**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG DỮ LIỆU**

**BÙI THANH HẢI**

**KHÓA LUẬN TỐT NGHIỆP**

**PHÁT TRIỂN GIẢI PHÁP GIÁM SÁT**

**NHÀ THÔNG MINH THỜI GIAN THỰC DỰA TRÊN**

**NỀN TẢNG ĐIỆN TOÁN ĐÁM MÂY**

**DEVELOPMENT OF A SURVEILLANCE SYSTEM FOR SMART HOUSES INTEGRATED OVER CLOUD PLATFORM**

**KỸ SƯ/ CỬ NHÂN NGÀNH TRUYỀN THÔNG VÀ MẠNG MÁY TÍNH**

**TP. HỒ CHÍ MINH, 2018**

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG DỮ LIỆU**

**BÙI THANH HẢI – 14520236**

**KHÓA LUẬN TỐT NGHIỆP**

**PHÁT TRIỂN GIẢI PHÁP GIÁM SÁT**

**NHÀ THÔNG MINH THỜI GIAN THỰC DỰA TRÊN**

**NỀN TẢNG ĐIỆN TOÁN ĐÁM MÂY**

**DEVELOPMENT OF A SURVEILLANCE SYSTEM FOR SMART HOUSES INTEGRATED OVER CLOUD PLATFORM**

**KỸ SƯ/ CỬ NHÂN NGÀNH TRUYỀN THÔNG VÀ MẠNG MÁY TÍNH**

**GIẢNG VIÊN HƯỚNG DẪN**

**PGS.TS LÊ TRUNG QUÂN**

**TP. HỒ CHÍ MINH, 2018**

DANH SÁCH HỘI ĐỒNG BẢO VỆ KHÓA LUẬN

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo Quyết định số …………………… ngày ………………….. của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

* 1. …………………………………………. – Chủ tịch.
  2. …………………………………………. – Thư ký.
  3. …………………………………………. – Ủy viên.
  4. …………………………………………. – Ủy viên.

NHẬN XÉT CỦA CÁN BỘ HƯỚNG DẪN

NHẬN XÉT CỦA CÁN BỘ PHẢN BIỆN

LỜI CẢM ƠN

MỤC LỤC

[Chương 1. TỔNG QUAN 14](#_Toc526000011)

[1.1. Lí do chọn đề tài 14](#_Toc526000012)

[1.2. Mục tiêu 14](#_Toc526000013)

[1.3. Đối tượng và phạm vi nghiên cứu 14](#_Toc526000014)

[Chương 2. XÂY DỰNG HỆ THỐNG GỬI NHẬN DỮ LIỆU THÔNG QUA API 15](#_Toc526000015)

[2.1. Các bước thực hiện 15](#_Toc526000016)

[2.1.1. Bước 1: Tạo một project mới bằng Visual Studio 15](#_Toc526000017)

[2.1.2. Bước 2: Cấu trúc thư mục 16](#_Toc526000018)

[2.1.3. Bước 3: Install các package cần thiết 17](#_Toc526000019)

[2.1.4. Bước 4: Tạo Model để định dạng dữ liệu và các phương thức truyền nhận 18](#_Toc526000020)

[2.1.4.1. PinModel 18](#_Toc526000021)

[2.1.4.2. UserModel 22](#_Toc526000022)

[2.1.4.3. DHTModel 25](#_Toc526000023)

[2.1.4.4. KitKey 28](#_Toc526000024)

[2.1.5. Bước 5: Tạo DBContext để tương tác với CSDL 29](#_Toc526000025)

[2.1.6. Bước 6: Chỉnh sửa file “Startup.cs” 29](#_Toc526000026)

[2.1.7. Bước 7: Tạo API 30](#_Toc526000027)

[2.1.7.1. PinController 30](#_Toc526000028)

[2.1.7.2. UserController 33](#_Toc526000029)

[2.1.7.3. DHTController 36](#_Toc526000030)

[2.1.8. Bước 8: Tạo dữ liệu mặc định 37](#_Toc526000031)

[2.1.9. Bước 9: Migration 39](#_Toc526000032)

[2.1.10. Bước 10: Chạy chương trình 40](#_Toc526000033)

[Chương 3. XÂY DỰNG HỆ THỐNG THU THẬP DỮ LIỆU 42](#_Toc526000034)

[Chương 4. XÂY DỰNG HỆ THỐNG GIÁM SÁT VÀ ĐIỀU KHIỂN 42](#_Toc526000035)

[Chương 5. TRIỂN KHAI HỆ THỐNG LÊN OPENSTACK 42](#_Toc526000036)

[Chương 6. KẾT LUẬN 42](#_Toc526000037)

[Chương 7. HƯỚNG PHÁT TRIỂN 42](#_Toc526000038)

DANH MỤC HÌNH VẼ

[Hình 2.1: Tạo một project (1) 16](#_Toc526000165)

[Hình 2.2: Tạo một project (2) 17](#_Toc526000166)

