Instituto Superior Técnico

Information Systems and Computer Engineering

Ambient Intelligence

# Home Control

Mafalda Ferreira
81613

Bárbara Ramalho
83523

Group 2

May 17, 2019

# Contents

# 1 Introduction

## 1.1 Context

In recent years there has been a social paradigm change that welcomes the use of technology into our lives in a way that 20 years ago would be hard to predict. This means that the existence of apps for controlling the most trivial things around us is not a scary novelty anymore, but is expected and anticipated even if it will take a few more years for more advanced technologies to be available to the average consumer.

One area that has called for new creative solutions that are still under development are SmartHomes. This refers to a system where house appliances and devices are interconnected into a common network in a way that they can interact with each other directly and/or remotely.

We want to contribute to the technological advancement in this area by creating an app that allows to control the electric devices of your house from one interface allowing for greater flexibility in the management of a home.

## 1.2 Motivation

Have you ever been in a situation where you leave the house and don't remember if you forgot to turn off the light or the oven? Or have you ever been home and wished you could turn on the heat without having to interrupt the movie you are watching? With HomeControl these things will be possible. Our app serves as a remote control panel for your house and you can change the settings anytime to fit your needs. We are considering a future of smart homes where appliances are connected to each other. Your lights and oven will turn off when you are not in the house or not using them. However sometimes smart appliances will not behave as you like in a certain situation and it is the role of the house owner to control the house and make sure everything is working as expected.

Our app will tell you which lights are on, and with a simple click of a button you can turn them off. You can also control the temperature or other settings of any appliances you connect to our app. Even in a smart home environment, you have to be the boss.

HomeControl is also a game changer for people with disabilities as it gives them the freedom to be independent in the control of a house or division, as long as they are capable of using a smartphone.

## 1.3 Objectives

The main objective is to offer an interactive interface that allows our users to remotely manage their house's appliances and/or devices settings.

Our idea of the final product would be an app that could be adaptable to mobile, tablets and pcs. This app will have an interface that allows you to choose which room of the house you want to control. In each room there will be a list of settings you can control, for example, in the bedroom you will have the option to switch on/off the lights, and control the temperature of the room. You will also have a control panel to manage settings for the whole house, such as turning off all the lights. This app is to be used remotely so you don't have to be inside the house to control it.

## 1.4 Main contributions

Our product will improve people's lives by giving them a feeling of control over their place. In a Smart Home environment it would be easy for people to feel like they have little control oven the environment. What if you wanted to take a nap during daytime, but the lights don't turn off? Or if you leave the house without starting the dishwasher and now you won't have clean dishes for diner? HomeControl gives you the possibility to force your environment to act as you like. Turn off your smart lights even when the motion sensors detect your presence and remotely turn on your dishwasher while at work.

It is also an important advancement for people with disabilities or reduced mobility as it can allow them the independence to take care of their house from one spot.

# 2 Related Work

## 2.1 Technologies Used

For the execution of this project, we used both hardware and software tools.

### 2.1.1 Hardware

In order to simulate the sensors installed in the house, we chose to use an Arduino [1]. We selected this technology because it is a rapid electronic prototyping platform with the capability of extend both software and hardware. The Arduino boards are able to read inputs - pushing a button and receive events from a server, in our case - and turn it into an output - turning on an LED, communicate with a server. We opted for the Arduino instead of the Raspberry Pi because we only needed to run one program at a time in a loop and didn't need such a powerful machine as the Raspberry Pi that would only add complexity.

In order to fulfill all the requirements of this project, an Arduino board alone wouldn't be enough. So, we added an Ethernet Shield - to communicate with the internet, a battery - so the Arduino wouldn't need to be connected to a computer, a button and some leds - to simulate the functionality. We will show the implementation of the circuit in figure 2.

### 2.1.2 Software

Since our project has a big component involving interoperability between different technologies, software tools were extremely essential.
For the WebApp, we used a know trio - HTML, CSS and JS. For the Arduino, we opted for the Arduino IDE [2], since it is an official Arduino software used mainly for writing and compiling the code into the Arduino Board.
For the Web Server, we used several tools. It was build using Node.js [6] and we opted to use this platform because this server will only do lightweight computation with two way connections. We used express.js [4], a web application framework and Socket.IO [9][3], a JavaScript library that enables real-time bidirectional event-based communication.
The Control Panel was built using React.js [8] since it was a interactive small component.

