MINISTRY OF EDUCATION

TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA, ROMANIA

FACULTY OF AUTOMATION AND COMPUTER SCIENCE
AUTOMATION AND APPLIED INFORMATICS

E.C.L.A.I.R.
(Entity Controller Limited At Infra-Red)

LICENSE THESIS

Graduate: Răzvan-Gheorghiță PĂDURARU
Supervisor: SI. dr. eng. Dan GOȚA

2022

MINISTRY OF EDUCATION

TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA, ROMANIA

FACULTY OF AUTOMATION AND COMPUTER SCIENCE
AUTOMATION AND APPLIED INFORMATICS

DEAN,
Prof. dr. eng. Liviu MICLEA

HEAD OF DEPARTMENT,
Prof. dr. eng. Honoriu VĂLEAN

Graduate: **Răzvan-Gheorghiță PĂDURARU**

E.C.L.A.I.R.
(Entity Controller Limited At Infra-Red)

1. **Project proposal:** *Design and implementation of a product which can control infrared devices using a mobile app*
2. **Project contents:** *The thesis will be structured as:*
   - *Introduction (Objectives, Motivation, Specification)*
   - *Theoretical Basis (History of Infrared, Applications of Infrared)*
   - *Analysis, design and implementation*
   - *Conclusions*
3. **Place of documentation:** *: Technical University of Cluj-Napoca, Automation and Applied Informatics Department*
4. **Consultants:** *SI. dr. eng. Dan GOȚA*
5. **Date of issue of the proposal:** October 1, 2021
6. **Date of delivery:** July 1, 2022

Graduate: PRăzvan

Supervisor: 

MINISTRY OF EDUCATION

TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA, ROMANIA

FACULTY OF AUTOMATION AND COMPUTER SCIENCE
AUTOMATION AND APPLIED INFORMATICS

Declaraţie pe proprie răspundere privind autenticitatea lucrării de licenţă

Subsemnatul(a) **Păduraru Răzvan Gheorghiță** legitimat(ă) cu **CI** seria **XB** nr. **668149** CNP **1990620046196**, autorul lucrării **Entity Controler Limited At Infra Red** elaborată în vederea susţinerii examenului de finalizare a studiilor de licenţă la Facultatea de Automatică şi Calculatoare, Specializarea **Automatică și Informatică Aplicată** din cadrul Universităţii Tehnice din Cluj-Napoca, sesiunea **Iulie 2022** a anului universitar **2021-2022**, declar pe proprie răspundere, că această lucrare este rezultatul propriei activităţi intelectuale, pe baza cercetărilor mele şi pe baza informaţiilor obţinute din surse care au fost citate, în textul lucrării şi în bibliografie.

Declar, că această lucrare nu conţine porţiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislaţiei române şi a convenţiilor internaţionale privind drepturile de autor.

Declar, de asemenea, că această lucrare nu a mai fost prezentată în faţa unei alte comisii de examen de licenţă.

În cazul constatării ulterioare a unor declaraţii false, voi suporta sancţiunile administrative, respectiv, *anularea examenului de licenţă*.

Data
30/06/2022

Nume, Prenume
Păduraru Răzvan Gheorghiță

Semnătura


MINISTRY OF EDUCATION

TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA, ROMANIA

FACULTY OF AUTOMATION AND COMPUTER SCIENCE
AUTOMATION AND APPLIED INFORMATICS

SYNTHESIS

Of the Diploma Thesis:

E.C.L.A.I.R.
(Entity Controller Limited At Infra-Red)

1. Requirements:

   The ability to save your remote infrared commands into a database and use a web application to replicate them afterward.

2. Solutions:

   The chosen solution for my project is the raspberry pi thee b+ and, as for the software application, a web application that makes the interaction with the raspberry pi user-friendly. Furthermore, an infrared receiver, an ir LED, a PN2222A transistor, and a 1kΩ resistor were chosen to be able to read light signals of the remote and emit them afterward.

3. Results:

   A web application and dedicated Android/IOS mobile applications which are cable of communicating with the raspberry pi through a web server. All the remote codes are stored in a database that can be accessed only by the user who created them.

4. Testing and Validation:

   Several tests were performed during the implementation of the project in order to ensure that codes are read, stored correctly, and replicated correctly.

5. Personal Contribution:

   Develop a fully functional web application; design a secure way for user registration and for creating entities in a database for each user's remotes and remote commands; find way to make the process of registering/replicating remotes commands very easily; keeping the data in a secure database;

6. Bibliography:

   The information was acquired from specialized articles, various research in the field of infrared light, web documents, and web pages.

Semnătura autorului: PRăzvan

Semnătura conducătorului științific: 

Contents

Chapter 1

Introduction

1.1 Project context

The infrared remote control technology is now used in many sectors of our daily lives and production. It is employed not just in high-tech applications such as aerospace, but as well as in everyday remote controls. [1]However, each product requires its own remote, and the user will be overburdened with a plethora of remotes to manage its gadgets. That's one of the factors why electronic waste is on the rise. Only in 2022, more than over 9 million tons of e-waste were generated, and it is anticipated to reach 120 million tons by 2050 if consumers do not modify their behaviors. [2].

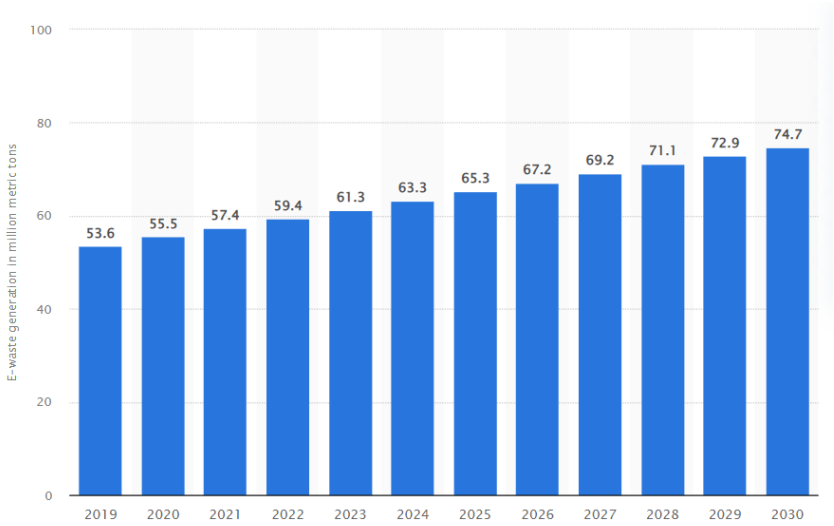

Figure 1.1: Projected electronic waste

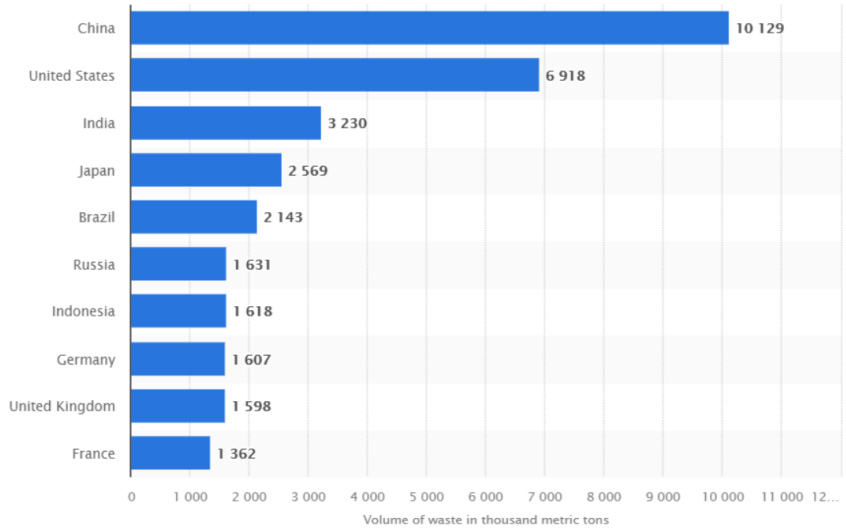

Figure 1.2: Electronic waste by country

As you can see in the figures 1.1 and 1.2, the numbers don't look so good. That's why I came with the idea of E.C.L.A.I.R. A device which can easily replace all your remotes that work with infrared light.

Commonly used infrared remote controls generally have two parts: emission and receiving. The part that emits infrared rays is mainly an infrared light-emitting diode, and the infrared-emitting diode at its core is a special light-emitting diode

Most consumer and consumer devices equipment, from video recorders to stereo systems, incorporate infrared remote controllers. Remote controls are almost usually supplied. Infrared remote controls are often used nowadays to operate video and audio equipment, pcs, and even lighting systems. The frequency of the carrier signal in such ir communications is generally about 36kHz. In serial format, the control code gets modulated to this 36 kHz carrier frequency. Multiple encoding techniques are in use, with different manufacturers using a wide range of encoding and data-rate algorithms.

In modern systems, typical frequencies for IR light modulation are 36-40k Hz. The 40 kHz frequency band is very susceptible to interference. One technique for decreasing interference is to utilize higher IR carrier frequencies. Several infrared systems currently employ transmitters in the megahertz region. [3].

1.2 Motivation

The main motivation of my thesis is to create an environment in which you can control your home appliances over the internet. Technology evolves at a high rate but many of the old home appliances weren't designed to be controlled over the internet (TVs, DVDs, air conditioners, entry barriers, garage gates, etc.). Rather than purchasing new

home appliances that would have wireless access but for a higher price, we could use a device that would be able to control them all.

1.3 Objectives

The primary goal of this thesis is to describe the tools and procedures used to create an infrared entity controller that might replace your home remotes.

The application should be able to let each user create remotes and for each remote to register commands. Several goals have been set to achieve the desired design and functionality:

- Researching the topic of infrared control to learn more and get a better concept of what my ultimate product should perform.
- Researching in the field of mobile and web applications in order to create a good and easy user experience
- Gathering the hardware specifications in order to build the project
- Learning the necessary technologies in order to create the software application
- Developing an algorithm to encode and decode infrared codes
- Creating a secure application that would let people register and login in order to access their remotes
- Include all of these features in a client-server application.

1.4 Specifications

The project's ultimate goal is to assist the user in accessing his gadgets through a third-party device and application that can operate numerous devices, code and decode all infrared remote control protocols, and connect with the transmitter and receiver. The application is divided into 3 parts:

- **Admin's part:** The admin should be able to have control over who is registered into the application, delete inactive users, send reset password links, and manually change the password to a temporary one when needed.
- **Remote's part:** The user should be able to see his registered remotes and create new ones.
- **Button's part:** For a selected remote the user should be able to see the registered buttons, send the specific commands to those buttons, deleting and creating new button commands.

Getting into the technical side of things, I used react native with typescript for the mobile and web application, .Net for the main server which communicates with the client and the SQL Server database due to its versatility and easy integration with Microsoft services, python for reading commands and for building a flask server to communicate with the .Net server in order to receive infrared commands from the database and C to decode those signals and sending them to the emitter. For the hardware, I used a raspberry pi 3 b+ as the main control board, KY-022 Infrared IR Sensor Receiver Module, an infrared receiver, an ir LED, a PN2222A transistor, and a 1kΩ resistor for emitting the light signals.

Chapter 2

Bibliographic research

2.1 The basics

We need to know the fundamentals of light in order to understand infrared. The power of a photon is proportional to its wavelength. In physics, a wave's wavelength is measured from the zero-crossing to the distance between two consecutive crests, or any two locations of the same phase on the wave (see Fig. 2.1). Both moving waves and stationary ones have a property known as wavelength. Energy is concentrated at shorter wavelengths. When comparing the energies of the many colors of light, violet has the most[4].

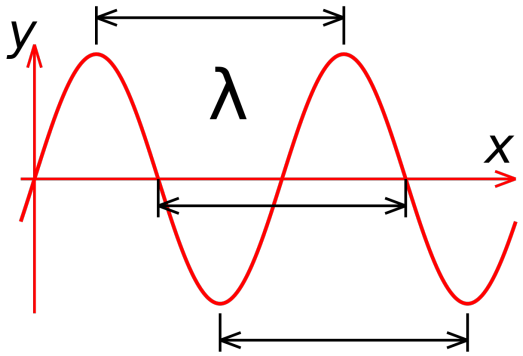

Figure 2.1: The wavelength of a sine wave

The wavelength λ can be defines as:

$$\lambda = \frac{v}{f} \tag{2.1}$$

The frequency of a wave is denoted by the symbol f, while the speed of a wave, v, is defined as the angle between the wave's leading and trail.

Infrared light is a form of electromagnetic radiation that sits between the visible portion of the electromagnetic spectrum and the microwave portion. Its wavelength extends between 800 μm and 1 mm, and it is mostly released by hot things[5].