[Hình 2.3: Cấu trúc thư mục của Project 18](#_Toc526000167)

[Hình 2.4: Cài đặt các packages cần thiết 19](#_Toc526000168)

[Hình 2.5: Cấu trúc thư mục của PinModel 19](#_Toc526000169)

[Hình 2.6: Định nghĩa thuộc tính của PinModel 20](#_Toc526000170)

[Hình 2.7: Định nghĩa phương thức của PinModel 20](#_Toc526000171)

[Hình 2.8: Triển khai phương thức của PinModel (1) 21](#_Toc526000172)

[Hình 2.9: Triển khai phương thức của PinModel (2) 22](#_Toc526000173)

[Hình 2.10: Triển khai phương thức của PinModel (3) 23](#_Toc526000174)

[Hình 2.11: Cấu trúc thư mục của UserModel 24](#_Toc526000175)

[Hình 2.12: Định nghĩa thuộc tính của UserModel 24](#_Toc526000176)

[Hình 2.13: Định nghĩa phương thức của UserModel 24](#_Toc526000177)

[Hình 2.14: Triển khai phương thức của UserModel (1) 25](#_Toc526000178)

[Hình 2.15: Triển khai phương thức của UserModel (2) 26](#_Toc526000179)

[Hình 2.16: Cấu trúc thư mục của DHTModel 27](#_Toc526000180)

[Hình 2.17: Định nghĩa thuộc tính của DHTModel 27](#_Toc526000181)

[Hình 2.18: Định nghĩa phương thức của DHTModel 27](#_Toc526000182)

[Hình 2.19: Triển khai phương thức của DHTModel (1) 28](#_Toc526000183)

[Hình 2.20: Triển khai phương thức của DHTModel (2) 29](#_Toc526000184)

[Hình 2.21: Nội dung file KitKey.cs 29](#_Toc526000185)

[Hình 2.22: Nội dung file ThesisAPIContext.cs 30](#_Toc526000186)

[Hình 2.23: Thiết lập kết nối với SQLite 30](#_Toc526000187)

[Hình 2.24: Tạo các Controller 31](#_Toc526000188)

[Hình 2.25: Nội dung file PinController (1) 31](#_Toc526000189)

[Hình 2.26: Nội dung file PinController (2) 32](#_Toc526000190)

[Hình 2.27: Nội dung file PinController (3) 32](#_Toc526000191)

[Hình 2.28: Nội dung file PinController (4) 33](#_Toc526000192)

[Hình 2.29: Nội dung file UserController (1) 34](#_Toc526000193)

[Hình 2.30: Nội dung file UserController (2) 35](#_Toc526000194)

[Hình 2.31: Nội dung file UserController (3) 36](#_Toc526000195)

[Hình 2.32: Nội dung file DHTController (1) 37](#_Toc526000196)

[Hình 2.33: Nội dung file DHTController (2) 37](#_Toc526000197)

[Hình 2.34: Nội dung file DHTController (3) 38](#_Toc526000198)

[Hình 2.35: Tạo SeedData 38](#_Toc526000199)

[Hình 2.36: Nội dung file SeedData.cs (1) 39](#_Toc526000200)

[Hình 2.37: Nội dung file SeedData.cs (2) 39](#_Toc526000201)

[Hình 2.38: Khai báo SeedData 40](#_Toc526000202)

[Hình 2.39: Chạy Migration 40](#_Toc526000203)

[Hình 2.40: Kết quả chạy chương trình 41](#_Toc526000204)

DANH MỤC BẢNG

[Bảng 1.1: Tên bảng 1 3](#_Toc367742567)

[Bảng 2.1: Tên bảng 1 4](#_Toc367742568)

DANH MỤC TỪ VIẾT TẮT

IoT: Internet of Things

API: Application Programming Interface

CSDL: Cơ sở dữ liệu

TÓM TẮT KHÓA LUẬN

MỞ ĐẦU

Internet of Thing là một hệ thống các thiết bị đồ dùng được kết nối với nhau thông qua mạng Internet.

