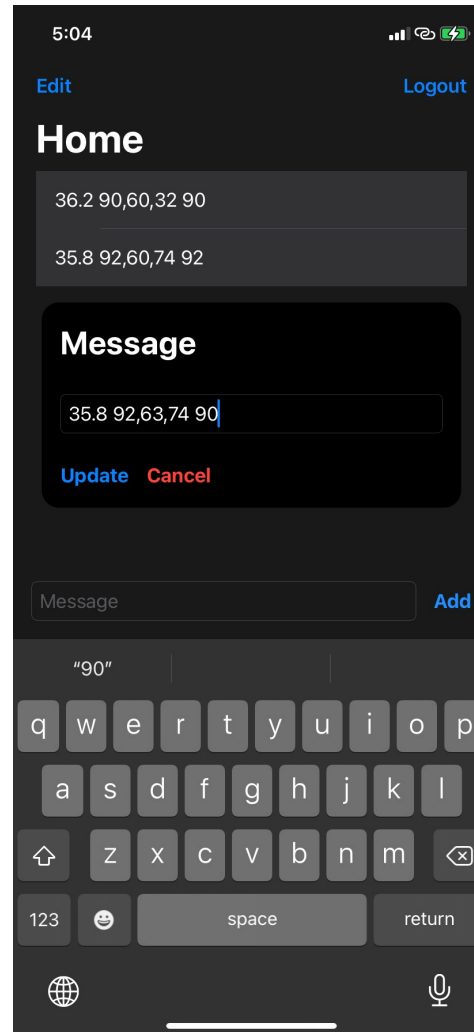
# 實際操作

- 開始輸入醫療數據資料



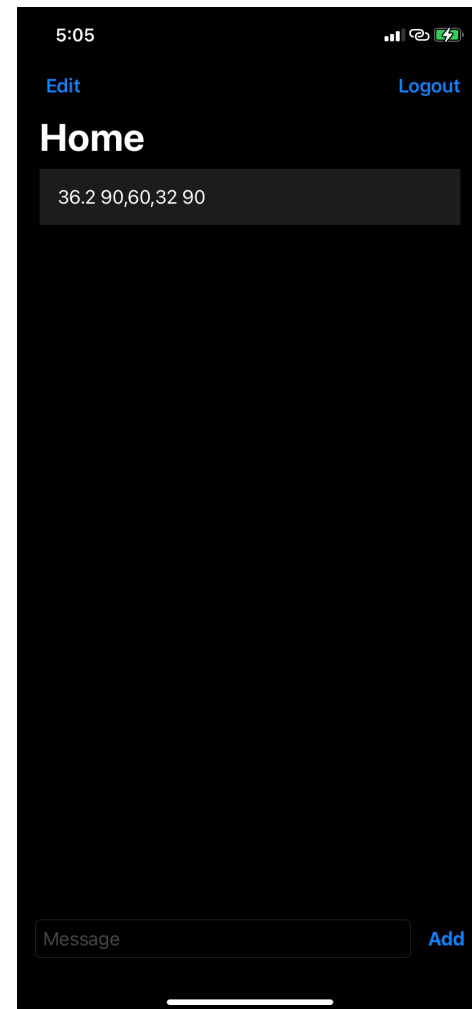
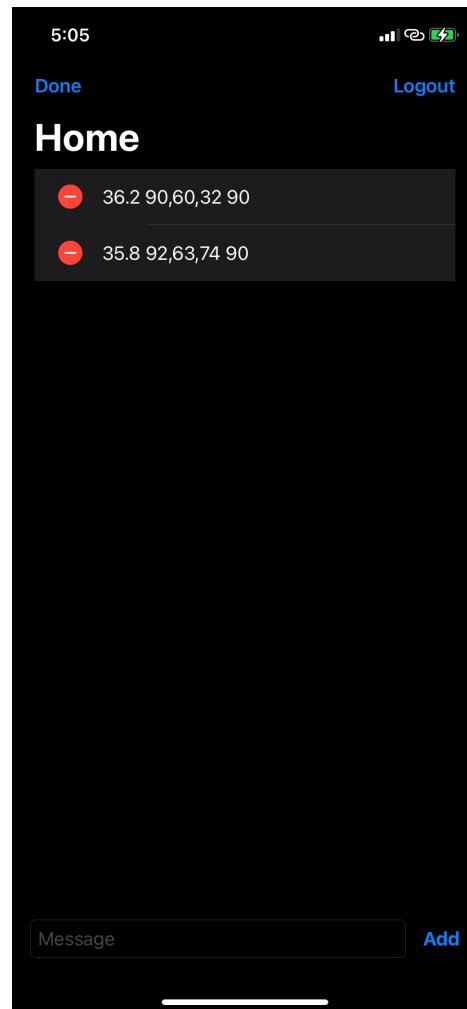
# 實際操作