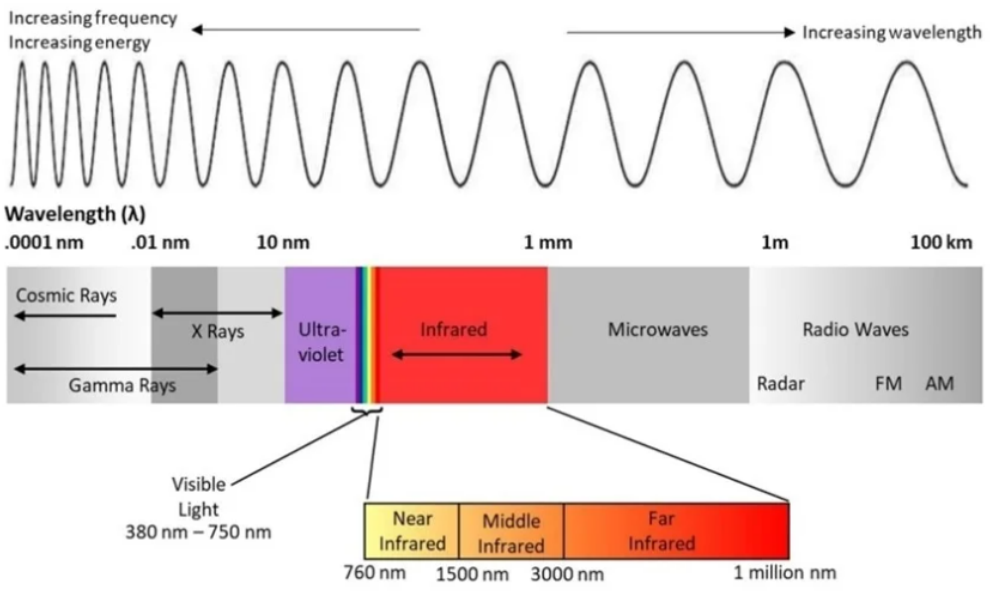

Figure 2.2: The electromagnetic spectrum

As can be seen in Figure 2.2, there are three distinct categories of infrared radiation:

- Near-infrared: Near-IR light, with wavelengths between 700 μm and 1500 microns μm, is the most similar to visible light.
- Mid-infrared: Medium-IR light has wavelengths between 1500 and 3000 μm. The near-IR and mid-IR spectrums are widely used in a wide variety of electrical applications, including remote controls.
- Thermal-infrared: The bulk of the infrared spectrum is comprised of the thermal-IR range of wavelengths, which extends from 3000 μm to 1 mm.

The primary distinction between thermal-IR and the other two is that thermal-IR is emitted by an item rather than reflected from it. A material emits infrared light as a result of what is going on at the atomic level[6].

2.2 A short history of how infrared was discovered

William Herschel discovered infrared radiation in the 19$^{\text{th}}$ century. In the year 1800, Herschel presented his results to the Royal Society of London. Herschel utilized a

prism to refract the sun's beams and observed infrared photons beyond the red portion of the spectrum as a result of the thermometer's temperature increase. Astounded by the outcomes, he dubbed them "Calorific Rays"[7]. In the late 19^{th} century, French scientist Edmond Becquerel invented the word infra-rouge to describe infrared radiation (infra-red)[8].

Figure 2.3 illustrates the research stand used by W. Herschel. The illustration was taken straight from his original work "Experiments on the refrangibility of the invisible rays of the sun"[7]. He determined that specific levels of the sun's light spectrum correlated to temperature impacts. He placed on a slidable surface, radiation detectors via mercury thermometers, along the spectrum, which was created by fracturing solar radiation via a glass prism. The energy of the incoming radiation was absorbed by the darkened containers, causing the thermometers placed in the spectrum to indicate a higher temperature than the ambient temperature. Figure 2.3 shows the results of his measurements of the temperature distribution throughout the spectrum of solar light (E) acquired from the experiment.

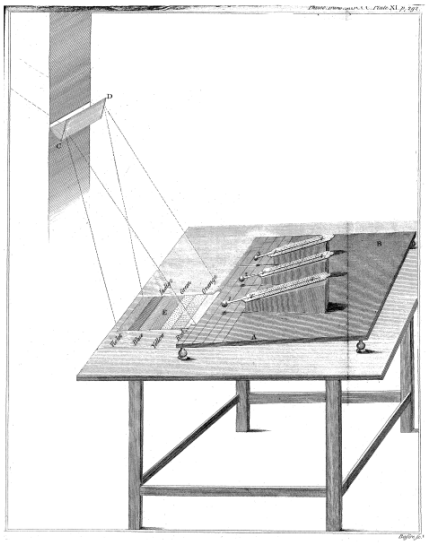

Figure 2.3: Stand for measurement

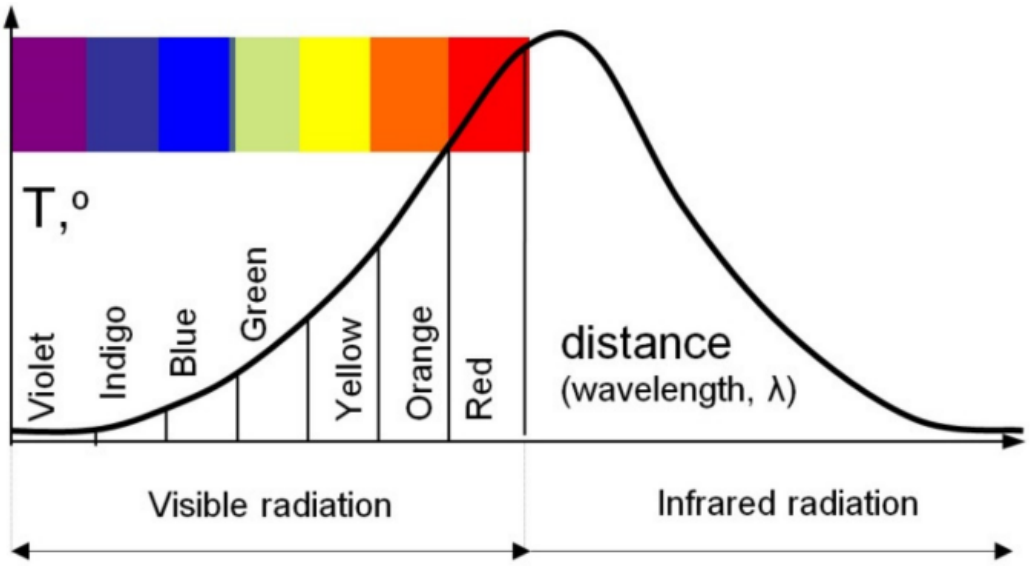

Figure 2.4: Temperature dispersion diagram

In the three decades that followed Herschel's discovery, infrared light's adherence to the rules of optics was the only significant advancement. Due to a lack of precise and sensitive detectors, infrared research was significantly hampered. However, in the 2nd decade of the 19th century, Thomas Johann Seebeck (1770-1828) initiated research on the connection behavior of conductive materials. In 1821 he discovered that if the junctions of two different metal conductors were kept at different temperatures, a small current would flow through their closed circuits [9] [10].

In 1829, L. Nobili invented the first thermal junction, now known as a thermocouple, and the improved electrical thermometer based on Seebeck's discovery of the thermoelectric effect. Four years later, M. Melloni suggested connecting several bismuth copper thermocouples in series, resulting in a higher and hence detectable output voltage. This was at least 40 times more accurate than the best thermometer on the market and could detect human heat from 30 feet away [11]. The output voltage of a thermopile like this grows linearly as the number of thermocouples connected increases. In figure 2.5(a) an example of thermopile prototype invented by Nobili is shown. Figure 2.5(b) shows an incomplete version of the Nobil-iMelloni thermopile, which was originally fitted with brass coneshaped tubes to capture radiant heat. This device was far more sensitive than prior thermometers, and it became the most extensively used IR radiation detector for the next half-century [12].

By joining two thin strands of platinum foil, Samuel Pierpont Langley (1834–1906) produced a Wheatstone bridge in 1880. Using this setup, he was able to investigate solar irradiance well into the infrared region as well as assess the intensity of solar radiation at different wavelengths [13]. Since then technology and infrared has come a long way and ifrared has a lot of applications in 21st century.

(a) (b)

Figure 2.5: The Nobili-Meloni thermopiles

2.3 Infrared in modern society

2.3.1 Remote Communication

IR data transmission is often used for short-distance communication between remotes and home devices. Typically, these devices comply to the Infrared Data Association standards. In remote controls and IrDA devices, infrared LEDs generate infrared radiation that may be focussed by a lens into a laser that the user aims at the detector. The light is modulated (switched on and off) based on a code that is decoded by the receiver. For practical reasons, near-IR wavelengths (below 800 μm) are often used. This wavelength is readily identified by inexpensive silicon photodiodes, which the receiver uses to convert the observed radiation into an electric current. This electrical current is then transmitted via a high-pass filter, which retains the fast pulsations of the IR transmitter while filtering out the gradually varying infrared radiation from ambient light. It is desirable to employ infrared communications inside in heavily populated areas. As IR cannot pass through walls, it does not interact with devices in adjacent rooms. Infrared is the most frequent way used by remote controls to operate home appliances. For infrared connections, standard control protocols such as NEV, RC-5, SANYO are used. In figure 2.6 you can see an infrared remote using the RC-5 protocol. The human eye cannot detect the light but a photo camera can detect it, hence the purple light.

Figure 2.6: Remote cotrol infrared

In 1980, the Canadian company Viewstar invented the first TV remote using infrared technology. This method, which allowed for broader and more complicated instructions than prior control systems, rapidly became the standard, particularly as cable providers extended their channel selections well beyond a TV dial. [14].

2.3.2 Night vision

Infrared is employed by night vision equipment because it can be detected even when regular light levels are low. Light from the environment is converted into electrons by a night vision system, which is then chemically and electrically amplified to produce more light. There is no requirement for a visible light source when infrared light sources are employed to supplement the ambient light for conversion by night vision equipment. [6]. There are two modes of operation for night vision:

- Image enhancement: This works by gathering light from the lower infrared light spectrum, which is undetectable to our eyes, and magnifying it such that we can see the picture. Fig 2.7 (a)
- Thermal Imaging: This works by collecting light from the higher region of the infrared light spectrum, which is released as heat rather than reflected as light. Humans and

animals radiate more heat than colder things such as trees or buildings and therefore are simpler to detect with thermal imaging. 2.7 (b)

Figure 2.7: Image enhancement vs thermal imaging

2.3.3 Astronomy

Astronomy is the study of space and the cosmos as a whole from a scientific perspective. A significant contribution of modern infrared technologies to astronomy has been made. Mirrors, lenses, and solid-state digital detectors are among the optical components used by astronomers to examine objects in the infrared range of the electromagnetic spectrum. The study of the universe using a telescope is known as optical astronomy. As a result, the detectors in an infrared telescope need to be cooled using liquid helium and the optical components need to be protected from any potential heat sources.

Many objects in the cosmos are too chilly and dim to be seen, yet they may be viewed in infrared light. Scientists are starting to solve the secrets of planets, cold stars, nebulae, as well as many other things in the cosmos by examining the infrared radiation released by cooler objects across the universe. Because infrared waves have longer wavelengths than visible light, they may travel through clumps of interstellar gas and dust with less absorption and scattering. As a result, infrared radiation has the potential to disclose things in the cosmos that optical telescopes cannot view in visible light. The James Webb Space Telescope (JWST) is outfitted with three infrared sensors to aid in the study of the universe's nature and the formation of galaxies, stars, and planets [15].

In the paper "The beginning of modern infrared astronomy"[16], the authors provide a brief history of the field. To make use of the new infrared detectors, astronomer Gerard Kuiper constructed a spectrometer in 1947 to get the IR spectra of the brilliant planets.

As a result, Jupiter's ammonium and methane absorptions were found to be rather large, and he also found considerable absorptions for Venus and Mars' CO2. Using data from the Kuiper discoveries, Peter Fellgett was able to measure 51 stars using an infrared photometer equipped with a lead sulfide detector in 1951. Despite the lack of follow-up to Fellgett's effort, Kuiper established the IR-friendly Lunar and Planetary Laboratory (LPL) at the University of Arizona to continue studying celestial objects using infrared. Nonetheless, only a small number of astronomers felt compelled to explore the vast spectral chasm that separated visual from radio observation [16].

2.3.4 Medicine

The ability to see temperatures in two dimensions in real time is a major advantage of thermal imaging. In a single snapshot taken with today's technology, hundreds of temperature readings may be taken in a split second.

