Project: Ameliorate

A Disaster Before We Touched It

[1]

Team:
SDDEC19-18

Jared Griffin
Nidhi Dalvi
Robert Guetzlaff
Siyuan Zeng
Tyler Borchert

Client:
Andrew Guillemette

Advisor:
Daji Qiao

Problem:

Elderly individuals fall into regimented routines. Changes in these routines could indicate that something in the life of the elderly person has changed. This project aims to track the routines of the elderly and detect when there is a change in their behavior.

Our client has been trying to get this project off the ground for two years, and we are the third and final group to work on it. Our task was to improve the base system created by the first group by replacing the wired sensors used previously with wireless sensors, integrate new sensors developed by the second group, add logic to process the sensor output, and develop a web application to display our health predictions.

Solution:

We have replaced the wired sensors with wireless sensors, integrated the second group's work, built a logic system that can detect if our test user has prepared a meal, and built a web application that can display whether or not a meal was eaten by the senior .

System Components

Wireless Sensor System:

- TI CC2650STK Sensortag
  - Uses an accelerometer and gyroscope to sense openings and closings of drawers and cabinets respectively
- Legacy Power Outlet System
  - Component Developed by previous group that was integral to our logic.
  - Used a backend server we could not gain access to.
  - Rewrote backend section to run on Raspberry Pi and forward data to logic server

Tools & Languages

- Python
- NodeJS
- PHP
- Gattool, pexpect, sqlite3

Intended Users

Intended to be used by healthcare professionals and loved ones to monitor the health of their senior citizens.

Behavioral Logic Server:

- Model
  - The server allows to decode the data from database to make the data readable by user.
- Repository
  - The server provides a repository for each type of data to make data well managed and easy to search and use.
- Service
  - The server provides the logic algorithms to analyze and calculate data so that the events can be determined.
- Controller
  - The server provides the path for web app to access so the web app can send requests to and get response from the logic server.

Frameworks & Languages:

- Java
- Spring Boot
- JVC

Web Application:

- For health professionals and loved ones to use when the want to verify the senior resident is eating their meals regularly
- Information about the senior resident's eating habits is displayed in a dashboard, including time each meal was began and how long it took to eat

Frameworks & Languages:

- JavaScript
- React JS
- Jest

Requirements

Functional

- Sensor data matches expected data
- Events are displayed through web app
- Sensors are wireless with 1-2 year life

Non-Functional

- Reuse previous groups work
- System is non-invasive
- System survives loss of power

Constraints

- Must be inexpensive
- Limitations of hardware
- Inherited components

Operating Envirnment

- Kitchen of an Elderly
- AWS
- Caretakers Mobile

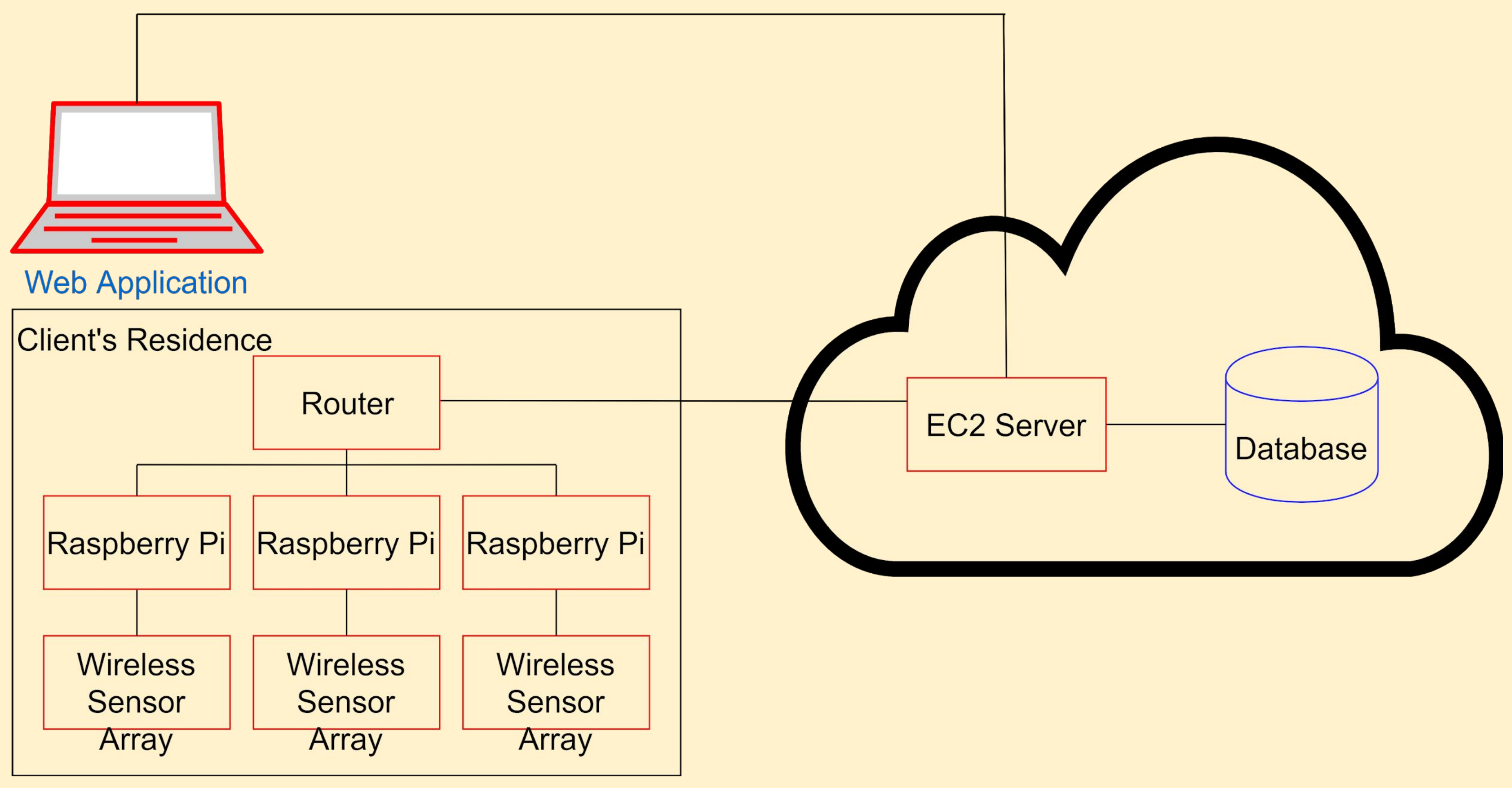
Standards

- APIs
- HTTP Requests

Project Resources

Gitlab
Sensor Tags
Wireless outlets
AWS
Raspberry Pi

Concept Diagram



Component Testing

Wireless Sensor System:
For the drawer and cabinet system, we used physical tests to simulate this part of the system.This involved connecting all the sensors to a script and opening and closing cabinets and drawers.
For outlets, we used both simulated data and real world power usage trials. For simulated data, we sent a post requests to the replaced backend with data that matched what we expected to see. We the left them for a plugged into devices we used for a week and monitored when they recorded a power usage event.

Behavior Logic Server:
We are using Junit & Postman for the unit testing. The Junit test was used to ensure each class of the Sping Project works correctly. And the Postman was used to mock the call requirements from web application so that we can do the integrating test of web app and logic server when the web app is during implement.

Web Application:
We used a testing framework called Jest to for unit testing. The code for the web application was built using Test Driven Development, and there is 100% coverage by our unit tests.This test suite is also ran in a continuous integration pipeline.

[1] https://www.kissclipart.com/guillotine-icon-clipart-computer-icons-guillotine-gldbwn/