- 更改醫療數據資料



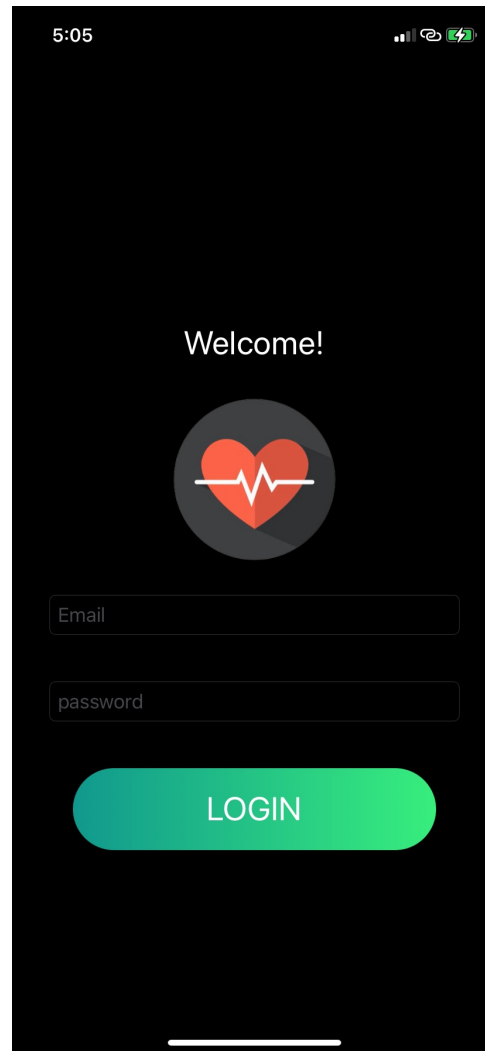
# 實際操作

- 刪除醫療數據資料



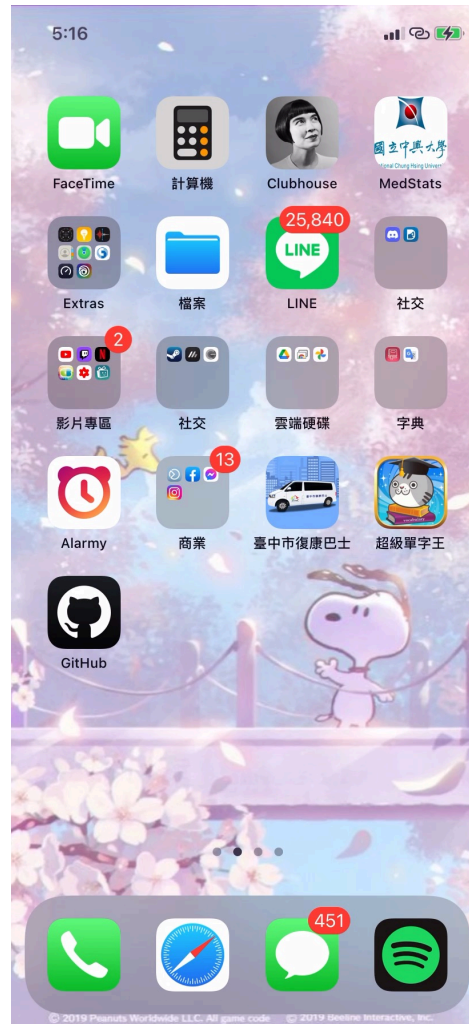
# 實際操作

- 登出

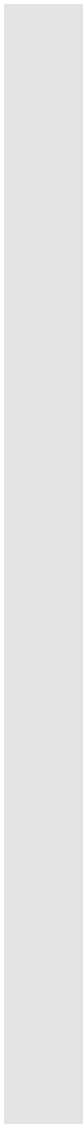



# 實際操作

- 實際操作影片







# 製作心得

## 製作心得

- 製作中發現Xcode很多問題，包含連結Framework都會有問題，串接API也會有很多找不到問題的錯誤，要自己去研究才能解決，可能也跟IOS更新速度快慢有關



## 製作心得

- 在寫的過程也發現自己的製作能力也有很多地方需要加強，可能還有很多地方可以寫得更漂亮一點





未來展望

# 未來展望

- 之後會製作使用者頁面，以及製作更多的欄位來分隔資料，最後再把Date欄位進行顯示，這部分就之後慢慢將它完成