# TỔNG QUAN

## Lí do chọn đề tài

## Mục tiêu

## Đối tượng và phạm vi nghiên cứu

Nội dung …………………

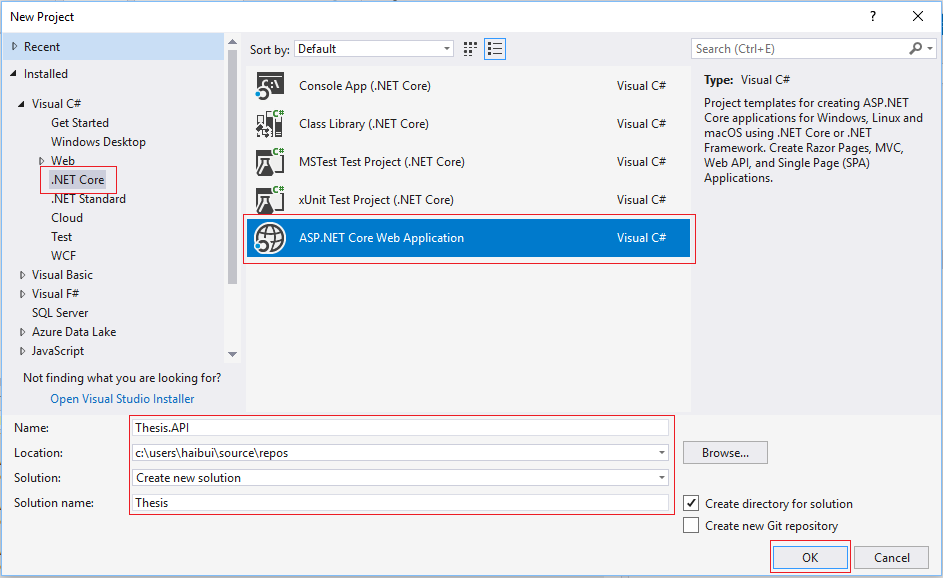
# XÂY DỰNG HỆ THỐNG GỬI NHẬN DỮ LIỆU THÔNG QUA API

## Các bước thực hiện

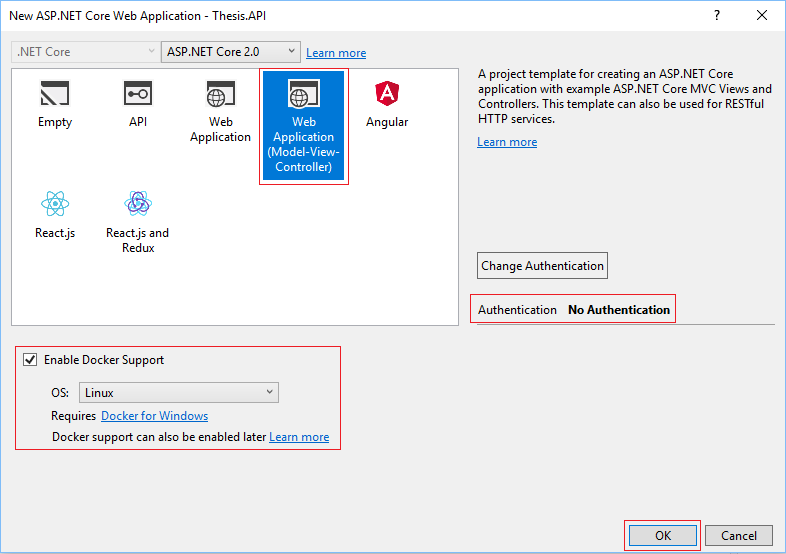
### Bước 1: Tạo một project mới bằng Visual Studio

Mở Visual Studio phiên bản nên lớn hơn 2017. Bấm tổ hợp phím “ctrl + shift + n” để tạo một project mới như hình bên dưới.

Sử dụng ASP.NET core 2.0[[1]](#footnote-1) để xây dựng server.



Hình 2.1: Tạo một project (1)

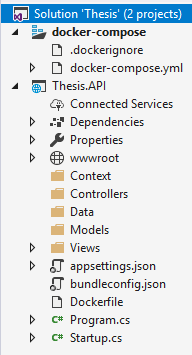


Hình 2.2: Tạo một project (2)

Chọn “Enable Docker Support” để hỗ trợ cho việc triển khai hệ thống lên Openstack dễ dàng hơn. Nếu chưa có Docker[[2]](#footnote-2) có thể lên trang chủ của Docker và cài đặt về máy.

### Bước 2: Cấu trúc thư mục

Tạo cấu trúc thư mục như hình bên dưới.



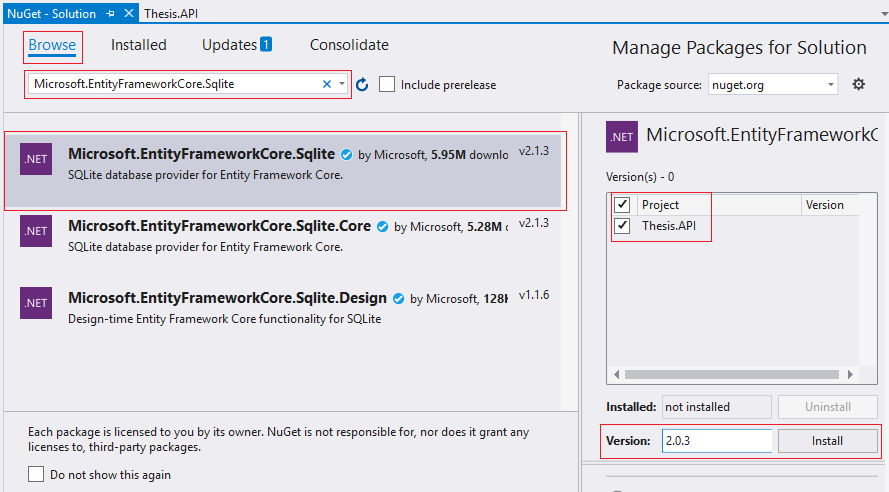
Hình 2.3: Cấu trúc thư mục của Project

### Bước 3: Install các package cần thiết

Vào Tool -> NuGet Package Manager -> Manage Nuget Packages for Solution…

Cần cài một số packages sau:

* Microsoft.EntityFrameworkCore.Tools
* Microsoft.EntityFrameworkCore.Sqlite
* Microsoft.EntityFrameworkCore.SqlServer



Hình 2.4: Cài đặt các packages cần thiết

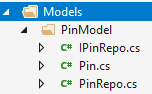
Với ASP.NET core 2.0 nên chọn package version 2.0.x và cài tương tự với các packages còn lại.

### Bước 4: Tạo Model để định dạng dữ liệu và các phương thức truyền nhận

#### PinModel

Mục đích của PinModel nhằm lưu trữ trạng thái của các cổng GPIO của Arduino. Chỉ có hai trạng thái duy nhất đó là 0 và 1.

Tạo cấu trúc thư mục cho PinModel như hình bên dưới.

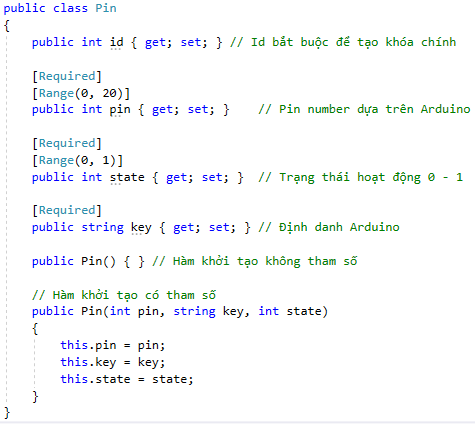


Hình 2.5: Cấu trúc thư mục của PinModel

Trong đó:

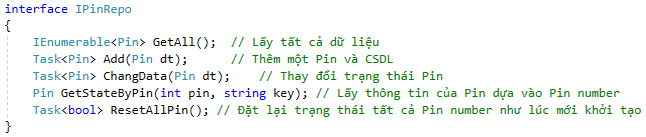
* IPinRepo.cs: Dùng để định nghĩa các phương thức sẽ được sử dụng.
* Pin.cs: Dùng để định nghĩa các thuộc tính cần thiết.
* PinRepo: Dùng để triển khai các phương thức đã được định nghĩa.

Định nghĩa các thuộc tính cần thiết như hình dưới.



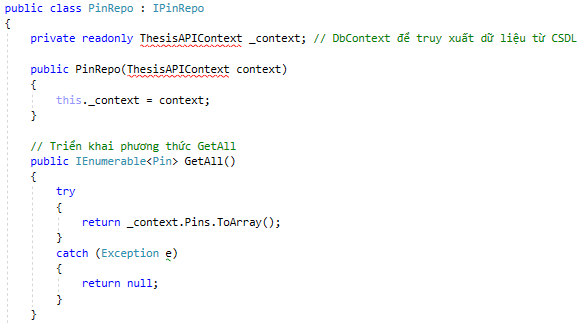
Hình 2.6: Định nghĩa thuộc tính của PinModel

Định nghĩa các phương thức dùng để truyền nhận dữ liệu.



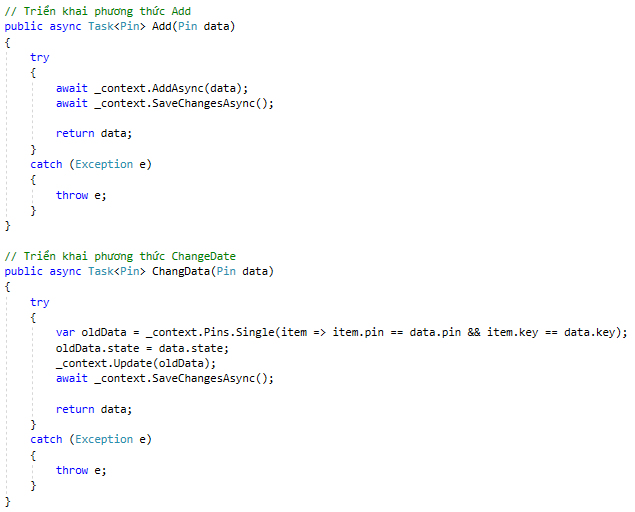
Hình 2.7: Định nghĩa phương thức của PinModel

Sau khi đã khai báo các thuộc tính và phương thức sẽ tiến hành triển khai các phương thức đã định nghĩa ở trên.