# 3 Description of the solution

## 3.1 Architecture and Global View

Our solution is composed by four main components that interact as we see in the following Figure 1. The User only interacts with the WebApp and the Arduino board and the server and the control panel are running independently .

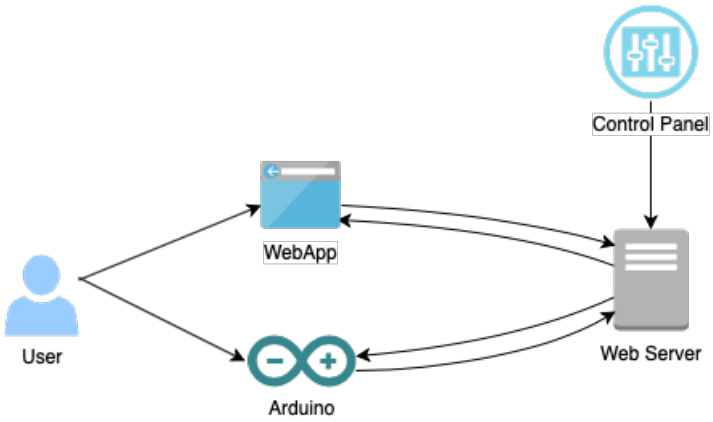


Figure 1: Solution Schema

The WebApp is the web interface that allows the user to interact with the house system and the Arduino represents an Arduino Board used to replicate the functionalities of an oven.

## 3.2 Main Components

As stated previously, the system is composed by four main components and we will describe them in this section.

### 3.2.1 Arduino

The main purpose of the Arduino is to simulate an oven functionality. As so, the Arduino board has available two inputs fot the user: an on/off switch and a knob.
The on/off switch is used to turn on and off the oven energy supply, just as a regular oven and the knob is used to simulate the oven current temperature.
The decision of using a knob to simulate the temperature instead of an actual temperature sensor was made because with an actual temperature sensor it would only read the room temperature and wouldn't be possible to demonstrate all the functionalities.
The Arduino interacts with the user since it's the user who controls the Arduino Board inputs and also interacts with the Server by receiving any state changes to the oven done through the WebApp and sending changes made to the Arduino Board input, such as turning the oven on and off.
This component is described in a more detailed way in section 4.1.1

### 3.2.2 Server

The Server is the central component of our solution, since it is responsible for the processing of all the requests from the other components. The server is a stateful component: it does not only receive requests and send events but also saves the state of the house. The state of the house comprises the current state of all the actuators, such as lights, temperature devices and general appliances.
This component is described in a more detailed way in section 4.1.2

### 3.2.3 WebApp

The WebApp makes the connection between the system and the users. It allows users to read and control the state of the system through an interactive screen.
This component is described in a more detailed way in section 4.2.1

### 3.2.4 Control Panel

The purpose of the Control Panel is to simulate other sensors that could be installed in the house. This panel allow us to simulate interactions with the sensors, such as a person entering a room or some timed events. This component essentially represents the smart part of the solution. For example, with this component we can simulate a person leaving the living room and see that the server processes this information and decides to turn off the lights, since no one is there. This component allows the system to be more responsive according to the state of the house and the interactions people made with it.
This component is described in a more detailed way in section 4.2.2

# 4 Details of the solution

## 4.1 Description of Main Components

The solution comprises four main components, described in a more general way in section 3.2. In this section, we will explain them using a much detailed approach.

### 4.1.1 Arduino

As stated before, the main purpose of the Arduino is to simulate an oven functionality. For the implementation of that functionality, we need a robust system with access to two different inputs: the on/off switch and the temperature knob.

The following materials were used:

- 1x Arduino Uno Rev3 ATmega328P board
- 1x Ethernet Shield W5100
- 1x Breadboard
- 3x LEDs (1x red, 1x green, 1x yellow)
- 3x Resistors 220 Ω
- 1x Resistor 10K Ω
- 1x Pushbutton switch
- 1x Buzzer
- 1x Ethernet Cable
- 1x Battery
- 1x Linear Potenciometer with a 10kΩ resistor - *Alps RK09D1130C2P*.
- 1x Precision button for the Potenciometer with an indicator trace.

**On/Off Switch** This input was implemented with the pushbutton switch. When the button is pressed, a state change occurs. This means that if the oven is off and we press the button, the oven is turned on and vice-versa.

