# Multi-room Space Heater Temperature Control

Eric Menq, Jie Sun, Rong Feng Ye

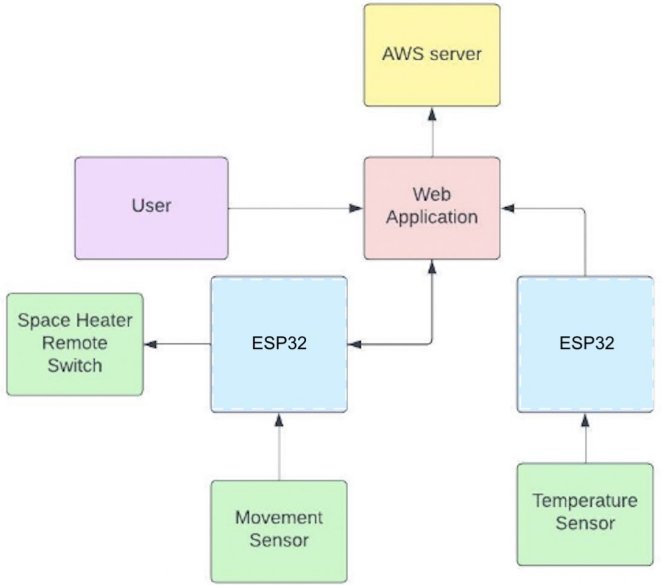
18500 ECE Design Review Presentation

# Use Case and Applications

- Our goal was to create remote controlled space heating system managed by a Web Application
- Our hope was to provide dynamic temperature control across different rooms for residents of older housing situations
- More specifically, allows for a safer, more financially efficient and flexible temperature regulation for off campus living situations
- With our web application, multiple users with different temperature preferences can use their corresponding space heaters to regulate personal temperature preferences at their space within a house

# Solution Approach

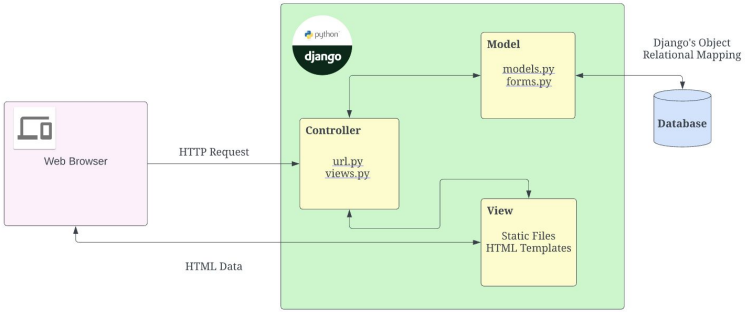
- Web App is the middle man between the AWS server and the devices
- A smart thermostat measures the temperature in the room.
- The WebApp maintains the temperature setting via a wireless smart plug
- Using infrared sensors to optimize load on breaker and save energy depending on whether someone is in the room
- Customizable temperature scheduling, setting for unoccupied rooms and breaker layouts on the Web app



