Teaching Case

An Introductory IS Course Culminating Project – Weather Easy as Pi

Anthony Serapiglia
anthony.serapiglia@stvincent.edu
St. Vincents College
Latrobe, PA

Abstract

Many introductory courses in CS/IS are modular by nature. This allows for multiple topics to be introduced to beginning students and to serve as a springboard to higher-level material in later courses. Unfortunately, in attempting to fit many topics in, connections between topics or a cohesive theme can be difficult to establish or maintain. The introduction of a final, multifaceted, project that incorporates the individual topics and concepts of the course can serve as the catalyst enabling the whole to gel within the nascent CS/IS student. The purpose of this paper is to present such a culminating project that is flexible in the amount of topics included, and difficulty level - a mini weather station Raspberry Pi project.

Keywords: Information Systems, Internet of Things, IT Project Management, EATPUT

1. INTRODUCTION

There have been many approaches to structuring introductory Computer Science (CS) and Information Systems (IS) courses. It can quickly become a "chicken or the egg" question when determining whether it is better to go deep and narrow or wide and shallow. How best to pack all of it in?

Many incoming freshmen or beginning students have only experienced a very narrow set of technologies or specialties in the discipline. As consumers of technology, they have been very proficient and accomplished. Whether for entertainment or within a classroom, students have been able to do what they have been asked to do, and at very high levels. They are Digital Natives after all. However, many have not been asked to create complex systems or go beyond single task/single technology exercises. Getting under the hood or straying from the guided path is not something that many high school aged students have been asked to do.

CS/IS are popular and sought-after degrees. According to the National Center for Education Statistics, in 2021-22 CS/IS degrees were the fifth highest category with 97,047 degrees issued (NCES, 2023). Many students are motivated by continued job growth and salary growth as projected by the Bureau of Labor Statistics. Both job openings and salaries have increased much faster than the average for all occupations and the sustained growth is expected to continue (USBLS, 2023).

The level of preparedness for a CS/IS major varies widely. According to a 2023 study by Code.org, only 8 states have legislation requiring public school students to pass a CS course for graduation, two of which were added in the 2022/23 academic year. The same study reported 14 states where less than half of high schools offer CS instruction. Nationwide, 57.5% of US public schools offer a foundational CS class. This leaves more than 10,000 high schools across the country that do not offer a single CS course. This number is also unevenly spread. Within the lowest 35 states, only 5.8% of students are enrolled in a foundation CS course (Code.org, 2023).

Without a head start in high school, many students arrive at a college already having declared for a CS/IS major with little to no formal introduction to the disciplines. This makes the role of an introductory CS/IS course even more important and valuable in filling in blanks, leveling the playing field, and establishing a solid foundation.

This paper describes the introduction of a capstone project into an introductory CS/IS course. The project incorporates subject areas of System Design, Hardware Integration, Networking, Python Scripting, Operating Systems, IoT, and System Lifecycle. Other topics of Cybersecurity, Privacy, and Data Management can easily be added.

Included in this paper is a background of the course and the evolution towards a capstone project, a description of the materials and environment necessary, an outline and instructions for each stage of project development, commentary and observations from implementation, and options for scaling the project up or down from the framework.

2. BACKGROUND

This project is part of a Computing and Information Systems department at a small private religious liberal arts college. The department supports three degree programs, Bachelor of Science degrees in Computer Science, Cybersecurity, and Information Systems.

Previous to 2012, the first course for all three programs was introductory programming. To more align the degree programs with model curriculum examples, and also in response to student surveys, a new course was added as a "CS-Zero" orientation class. This course was designed to be complimentary to the introductory programming course and to include a multitude of subjects as a stepping stone to more advance courses. As such, a new introductory course was designed in a very modular fashion to allow for flexibility and evolution of topics as the years progressed.

Feedback from student course evaluations remained positive for the next decade. For many students the broadness of the introductory material was comforting. Some were thankful that a stated goal of the course was to create a level start for everyone, knowing that there was a wide disparity of experience within each incoming freshman class.