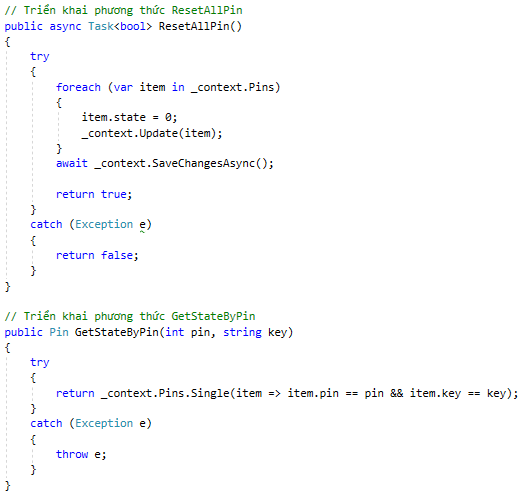


Hình 2.8: Triển khai phương thức của PinModel (1)

Ở bên trên báo lỗi không tìm thấy “ThesisAPIContext”, phần này sẽ khởi tạo bên dưới sau khi đã định nghĩa tất cả Model.



Hình 2.9: Triển khai phương thức của PinModel (2)



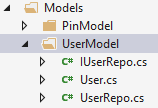
Hình 2.10: Triển khai phương thức của PinModel (3)

#### UserModel

Mục đích của UserModel nhằm lưu thông tin của người dùng bao gồm các thông tin cơ bản như tên người dùng và mã số thẻ RFID tương ứng.

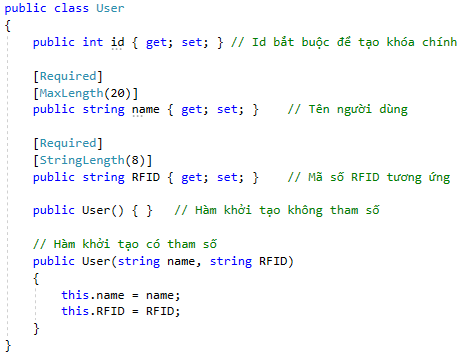
Các bước thực hiện tương tự PinModel nên sẽ không giải thích lại.

Tạo cấu trúc thư mục.



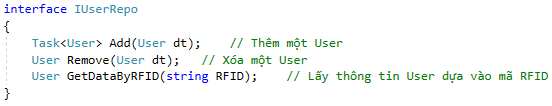
Hình 2.11: Cấu trúc thư mục của UserModel

Định nghĩa các thuộc tính.



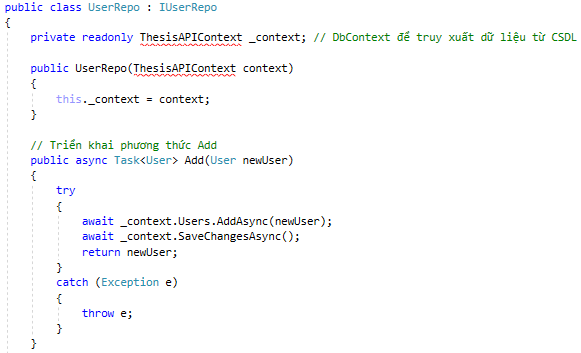
Hình 2.12: Định nghĩa thuộc tính của UserModel

Định nghĩa các phương thức.

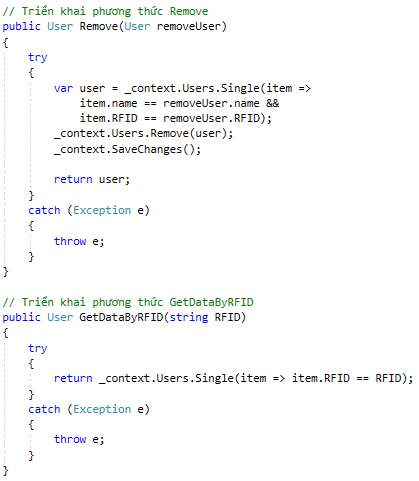


Hình 2.13: Định nghĩa phương thức của UserModel

Triển khai các phương thức.



Hình 2.14: Triển khai phương thức của UserModel (1)



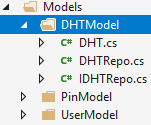
Hình 2.15: Triển khai phương thức của UserModel (2)

#### DHTModel

Mục đích của DHTModel nhằm lưu trữ thông tin về nhiệt độ và độ ẩm thu thập được từ cảm biến DHT11[[3]](#footnote-3).

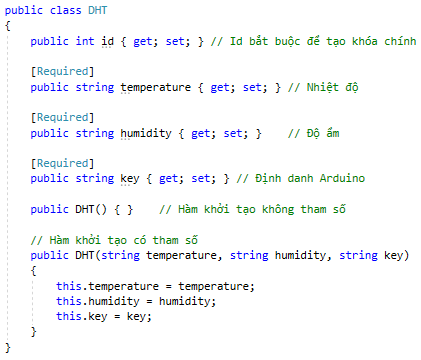
Các bước thực hiện tương tự PinModel nên sẽ không giải thích lại.

Tạo cấu trúc thư mục.



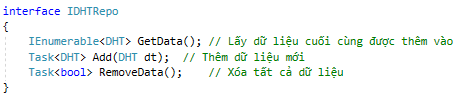
Hình 2.16: Cấu trúc thư mục của DHTModel

Định nghĩa các thuộc tính.