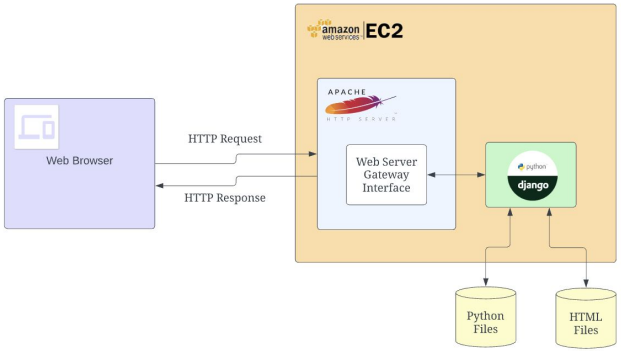

Original High Level Approach

# Solution Approach: Software System Specification

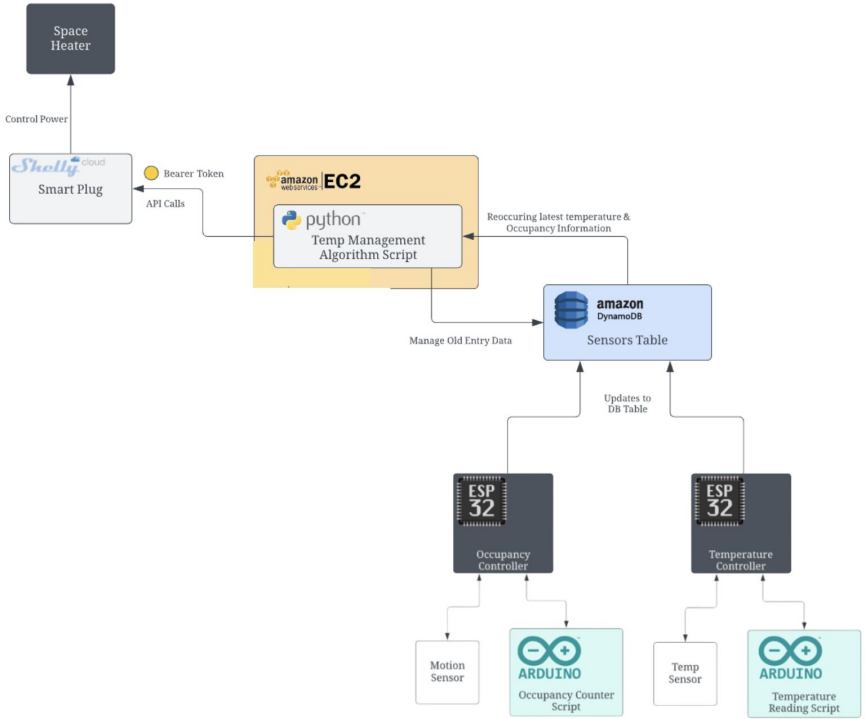
Web Application MVC-Architecture

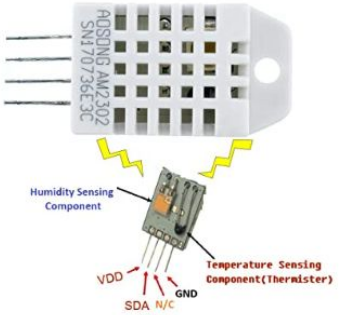


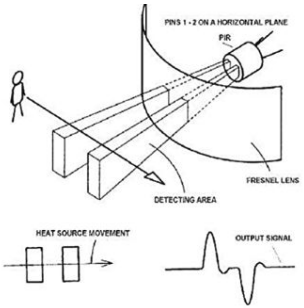

Cloud Deployment of Web Application



# Solution Approach: Hardware System Specification

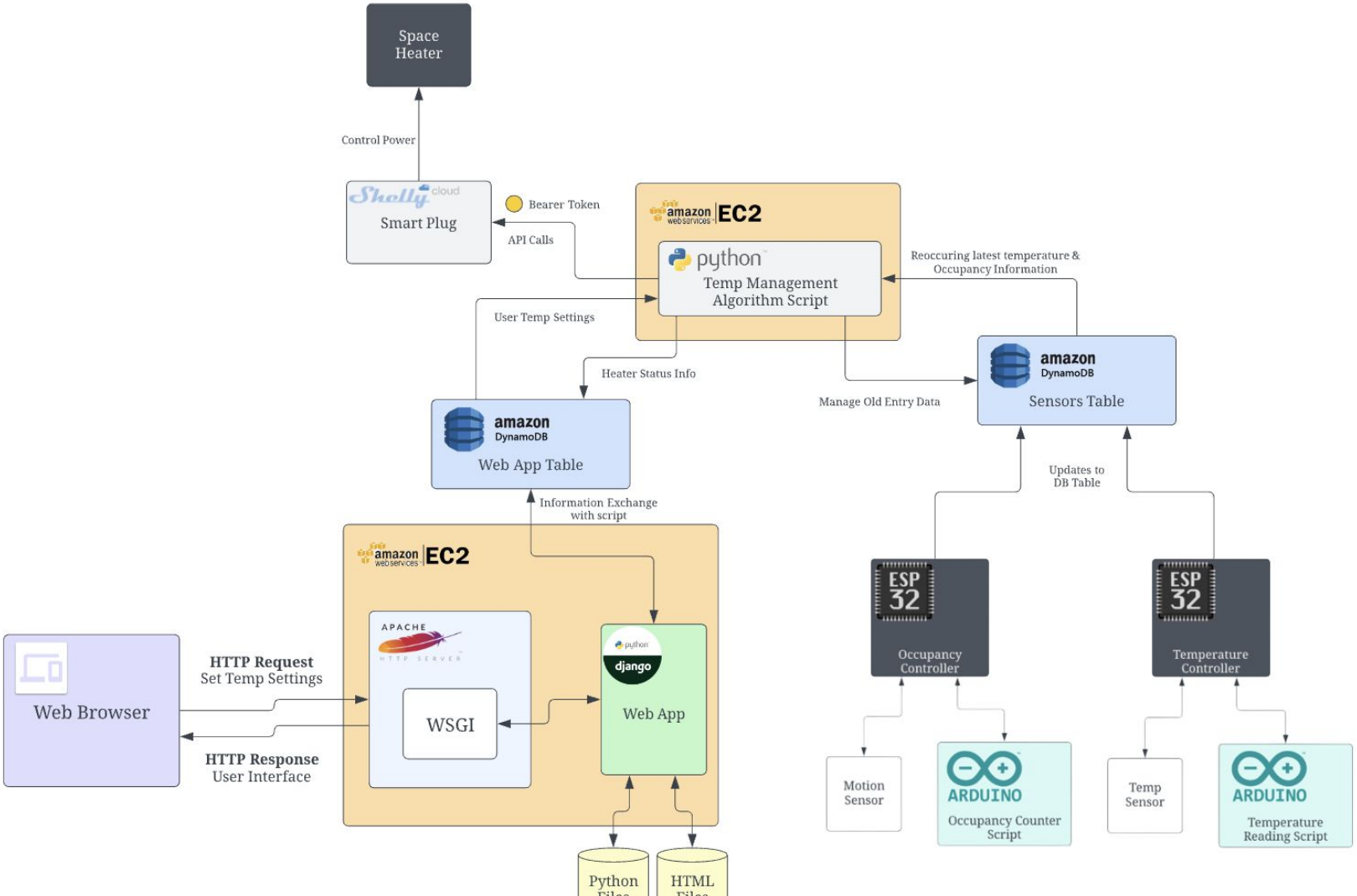


# Solution Approach

- Through class discussions and deeper reflection over the course of the semester, we have identified additional considerations regarding public health, safety, and welfare, as well as global, cultural, social, environmental, and economic impact from our project
- Ethical issues like
  - Electrical fires caused from misuse/accidents
  - Unintentional increase in utility bills if preferences tampered with
  - Occupancy data can be used in a manner that invades user privacy concerns
- We have catered our implementation choices to reduce the possibility of these issues by
  - Simplifying our WebApp UI for less technological literate users
  - Choosing secure data storage options on the cloud like EC2 and DynamoDB


Space Heater
Control Power
Shelly cloud
Smart Plug
Bearer Token
API Calls
amazon web services EC2
python
Temp Management Algorithm Script
Reoccuring latest temperature & Occupancy Information
User Temp Settings
Heater Status Info
Manage Old Entry Data
amazon DynamoDB
Sensors Table
amazon DynamoDB
Web App Table
Updates to DB Table
Information Exchange with script
amazon web services EC2
APACHE HTTP SERVER
WSGI
python
django
Web App
Web Browser
HTTP Request
Set Temp Settings
HTTP Response
User Interface
ESP 32
Occupancy Controller
ESP 32
Temperature Controller
Motion Sensor
ARDUINO
Occupancy Counter Script
Temp Sensor
ARDUINO
Temperature Reading Script
Python Files
HTML Files