In order to maintain life, the human body is homeothermic, meaning it produces and controls its internal temperature. We, humans, try to keep our comfort by layering up during the colder months and cooling down in the warmer months. The core of a person's body maintains a relatively stable temperature, although the skin and other superficial tissues play a role in maintaining a comfortable external temperature. Human skin is practically a black body because of its low transmittance of 0.96 to 0.98. In 1934, American physiologist J D Hardy showed that the skin's emission peaked between 9 and 12 μm.

In contrast, bolometer devices operating up to 15 m have been found to be just as effective in clinical uses as detectors acting at 2-5 μm. For the last half-century, thermal imaging has been primarily used in academic settings. Many diseases have been studied using this method because skin temperature changes might indicate inflammation in human tissue or an abnormality in blood flow. Medically, thermal imaging may be utilized for both diagnosis and as an outcome measure in research [17].

To evaluate the physiological processes involved in controlling body temperature at the skin level, Medical Infrared Thermography (MIT) is utilized. Because of its effectiveness, safety, and low cost, MIT is a useful supplementary instrument for detecting and locating thermal abnormalities, such as increases or decreases in skin surface temperature. Useful in veterinary medicine for monitoring the health of racehorses and diagnosing damage to their locomotor systems. The vast majority of medical diagnostic imaging techniques make use of electromagnetic spectrum components (figure 2.8). However, unlike other medical equipment, MIT employs non-ionizing radiation, making it safe for patients to use without restrictions. The quantity of heat radiated from a surface is used by infrared cameras to create thermal pictures. Human skin emits infrared light, the majority of which is in the μm wavelength range. [18].

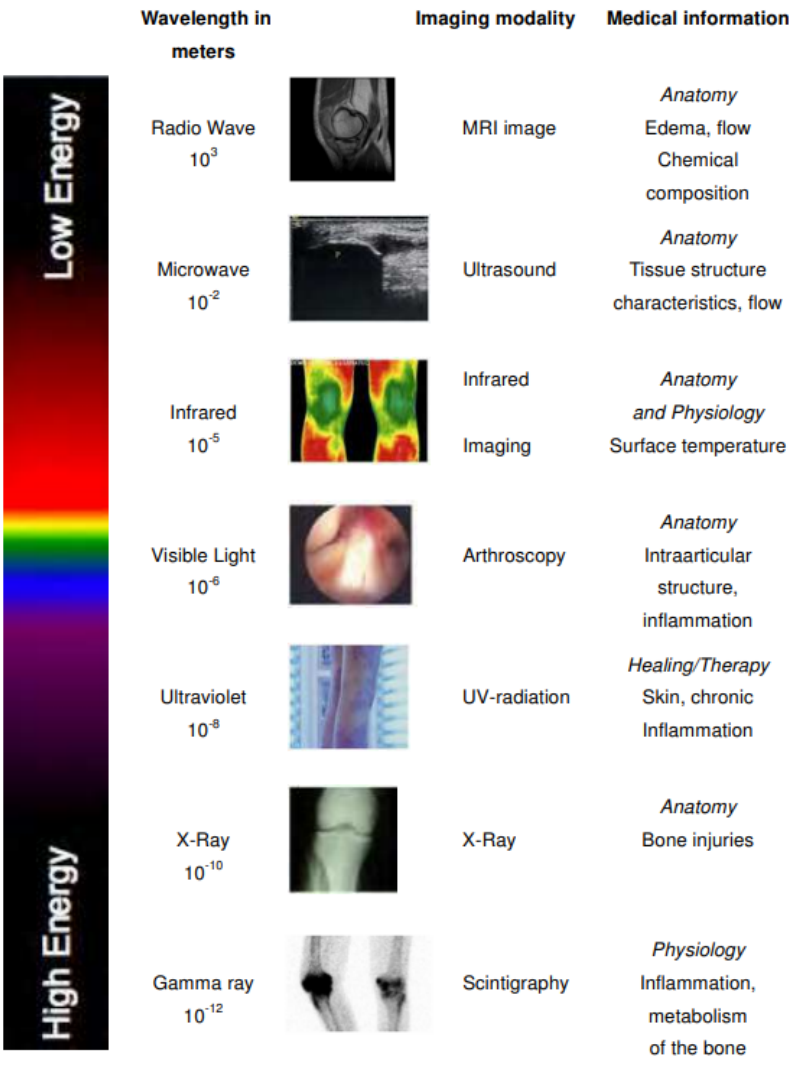

Figure 2.8: Electromagnetic spectrum-based medical imaging techniques [18]

MIT is employed in a range of medical specialties, including neurology, cancer, orthopedics, and dermatology. The approach is now widely used in breast cancer research. Tumors have enhanced angiogenesis and hence greater metabolic activity, resulting in larger temperature gradients as compared to surrounding tissue. Furthermore, MIT is widely acknowledged in the field of surgery. It is feasible to monitor the resumption of blood flow via the coronary blood arteries after aortic-coronary bypass surgery. An infrared camera may be used in plastic surgery to assess the reperfusion of perforator flaps [18].

Chapter 3

Software and Hardware specifications

All of the specific hardware and software components will be broken down and detailed in this section.

3.1 Web applications

3.1.1 What is a web application

In most cases, a web app will have two main components: a client and a server. The client is the item that initiates communication with the server, such as a computer's browser or a smartphone app. The server is the component that watches for requests, handles them, and returns the results. HTTP is one of the most widely used protocols for transmitting requests between a client and a server over the internet (Hypertext Transfer Protocol). Properly formed HTTP requests include a Uniform Resource Locator (URL), an HTTP method (GET, POST, PUT, DELETE...), a set of HTTP headers (also known as header fields), and a message body, if necessary. A well-formed HTTP response will also include the status line (200 OK for HTTP/1.1), a set of HTTP headers (also known as header fields), and a message body. Fig 3.1 shows a client-server interaction.

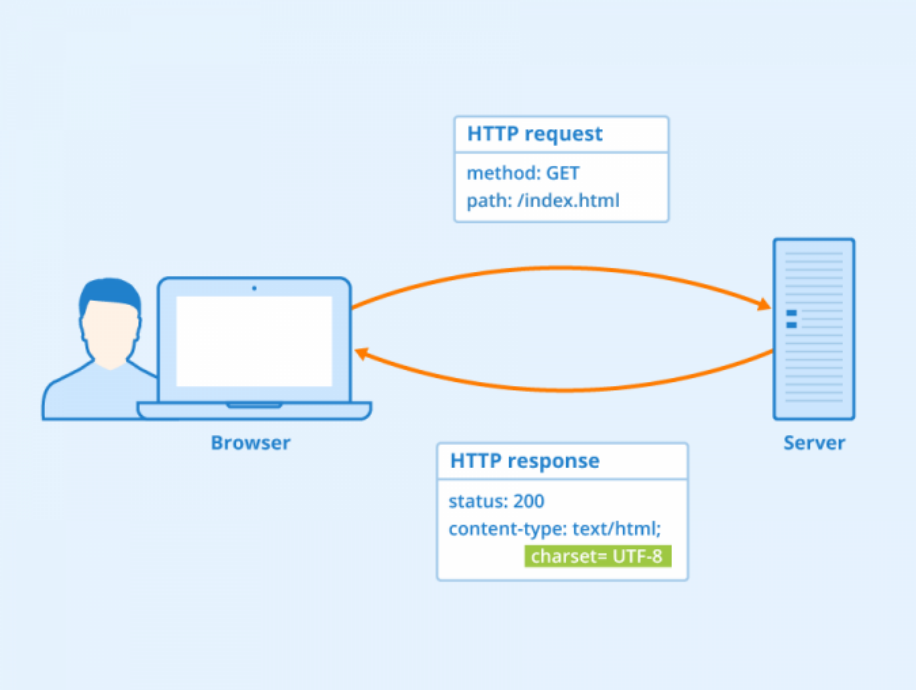

Figure 3.1: Client-Server interaction [19]

3.1.2 What is REST Service

Representational State Transfer (REST) is an architectural approach used in the area of distributed hypermedia systems. Here are some of the core ideas behind what makes up a RESTful architecture: [20]:

- Client-server: The client-server design relies on the separation of concerns concept. The portability of the user interface is enhanced across platforms when it is decoupled from the data storage. It also has the advantage of allowing different components to be developed independently of one another, increasing the scalability and portability for different platforms.

- Statelessness: Statelessness implies that the communication between client and server always includes all of the information required to carry out the request. There is no state for the session on the server, it is fully maintained by the client. If access to a resource needs authentication, the client must do so with each request.

- Uniform interface: To connect with one another, all components of a RESTful API must adhere to the same set of rules. This also makes it easy to understand relationships between system components.

- Layered System: This implies that a client connecting to an intermediary component,

such as a proxy, has no idea what is going on behind the scenes. As a result, components may be replaced or extended independently of one another.

- Caching: To boost speed, the client, server, and any intermediary components may cache all resources. The data is classed as cacheable or non-cacheable.

- Code-on-demand: To increase client capabilities, additional code may be downloaded. This is optional, since the client may be unable to download or run this code.

3.2 Client technologies

3.2.1 JavaScript

JavaScript is a scripting or programming language that enables you to create complicated features on web sites, such as presenting timely content updates, interactive maps, animated 2D/3D visuals, scrolling video jukeboxes, and so on. It is the third layer of the layer cake of standard web technologies (see Figure 3.2), the other two are HTML and CSS [21].

Websites with complex features, such as dynamic content updates, animated 2D/3D graphics, interactive maps, and scrolling video jukeboxes, may be developed using the scripting or programming language JavaScript. When compared to HTML and CSS, which make up the first two layers of the web's "layer cake," it represents the third and final layer (Figure 3.2).

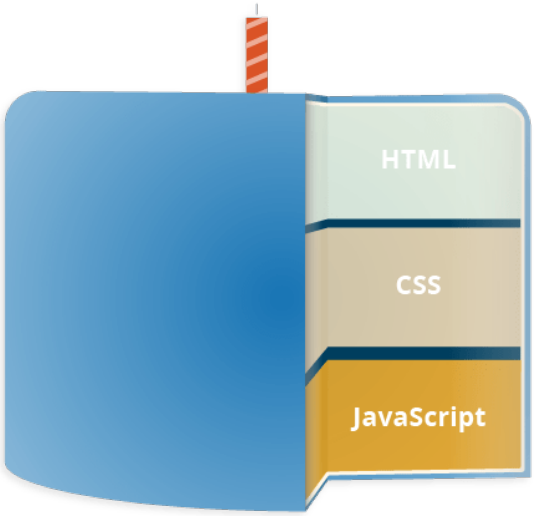

Figure 3.2: Standard web technologies [21]

3.2.2 TypeScript

Comparing TypeScript to JavaScript, the latter is superior. Typescript is a superset of Javascript, and Microsoft created and continues to support this programming language as open-source software. It all starts with writing the TypeScript code. A TypeScript compiler takes the TypeScript code and compiles it into standard JavaScript. After you have the JavaScript code, you may use the core functionality on any platform that allows for the execution of JavaScript. A TypeScript program is one that has no syntax mistakes and is written in JavaScript. Therefore, all JavaScript programs are TypeScript programs. This is really useful when transitioning an existing JavaScript codebase to TypeScript. TypeScript is equally fine in client- and server-side contexts.

3.2.3 React Native

React Native is a framework for building natively rendered iOS and Android mobile applications. It is based on ReactJs, Facebook's JavaScript framework for developing user interfaces. However, it is designed for mobile devices rather than browsers. It is a great tool, especially for web developers, because they can now construct native-looking and feeling mobile apps using a JavaScript framework that they are familiar with. In addition, since much of the code you build can be shared between platforms, React Native makes it easy to develop for Android, iOS and web simultaneously.

React Native applications, similar to React for the Web, are built using JSX, a combination of JavaScript and XML-like syntax. The React Native "bridge" then calls the native rendering APIs in Objective-C (for iOS) or Java (for Android). As a result, your application will appear and feel like any other mobile application because it will be rendered using genuine mobile UI components rather than webviews. React Native exposes JavaScript interfaces for platform APIs, allowing your React Native apps to utilize platform capabilities such as the phone camera or the user's location [22].

Similar to React for the Web, React Native apps are written in JSX, a hybrid of JavaScript and an XML-like syntax. The Java and Objective-C (Android and Ios) native rendering APIs are then called via the React Native "bridge". Due to the fact that it is produced using native mobile UI components rather than webviews, your app will have the look and feel of a native mobile app. Using the JavaScript interfaces exposed by React Native, your applications may access platform features like the camera or the user's location.