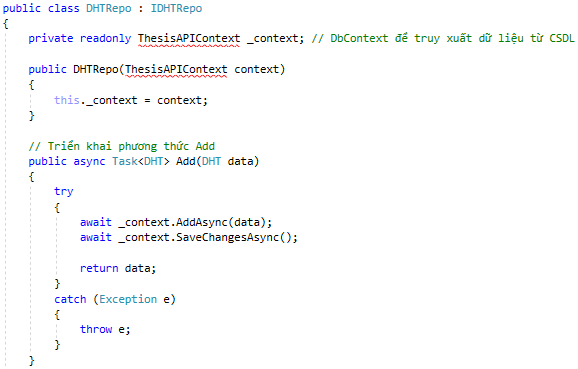
Hình 2.17: Định nghĩa thuộc tính của DHTModel

Định nghĩa các phương thức.

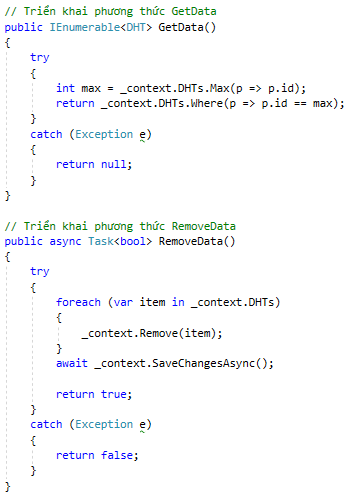


Hình 2.18: Định nghĩa phương thức của DHTModel

Triển khai các phương thức.



Hình 2.19: Triển khai phương thức của DHTModel (1)

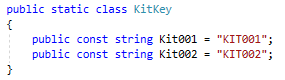


Hình 2.20: Triển khai phương thức của DHTModel (2)

#### KitKey

Định nghĩa “key” sẽ được dùng để định danh cho từng thiết bị Arduino.

Tạo một file “KitKey.cs” bên trong thư mục Models.

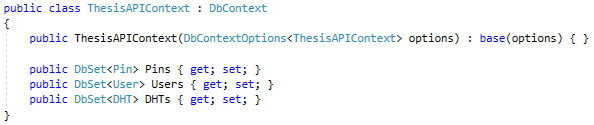


Hình 2.21: Nội dung file “KitKey.cs”

Ở đây khai báo theo số lượng thiết bị hiện tại. Nếu nhiều hơn hai thiết bị thì có thể khai báo thêm.

### Bước 5: Tạo DBContext để tương tác với CSDL

Trong thư mục context, tạo một file “ThesisAPIContext.cs” có nội dung như sau.

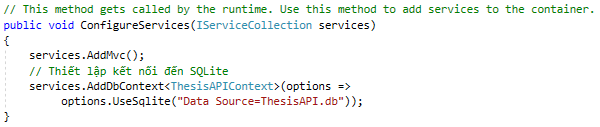


Hình 2.22: Nội dung file “ThesisAPIContext.cs”

Sau khi tạo file “ThesisAPIContext.cs”, tiến hành cập nhật cho các file “PinRepo.cs”, “UserRepo.cs”, “DHTRepo.cs”.

### Bước 6: Chỉnh sửa file “Startup.cs”

Trong file “Startup.cs”, thiết lập kết nối với SQLite[[4]](#footnote-4).



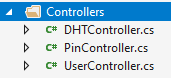
Hình 2.23: Thiết lập kết nối với SQLite

### Bước 7: Tạo API

Tiến hành tạo các Controller để có thể gửi nhận dữ liệu thông qua các API mà Controller cung cấp.

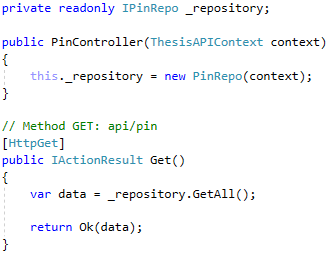
Để tạo Controller, chuột phải vào thư mục Controllers -> Add -> Controller… -> API Controller – Empty -> Add, đặt tên cho Controller và chọn Add.

Tạo Controller tương ứng với các Model như hình.

