Contrôle d’une porte d’aération d’une serre à distance

par

Christian Castaneda

Présenté à :

Dhafer Ben-Khedher

Dans le cadre du cours :

420-6GP-BB Objets Connectés

Cégep du Bois-de-Boulogne

15/05/2022

Description du project:

* Un contrôleur d’une porte d’aération coulissante qui se fermer ou ouvre selon la température ambiante. Il existe deux modes dans le contrôleur : Automatique et manuel. Dans cette version, une interface utilisateur dans le web a été implémenté. Il est possible de contrôler le périphérique intelligent qui gère l’ouverture de la porte à distance. En plus, une application mobile pourrait permettre de contrôler cette porte à distance.
* Dans le mode automatique, le contrôleur ouvrira la porte coulissante en fonction de la température ambiante.
* Dans le mode manuelle, l’utilisateur pourra spécifier le pourcentage d’ouverture de la porte.
* Les données collectées par l’application de l’objet (Raspberry Pi) sont sauvegardées dans une base de données.

Descriptions des étapes dans Azure:

1. Après avoir activé mon compte étudiant dans Microsoft Azure, j'ai procédé à la création d'une ressource IoT Hub :
2. J'ai ensuite procédé à l'ajout d'un périphérique à mon IoT Hub (c'est-à-dire mon Raspberry Pi) :

Une image contenant texte, moniteur, capture d’écran, télévision

Description générée automatiquement

1. J'ai ensuite dû créer une base de données SQL pour stocker les données envoyées par le Raspberry Pi au Hub IoT :
   * J’ai utilisé SSMS (SQL Server Management Studio) pour pouvoir configurer les tables dans la base de données à distance d’Azure.
2. Pour réaliser le processus de sauvegarde des données de IoT Hub (qui reçoit les données du Raspberry Pi) dans la base de données, j'ai dû créer une Azure Function qui allait être exécutée chaque fois que IoT Hub recevrait de nouvelles données. J’ai crée un project à l’aide de Visual Studio. Ce project contient tous mes Azure Functions.
3. J’ai également ajouté deux autres fonctions : Une qui me permettra de récupérer les données depuis la base de données SQL dans Azure et une autre fonction pour pouvoir envoyer des messages vers ma Raspberry Pi (voir Description des codes de l’application).
4. Une fois Azure configuré, j’ai commencé à programmer ma Raspberry Pi, mon application web et mon application mobile.

Capture d’écran:

Groupe des ressources : Certains ressources ont été implémentés, car se sont de dépendances d’autres ressources.

A screenshot of a computer

Description automatically generated with medium confidence

Azure Functions :

A screenshot of a computer

Description automatically generated with medium confidence

IoT Hub : Je n’ai pas pu prendre une capture de mon périphérique, car mon crédit d’étudiant a été finit.

Graphical user interface, application

Description automatically generated

Description des codes de l’application :

Raspberry Pi :

* La Raspberry Pi est celui qui collectera les données depuis ses senseurs et les envoyera au IoT Hub. Grâce à une Azure Function, je suis capable de stocker les données reçues dans une base de données.
* L’application de la Rapsberry Pi a été codé en Python. J’ai utilisé la librairie TKinter pour faire une interface graphique. J’ai utilisé le patron MVC.
* Pour des fins de tests, j’ai créé un simulateur en C# .NET pour éviter de coder directement sur la Raspberry Pi à chaque instant. Alors, dans le cas que la Raspberry Pi ne fonctionne pas, on est capable de la simuler grâce à cette application.

Azure functions :

* Les functions Azure ont été codés en C# .NET. En utilisant Visual Studio, j’ai publié le projet dans ma ressource Azure Function dans Azure.
* J’ai utilisé :
  + Une fonction Event Hub Trigger pour sauvegarder les données reçus par le IoT Hub dans la base de données SQL dans Azure.
  + Une fonction http Trigger pour pouvoir récupérer les données de la base de données par une application.
  + Une fonction http Trigger pour pouvoir « post » un message qui va être envoyé à la Raspberry pi. La Raspberry Pi interceptera le message et adaptera sa fonctionnalité en fonction du message.

Base de données :

* + Le système de gestion de base de données dans Azure est SQL Server. Pour configurer cette base de données, j’ai utilisé SSMS (SQL Server Management Studio).
  + En ce qui concerne la Raspberry Pi, j’ai utilisé MariaDB comme SGBD. Pour configurer cette base de données, j’ai utilisé PHPMyAdmin.

Application Web :

* L’application web a été construit en utilisant Angular.

Application Mobile :

* L’application mobile a été construit pour Android avec Android Studio.
* Pourtant, celle-ci n’as pas été finit, car elle était optionnelle.

Manuel d’utilisation :

* Application web :

Graphical user interface, text, application

Description automatically generated

Nous pouvons contrôler la porte en spécifiant : le mode et le pourcentage d’ouverture de la porte si le mode manuel est choisi.

Il est possible de consulter les informations de la porte qui proviennent de la Raspberry Pi.

Autres détails:

* Malheureusement, j’ai pas pu faire une démonstration du fonctionnement, car Azure m’indique que j’ai dépassé mon crédit d’étudiant.