CEN4010 Principles of Software Engineering, Summer 2023

IOT Smart Thermostat

Group: 6

Repository: https://github.com/AbirFaisal/CEN4010

Trello: https://trello.com/b/2uNrNM0m/project-board

Firebase: https://console.firebase.google.com/u/0/project/cen4010-group-6/overview

Team members:

Abir Faisal	afaisal2018@fau.edu
Isaiah Cruz	icruz2020@fau.edu
Gabriel Giani	ggiani2022@fau.edu
Brandon Hyppolite	bhyppolite2020@fau.edu
Kevin Pham	kpham2019@fau.edu

Milestone 3

7/18/2023

Revision 1

TABLE OF CONTENTS

1. Executive Summary

IOT Smart Thermostat - Most smart thermostats are not really IOT devices, they connect to a server and the server mediates communication between the user and the device. This requires you to have an active internet connection to use the device wirelessly. It also requires you to create and manage an account with the service provider. More importantly, the network functionality of these devices cannot operate independently of their service providers.

Our solution to this problem is to create a smart thermostat with off-the-shelf components that can be purchased online for a reasonably low cost.

The interface will be open and not locked into any ecosystem such as Google's ecosystem for their smart Nest thermostats. It will also not require an internet connection. It will not require the user to create an account or download an app. It will be able to take multipoint temperature measurement, a feature usually found only in commercial thermostats.

Our target market sector is the home user and small businesses that are interested in independently operating smart devices that do not require any additional infrastructure other than a common wireless router.

Competitive analysis

Summary:

Our IoT Smart Thermostat offers a range of advantages that make it a compelling choice for customers seeking a versatile and user-friendly smart thermostat solution.

It provides independence from internet and cloud services, ensuring uninterrupted control and eliminating privacy concerns. With a seamless user experience, intuitive interface, and no need for complex setup or user accounts, it offers hassle-free operation.

Advanced features like multi-point temperature measurement and bed temperature sensing enhance accuracy, comfort, and energy efficiency.

Additionally, our thermostat is affordable, utilizing off-the-shelf components.

These combined advantages position our IoT Smart Thermostat as a comprehensive solution, catering to customers' preferences for independence, user-friendliness, advanced functionality, and cost-effectiveness.

Key Features of Competitors vs. Our Thermostat Application

Our Thermostat	Google Thermostat
No internet or cloud services required.	Temperature presets/preferences: Comfort is the temperature setting for when you're home, Eco is the temperature you'd like to use when you're away,
No need to make accounts or download any apps.	Create custom presets for temperatures you'd like to use. Ex. If you're working in your office or cooking in the kitchen.
Multi-point temperature measurement is usually only available in commercial thermostats.	Mode selection, switch between Heat, Cool, Heat and Cool, or to turn your system off.
You can measure your bed's temperature and lower the temperature when it senses that your bed is getting warmer.	Hold Temperature, when you want to set a specific temperature, select the temperature you'd like and choose the length of time.
	Fan Timer, you can run the fan for 15 minutes or up to 12 hours.

2. Data Definitions

Name	Meaning	Usage	Comment
Local Temperature	data	Use Case scenarios	Temperature from internal Sensor
Outside Temperature	data	Use Case scenarios	Temperature from Weather API
Remote Temperature	data	Use Case scenarios	Temperature from sensor on local network.
Bed Temperature	data	Use Case scenarios	Temperature from a sensor near bed.
RP2040	hardware	Controller	This is the controller we are using.
AHT20	hardware	Sensor	This is a temperature and humidity sensor.
SSD1309 OLED 0.91inch	hardware	Display	This is the display we are using for our device.
Mode Select Switches	Human interface	Select Operation Mode	Select between Auto, Configure, Fan, Heat, Cool, and Off.
Setpoint Buttons	Human interface	Set temperature	Up and down buttons for quick temperature adjustment.

3. Overview, Scenarios, and use cases

4a. Overview:

Our IoT Smart Thermostat provides convenient temperature control. You can make direct adjustments on the thermostat itself, but our software enhances the user experience by offering a user-friendly interface for managing temperature settings.

Using our software, you can easily control and fine-tune the thermostat's temperature settings from within your home network. This means that you need to be on the same network, typically within your home, to make changes to the thermostat.