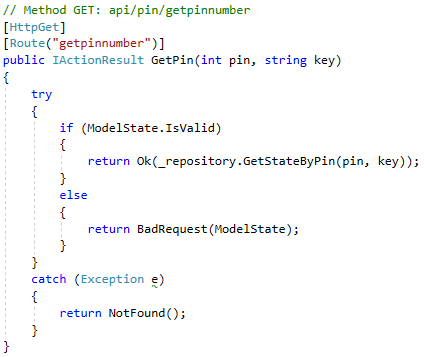


Hình 2.24: Tạo các Controller

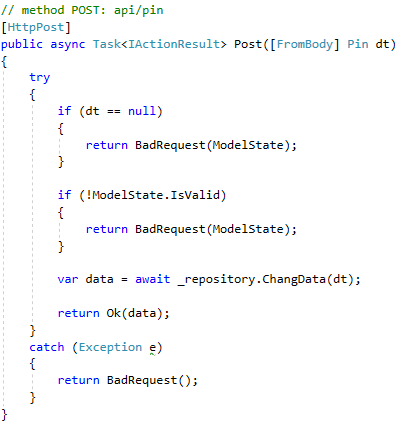
#### PinController



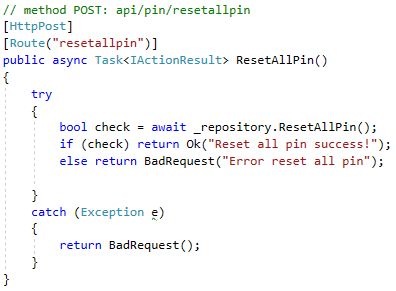
Hình 2.25: Nội dung file “PinController.cs” (1)



Hình 2.26: Nội dung file “PinController.cs” (2)

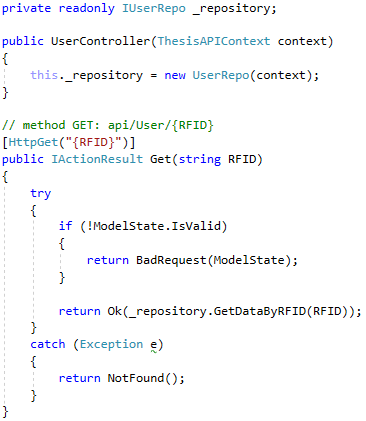


Hình 2.27: Nội dung file “PinController.cs” (3)

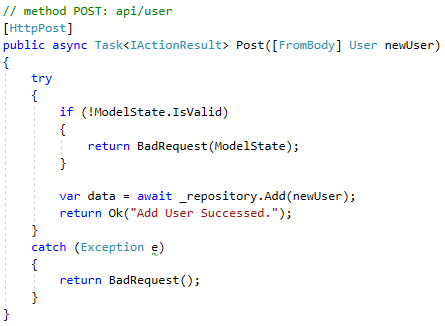


Hình 2.28: Nội dung file “PinController.cs” (4)

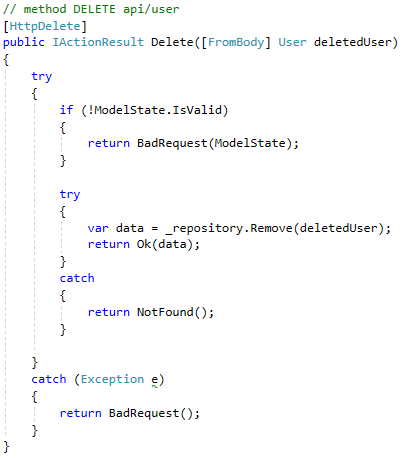
#### UserController



Hình 2.29: Nội dung file “UserController.cs” (1)

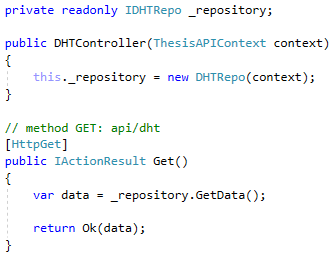


Hình 2.30: Nội dung file “UserController.cs” (2)

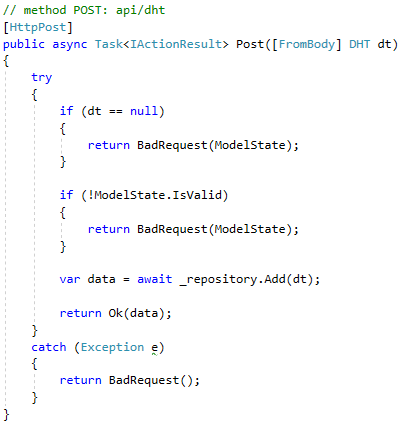


Hình 2.31: Nội dung file “UserController.cs” (3)

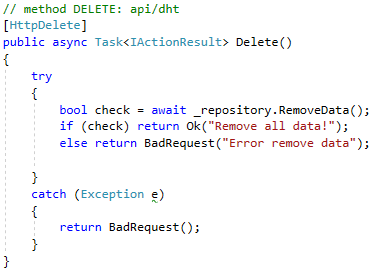
#### DHTController



Hình 2.32: Nội dung file “DHTController.cs” (1)



Hình 2.33: Nội dung file “DHTController.cs” (2)



Hình 2.34: Nội dung file “DHTController.cs” (3)