**Temperature Knob** The temperature knob was implemented with a potenciometer. In order to regulate the oven simulated interior temperature, the user only needs to rotate the button, as in a regular oven. This means if the button is rotated to the right, the temperature rises and vice-versa.

The circuit was implemented as showed in figure 2.

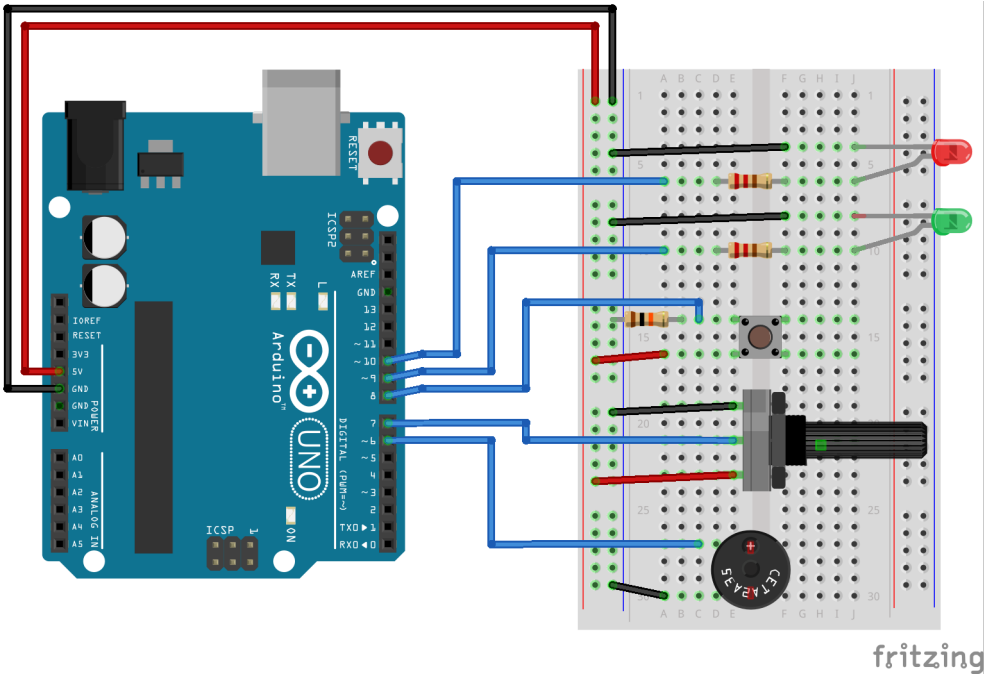


Figure 2: Circuit implemented in the Arduino Board

### Arduino ↔ Server

The arduino interacts with the server via a websocket interface. This allows the Arduino to send requests to the Server and to receive events from it. This communication has two types of interactions:

- The state of the Arduino changes (on/off switch pressed or temperature regulated) and the Arduino updates the server state with the new changes. The server acknowledges the new changes and modifies the state (and file) accordingly. For example, in the WebApp we have the oven information displayed, if the oven state changes, we need to inform the server of this change, so the server can update the information on the WebApp.
- Server sends new state to the Arduino. This type of interaction happens if a state change is made by another component such as the WebApp. For example, if a user decides to turn the oven off via WebApp, we need to inform the oven that a new state was requested and it needs to act accordingly and coherently.

### User ↔ Arduino

The user interacts with the Arduino by changing the inputs (on/off switch and temperature regulator) as explained in the beginning of this section.

### 4.1.2 Server

As stated before, the server is the central component of our solution. It is responsible for receiving requests, sending events and saving the house state. It only has 2 functions: change the house state and return the state to whoever requests it.

It was implemented using Node.js, express.js and socket.IO. We opted to use Heroku [5] to run our server in order to scale our solution. This way, our server is running in a well-known url - `http://homecontrolserver.herokuapp.com` - which allows the communication with the other modules to be done remotely, approximating our solution as much as possible to a production one.

#### Server ↔ WebApp

The Server and the WebApp comunicate over a websocket. This type of channel not only allows the WebApp to send requests to the Server but also to receive events from it, so we have two types of interactions:

- User requests some change in the WebApp - p. ex: *Turn off the tall lamp in the master bedroom*. The user requests a change in the WebApp via a button, switch or other input and that triggers the sending of a request to the server with the new changes. The server acknowledges the new changes and modifies the state (and file) accordingly.
- Server sends new state to the WebApp. This type of interaction happens if a state change is made by other component, such as the Control Panel or the Arduino and the server sends the new change to the WebApp in order to keep it updated and coherent with the server state.