3.2.4 React Component

In React, a component is a crucial building block. React Components are the building blocks from which your React applications are constructed. Components simplify the process of developing user interfaces significantly. [23]. There are primarily two kinds of components in React.:

1. Functional Components: Javascript functions are used to create functional components. By writing a javascript function, we can create a functional component in React. These functions may or may not be provided with data as parameters. In figure 3.3 you can see how a function component looks like:

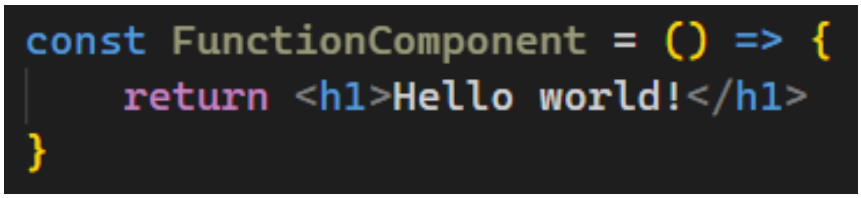

Figure 3.3: React Function Component

2. Class Components: In comparison to functional components, class components have a few more layers of complexity. The functional components of your software don't know about the other parts, but the class parts may work together. Data can be passed from one class component to another. In figure 3.4 you can see how a function component looks like:

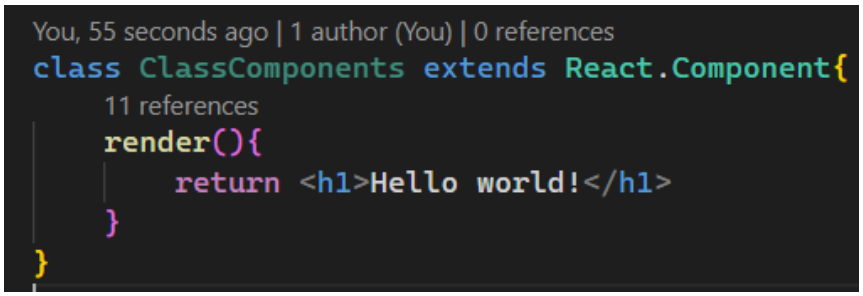

Figure 3.4: React Class Component

3.2.5 VDOM

VDOM (virtual document object model) is a programming concept that can be achieved by using a library like ReactDOM, a "virtual" version of a user interface (UI) may be created in memory and kept in sync with the "real" DOM. This is called "reconciliation". The declarative API of React is made possible by this technique. By specifying the desired UI state, React will then guarantee that the DOM is properly updated. This eliminates the need for you to directly deal with attribute manipulation, event handling, and manual DOM modifications when building your application [24].

3.3 Server technologies

3.3.1 .NET

.NET is a free and open-source framework for building software for several platforms. The platform was created by Microsoft and may be used to build apps for the web, mobile devices, desktop computers, and even the Internet of Things using any of a number of different programming languages and libraries. Your source code is compiled into a Common Intermediate Language (CIL) regardless of the programming language used (C#, F#, Visual Basic, etc.) because.NET supports the Common Language Infrastructure (CLI). This guarantees high-quality translation between the many supported languages on the site.

The .NET architecture is based on two main components [25]:

- CoreFX: CoreFX is the platform's API, often known as UBCL (universal base class library). This is how the CLI Standard Libraries are implemented. These are libraries that offer common functionality including file system administration, exception handling, network connection, threading, and reflection.

- CoreCLR: This is the.NET runtime environment. It is in charge of running CLI applications and contains a just-in-time compiler.

.NET also offers different application model frameworks supporting for developing different types of applications:

- Blazor: A C# framework for creating web-based programs with a client-side component. Further, it provides the opportunities for sustainable development of WebAssembly-based client web applications.

- Windows Presentation Foundation (WPF): A user interface framework for making windows software.

- ASP.NET: Web applications, Internet of Things applications, and mobile backends may all be built with ASP.NET Core since it is cross-platform framework. It may function both in the cloud and locally.

- Xamarin: An open-source framework for creating high-performance apps for Windows, macOS, and mobile platforms.

- ML.NET: Without specialized knowledge in model development or tuning,.NET developers may utilize ML.NET to create their own models and incorporate custom ML into their applications.

I chose ASP.NET for as the framework for the relation between the client and the database because it is reliable, fast, easy to use, free, easy to integrate with other Microsoft

tools such as SQL Server. Using Entity Framework as an object object-database mapper allows performing CRUD operations without having to write SQL queries and it has great scalability.

3.3.2 Python

For creating the raspbery pi server I chose to use python as a programming language. Python is an interpreted high-level programming language that is object-oriented and has dynamic semantics. Because of its high-level built-in data structures, dynamic typing, and dynamic binding, it is well-suited for use as a scripting language to facilitate the integration of pre-existing parts. Software maintenance expenses are reduced because to Python's succinct, easy-to-learn syntax, which places a premium on readability. Python's modules and packages facilitate the separation of concerns in software and the reusability of its components. Python and its extensive standard library may be downloaded and distributed without cost on all major operating systems[26].

The reason for which I chose python is its multitude of libraries which could be integrated with my application. FLASK for building the raspberry pi server which communicates with the .net server, python GPIO to control the GPIO interface on the raspberry pi.

3.3.3 Flask

Flask is a lightweight web framework written in Python. Since it does not need the use of any special tools or libraries, it is called a "microframework." Unfortunately, it is missing certain key features that are often provided by third-party libraries, such as a database abstraction layer, data validation, and so on. Alternatively, Flask supports extensions that, when added to an app, work just as they were part of Flask from the start. Extensions are available for a wide variety of framework-related technologies, including ORMs(object-relational mappers), form validation, upload processing, and many open authentication protocols.

I chose flask because it is written in python and it doesn't depend on any other library or additional extensions, which makes it easier to use. It is particularly accessible due to the principle on which it was designed, namely that it may be extended to other tasks. At various levels a significant advantage is the simplicity with which it may be learned, thanks to its simple but unambiguous syntax. For example, in figure 3.5 you can see how easy it is to write and setup a web application to print "Hello World" on the screen.

```
from flask import Flask
app = Flask(__name__)


@app.route('/')
def hello_world():
    return 'Hello, World!'

if __name__ == '__main__':
    app.run()
```

Figure 3.5: ”Hello World” in Flask [27]

3.3.4 Ngrok

An easy way for developers to publish a local development server to the Internet, Ngrok is a tool that works on several platforms. Your web server may now run locally without needing a public Ip or domain name by disguising as a subdomain of ngrok.com. While reverse SSH tunneling offers comparable functionality, it requires additional configuration and the hosting of a remote server.

To get over NAT (network address translation) Mapping and firewall restrictions, Ngrok creates a persistent TCP (Transmission Control Protocol) tunnel from a randomly generated subdomain on ngrok.com (for example, cd90-78-97-210-25.ngrok.io) to the local system. The ngrok client program connects securely to the ngrok server after you've specified the port number on which your web server listens, allowing anybody with the ngrok tunnel address to send requests to your local server [28].

Figure 3.6 shows how simple it is to get started using ngrok. To expose your localhost server port to the internet, just download the ngrok.exe executable from their official website. After launching it, you will be given a temporary unique endpoint to use for routing requests to the localhost server. The drawback of the free plan is that each time you get a new random url, you must alter your configuration files in order to send the requests to the correct location.

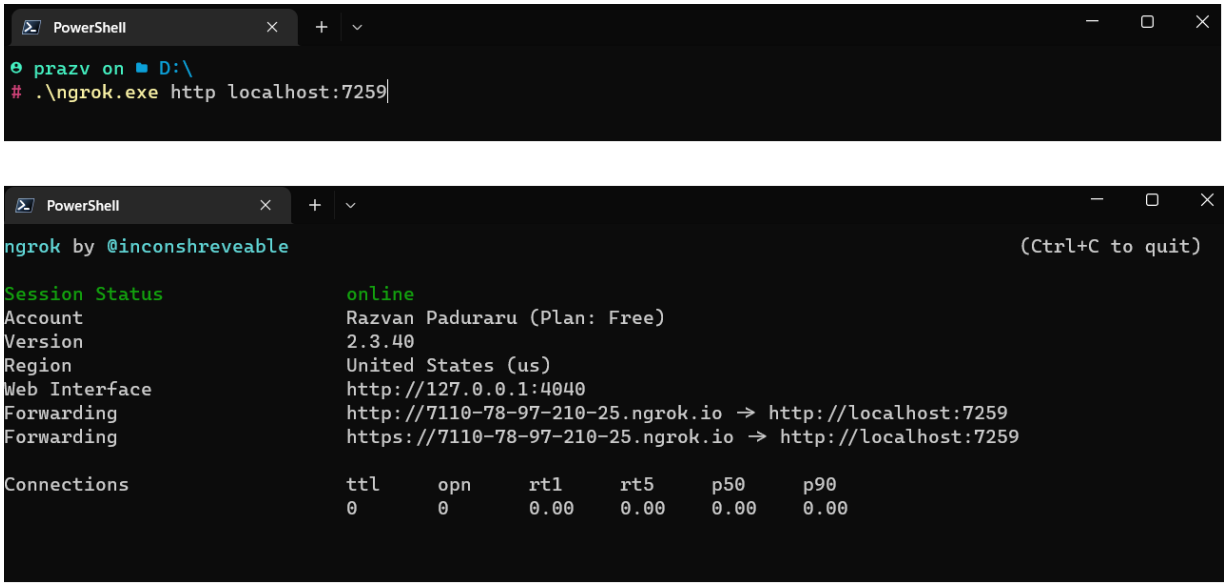

Figure 3.6: Running Ngrok

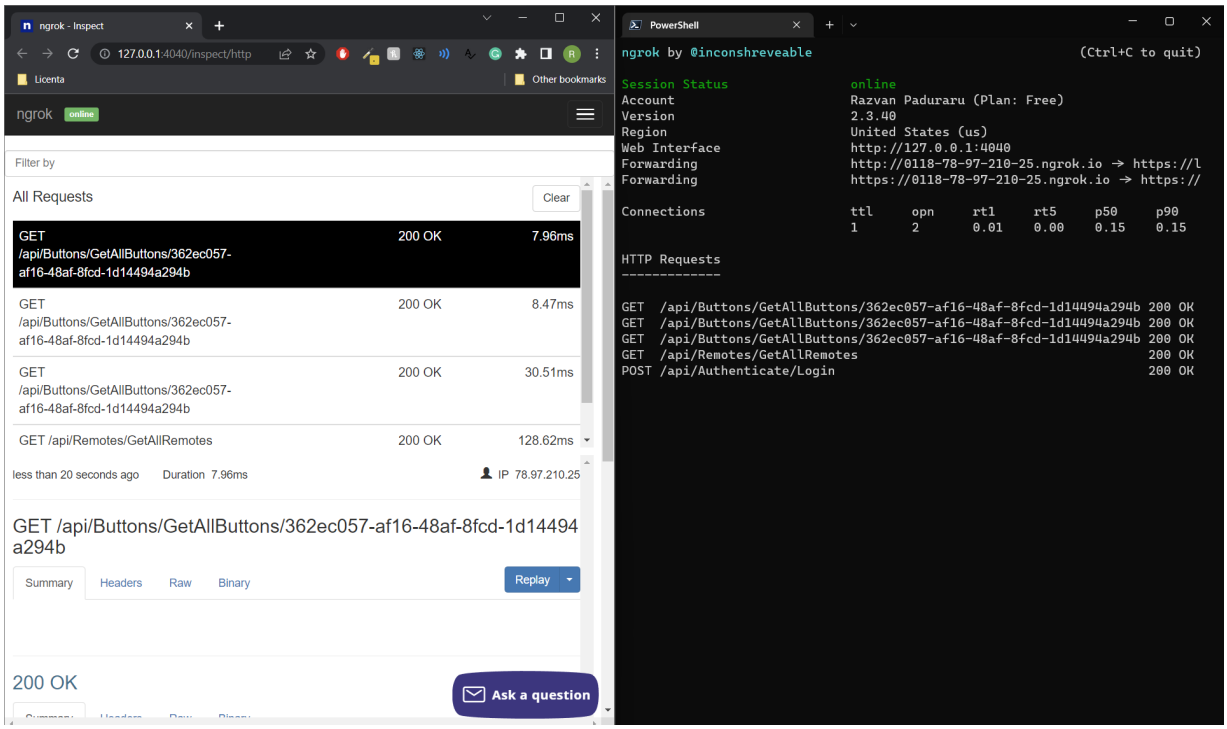

Figure 3.7: Ngrok Interfaces

Ngrok makes it incredibly simple to test the mobile application on a real iOS device

without the requirement for a Mac or Xcode. Ngrok removes the requirement for an android emulator on windows to test a mobile application and makes it much simpler to access the API outside of your local network. Figure 3.7 shows that ngrok provides both a web interface and a console interface for viewing requests.