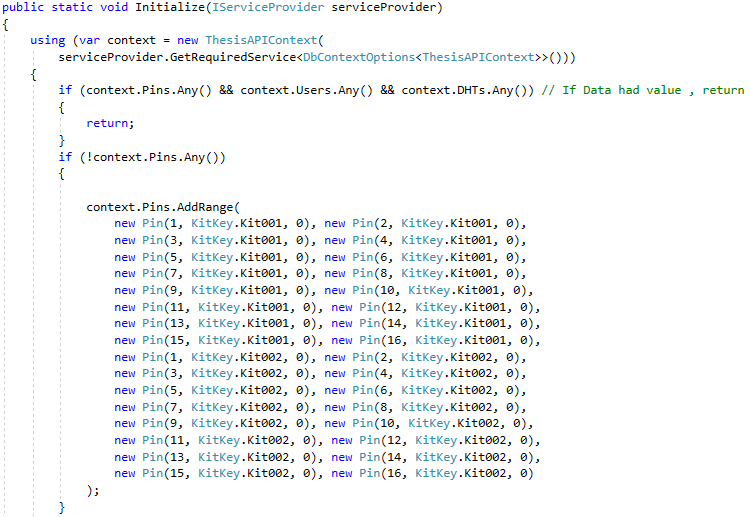
### Bước 8: Tạo dữ liệu mặc định

Trong thư mục Data, tạo một class mới tên SeedData.cs

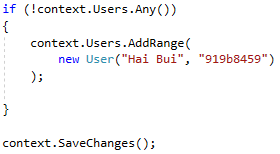
Mục đích là tạo ra những dữ liệu mặc định trong CSDL khi chạy chương trình lần đầu tiên, hoặc sau khi xóa CSDL và tạo mới CSDL.



Hình 2.35: Tạo SeedData

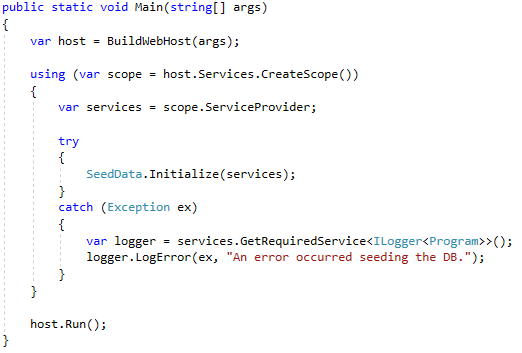


Hình 2.36: Nội dung file “SeedData.cs” (1)



Hình 2.37: Nội dung file “SeedData.cs” (2)

Khai báo dữ liệu vừa tạo vào file Program.cs -> hàm Main()



Hình 2.38: Khai báo SeedData

### Bước 9: Migration

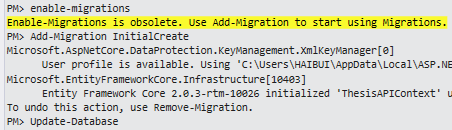
Chuột phải vào Solution -> Properties -> Common Properties -> Startup Project -> Single startup project. Chuyển “docker-compose” thành “Thesis.API”.

Chọn Tool -> Nuget Package Manager -> Package Manager Console

Chạy từng lệnh sau:

* enable-migrations
* Add-Migration InitialCreate
* Update-Database

Sau khi chạy xong ta có kết quả như sau.

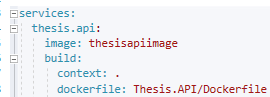


Hình 2.39: Chạy Migration

Ngoài ra trong project sẽ thấy một thư mục Migrations được tự động sinh ra. File này sẽ tạo bảng tự động trong CSDL khi chạy chương trình.

### Bước 10: Chạy chương trình

Vào file “docker-compose.yml” thay đổi dòng image như sau.



Hình 2.40: Cập nhật docker-compose.yml

Có thể chạy chương trình bằng nhiều cách như : docker-compose, IISServer, console.

Sau khi chạy chương trình, kiểm tra đường dẫn [http://localhost:<port>/api/pin](http://localhost:%3cport%3e/api/pin) được kết quả như sau.



Hình 2.41: Kết quả chạy chương trình

Ngoài ra, kiểm tra trong source code của project sẽ thấy một file “ThesisAPI.db”. Đây là file SQLite, tự động sinh ra khi chạy chương trình.

Kiểm tra tương tự với các API còn lại.

# XÂY DỰNG HỆ THỐNG THU THẬP DỮ LIỆU

# XÂY DỰNG HỆ THỐNG GIÁM SÁT VÀ ĐIỀU KHIỂN

# TRIỂN KHAI HỆ THỐNG LÊN OPENSTACK

# KẾT LUẬN

# HƯỚNG PHÁT TRIỂN

TÀI LIỆU THAM KHẢO

Theo chuẩn IEEE

1. ASP.NET core 2.0: <https://docs.microsoft.com/en-us/aspnet/core/getting-started/?view=aspnetcore-2.0&tabs=windows> [↑](#footnote-ref-1)
2. Docker: <https://www.docker.com/> [↑](#footnote-ref-2)
3. DHT11: <https://vngiotlab.github.io/vbluno/vi/mydoc_arduino_tut10_vi.html> [↑](#footnote-ref-3)
4. SQLite: <https://www.sqlite.org/index.html> [↑](#footnote-ref-4)