This local app-based approach ensures a secure and reliable connection between your device and the thermostat. Within the comfort of your home, our software allows you to adjust the temperature to your preferences with ease with just a few simple taps on your device.

4b. Scenarios and use cases:

Use Case 1 – Change Temperature Remotely

For the user to continue to be comfortable without leaving the bed, they can conveniently adjust the room temperature using their mobile device.

Description:

Use case describes how a user will use the web interface to change the system's temperature.

Actors:

- User
- System

Preconditions:

- User has an active connection to their network
- System is available
- User does not want to be disrupted
- Stay in their comfort zone

Primary Flow of Events:

- Temperature is slightly unpleasant, too cold or warm
- User arrives on web page
- User changes temperature using web interface
- Terminate Use Case: Change Temperature

Alternative Flows:

Use Case 2: Change temperature Locally

The user is physically near the thermostat and does not want to open their phone to change the temperature. They can do so using the buttons on the thermostat controller.

Actors:

- User
- System

Preconditions:

- User is near thermostat
- User is not satisfied with room temperature

Alternative Flow of Events:

- Temperature is slightly unpleasant, too cold or warm
- User is near or walks to thermostat controller
- User changes temperature using buttons on the control panel
- Terminate Use Case: Change Temperature

4. List of high-level functional requirements:

User Functions:

1. **<u>Allow the user to access the thermostat using a passcode. - 2</u>**

The system should allow the user to access the thermostat from the web interface. If a passcode is set, then the system should prompt the user for it before allowing the user to access the web interface.

Stimulus/Response Sequence:

1. User visits login page
2. System prompts user with passcode if one is set
    a. If passcode is set
        i. System prompts for passcode
        ii. User enters passcode
3. System shows user interface (home page).

Functional Requirement Label: REQ 1 Access Web Interface

2. **<u>Allow the user to set the temperature. - 1</u>**

The system should allow the user to set the temperature from the web interface.

Stimulus/Response Sequence:

1. User visits home page
2. User changes temperature setpoint using “dial”? (Kevin what is it called?)
3. System adjusts temperature to new setpoint.

Functional Requirement Label: REQ 2 Set Temperature

3. Allow the user to set a schedule for the temperature. - 2

The system should allow the user to set a schedule.

Stimulus/Response Sequence:

1. User navigates to schedule page.
2. System displays an interface with days of the week

Functional Requirement Label: REQ 3 Set Schedule

4. Allow the user to select operating mode (heating/cooling) - 1

The system should allow the user to select heating mode or cooling mode.

Stimulus/Response Sequence:

1. User navigates to home page.
2. System displays an interface to with selection choice of heat or cool

Functional Requirement Label: REQ 4 Set Operating Mode

System Functions:

1. **Sense and maintain the temperature of the room. - 1**

The system will sense and maintain the temperature of the room using a feedback loop.

Stimulus/Response Sequence:

1. System measures room temperature.
2. System calculates difference between setpoint and measurement.
3. If the difference is larger than 1 degree, the system will turn on the A/C.

Functional Requirement Label: REQ 5 Measure Temperature

5.Non-functional requirements:

1.Performance: Program should perform well given the limited hardware.

2. Usability: Easy, simple to use interface that should require minimal instruction on how to use it.

3. Accessibility: Program should be accessible from most modern mobile devices and desktop computers.

4. Expected load: Low expected load as data being sent and stored will be minimal and the number of users will be low.

5. Security requirement: Users login info to the device will be computed into a hash to ensure the password is not compromised if someone has physical access to the device. The device will support connections to secured wireless networks.

6. Storage: Onboard flash memory to store programs and configuration files and in memory data structure to information such as, thermostat statistics, performance statistics, etc.

7. Availability: Must be available 24/7 because a thermostat must monitor and maintain temperatures.

8. Fault tolerance: High fault tolerance due to multiple sensors.

6. High-level system architecture and database organization

Database organization

Tables:

- USERS, stores information about the user

- SENSOR_READINGS, stores readings from the sensors in a time series format.

Login Information Database:

Stores email address and password used to login and authenticate user.

User Information Database:

Stores the first name, last name, date of birth, and email

address of users.

Media Storage: Images are stored on file system, we are only using simple icons that do not need to be backed with a database.

Search filter Architecture: (None) Not required.

APIs: None (Not required)

Non-Trivial Processes and Algorithms: System will search for sensors on network and average sensor readings from them to get a more accurate room temperature and humidity.