3.4 Hardware Specifications

3.4.1 Raspberry Pi

As a development board for my project, I utilized a Raspberry Pi Model 3b (Figure 3.8). I chose it because it is a low-cost Linux computer with GPIO (general purpose input/output) ports that allow you to operate electronic systems for physical computing and it is a great board and starter to get into IoT (internet of things).

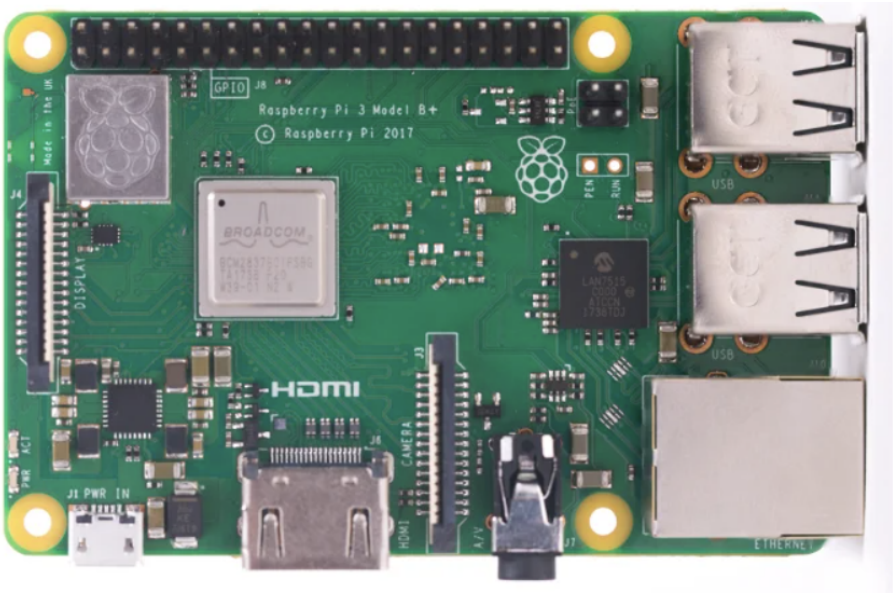

Figure 3.8: Raspberry PI model 3b

A Raspberry PI may be used as both a computer and a micro-controller board, similar to an Arduino. Its key benefits over an android are are as follows:

- CPU power
- Because the Raspberry Pi runs Linux, it has access to a wide range of programming languages, some of which may be used in conjunction with the GPIO. Python and Scratch are two prominent examples of GPIO-compatible languages, but there are many more, including Node-RED, Ruby, and C. In contrast, Arduino exclusively uses C/C++ routines that are built and flashed on a board.

- The Raspberry PI already as a network board and can be connected to the internet through wifi or a network table while Arduino needs extra modules for this.

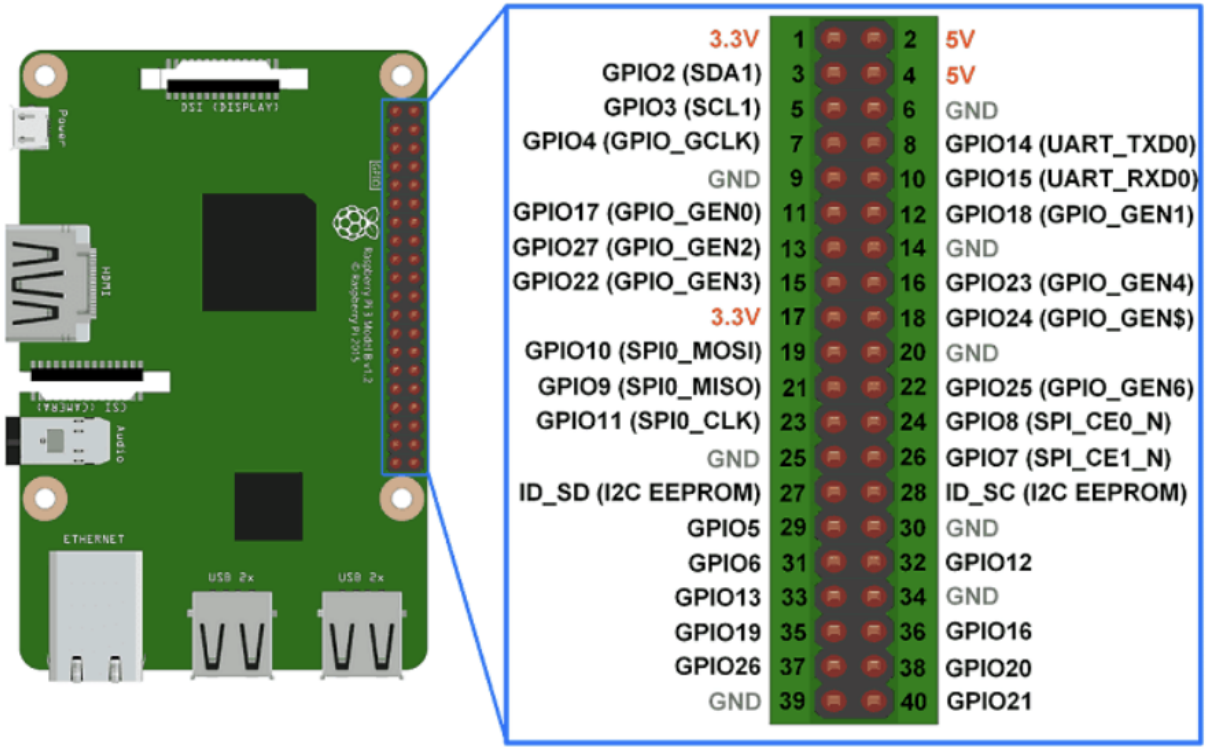

Figure 3.9: Raspberry PI GPIO [29]

In figure 3.9 you can see the gpio distribution in details. For my project I used the pin number 4 as the VCC, pin number 9 for the ground, pin number 12 as the output pin for the infrared emitter and pin number 11 as the input pin for the infrared receiver.

3.4.2 KY-022 Infrared Receiver Module

The KY-022 Infrared Receiver module (see figure 3.10) responds to IR light at 38kHz. It can accept commands from IR remote controls on TVs, stereos, and other devices. This module is made up of an 1838 infrared receiver, a 1k resistor, an LED, and three male header pins. The KY-022 Infrared Receiver module has the following specifications [30]:

- Operating Voltage : 2.7V to 5.5V
- Operating Current : 0.4mA to 1.5mA
- Reception Distance : 18m $ReceptionAngle$: $\pm 45°$

- Carrier Frequency : 38KHz
- Low Level Voltage : 0.4V
- High Level Voltage : 4.5V
- Ambient Light Filter : up to 500LUX

Figure 3.10: KY-022 [30]

3.4.3 Infrared Emitter

Due to the need of a resistance at the base pin of the transistor, I used a PN2222A, an infrared LED, and a 1k resistor to create an infrared light source. Basically, we need to turn the 5v GPIO pin on and off in order to generate the needed infrared modulations.

A solid state lighting (SSL) device that generates light in the infrared band of the spectrum of electromagnetic is known as an IR LED (infrared light emitting diode). IR LEDs enable the low-cost, high-efficiency generation of infrared light, which is electromagnetic waves with wavelengths ranging from 700 μm to 1mm.

Typical NPN transistor applications include VHF (very high frequency) amplifiers and switches. This transistor's design can be implemented in silicon. When used in place of an NPN transistor, this kind is treated as if it were just another regular transistor. A popular NPN bipolar junction transistor(BJT), the 2N2222A transistor is often put to use for low-power switching and amplification. This transistor operates at high speeds for its class, requires little to no maintenance, and can handle voltages up to a few hundred volts [31].

I needed a strong resistor to restrict the power traveling through the LED since I was pulsing it rather than just turning it on and off for long periods of time. I picked the 1 1kΩ resistor after experimenting with many others. I needed the strongest signal possible while without reducing the brightness of the LED too far, and this one provided the greatest results for me.

The electrical circuit for the infrared module is shown in figure figure 3.11. The black wire is attached to ground, the red wire is connected to 5V, and the greed wire is the signal connected to board pin number 12.

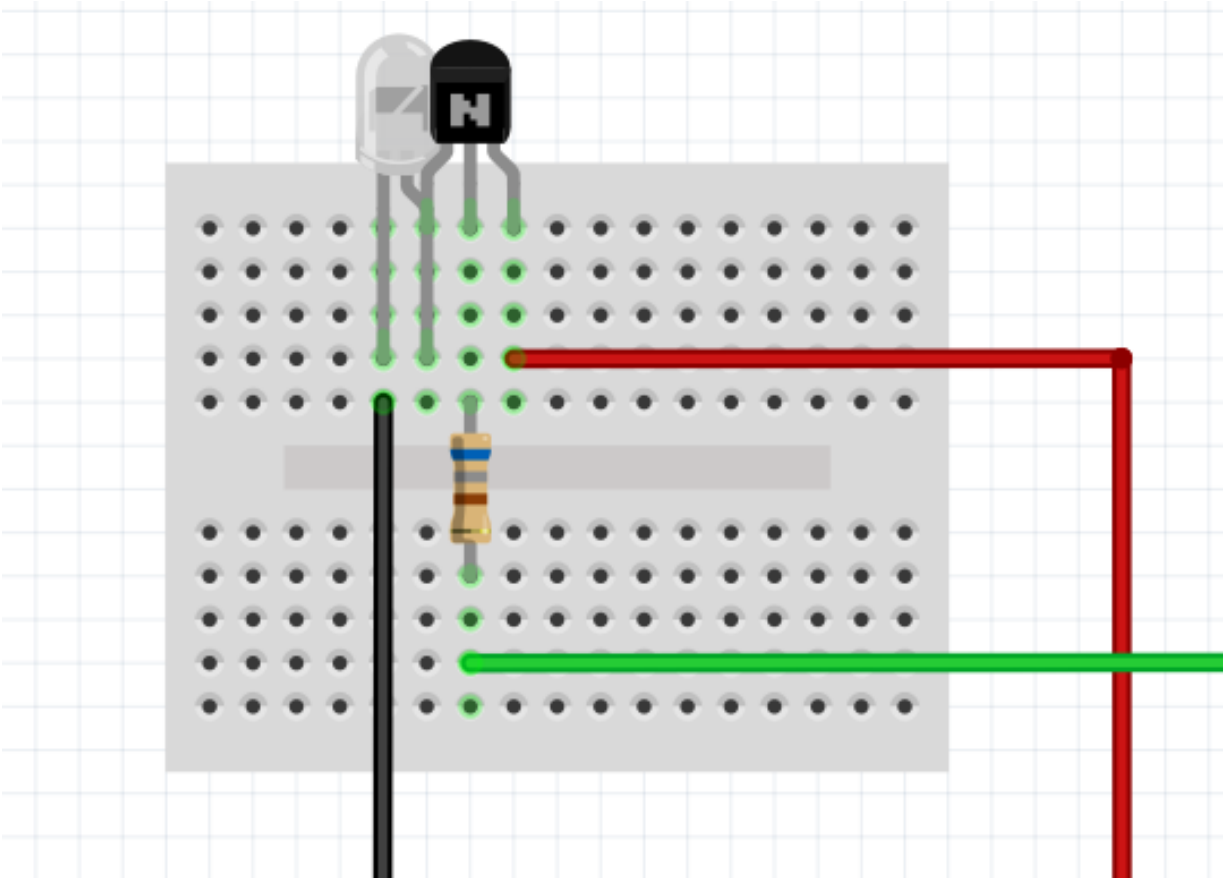

Figure 3.11: Infrared emitter circuit

Chapter 4

Design and Implementation

In this chapter I will present the implementation of E.C.L.A.I.R. step by step.

4.1 Design of the Project

4.1.1 Hardware circuit

When I first started working on my project, I wanted to see how my final circuit will appear, so I produced a prototype design in Fritzing. Figure 4.1 depicts how I envisioned my connections, sensors, and actuators. It was much simpler to maintain consistency across the program and not get lost along the way by first planning and then executing.