Beginning in 2021, feedback from students began to evolve to include several comments from students that the course was starting to become too fragmented. Students noted that the connections between topics were not always visible or emphasized. In a course that emphasizes systems, the greater picture as a whole was being missed.

Reflecting upon the original intent of the introductory course, the primary goal was for the content of the course to be a stepping stone to other, deeper, experiences through the degree program. This was accomplished with introductory material for topics such as Networking and Databases, but a primary area not addressed was project management and preparation for classes such as Software Engineering, Systems Analysis and Design, or the Senior Project capstone.

Introducing a mini-capstone project within the introductory course would serve to provide a platform for cohesively bringing together the topics of the course into one as well as preparing students for the complexity of larger projects later in the degree program.

The EATPUT Information Systems Model
Dr. Anthony Debons was an experimental psychologist and early pioneer in Information Science. Debons worked closely with the Army Air Corps and US Air Force in the years after World War II developing command and control systems. These were heady days of advancements in Information Systems, Decision Support Systems, and ADIK (Advanced Data Information Knowledge) systems. While these specific labels may have gone out of favor, their core simplicity in structure and framework is worth revisiting as models for modern "complex" systems. (Serapiglia, 2022)

Beginning in 1960, Debons led a project to establish a conceptual framework for the design of an information system to support command and control for the Strategic Air Command. This project was a contemporary of the group led by J.C.R. Licklider at DARPA, with Debons and Licklider both having backgrounds in psychology and wide interdisciplinary views of information systems. According to Debons, they conferred on a number of occasions, including consultations on funding devoted to projects focused on the development of better software to train computer programs that would benefit both of them (Asprey, 1999). These efforts in developing command and control systems for the military had great influence on the development of early management information systems and decision support systems that have evolved into the systems of today (Asprey, 1999).

Debons' work culminated in an abstract model of an Information System – EATPUT. Consisting of six basic components, the first letters of which produce the acronym. The six components of EATPUT are:

Event World – The occurrences that are relevant to the objective and functioning of the information system. It includes the classifying and categorizing of events and the representation of them in symbolic form.

Acquisition – The initial physical component of the system, used to capture matter and energy describing an event from the external environment (data).

Transmission – The actual movement of signals (data) within and between components of the system.

Processing – The ordering, storage, and retrieval of data for the ultimate purpose of applying it to problem solving, decision making, or general development (knowledge formulation).

Utilization – The component that represents the evaluative, interpretive requirement of information systems

Transfer – the action component of the system; the implementation of the decider function through the system's transfer medium. The Transfer function in this model can be seen as communication or information transfer (Debons, Horne, Cronenweth 1988).

Internet of Things Six Layer Model
In 1999, Kevin Ashton coined the term Internet of Things (IoT) in describing a system of uniquely identifiable sensors connected to the Internet (Ashton, 2009). Since that time, there have been many variations of models of Internet of Things (IoT) structure. Some may be as few as three layers (Khajenasiri et al., 2017), and others as large as seven (Padmanabhan, 2022). As an introduction, this exercise builds from a six-layer IoT model that is derived from Dr. John Barrett (Barrett, 2012). Dr. Barrett's original model included five layers. A sixth wrapper layer of security has been added to emphasize the need to incorporate Cybersecurity consciousness in the system lifecycle. The basic layer of Dr. Barrett's original model are:

Identity Layer – The need to uniquely identify each system/sensor. This can be accomplished through existing network identifiers, such as MAC or IPV6 addresses.

Sensing Layer – This may also be termed the perception layer. This layer includes any device that enables the gathering of data from outside of the system.

Communications Layer – This may also be termed the network layer. This layer allows for the movement of data throughout the system and beyond. This layer may utilize multiple technologies, Including Bluetooth, Ethernet, Wi-Fi, and more.

Data Storage Layer – This layer determines how data is handled both in use and at rest. Technologies that may be utilized in this layer include MicroSD cards and High Bandwidth Memory.

