Soufan Weather Station

Weather Station Application with Configuration Mobile Application

Mohamad Soufan

December 13, 2017

Project Advisor & Sponsor: Dr. Roberts

ABSTRACT:

This project forms a configuration mobile application for a personal weather station. The mobile application will allow users to configure their station’s wifi access using only their phones. No actions need to be taken on the hardware (no displays, numpad, pins ...etc). The application can support multiple weather stations at a time. The application allows the users to switch theirs stations from using one wireless network to another. The project aims to find a user-friendly manner to configure personal weather stations with minimum effort.
AKNOWLEDGEMENTS:

I thank the project sponsor and advisor Dr. Roberts for his help in supporting resources and for his critical suggestions regarding the hardware and software part of this project. I also appreciate Dr. Hwang help in mentoring and advising the seminar section of the project. Thanks for her supervision and for her suggestions and recommendations that kept the project formal and professional on all aspects.

FIGURES

1. Backend Section: Overview Diagram
2. Raspberry Pi Model Zero W
3. Mobile App Pages Design (1-3)
4. Mobile App Pages Design (4-6)
5. Mobile App Pages Design (7-8)
6. Screenshots of Network Page

TABLES

1. API Content

INTRODUCTION

Weather data is an important parameter for humans. Individuals tend to learn about their weather from several sources such as the weather forecast or the internet. However, with the emerge of the new technologies PWS (Personal Weather Stations) have become more and more common to support more accurate and personalized weather data. PWS form a powerful and an easy-to-use system for customers. However, when installing these units one issue might show up which is internet/wifi configuration. Many methods have been suggested by commercial vendors to
overcome this obstacle, but each had a cost. This project proposes a user-friendly solution to configure wifi for PWS units.

**PROBLEM STATEMENT**

Personal weather stations are a modern and an efficient method to track weather data. The entire system in which these stations function in is maintained and established via internet access. Vendors for PWS spend effort to come up with ideas to facilitate internet configuration for their system. Here is a list of methods that has been suggested:

1. Use of ethernet cables (can become expensive very quickly)
2. Numpad or hardware keyboards built on the stations itself (complicated to use)
3. Programmatical methods using Python (Not convenient for all users)

Obviously, each of these current methods have advantages and disadvantages. However, some of the drawbacks are not sustainable and can cause major harm to the vendor’s business. An ideal solution must overcome the financial barrier and the complex practices that some vendors depends on.

**REQUIREMENTS & SPECIFICATIONS**

The sponsor for this project has given these requirements:

1. The project must include an easy to install and use weather station hardware
   
   GroveWeatherPi sensor station.

2. Weather station system configuration must be user friendly.

3. Cross-platform configuration mobile app (Targets wide range of users)

4. Solution must be inexpensive
5. Weather station hardware used must not include any mechanical side (i.e. fully programmable and accessible by connecting it to a small computer).

GroveWeatherPi is a programmable open-sourced personal weather station. As a computer science project, the mechanical design aspect of any hardware to be used must be minor, and the GroveWeatherPi station completely fulfills this requirement for both the installation and programming stages. However, the configuration stage for GroveWeatherPi or any other personal weather station in the market requires a sort of interface to maintain internet connection which sometimes could be difficulty for most users. The project includes and easy-to-use mobile app that configures the internet connection seamlessly without requiring any programming background or complicated wiring and allow an average user to configure the station quickly and simply.

The weather station collects different parameters that include: Daytime and nighttime temperatures, humidity, atmospheric pressure, air quality, liquid precipitation, and wind speed and direction. The hardware includes a weather board that has the following connectable sensors: Thermometer, hygrometer, barometer, anemometer, and rain gauge. This hardware configuration is essential to make the station more efficient and reliable.

The project includes software to drive the hardware and take readings. In addition, the software is able to take readings periodically by each specific sensor. This dedicated software is established to drive the hardware and utilize the full potential of its sensors to make their valuable data accessible to the programmer. The driving software connects to a server using an API that allows the software to send data and store it in a database on the server.