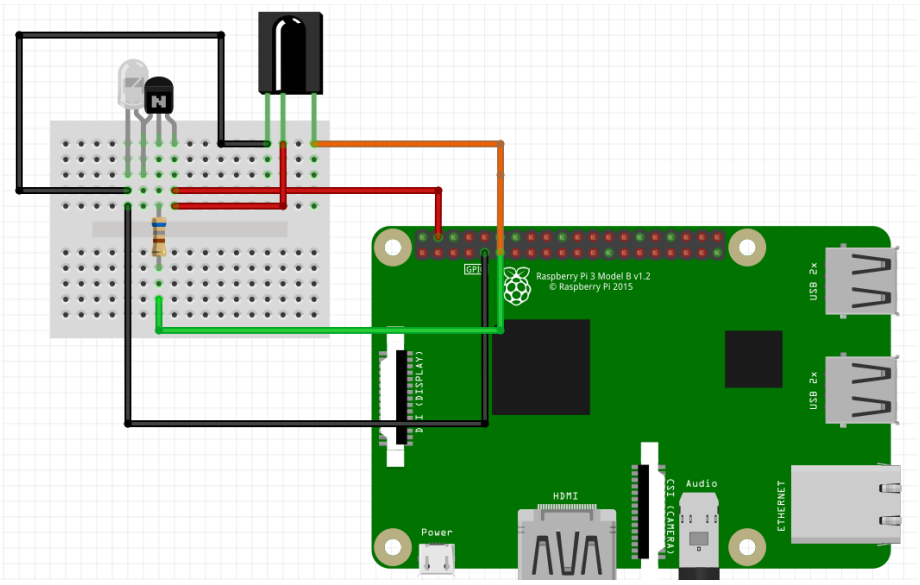

Figure 4.1: Final circuit

4.1.2 Application Architecture

In this section, we will examine the application's architecture. The components can be seen in figure 4.2. I chose a typical client-server design since we may have numerous clients accessing our program from many devices, and resources may be shared at times. Having two servers, one for communication with the client and the database, and one for communication with the raspberry pi, which performs all infrared control. For this use case, I'll offer the client is an iOS mobile application.

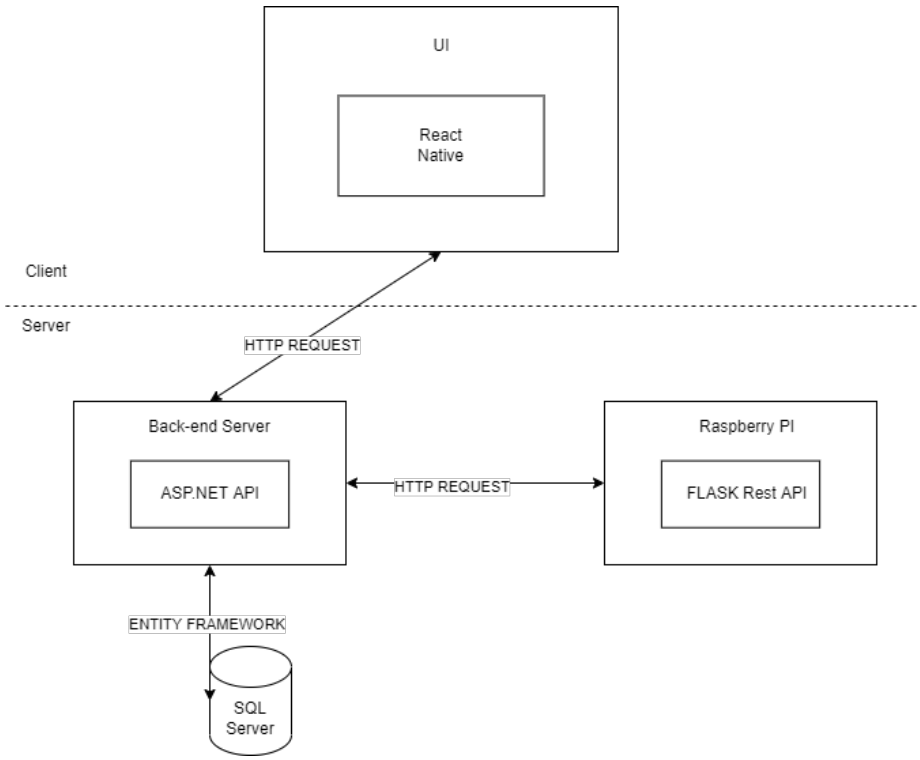

Figure 4.2: Project components

4.1.3 Use Case Diagram

A good use case diagram presents a high-level picture of the interaction between use cases, actors, and systems. It is also known as a behavior diagram since it is used to represent the behaviors of all system users.

The Use case diagram (figure 4.3) shows that there are two categories of app users. The app user might be an existing user or a new user. If the user is unregistered, he may register but cannot access anything related to remote control. If the user is already enrolled, he may see new remotes, register new remotes, update existing ones, examine the list of buttons for each remote, register a new command, or remove existing ones.

Furthermore, we can see the admin on the right side of the figure, who had the following options: verify existing accounts, remove users, and manually reset passwords.

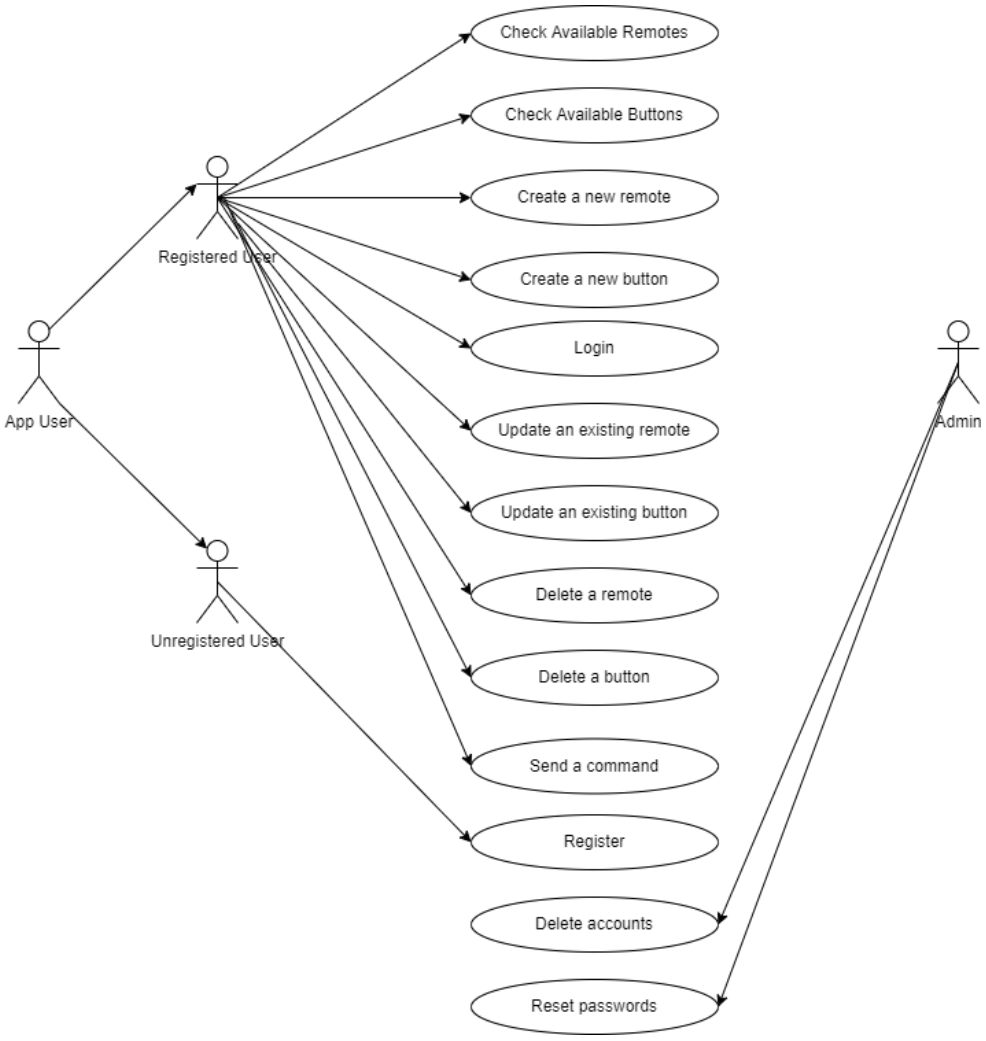

Figure 4.3: Use case diagram

4.1.4 Flowchart

Flowcharts show the steps of a process in logical sequence. A manufacturing process, an administrative or service procedure, or a project plan might all benefit from the usage of this versatile tool.

My flowchart on page 4.4 shows the fundamental flow of the program and the actions you may take at each phase.

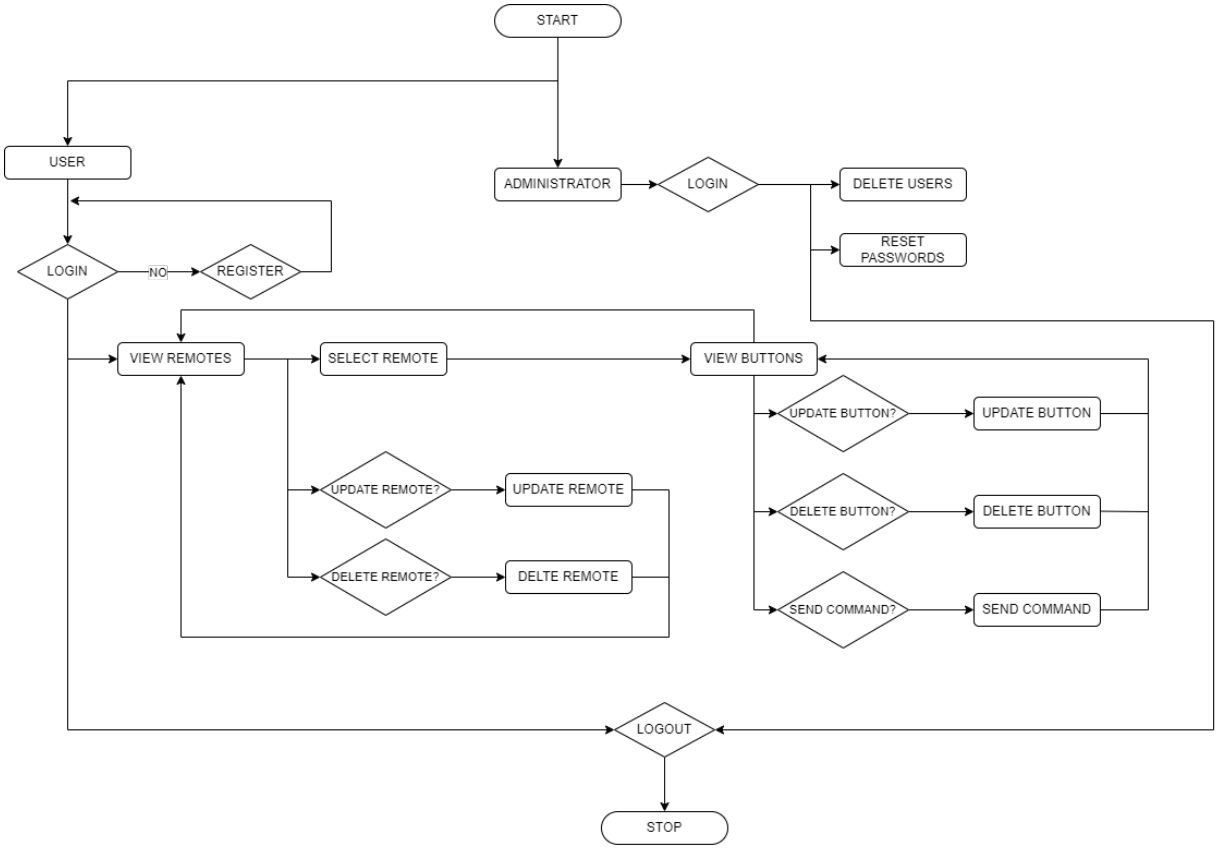

Figure 4.4: Flowchart

4.2 Implementation of the Project

4.2.1 Infrared signal decoding

When I began this project by first goal was to switch on and off an RGB light bulb. I was clueless about infrared, but I figured there must be an easy way to record the IR signal and repeat it using an infrared LED. I did some research since I didn't know what resources would be required. Here are a few key points when it comes to infrared signal decoding: [32]:

- For most goods, an infrared remote simply sends a series of 1s and 0s utilizing non-visible light.
- A photo diode (to catch the infrared frequency) or an infrared receiver may be used to collect an IR signal
- Many items employ the NEC protocol and run at 38 kHz (we'll get to that later in this section).