Application Layer – This layer can also be termed as a middleware, business, or management layer. As an application layer, this is a multifaceted catch-all for working with collected data and allowing user access.

Security Layer – This layer is not often included in IoT models. It is important to stress to students the need to ingrain cybersecurity concepts into all systems. This layer can include the acknowledgment of encryption for data, access control models, backup/recovery methods, and much more.

3. WEATHER STATION PROJECT

Project goal: This project was designed to engage students in the process of developing an information system to solve a need or problem. Students build a working information system that will gather measurements of the environment and display them on an accessible website within the department network. A successful project results in the ability of the instructor to access a webpage served by the student's system that displays the temperature, humidity, and time.

This project was originally spread across a two-week window in a course delivered in a Monday/Wednesday/Friday fifty-minute format.

Project Stages

1. Project overview, environment analysis, solution identification, project planning
2. Proposal
3. Base OS installation
4. Apache installation and verification
5. Sensor installation, script configuration, and output verification
6. Web page creation and accessibility verification
7. Documentation

The timing of the project was designed such that the initiation and introduction took place on a Friday. This allowed for students to have the weekend to develop and submit a project proposal that was due the following Monday. This arrangement also allowed for the final project day presentations to take place on the Wednesday of the last week of classes. Friday, the last class of the semester, was able to be reserved for review.

While students were given the opportunity to work outside of the classroom, most students worked within the class time and computer lab environment.

Project Kit

The following seven items were bundled together in resealable gallon-sized bags and provided to students:

Raspberry Pi Model 4B computer board
MicroSD card
DHT11 Temperature Humidity Sensor Module
GPIO pin jumper wire
Micro HDMI to Display Port cable
USB cable for power
USB power block

Working within the department computer lab, students were able to utilize the existing monitor, keyboard, mouse, and network cable at each station.

Based on July 2023 prices, each kit costs approximately $82 with most components available through Amazon.

Project Overview

Stage1:

At the beginning of the project, students are reminded of the course topics that have preceded this capstone exercise. At the conclusion of the recap, a scenario is developed with class discussion.

For this project, a problem scenario was initiated related to the specific room in which the class is held. Discussion is started based on the Internet of Things sections that precede the final exercise. With a focus on visibility into the environment, students are asked what variables within the room would be most valuable to monitor. The most common responses have been related to occupancy. Students immediately identify the need to know if the room is occupied and if any computers are available for use. Beyond those variables, with some prompting other environmental aspects are identified. This classroom has a set of windows facing west, and at various times during the year can become relatively warm as the sun sets. Several years previously, there was also an emergency situation within this room where a pipe had burst in the ceiling, resulting in some flooding.

As an introductory course, the layout of the system is provided to the students in a general outline: The project will consist of students creating a temperature and humidity reporting system. The system will be comprised of a sensor and a Raspberry Pi computer. The sensor will collect data through a Python script on the Raspberry PI and save the readings to a text file. The Raspberry Pi will also support an Apache web server which will display a web page configured to display the contents of the text file of collected readings and refresh on a determined interval. The Raspberry Pi/Apache server will be connected to the school network such that the page displaying the collected data will be accessible to other computers on the school network.

Given the outline of the system, the first assignment for students is two-fold. First, students are asked to re-state the problem and solution in their own words. This paper should include a description of the environment, resources available, restrictions, and other factors of note (such as traffic of rotating classes throughout the day). An inventory of the devices, software, and other resources is required. This inventory list should also contain a preliminary budget with links to possible purchase locations and pricing. In re-stating the solution, students are directed to project the proposed system to the EATPUT Information System model, as well as a six-layer IoT system model.

Stage 2:

Stage two is a gateway assignment. Students are required to develop and submit a one-page executive summary proposal to request permission to proceed with the project. The proposal must include a summary of the situation, a description of the system, and a bottom line estimated cost. Upon submission and review of the proposal, students receive the project kit containing the equipment needed to build the system.