# Test, Verification, and Validation: Use Case Requirements

| Feature Tested | Quantitative Methodology | Result |
|---|---|---|
| Temperature Maintenance | Space Heater must maintain a temperature within a +- 1.5F range | Success |
| User functionality | Manual and scheduling control | Success |
| Web App Usability | Satisfactory experience on survey questions with (1-5) ratings | Average 4.6 feedback |

# Test, Verification, and Validation: Design Requirements

| Feature Tested | Quantitative Methodology | Result |
|---|---|---|
| Occupancy Sensors | Sensors must be able to detect a physical presence in the personal space with >95% accuracy | ~90% reliability |
| Temperature Sensors | Sensors must be accurate within +-0.5 F, tested with thermometer gun | Success, although 10-15 second delay in sensing temp changes |
| Toggling Heater Power depending on temperature | The heater must be turned on or off when the room temperature exceeds the user's preference within reasonable time limit. | Success, the heater is turned on or off within 5 seconds of temperature sensing |

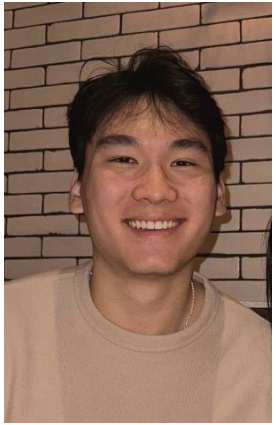
# Test, Verification, and Validation: Design Requirements

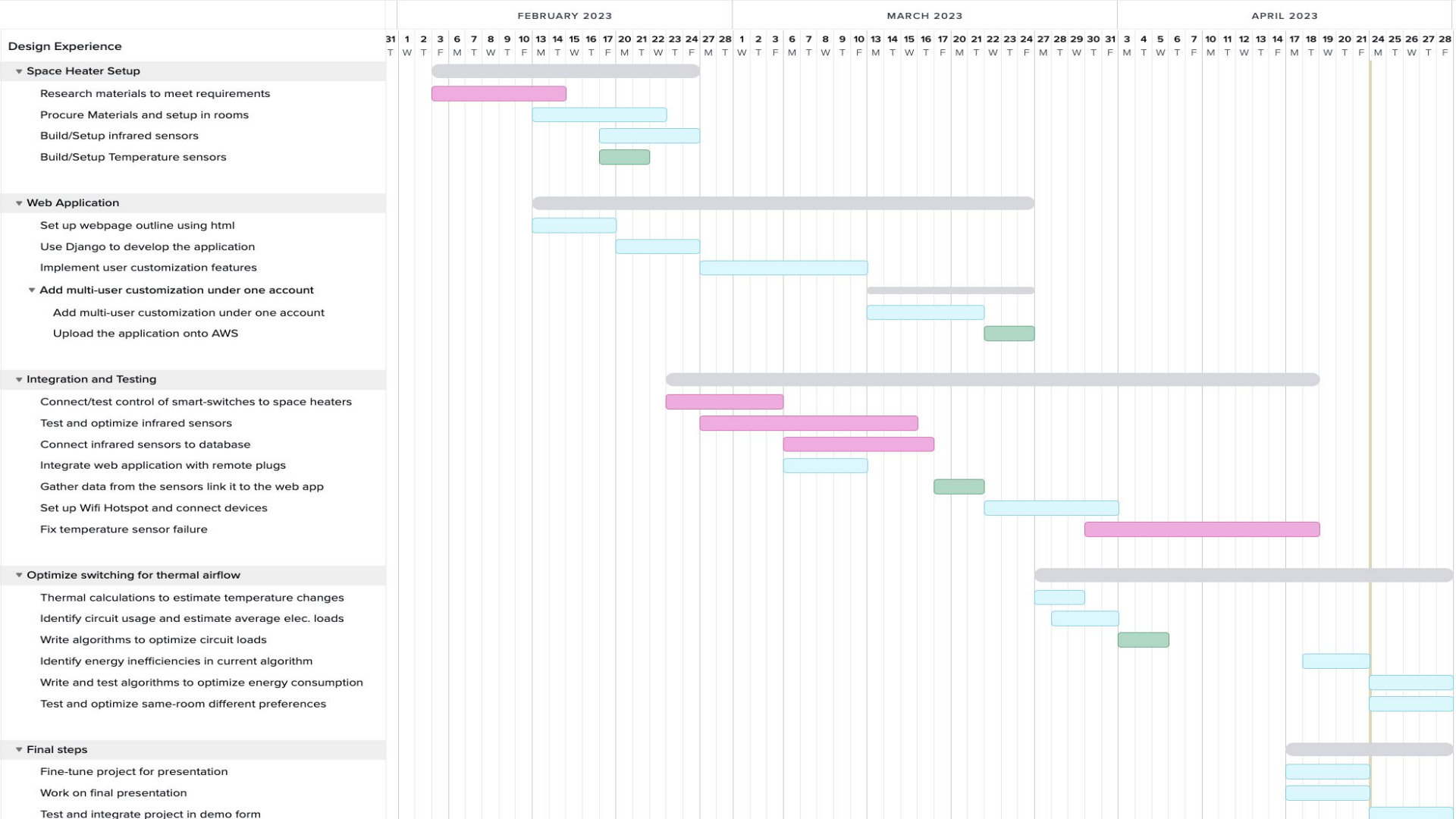
| Feature Tested | Quantitative Methodology | Result |
|---|---|---|
| Web App Capacity | Must be able to simultaneously support 50 accounts | Still Testing |
| Web App and Script integration | The web app and script database must sync within 5 seconds of new information on either end | Success, the script and web app was able to update each other. |
| Breaker Circuit Performance | When 2 heaters turned on within the same circuit, the next heater will be added onto the queue instead of turning on | Success, the script was able to add heater to queue within 2 secs. |