- For our purposes, a ”carrier wave” is just an infrared light that flashes on and off extremely fast. 38,000 times per second in this scenario
- 1s and 0s are encoded by switching the LED on and off at 38 kHz for a specific length of time, until the entire signal is received

I attached my KY-022 Infrared Receiver module to the board after learning all of this and started decoding the signal. To read the values of infrared receiver pins, I created a Python script. I found that it moves quickly enough to accomplish our goals. Finding the precise 1s and 0s that the signal represents as well as the estimated timings of the pulses are the objectives of capturing our remote. To do this, we shall tally the amount of pulses and pauses in the signal. The infrared receiver consistently outputs 1. (no signal). When we read the input pin and get a 0, it means that the ir receiver picked up a 38 kHz infrared pulse. We'll record when the receiver initially emits a 0 and time how long it takes until we get another 1. After running the script I was pleased to find out that the remote that I was testing follows the NEC protocol. The message bits in the NEC IR transmission protocol are encoded using pulse distance. Each pulse burst lasts 562.5 μs and has a carrier frequency of 38kHz. The following logical bits are transmitted[33]:

- Logical '0' – This is a 562.5 μs pulse burst followed by a 562.5µs space, with a total transmit time of 1.125ms
- Logical '1' – This is a 562.5 μµs pulse burst followed by a 1.6875ms space, with a total transmit time of 2.25ms

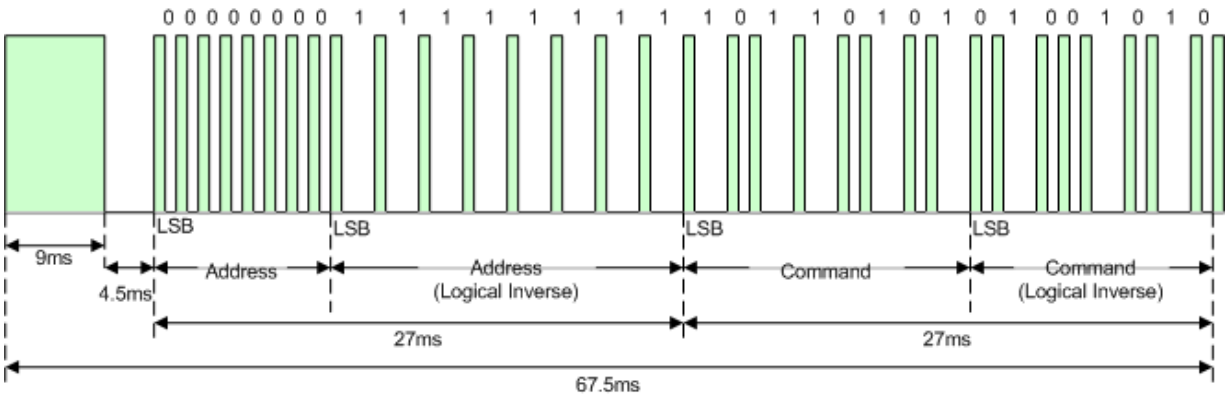

Figure 4.5: Example of how the NEC IR transmission protocol is used to send a message frame [33]

You can see in Figure 4.5 that:

- Both the 16-bit address (address + inverse) and the 16-bit command (command + inverse) take 27ms to send out. This is because each 16-bit block will hold eight '0's and eight '1's in the end, which adds up to (8 * 1.125ms) + (8 * 2.25ms).

- Transmission of the whole message frame occurs in 67.5 milliseconds (With the exception of the last 562.5-second pulse burst signaling the end of transmission)

4.2.2 Transmitting the IR signal

At first, I attempted to send the IR signal in the same way I had received it. For infrared to work properly, a stable 38 kHz signal must be generated. That means you'll need to toggle the light on and off again every 0.00002631578 seconds. There was enough of processing power in the Raspberry Pi's 1.2GHz CPU to toggle a pin at 38 kHz. You don't have access to the whole 1.2GHz of computing power, however. You need to run the operating system, which eats up a lot of resources. Furthermore, I was working in Python, which is an interpreted language that randomly triggers a garbage collector and hence might slow down your progress. As your application is not the only one running, the OS may pause your program to enable other applications to continue, and this is true even for compiled programs written in C++ or C. You can get close to the required speed in C, but the signal won't function reliably due of the OS disruptions and you won't get a steady stream of data. After some time, I discovered LIRC. Since LIRC is a kernel module, it operates on the same level as the OS. This way, it won't be killed by the Pi's OS and will be able to maintain a constant signal on the GPIO pins.

LIRC is a program that decodes and sends infrared signals from many (but not all) regularly used remote controllers. Some IR remote controllers may now be used as conventional input devices with recent Linux kernels. This sometimes renders LIRC obsolete. However, LIRC provides more flexibility and versatility and is still the best tool in many situations. The lircd daemon, which decodes IR signals received by device drivers and transmits the information on a socket, is the most crucial component of LIRC. It also takes orders to send IR signals if the device supports it [34].

With LIRC I managed to send my first successful to my lightbulb. But unfortunately, setting up LIRC was quite tedious. The config files were very confusing, the long process of setting it up was needed on each new installation of raspbian and the main problem was that I had to create a config file for each remote in the raspberry pi file system and this was against of how I wanted my project to behave (storing the pulses in a database, sending them to the raspberry pi and have it do the work of transmitting them without the user interfering). After failing with LIRC I came across the pigpio library for C.

Pigpio is a C library made by joan2937 for the Raspberry which allows control of the General Purpose Input Outputs (GPIO) [35]. Out of its features the one that is most important is "Construction of arbitrary waveforms to give precise timing of output GPIO level changes (accurate to a few microseconds)" which is exactly what I needed. Using it and Brian Schwind's ir-slinger [36] I managed to get my desired results in transmitting infrared signals.

4.2.3 Implementing the appliction

When it came to deciding how my application should appear and behave, React was the clear winner. React is an extremely versatile framework, and React Native provided me with all of the tools I needed to create an uniform UI and native apps for Android and iOS. React native acts similarly to HTML, CSS, and JAVASCRIPT all in one file. I could put my page template, stylings, and logic in each file, and then use the axios library to send requests to my server. .NET with Entity Framework made my task of querying the database much simpler. Entity Framework is an excellent ORM (Object–relational mapper) because it allows me to create model classes, map them to the database, and get them as generic lists using a database context.

My database diagram may be seen in figure 4.6. Identity Framework was quite useful in developing safe authentication. The ASP.NET Core Identity framework manages and stores user accounts in ASP.NET Core applications. Identity takes advantage of Entity Framework data models and generates highly helpful tables like as "AspNetRoles" or "AspNetUserRoles" that make it very simple to create users with various roles in an application such as regular users, administrators, managers, and so on.

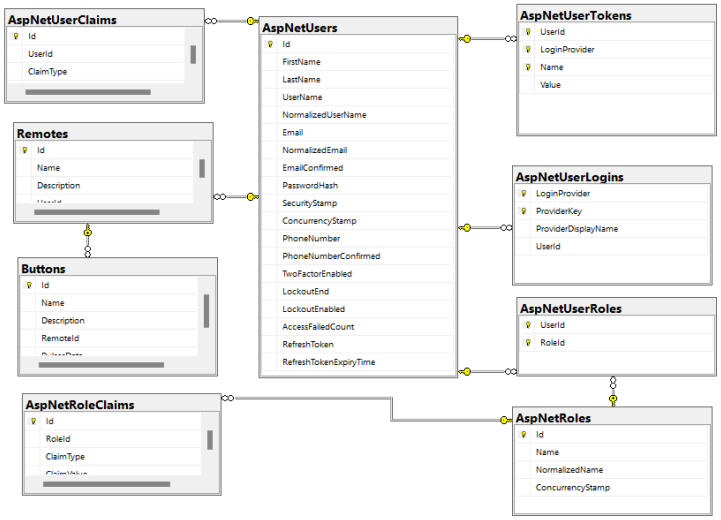

Figure 4.6: Database Diagram

4.3 Application Overview

In this section, I'll walk you through a whole application use scenario. I'll go through each of its primary features step by step.

When you first open the application you will be greeted by a splash screen. Splash screens are often used by especially big apps to warn the user that the program is loading.

They inform you that a long procedure is in progress. A progress bar inside the splash screen reveals the loading process on occasion. Whenever the application's main screen appears, the splash screen vanishes. Splash screens can be applied for a limited time and then removed. I created it because I didn't want the user to be greeted directly by a login page. It can be enhanced with animation with a log or the identity of the app.

Following the splash screen, the next screen is the login screen, where you have the following options:

- "Forgot Password?" This takes you to the form in which you need to enter your email in order to receive a password reset link
- "Contact" Takes you to the contact form
- After completing your correct credentials you can use the "Sign In" button to access the application
- "Sign Up" will take you to the register form.

The splash screen and the first login form are seen in 4.7.

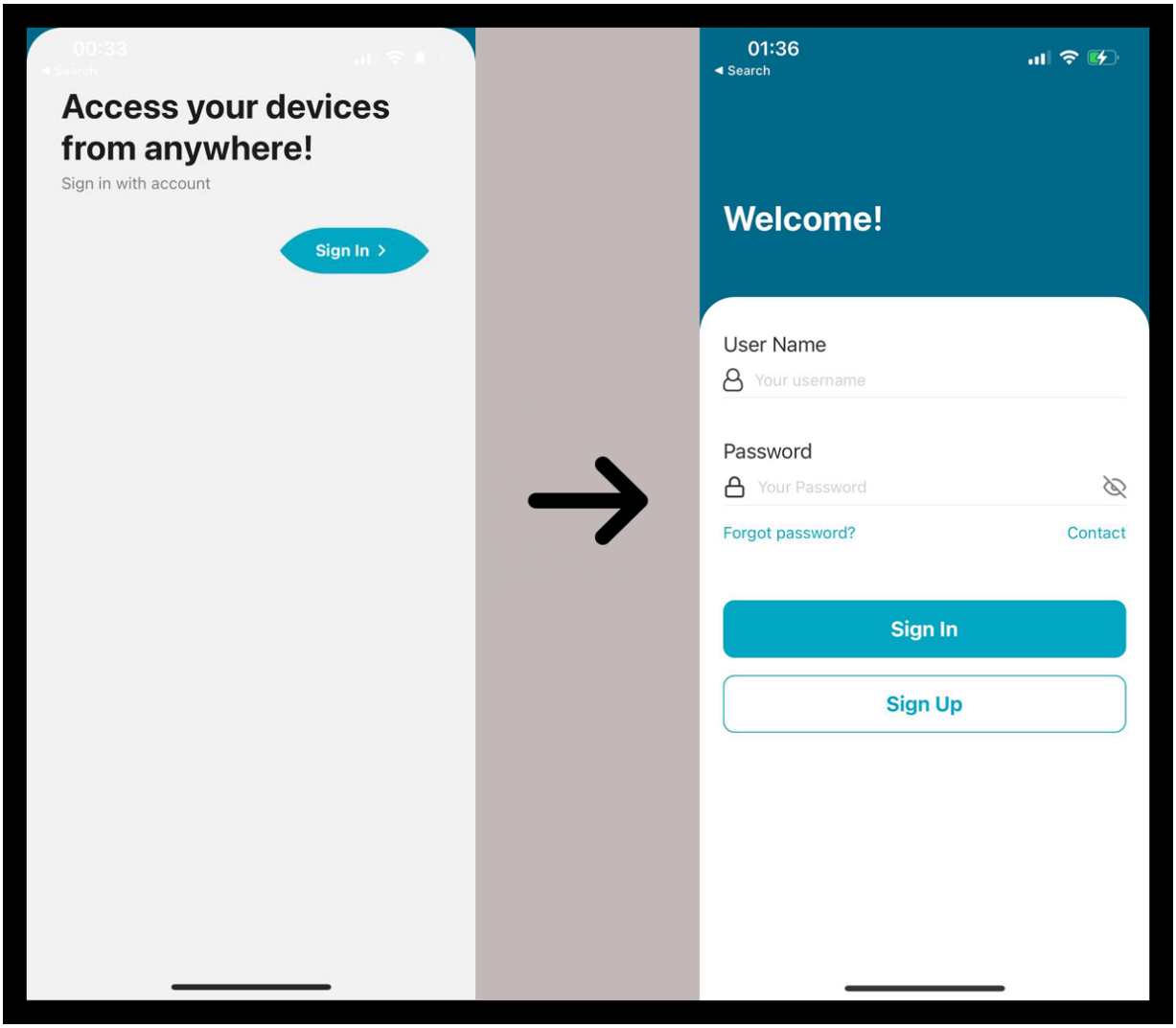

Figure 4.7: Splash Screen and Log in

Register

Consider the following scenario: the user is an unregistered user who has to register in order to access the program. The registration form is shown in 4.8. To register, you must fill out all of the required information. If a user with the identical email or username already exists, the account cannot be created. After entering all of the required information, we hit the sign up button, and a new account is established with a confirmation link delivered to your email. The account will be inaccessible until the verification link is clicked. After creating and confirming the account we can go back to the log in screen and sign in.