Stage 3:
Once students have a project kit, the first action stage is to establish the Raspberry Pi as a working computer. Initially, as a way of reducing possible failure points, students were provided a microSD card pre-flashed with the latest version of the Raspberry Pi OS. The OS had not been setup, requiring students to still work through the initial installation info screens. Students are directed to the official Raspberry Pi website for documentation and initial Getting Started instructions. Once at a desktop, students are required to verify connectivity and to update the OS through a terminal command.

The artifact for Stage 3 is a dialog/journal of installation steps that includes a final screenshot of the Raspberry Pi displaying a terminal window showing a successful update of the OS.

Stage 4:
Once the Raspberry Pi is updated, the next stage is the installation of Apache a web server. Again, students are directed to install Apache through a terminal window. While the process for this stage is short compared to others, it is stressed to students that it is important to separate and isolate different tasks.

The artifact for Stage 4 is a dialog/journal of the installation steps required to install Apace on their Raspberry Pi. Screenshots are required to show the installation success in the terminal window. A screenshot is also required to show the default Apache landing page accessible through the network IP address and the local loopback address.

Stage 5:
Stages 5 and 6 combined may be the most challenging for introductory students. Each requires some programming. Depending on the level of the students, some initial code may be provided and then modified by the students.

The focus of stage 5 is to attach the DHT11 Temperature/Humidity sensor. There are several ways to interact with the DHT11 sensor. The DHT11 sensor is a popular device and has many libraries associated with it for free use. The library utilized by this project is adafruit-circuitpython-dht available through the Adafruit Industries GitHub repository. The DHT11 is also a digital device, so there is no need for an extra analog-digital convertor. The sensor can attach directly to the General Input Output Pins (GPIO) of the Raspberry Pi.

The first part of Stage 5 is connecting the sensor to the Raspberry Pi. The DHT11 sensor has three connectors. These are labeled with "+", "out", and "–" markings.

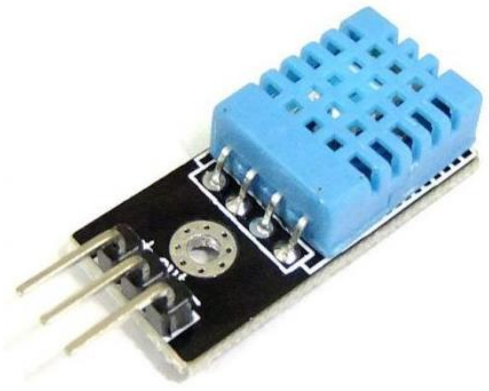


Figure 1: DHT11 Temperature/Humidity Sensor

Connecting to the Raspberry Pi GPIO pins is simple when utilizing female to female jumper cables. The "+" connector will be connected to pin 2 on the Pi. The "out" connector will be connected to pin 11, GPIO17 within the Pi. The "-" pin will be connected to pin 6 on the Pi.

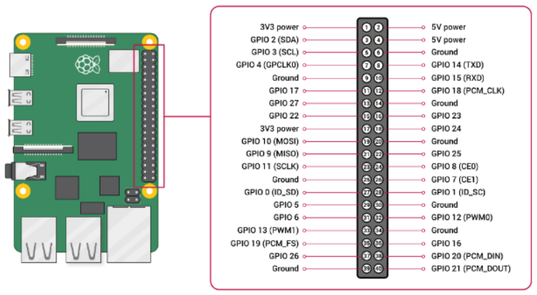


Figure 2: Raspberry Pi GPIO Pin Assignments

Take special note of the orientation and labeling of the pins on the DHT11 and the corresponding pins on the Raspberry Pi. Improper connection, the reversal of the plus and minus, can lead to the destruction of the sensor.