The state of the house is stored in the server in a .json file with the hierarchy of the objects as we can see in the example 1.

## 4.2 User interface

### 4.2.1 WebApp

The WebApp was developed using HTML, CSS and Javascript. It has an initial general screen with setting related to the whole house. You can see who is in the house at the moment, receive notification from the appliances or select each of the rooms individually. When one room is selected another screen opens with each of the intelligent appliances of the house. Here you can control them through a series of toggle buttons and sliders.

The WebApp was designed to be User-Oriented, taking into account usability heuristics. We payed special attention to the "recognition rather than recall" and "minimalist design" parameters so the app would blend well with others apps that are used nowadays.

```
{
  "masterbedroom": {
    "lights": {
      "tall_lamp": "1",
      "nightstand_lamp": "0",
      "ceiling_lamp": "1",
      "desk_lamp": "0"
    },
    "temperature": {
      "heating_device": "22"
    },
    "appliances": {
      "tv": "0",
      "music": "100"
    }
  }
}
```

Listing 1: House state .json file example

Our focus group will be any person who lives in a home with an intelligent environment and wishes to interact with it through an intuitive interface medium. This includes both males and females from all ages. For the final product, HomeControl would include a special interface for children as early as 1 year olds. However the current prototype focuses only on accommodating adults from 18 years old forward.

To complete our product we also went as far as to make a brand for ourselves. With HomeControl we wanted to build a system that could go beyond this project. Here is the logo we drew that serves as a package to our brand:


Figure 3: HomeControl's logo

**WebApp ↔ User**

The user interacts directly with the WebApp. For this prototype we are using the mouse as the main control but in future improvements the app could be used via touchscreen.

- The user decides on what he wants to change in the house environment. He proceeds to click on the appropriate buttons that make that happen.
- The app should informs the user of the current state of the house through appropriate feedback given by button colors and position. The app also has a notification panel to serve this purpose. There should also be a match between the system and the real world that reassures the user everything went as planned.

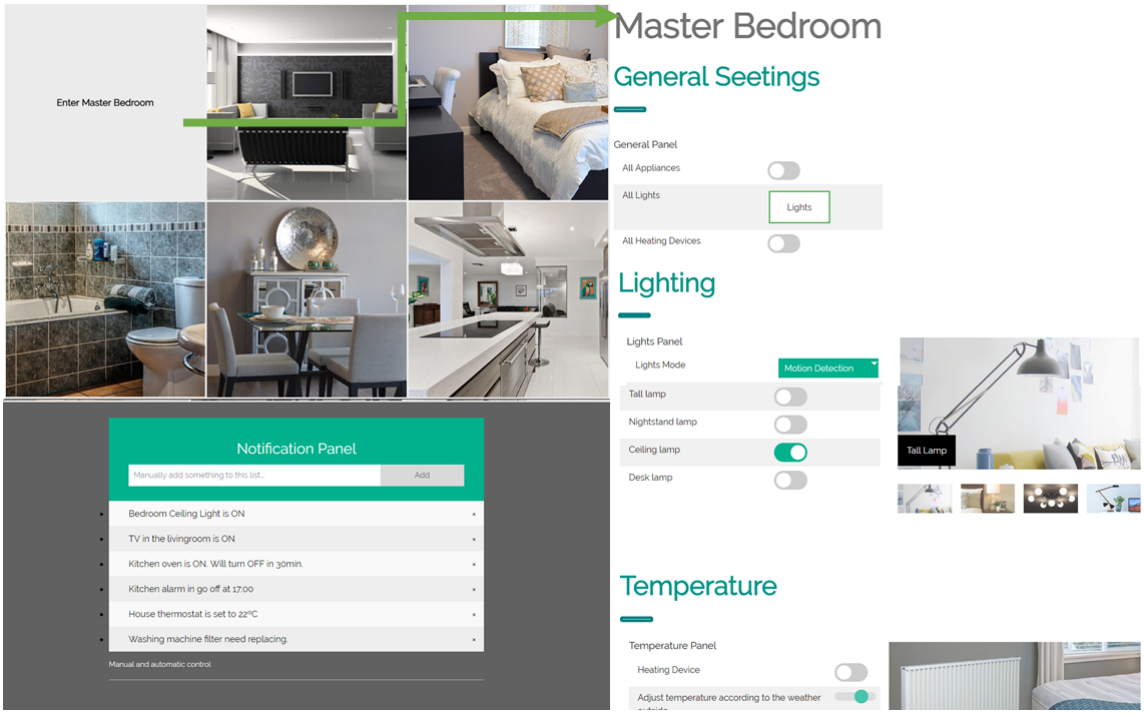


Figure 4: Storyboard path from the Main menu's screen to the Master bedroom's screen

### 4.2.2 Control Panel

The control panel is composed by switches - to simulate on/off interactions, such as if there are people in a room or being night time (timed event), as we can observe in Figure 5.