Software Tools:

- Microsoft VSCode (Code Editor)
- Microsoft Excel (Spreadsheet)
- Microsoft Visio (Charting and Diagrams)
- Draw.io (Charting and Diagrams)
- GitHub (Versioning System)
- Simens NX12 (Mechanical CAD)
- UltiMaker Cura (3D Printing)
- Davinci Resolve (Video Editor)
- KiCAD (Electronic CAD)
- Firebase (Cloud Service)

Languages:

- Micropython
- HTML
- CSS
- Javascript

Libraries and Frameworks:

- Bootstrap Web Framework
  - License: MIT License
  - URL https://getbootstrap.com/
- AHTx0 Temperature and Humidity Sensor Library
  - License: MIT License
  - URL: https://pypi.org/project/micropython-ahtx0/
- External Systems:
  - Wireless Router
  - Web Browser

7. Team List student group names, name of Scrum master, product owner and initial roles for each member.

Front End:

- Gabriel
- Brandon
- Isaiah

System Software:

- Kevin

Hardware and Drivers:

- Abir

GitHub

- Abir

Trello and Firebase

- Brandon

8. **Checklist - For each item below you must answer with only one of the following: DONE, ON TRACK (meaning it will be done on time, and no issues perceived) or ISSUE (you have some problems, and then define what is the problem with 1-3 lines). Reflect these items in your Trello project space:**

a) Team decided on basic means of communications - DONE

b) Team found a time slot to meet outside of the class - DONE

c) Front and back-end team leads chosen - DONE

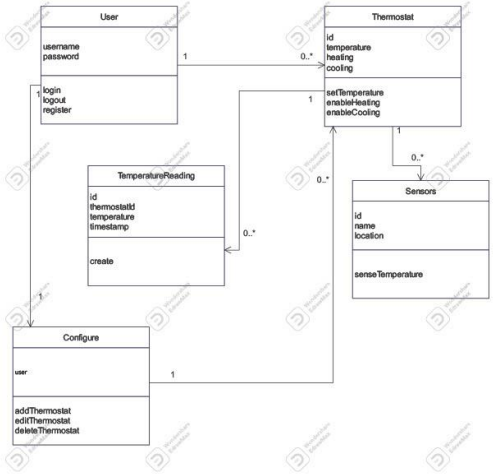
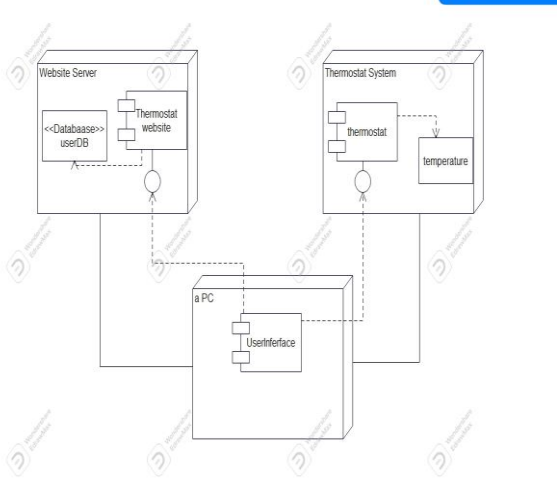
d) GitHub master chosen - DONE

e) Team ready and able to use the chosen back and front-end frameworks - DONE

f) Skills of each team member defined and known to all – DONE

g) Team lead ensured that all team members read the final M1 and agree/understand it before submission - DONE

9. High-Level UML Diagrams:

Class	Component/Deployment
	

10. Identify Risks and Actions

1. Skills Risks
    a. Our team has no experience with Firebase.
2. Schedule Risks
    a. Need more time for backend development, we only had time to develop the front end, and demo.
    b. Need more time for hardware development, hardware takes a lot of time and money to develop.
3. Technical Risks
    a. Usage of firebase requires an internet connection and making an account, both of which go against the original goals of the project which is to create a smart thermostat that does not require any external services.
4. Teamwork Risks
    a. No set schedule of daily meetings, instead a queue of tasks is created. This is risky but it has worked out so far.
    b. Most communication and tasking are done through discord, with some use of Trello as well.
5. Legal/Content Risks
    a. None, the system is self-contained, libraries are open source.
    b. Patent on smart thermostat is expired.