屈 卿晨

软工196 2019081300

实验二

实验二

1. 实验目的

使用SQLite存储数据，进行增删改查

1. 实验内容
2. 流程图

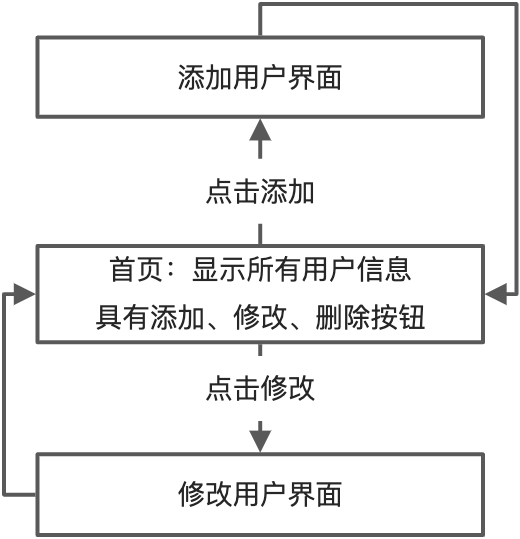


图 1 功能流程图

1. 代码实现（本次实验采用Swift UI）

新建工程

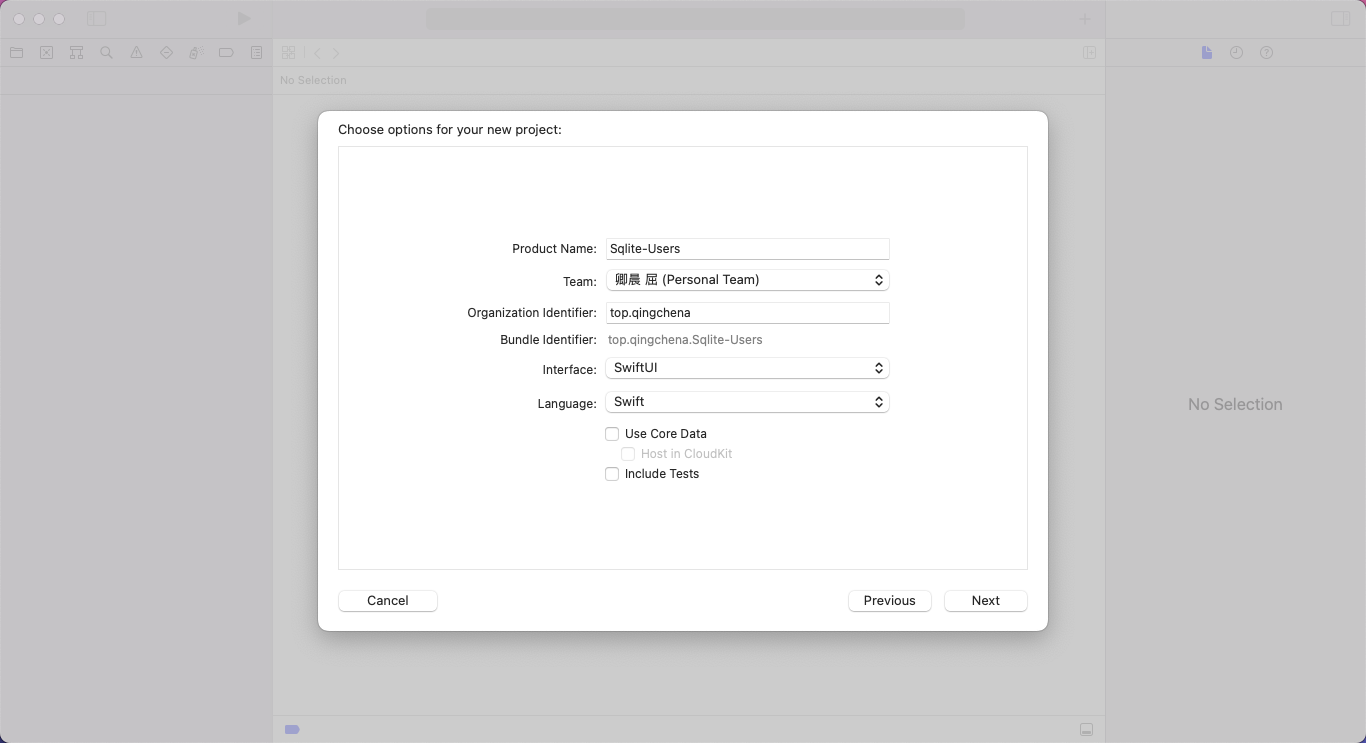


图 2 新建项目

新建SQLite工具类 DB\_Manager 并引入SQLite 库

建立db、users、id、name、email、age属性变量，更改init方法，在项目本地工程文件下生成sqlite数据库文件。并且创建User表。工具类有插入用户、获取所有用户、获取单个用户、删除用户、更新用户方法。如下所示：

|  |
| --- |
| //  // DB\_Manager.swift  // Sqlite-Users  //  // Created by Qingchen on 2022/4/18.  //  **import** Foundation  **import** SQLite  **class** DB\_Manager {    **private** **var** db: Connection!  **private** **var** users: Table!  **private** **var** id: Expression<Int64>!  **private** **var** name: Expression<String>!  **private** **var** email: Expression<String>!  **private** **var** age: Expression<Int64>!    **init**() {  **do** {  **let** path: String = NSSearchPathForDirectoriesInDomains(.documentDirectory, .userDomainMask, **true**).first ?? ""  db = **try** Connection("\(path)/my\_users.sqlite3")  print("\(path)")  users = Table("users")  id = Expression<Int64>("id")  name = Expression<String>("name")  email = Expression<String>("email")  age = Expression<Int64>("age")    **if**(!UserDefaults.standard.bool(forKey: "is\_db\_created")) {  **try** db.run(users.create {(t) **in**  t.column(id, primaryKey: **true**)  t.column(name)  t.column(email, unique: **true**)  t.column(age)  })  UserDefaults.standard.set(**true**, forKey: "is\_db\_created")  }  } **catch** {  print(error.localizedDescription)  }  }    **public** **func** addUser(nameValue:String, emailValue:String, ageValue:Int64) {  **do** {  **try**(db.run(users.insert(name <- nameValue, email <- emailValue, age <- ageValue)))  } **catch** {  print(error.localizedDescription)  }  }    **public** **func** getUsers() -> [UserModel] {  **var** userModels:[UserModel] = []  users = users.order(id.desc)  **do** {  **for** user **in** **try** db.prepare(users) {  **let** userModel: UserModel = UserModel()  userModel.id = user[id]  userModel.name = user[name]  userModel.email = user[email]  userModel.age = user[age]    userModels.append(userModel)  }  } **catch** {  print(error.localizedDescription)  }  **return** userModels  }    **public** **func** getUser(idValue: Int64) -> UserModel {  **var** userModel: UserModel = UserModel()  **do** {  **let** user: AnySequence<Row> = **try** db.prepare(users.filter(id == idValue))    **try** user.forEach({ (rowValue) **in**  userModel.id = **try** rowValue.**get**(id)  userModel.name = **try** rowValue.**get**(name)  userModel.email = **try** rowValue.**get**(email)  userModel.age = **try** rowValue.**get**(age)    })  } **catch** {  print(error.localizedDescription)  }  **return** userModel  }    **public** **func** updateUser(idValue: Int64, nameValue: String, emailValue: String, ageValue: Int64) {  **do** {  **let** user: Table = users.filter(id == idValue)  **try** db.run(user.update(name <- nameValue, email <- emailValue, age <- ageValue))  } **catch** {  print(error.localizedDescription)  }  }    **public** **func** deleteUser(idValue: Int64) {  **do** {  **let** user: Table = users.filter(id == idValue)  **try** db.run(user.delete())  } **catch** {  print(error.localizedDescription)  }  }  } |

创建用户模型类 UserModel，有id、name、email、age属性，并且其类型是Identifiable

|  |
| --- |
| //  // UserModel.swift  // Sqlite-Users  //  // Created by Qingchen on 2022/4/18.  //  **import** Foundation  **class** UserModel: Identifiable {  **public** **var** id: Int64 = 0  **public** **var** name: String = ""  **public** **var** email: String = ""  **public** **var** age: Int64 = 0  } |