Once connected, Python and the necessary libraries can be installed through the following commands:

```
sudo apt install python3 python3-pip
python3-venv
mkdir ~/dht11
cd ~/dht11
python3 -m venv env
source env/bin/activate
python3 -m pip install adafruit-
circuitpython-dht
```

A simple way of saving the data is to place the text file in a subfolder of Apache. In a terminal window, browse to the default html folder for Apache (/var/www/html). Create a subfolder for the output file name DHT11. Assign RWX permissions appropriately (… I.E. NOT 777). Open a second terminal window for the following commands:

```
cd /var/www/html
mkdir DHT11
cd DHT11
sudo chmod -R 327 ./
```

Configuring the Python script
The Raspberry Pi OS has a few preinstalled scripting tools to help create and edit code. From the top Pi menu, go to the Programming section and open Thonny. Thonny is an editing environment that will allow you to create or edit Python scripts.

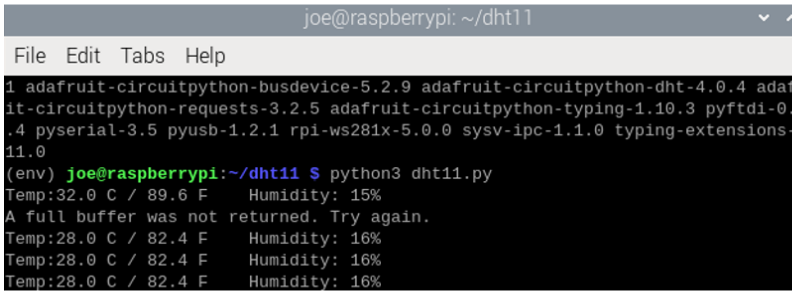
Once Thonny opens, first save the opened blank document as dht11.py in the "dht11" folder created in the home folder during the previous step. Sample code has been provided in Appendix A. Copy the Python code into the editor. Save the file.

Returning to the original terminal window. Verify that it is in the "dht11" folder and the environment is still active.

Run the script:

```
        python3 dht11.py
```

The output of the script shows the temp in Celsius, Fahrenheit, and Humidity:



```
1 adafruit-circuitpython-busdevice-5.2.9 adafruit-circuitpython-dht-4.0.4 adaf
it-circuitpython-requests-3.2.5 adafruit-circuitpython-typing-1.10.3 pyftdi-0.
.4 pyserial-3.5 pyusb-1.2.1 rpi-ws281x-5.0.0 sysv-ipc-1.1.0 typing-extensions-
11.0
(env) joe@raspberrypi:~/dht11 $ python3 dht11.py
Temp:32.0 C / 89.6 F    Humidity: 15%
A full buffer was not returned. Try again.
Temp:28.0 C / 82.4 F    Humidity: 16%
Temp:28.0 C / 82.4 F    Humidity: 16%
Temp:28.0 C / 82.4 F    Humidity: 16%
```

Figure 3: Script Output

Verify that the "output" file exists in the html folder. Move to the second terminal window. Show the directory contents (ls) of the DHT11 folder and verify the output file exists.

The Stage 5 artifact is a dialog/journal of the installation steps required to install Python, configure directories, create/modify the DHT11 Python script, and to run/verify the DHT11 Python script. Screenshots are required to show the script running in a terminal window, validation of the output file creation, and contents written to the output file.

Stage 6:
With a working sensor successfully writing data to an output file, the next step is to configure a web page that will display the contents of the output file in a web browser.

With an introductory class, sample code has been provided. See Appendix B for the sample html code. There are multiple approaches to displaying the contents of a text file from an HTML document, this sample utilized fetch. Depending on the level of the course, students could be asked to develop their own code and pursue other methods.

To utilize the provided sample HTML code, open a terminal window on the Raspberry Pi and browse to the /var/www/html directory. Rename the existing index.html page with the following command:

```
mv index.html index.ori
```

Open the Thonny editor that was utilized in Stage 5. In Thonny, create a new file. Save this file to the default location for Apache by browsing to other locations – computer – var – www – html. Save the new file as index.html. Copy the contents of the index file sample code into Thonny. For personalization, have students add their name as the top displayed line to the index page. Save the file.