# Test, Verification, and Validation: Trade Offs

| Design Choice | Trade Off |
| --- | --- |
| Modularized python control script | Efficiency VS Cost |
| Extending motion sensors to side of door | Success rate VS more setup and points of failure |
| Adding 5V power module to temperature circuit instead of using 3.3V ESP output | Sensor actually working VS additional device cost and technically higher voltage than ESP input specs (potential but unlikely damage to ESP pins) |
| Using a Wifi Hotspot | Easier setup/backup for internet outage VS additional device cost |
| Adding max of two active heaters on one circuit | Preventing breakers going off in older houses VS potentially limiting heater output -> takes longer to reach desired heat |

# Project Management




FEBRUARY 2023
MARCH 2023
APRIL 2023
Design Experience
Space Heater Setup
Research materials to meet requirements
Procure Materials and setup in rooms
Build/Setup infrared sensors
Build/Setup Temperature sensors
Web Application
Set up webpage outline using html
Use Django to develop the application
Implement user customization features
Add multi-user customization under one account
Add multi-user customization under one account
Upload the application onto AWS
Integration and Testing
Connect/test control of smart-switches to space heaters
Test and optimize infrared sensors
Connect infrared sensors to database
Integrate web application with remote plugs
Gather data from the sensors link it to the web app
Set up Wifi Hotspot and connect devices
Fix temperature sensor failure
Optimize switching for thermal airflow
Thermal calculations to estimate temperature changes
Identify circuit usage and estimate average elec. loads
Write algorithms to optimize circuit loads
Identify energy inefficiencies in current algorithm
Write and test algorithms to optimize energy consumption
Test and optimize same-room different preferences
Final steps
Fine-tune project for presentation
Work on final presentation
Test and integrate project in demo form

# Lessons Learned

- Perform more research on the resources and equipments before purchasing them
- Thoroughly understand all devices we are working with instead of trial and error
- Utilize other’s experiences that are shared online
- Expect errors to occur and have a mitigation plans