Its interface it's composed by a unique screen separated by divisions, where several sensors are available and can be manipulated. For example, if a user turns on the switch *People In The Room*, it simulates that some house sensors recognize that there are people in the correspondent room; if a user turns on the switch *Night Time*, it simulates that some outside and inside house sensors recognize that it is night.

This application was created using React.js and essentially it just receives inputs from a user and sends the appropriate request to the server.

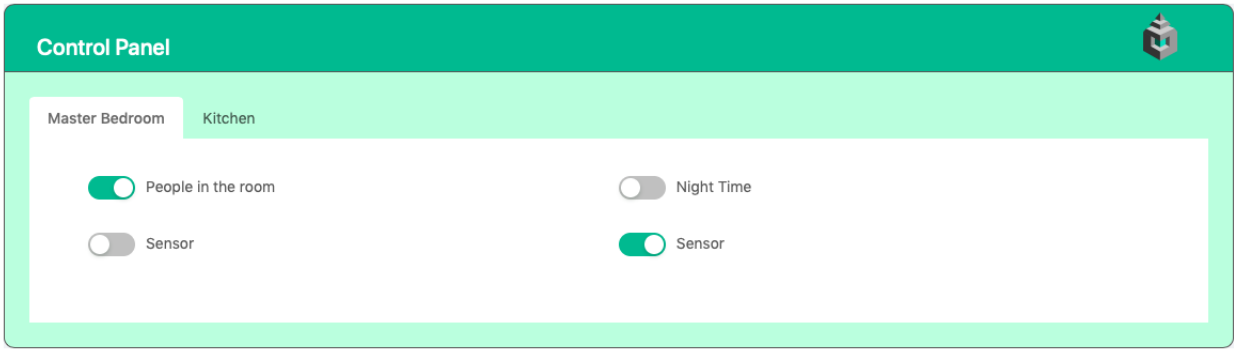


Figure 5: Control Panel View

### Control Panel ↔ Server

This interaction is pretty staightforward since it's unidirectional. The Control Panel only sends requests with the new sensor information (simulated by the switches) to the server.

In the figure 6 it's possible to observe what type of flux the system has when the switch *People in the Room* changes its state.