Open a web browser and enter 127.0.0.1 as the loopback address. Confirm that the page now appearing is the new index.html page. If the original Apache page is still showing, try a "hard refresh" by hitting the Ctrl-F5 buttons together.

Open a terminal window and find the IP number assigned to the Raspberry Pi utilizing the "ip a" command. Enter the IP number into a web browser to confirm that the temperature and humidity data are displayed. Share the IP number with another student and the instructor to confirm that the index page can be viewed by others on the network.

The stage 6 artifact is a dialog/journal of the steps required to rename the existing index.html page, creation of a new index.html page, the modification/configuration of the index.html page, and the confirmation that the index.html page displaying the temperature/humidity data is accessible on the network. A screenshot of the index.html page accessed through the LAN address with the student's name and current data displayed must be included.

Stage 7:
The final piece of the project is documentation. With a limited time window to complete the project at the end of a semester, the emphasis is placed on documenting as you go. If students have been diligent in keeping a running dialog/journal during the stages of system development, this final stage is an exercise in combining those artifacts into a cohesive whole. Introducing organization, note-taking, and documentation concepts in the early stages of a student's journey is just as important as any technical skill. The idea of a twenty-page report on a project will seem very intimidating to first-year students. However, once they add together the amount of pages from the combined assignments leading up to the final report, they quickly realize that they are most likely past that number already.

At the low end of the rubric are reports/documentation that simply package the previous sections together with a table of contents. More developed reports will revisit and revise sections with broader observations, details, and commentary. Fully developed documentation will include all sections, a cohesive narrative of the project, and appropriate reflection with tips for improvement and further development paths.

4. PROJECT REFLECTION

This mini weather station Raspberry Pi project was designed to serve as a culminating capstone exercise in an introductory CS-Zero or Introductory Information Systems course. There are many goals for the project, and they can be managed in a flexible fashion to emphasize sections as needed or desired. The level of difficulty can also be adjusted by managing the amount of prepared code provided to students in different stages.

This project has been delivered in two semesters with twenty-five students during the fall semester and twelve students in the spring. Overall student response to the project was positive. A short student survey of five questions was conducted at the end of the project each semester. Combined, thirty-one students responded. Each question was ranked on a five-point scale with 1 as low and 5 as high. For each question, the lowest response was a 3.

Question 1 – This project helped me understand the different topics of the course and how they interrelate to each other. Responses to question one averaged 4.52.

Question 2 – This project helped me understand Information Systems in general. Responses to question two averaged 4.55.

Question 3 – This project helped me understand the stages of IT project management. Responses to question two averaged 4.42.

Question 4 – After completing this project, I feel more confident about how different technologies interact with each other. Responses to question two averaged 4.52.

Question 5 – Overall, this project increased my interest in Information Systems. Responses to question two averaged 4.58.

During the project, students expressed that they enjoyed being able to physically work with the Raspberry Pi and the sensor. While the hardware interactions are very simple, many students have not had the experience of piecing a system together. Several students also expressed their gratitude for how the project was broken into stages. They had originally felt overwhelmed by the description of the project, but that anxiety diminished as they saw how the work was broken out into manageable pieces. At the end of the project, several students began discussing ideas on how to expand their system, expressing interest in incorporating authentication methods and cloud hosting.

4. REFERENCES

2023 State of CS Report. Code.org. (2023, January 1). https://advocacy.code.org/stateofcs. Retrieved June 14, 2024, from https://advocacy.code.org/stateofcs.

Ashton, K. (22 June 2009). "That 'Internet of Things' Thing". rfidjournal.com. Retrieved 9 May 2024. https://www.rfidjournal.com/expert-views/that-internet-of-things-thing/73881/

Aspray, W. (1999). Command and control, documentation, and library science: The origins of information science at the University of Pittsburgh. *IEEE Annals of the History of Computing*, *21*(4), 4–20. https://doi.org/10.1109/85.801528

Barrett, J. The Internet of Things [Video]. Ted Conferences, TEDxCIT. September 6, 2012. Cork, Ireland. Retrieved fro: https://www.youtube.com/watch?v=QaTIt1C5R-M

Debons, A., Horne, E., & Cronenweth, S. (1988). *Information science an integrated view*. G.K. Hall.