编写ContentView界面

需要有一个存放所有用户的数组，类型是UserModel，还有一个是否选择了用户的一个变量，一个记录选择用户的id。具有添加、修改、删除、编辑功能。具体代码如下

|  |
| --- |
| //  // ContentView.swift  // Sqlite-Users  //  // Created by Qingchen on 2022/4/18.  //  **import** SwiftUI  **struct** ContentView: View {    @State **var** userModels: [UserModel] = []  @State **var** userSelected: Bool = **false**  @State **var** selectedUserId: Int64 = 0    **var** body: **some** View {  NavigationView {  VStack {  HStack {  Spacer()  NavigationLink(destination: AddUserView(), label: {Text("Add user")})  }    NavigationLink (destination: EditUserView(id: **self**.$selectedUserId), isActive: **self**.$userSelected) {  EmptyView()  }    List(**self**.userModels) { (model) **in**  HStack {  Text(model.name)  Spacer()  Text(model.email)  Spacer()  Text("\(model.age)")    Button(action: {  **self**.selectedUserId = model.id  **self**.userSelected = **true**  }, label: {  Text("Edit")  .foregroundColor(Color.blue)  }).buttonStyle(PlainButtonStyle())    Button(action: {  **let** dbManager: DB\_Manager = DB\_Manager()    dbManager.deleteUser(idValue: model.id)  **self**.userModels = dbManager.getUsers()  }, label: {  Text("Delete")  .foregroundColor(Color.red)  }).buttonStyle(PlainButtonStyle())  }  }  }  .padding()  .onAppear(perform: {  **self**.userModels = DB\_Manager().getUsers()  })  .navigationBarTitle("Users")  Spacer()  }  }  }  **struct** ContentView\_Previews: PreviewProvider {  **static** **var** previews: **some** View {  ContentView()  }  } |

添加用户界面：email、name、age变量，并且添加成功返回首页。

|  |
| --- |
| //  // AddUserView.swift  // Sqlite-Users  //  // Created by Qingchen on 2022/4/18.  //  **import** SwiftUI  **struct** AddUserView: View {    @State **var** name: String = ""  @State **var** email: String = ""  @State **var** age: String = ""    @Environment(\.presentationMode) **var** mode: Binding<PresentationMode>    **var** body: **some** View {  VStack {  TextField("Enter name", text: $name)  .padding(10)  .background(Color(.systemGray6))  .cornerRadius(5)  .disableAutocorrection(**true**)    TextField("Enter email", text: $email)  .padding(10)  .background(Color(.systemGray6))  .keyboardType(.emailAddress)  .autocapitalization(.none)  .cornerRadius(5)  .disableAutocorrection(**true**)    TextField("Enter age", text: $age)  .padding(10)  .background(Color(.systemGray6))  .cornerRadius(5)  .keyboardType(.numberPad)  .disableAutocorrection(**true**)    Button(action: {  DB\_Manager().addUser(nameValue: **self**.name, emailValue: **self**.email, ageValue: Int64(**self**.age) ?? 0)    **self**.mode.wrappedValue.dismiss()  }, label: {  Text("Add User")  })  .frame(maxWidth: .infinity, alignment: .trailing)  .padding(.top, 10)  .padding(.bottom, 10)  }.padding()  Spacer()  }  }  **struct** AddUserView\_Previews: PreviewProvider {  **static** **var** previews: **some** View {  AddUserView()  }  } |