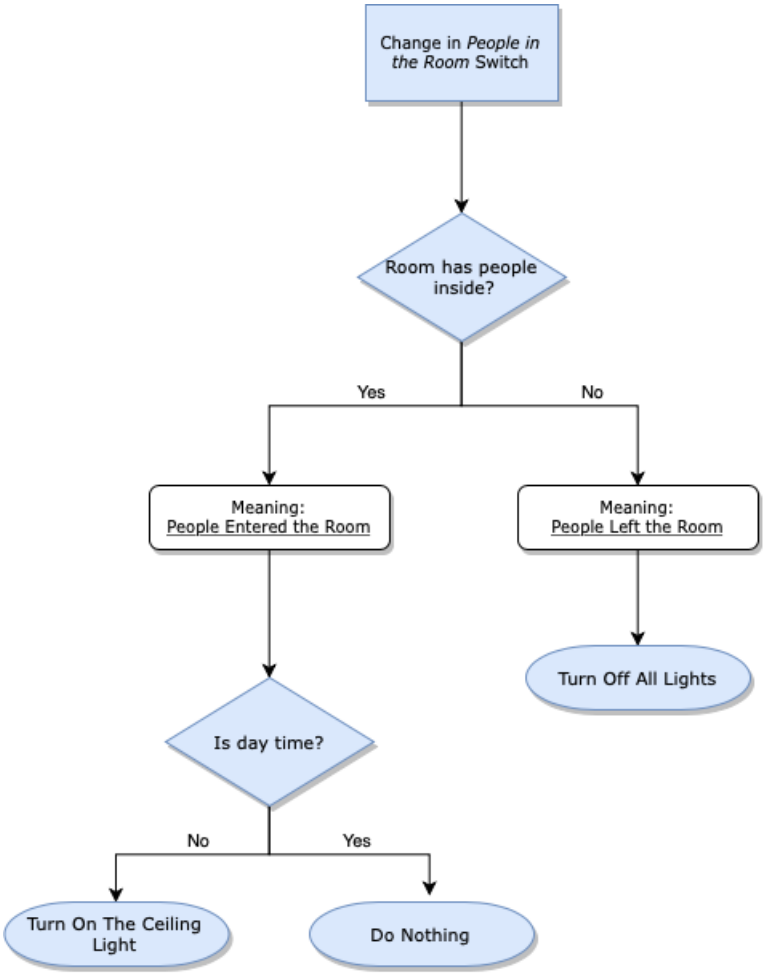


Figure 6: Example of a Flux of Interaction

## 4.3 Examples of Use

**Scenario 1** In his smart home, Daniel's favourite artist starts playing as he arrives to his room. However, it's way too loud and he only wanted it playing in the background. Daniel adjust the volume by lowering the slider on his rooms entertainment interface in the HomeControl app. The volume lowers accordingly. He continues his routine.

**Scenario 2** Josh arrives to work in a rush. Just as he walks through the door he remembers he forgot to run the washing machine. Josh really needed that done asap, or his wife will throw a fit when he get home. Josh groan at himself as he selects the HomeControl icon on his phone. In the kitchen interface he presses the toggle button corresponding to the washing machine. Back home the machine starts to work.He receives a notification reassuring his of that. He considers that task done.

# 5 Evaluation

## 5.1 Evaluation description

Since users will have direct contact with HomeControl's system it is critical to get feedback from our focus groups during the development process. User Research helps us understand how users interact with the system and correct possible flaws that would disturb the usability flow. So far we have been through one iteration of User Research using four people from our focus group, three females and one male all belonging to the 20-24 age gap.

## 5.2 Functionality testing

All the functionalities were extensively tested recurring to observed behaviour of the components and using Postman [7], which allowed us to test all the possible interactions between components.

## 5.3 Testing methods

For the User Research we used the Think Aloud method and guided observation. We asked the users to perform specific tasks such as switching off all bedroom lights and lowering the temperature of the heater in the kitchen. While they performed these tasks we asked them to voice their thoughts so we could better follow their behaviour. We collected information based on the time they took to complete each task and the number of mistakes made. After that we also performed a non-structured interview with open questions to get feedback on what we could do to improve the app.

## 5.4 Results

The users took an average of 36 seconds to perform each task and an average of 1 mistake. This is more time than we expected for this test, however there is a learning curve to overcome and be believe the times would improve with more tests. The number of mistakes corresponds to what we believed the tests would show. From the feedback we got the toggle button for the general panel that switches off all the lights should be replaced by a single press button.The images below show that improvement already implemented. Another feedback we took into account was removing a check list feature from the notification panel as it was confusing and not really a needed feature.