One of the most important pieces to this project is the hardware-server connection protocol. This medium is established through an API that forms an easy route for the data to be transferred
and sent to the server. Only the measurements that differ from their previous values are sent over the API. A handling mechanism on the server is going to organize the SQL database and observe which parameters have been changed and which have not. Consequently, the mechanism establishes a data report of all the weather measurements and sends it thereafter to the mobile device. A different API is established to manage the connection protocol between the server and the mobile app. The API directs and updates the data being reported to the user on the mobile app. This API is a beneficial resource for any mobile developer because it facilitates the connection between a mobile app and the weather data server.
Figure 1: System Overview Diagram [1][2][3][4][5][6]
The overall system from one end to the other consists of three main sections:

1. A hardware sensors station (GroveWeatherPi) and a Raspberry Pi Zero W.
2. A web server that has the storage facility.
3. Mobile configuration application.
4. A dedicated mobile app for the users (developed by another engineer).

The first two sections form the data engine for the weather station (i.e. backend). The mobile app is the interface with the user (i.e. frontend). There are two parts for the API that manages the connection between the parts. One part is responsible to send the different collections and variations of weather from the Pi to the server. The second part’s facilitates the data requesting process that goes to the server from the mobile app and provides a straightforward way for the frontend developer to communicate with the backend engine of this project. This part is mainly involved with the backend section (parts one, two and three) while the frontend section (part four) is to be accomplished by another developer.

At the backend side, there are three important parts. One is the sensors station such as the GroveWeatherPi. Two is the small computer Raspberry Pi Zero W. The GroveWeatherPi is a reliable station that features four different sensors: thermometer, hygrometer, anemometer, rain gauge. The GroveWeatherPi provides a very efficient hardware container for the sensors because of its extensibility. This container allows the replacement and the addition of new sensors that may not be part of the existing collection such as a barometer.
The second piece of hardware this project requires is a micro-computer or a small portable processing unit that can receive the readings from the different sensors and report them. Raspberry Pi Zero W, shown in Figure 2, is adequate for this project. The Raspberry Pi is an open-system computer that allows connections through various protocols such as Wifi, spi, i2c, etc. In addition, the Raspberry Pi can communicate easily with the sensors and organize the readings data coming from them.

![Figure 2: Raspberry Pi Model Zero W](image)

Utilizing this small machine, the project will be able to perform several tasks. For example, establish a stable connection with the sensors, receive the periodic readings from the station, connect to the web server, and group the data and prepare it for sending to the server. However, in order for the Pi to accommodate all these tasks, corresponding software must be written. A mobile application is written to facilitate the configuration stage. The Pi has a Bluetooth and Wifi chip that allows outer communication. A mobile phone can be paired to the
Pi via Bluetooth, however, a software in Python and ShellScript is written to surpass the Bluetooth PIN Authorization stage. Then once pairing is established a dedicated mobile application allows the user to view the Pi/Station and manages the communication over a Bluetooth socket by sending data packets of 39 KiloBytes. Figures 3-5 has a basic design for mobile app. Then a Shell script runs on the Pi to turn on Wifi. The user on the mobile phone can send network SSID and Password over the Bluetooth channel. A receiving script is written on the Pi which permits the Pi to establish Wifi communication and start sending weather data to the server. After the Wifi is fully established a script to send success message to the mobile phone and prompt the user to restart the system. Then user send a Bluetooth packet requesting a reboot. After successful reboot a script runs to start collecting and sending weather data to the online server. This is an essential part since the weather station is a power sensitive project. The developer should pay attention for power consumption that limits the usage of unneeded resources (such as the Bluetooth chips). Another feature that is added to the program is the ability to edit an already established connection or device. Hence, the user can change the wifi network that is connected to or edit the device name. The initial design for the configuration mobile app is to be cross platform written using the Xamarin technology which facilitates writing apps for all common mobile platforms (iOS, Android, and Windows). Xamarin is suitable technology for the configuration application for several reasons: One, on the frontend side, Xamarin supports easy-to-use UI element for non-heavy frontend projects (for a config app, no fancy animation is required). Two, on the backend side, Xamarin also provides the programmer with full access to the native platform API which is a required feature for this app since Bluetooth communication programming is part of the native API. Lastly, application production
and deployment are very simple and quick using Xamarin studio and won’t require any additional 3rd party servers.