编辑用户界面：id、email、name、age变量。更新成功返回首页。

|  |
| --- |
| //  // EditUserView.swift  // Sqlite-Users  //  // Created by Qingchen on 2022/4/18.  //  **import** SwiftUI  **struct** EditUserView: View {    @Binding **var** id: Int64    @State **var** name: String = ""  @State **var** email: String = ""  @State **var** age: String = ""    @Environment(\.presentationMode) **var** mode: Binding<PresentationMode>    **var** body: **some** View {  VStack {  TextField("Enter name", text: $name)  .padding(10)  .background(Color(.systemGray6))  .cornerRadius(5)  .disableAutocorrection(**true**)  TextField("Enter email", text: $email)  .padding(10)  .background(Color(.systemGray6))  .cornerRadius(5)  .keyboardType(.emailAddress)  .autocapitalization(.none)  .disableAutocorrection(**true**)  TextField("Enter age", text: $age)  .padding(10)  .cornerRadius(5)  .background(Color(.systemGray6))  .keyboardType(.numberPad)  .disableAutocorrection(**true**)    Button(action:{  DB\_Manager().updateUser(idValue: **self**.id, nameValue: **self**.name, emailValue: **self**.email, ageValue: Int64(**self**.age) ?? 0)    **self**.mode.wrappedValue.dismiss()  }, label: {  Text("Edit User")  }).frame(maxWidth: .infinity, alignment: .trailing)  .padding(.top, 10)  .foregroundColor(Color(.systemTeal))  }.padding()  .onAppear(perform: {  **let** userModel: UserModel = DB\_Manager().getUser(idValue: **self**.id)    **self**.name = userModel.name  **self**.email = userModel.email  **self**.age = String(userModel.age)  })  }  }  **struct** EditUserView\_Previews: PreviewProvider {  @State **static** **var** id: Int64 = 0  **static** **var** previews: **some** View {  EditUserView(id: $id)  }  } |

实现效果图

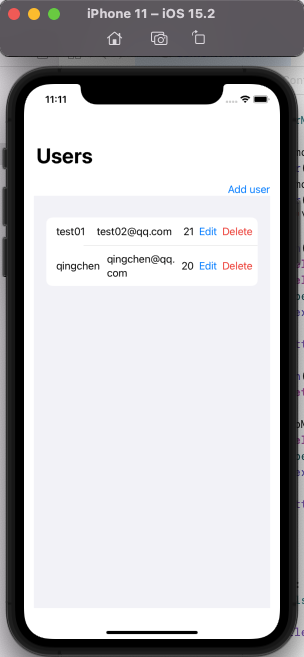
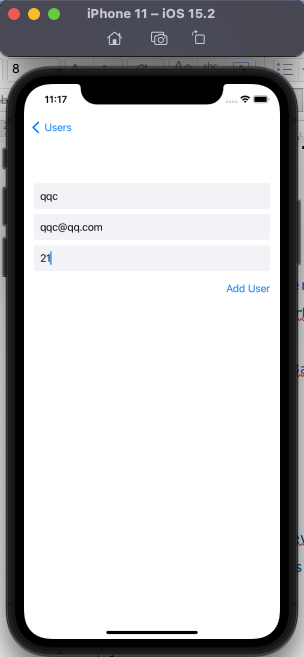
 

图 3 首页 图 4 添加用户

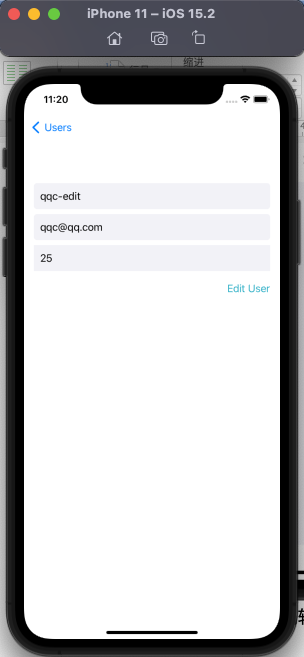
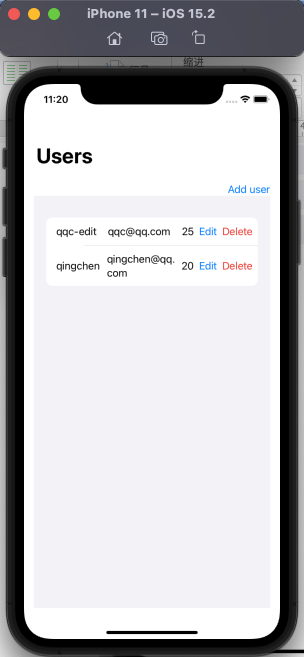
 

图 5 编辑界面 图 6 删除用户

1. 问题

不熟悉SQLite的使用方式、对于DB\_Manager工具类的编写很吃力，架构相对来说比较复杂。需要对创建文件，写入文件。增删改查等方法等编写。

通过自学，网上查阅资料。不断的更改代码。使用Storyboard也做过，但是不知道如何在渲染数据的同时添加编辑和删除按钮。