Khajenasiri, I., Estebsari, A., Verhelst, M., & Gielen, G. (2017). A review on internet of things solutions for intelligent energy control in buildings for Smart City Applications. *Energy Procedia*, *111*, 770–779. https://doi.org/10.1016/j.egypro.2017.03.239

National Center for Education Statistics. (2023). Undergraduate Degree Fields. *Condition of Education.* U.S. Department of Education, Institute of Education Sciences. Retrieved May 30, 2024, from https://nces.ed.gov/programs/coe/indicator/cta.

Padmanabhan, A. (2022, February 15). *IOT security model*. Devopedia. Retrieved March 2, 2024, from https://devopedia.org/iot-security-model

Serapiglia, A., (2023). Command and Control – Revisiting EATPUT as an IS Model for Understanding SIEM Complexity. *Cybersecurity Pedagogy and Practice Journal*2(1) pp 77-84. http://CPPJ.org/2023-1/ ISSN : 2832-1006. A preliminary version appears in The Proceedings of EDSIGCON 2022

The NCES Fast Facts Tool provides quick answers to many education questions (National Center for Education Statistics). National Center for Education Statistics (NCES). Retrieved June 14, 2024, from https://nces.ed.gov/fastfacts/display.asp?id=37

U.S. Bureau of Labor Statistics. (2023, September 6). *Field of degree: Computer and Information Technology*. U.S. Bureau of Labor Statistics. Retrieved June 14, 2024, from https://www.bls.gov/ooh/field-of-degree/computer-and-information/computer-and-information-technology-field-of-degree.htm

ISSN: 2473-4901
v10 n6175

APPENDIX A
Python Code

The following is Python code that will collect data from a DHT11 temperature/humidity sensor. It will display data to the screen with the temperature in Celsius and Fahrenheit, with humidity. This script will also write the data to an output file "output" located in /var/www/html/DHT11/. The data written to the output file also contains a time stamp for each line written.

```
import time
import adafruit_dht
import board

dht_device = adafruit_dht.DHT11(board.D17)

while True:
    try:
        temperature_c = dht_device.temperature
        temperature_f = temperature_c * (9 / 5) + 32

        humidity = dht_device.humidity
        currentTime = time.ctime()

        print("Temp:{:.1f} C / {:.1f} F    Humidity: {}%".format(temperature_c, temperature_f, humidity))


        if humidity is not None:
            with open('/var/www/html/DHT11/output','a') as w:
                w.write('Temp={0:0.1f}*  Humidity={1:0.1f}%  '.format(temperature_c, temperature_f, humidity))
                w.write(f"{currentTime}")
                w.write('\n')
        else:
            print('Failed to get reading. Try again!')

    except RuntimeError as err:
        print(err.args[0])


    time.sleep(10.0)
```

ISSN: 2473-4901
v10 n6175

APPENDIX B
HTML code

The following code for a basic HTML page that will display the contents of an output file containing data. It assumes an output file exists in a subdirectory named "DHT11".

```
<html>
<head>
  <title>Temp in W214</title>
  <meta http-equiv="refresh" content="60">
</head>
<body>
  <h1>Contents of "output" file:</h1>
  <pre id="fileContents"></pre>
  <script>
    // Function to fetch file contents using Fetch API
    function fetchFileContents() {
      fetch('./DHT11/output')
        .then(response => {
          if (!response.ok) {
            throw new Error('Network response was not ok');
          }
          return response.text();
        })
        .then(data => {
          document.getElementById('fileContents').textContent = data;
        })
        .catch(error => {
          console.error('Error fetching file:', error);
          document.getElementById('fileContents').textContent = 'Error fetching file. Check console
for details.';
        });
    }

    // Call the function when the page loads
    window.onload = fetchFileContents;
  </script>
</body>
</html>
```