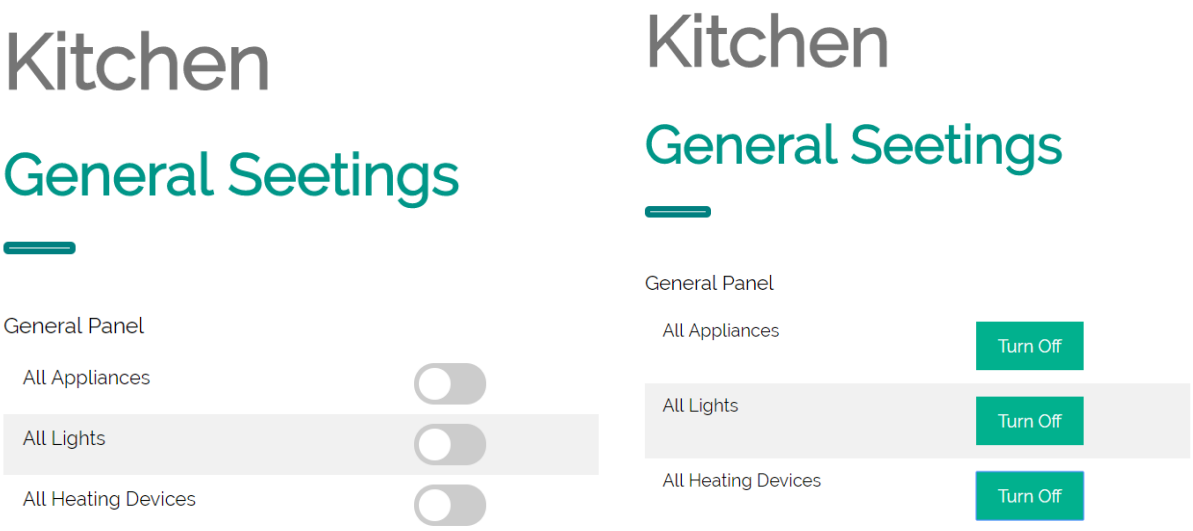


Figure 7: Toggle button improvement in the kitchen interface

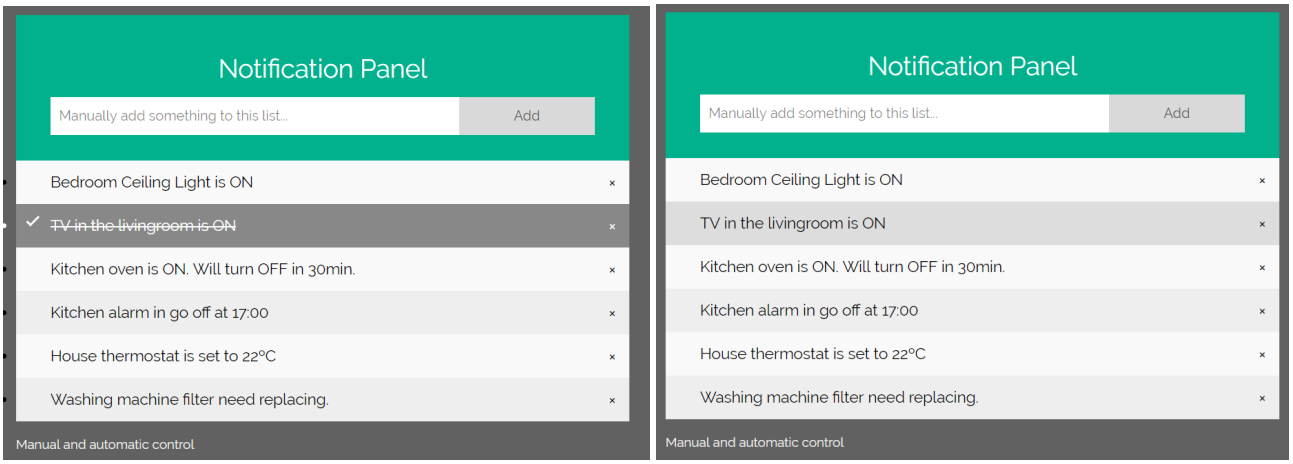


Figure 8: Improvement in the design of the Notification Panel

# 6 Conclusion

## 6.1 Global Balance

With our current prototype we are able to connect the webApp, arduino, webserver, and control panel in a system that enables the users to control the state of their intelligent house. This means we achieved our initial proposal of creating a prototype for an interactive house control system.
So far we have only built a low-functionality prototype. The goal for the fully-functional prototype of HomeControl would be to connect the app to real house appliances and to have users directly control them through the interface, instead of simulated with the arduino.
For this prototype we only developed the simulation mechanisms for the master bedroom and kitchen areas. The house simulated by the interface has six rooms (two bedrooms, a bathroom, a living-room, dining-room and kitchen), but because of the time we had we decided to only work on two of them. However the system could easily be replicated to work in any room.

## 6.2 Future Improvements

In the future we could change the mouse control for a touch screen.
Another good improvement would be to have separate log-in accounts for children and for adults in the house.This implies implementing log-in screens as well.
Our app is also very versatile so it would be possible to implement more functionaries such as a house security system.

# References


[1] *Arduino*. URL: https://www.arduino.cc/en/Guide/HomePage.

[2] *Arduino IDE*. URL: https://www.arduino.cc/en/Main/Software#downloads.

[3] *Arduino Socket.IO*. URL: https://github.com/Links2004/arduinoWebSockets.

[4] *Express.js*. URL: https://expressjs.com.

[5] *Heroku*. URL: https://www.heroku.com.

[6] *Node.JS*. URL: https://nodejs.org/en/.

[7] *Postman*. URL: https://www.getpostman.com.

[8] *React.JS*. URL: https://reactjs.org.

[9] *Socket.IO*. URL: https://socket.io.