Figures 3-5 describe the actual design of the mobile app pages.
Figure 4: Mobile App Pages Design 4-6 [9]
Another part of this project is, connecting the sensor station to the Pi will require a set of programs to connect to each sensor separately. All the sensors that are provided with the GroveWeatherPi station are connectable through I2C, which is a medium that is already supported by the Pi. Thus, the software to drive the thermometer, anemometer, hygrometer and rain gauge must establish a I2C connection with the different sensors.
The second part of the project is the server. The main method for storing the data -and displaying it for the user is implemented through a storage facility that lives on the Pi that users and programmers can connect to and use. An SQL server is constructed and the software on the Raspberry Pi reports to the server periodically. An API is established to administrate the usage of this SQL local server, which has a comprehensive set of functions that allows data registration to the storage facility. The API is a library of programmed functions that allows the Raspberry Pi to use the database and store data in it. An example of these functions is to be provided in Table 1. The data registration intervals are dictated by the Python code on the Pi. Only the parameters that have been changed will be transferred to the remote server. Extra functionalities are added to the API such as data grouping. This functionality allows the sending of multiple parameters at once which makes the system faster. The group is a list of floating points numbers that reflect the set of changing parameters. The functions in the Table 1 wraps the SQL code that stores the data into the database. The database code consisting of SELECT and INSERT functions is a separate piece of software. However, this API makes the programmer job easier and hides all the SQL and database code, so the only code the Raspberry Pi is running is Python code.

<table>
<thead>
<tr>
<th>Functions:</th>
<th>Parameters:</th>
</tr>
</thead>
<tbody>
<tr>
<td>StoreTemperature</td>
<td>double tempr</td>
</tr>
<tr>
<td>StoreHumidity</td>
<td>double humid</td>
</tr>
<tr>
<td>StoreWindSpeed</td>
<td>double windspeed</td>
</tr>
<tr>
<td>StoreWindDirection</td>
<td>double windDirec</td>
</tr>
<tr>
<td>StoreRainGauge</td>
<td>double raindrop</td>
</tr>
<tr>
<td>StoreAirPressuer</td>
<td>double press</td>
</tr>
</tbody>
</table>

*Table 1 API content*

The last part of the system, that is related to the backend side of this project, is the second API. In this project WeatherUnderground developer API is used to fulfill the purpose of this part.
RESULTS

GroveWeatherPi hardware station is a light weight, extensible, and reusable system. The Raspberry Pi acts as the network administrator and data organizer, while the API facilitates the flow of the data from one end of the system to the other. All parts are important, and the possibility of an efficient data reporting is dependable on the establishment of a correct and an effortless connection among them.

Having a configuration mobile application is an unprecedented feature for personal weather stations. It eliminates the need for a long ethernet cable or the programming background that a user must have to configure the wifi connection of the Pi. This property is essential for a more seamless user experience.

The completed features in this project:

- User can successfully connect the weather station to wifi
- Users have the ability to manage multiple weather stations at a time
- The application allows switching between wireless networks
- The application is fully supported on all three main platforms (iOS, Android, Windows)
- Fully functional and an inexpensive method to configure the PWS wifi/internet access
FUTURE WORK

- Add tablets support for the mobile application
- Allow wifi to be configured for multiple networks and support automatic network adapter that will pick the network with the more powerful signal and auto connect to it
- Add monitor mode to the mobile app which can list all weather parameter being sent over wifi/internet
• Add credentials for Bluetooth communication to secure the channel between the phone and the Pi

BIOGRAPHY

My name is Mohamad Soufan, I am a CS student at the University of Evansville. I am from Damascus, Syria. I am 23 years old. I have successfully completed five computer science internships and all required and elective computer science classes. I have been developing code for 4 years so far. After my graduation I will start a position as a .NET developer at ADESA in Carmel IN.
REFERENCES

   Acurite. Web. 29 Apr. 2017. URL:

   <https://arqspin.com/tag/api/>

   <https://clipartfest.com/categories/view/f3215aa4cb7b88dab94aae26dc2fd4a0e57af773/web-server-clipart.html>

   <http://madisonbeckerman.com/Weather-App>

   <https://www.raspberrypi.org/blog/raspberry-pi-2-on-sale/>

   <https://thekenyonthrill.com/2014/01/14/vive-la-difference/>