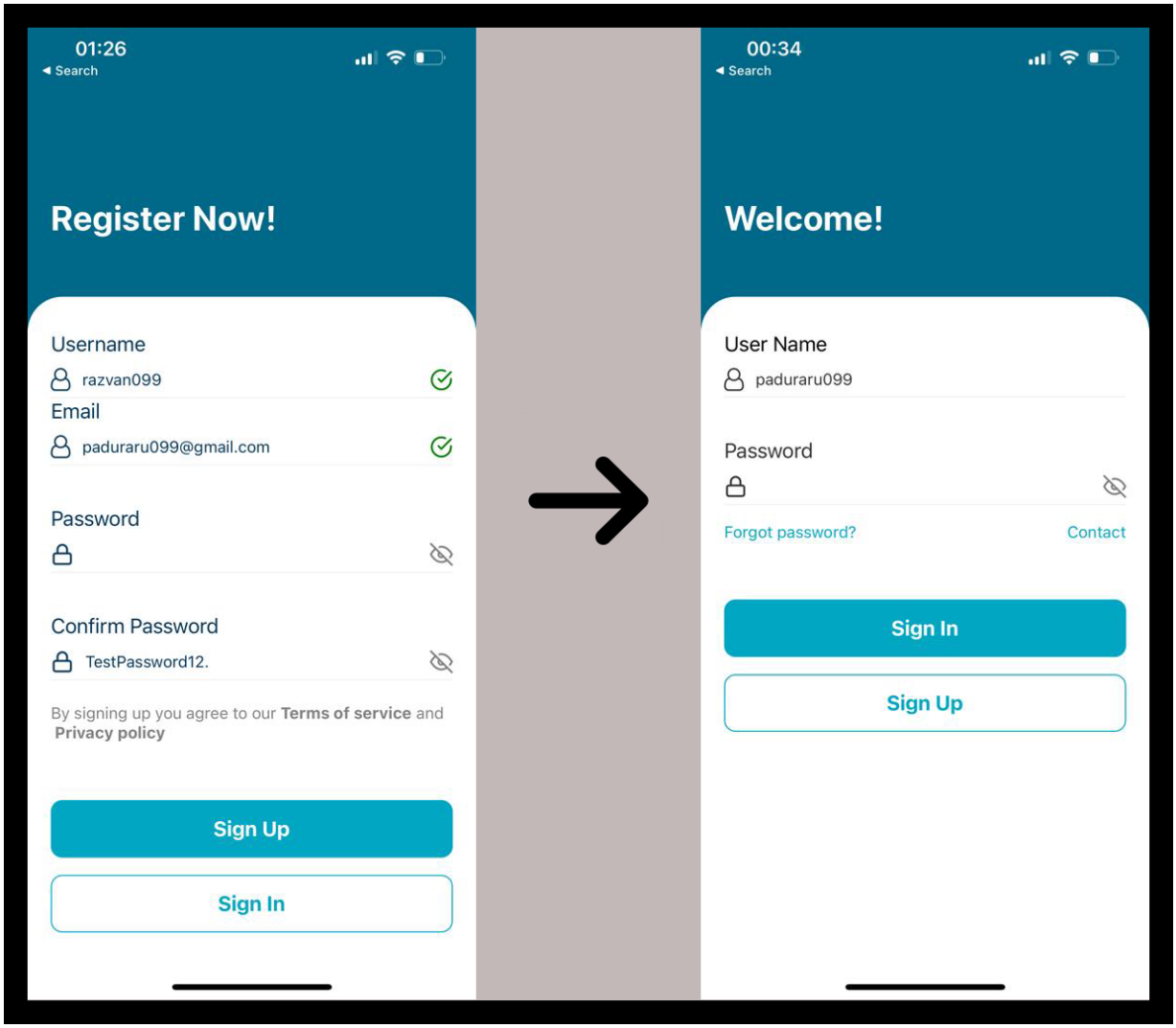

Figure 4.8: Register and Log In

Home page

When we log in, we are met with a home screen and a brief summary of what E.C.L.A.I.R. they are. Swiping right or tapping the top left button reveals the navigation drawer menu, as seen in figure 4.9 , which includes the following menu options:

- Home: the initial landing page after being signed in. Can be seen in figure 4.9.
- Remotes: The screen displaying the user's list of remotes.
- Profile: Profile page in which he can change is password.
- Support: Contact form.

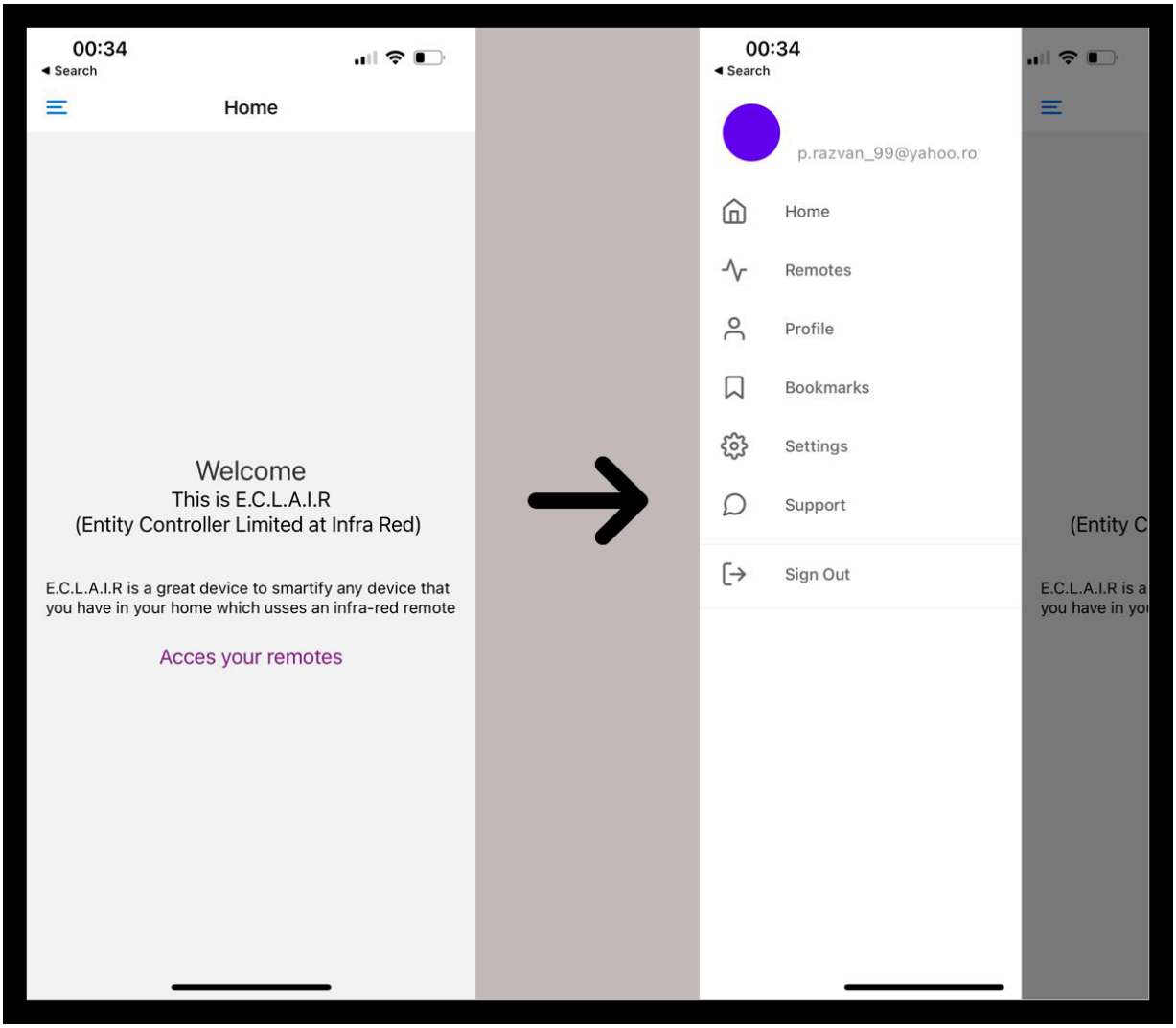

Figure 4.9: Home screen and app drawer

Remotes page

The remotes screen is shown in 4.9. The user may see its list of remotes, their descriptions, and by hitting the ”Add remote” button, a modal will be created to complete the criteria for a new remote in the database. If both fields (Remote Name and Description) are filled out and the ”Add” button is pushed, a new remote will be created in the database for that user. From each screen you can use the previously mention menu drawer to access any other screen of the application.

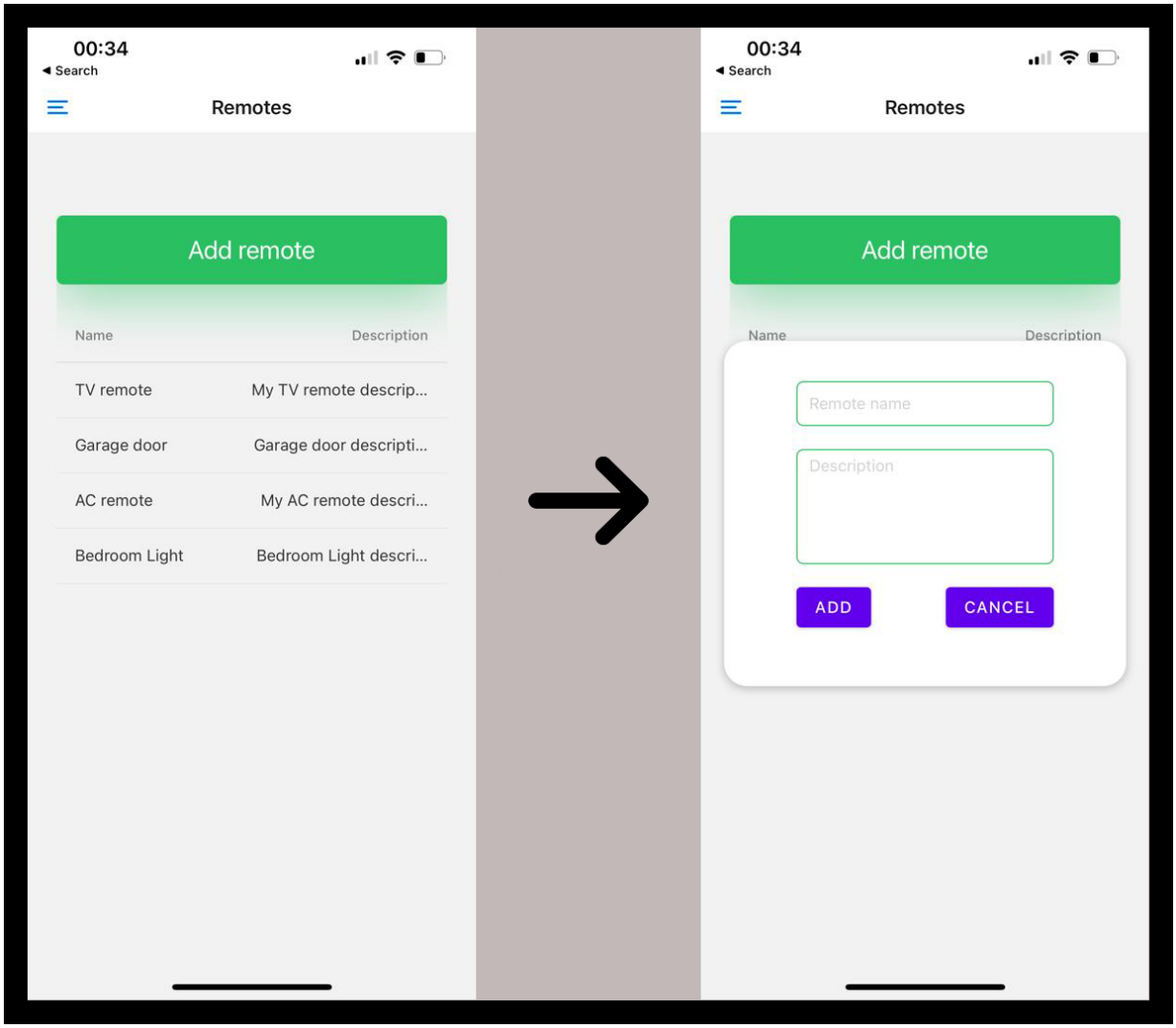

Figure 4.10: Remotes Page

Buttons page

The program user may access the buttons screen for any remote by clicking on it in the remotes screen. Figure 4.11 depicts how the buttons screen appears. It has a similar user interface and functionality to the remotes screen, with a few important distinctions. From this first screen, you may send instructions to the raspberry pi to cause the infrared pulses associated with that button to be emitted by clicking on that icon which is at the end of the row for each button.

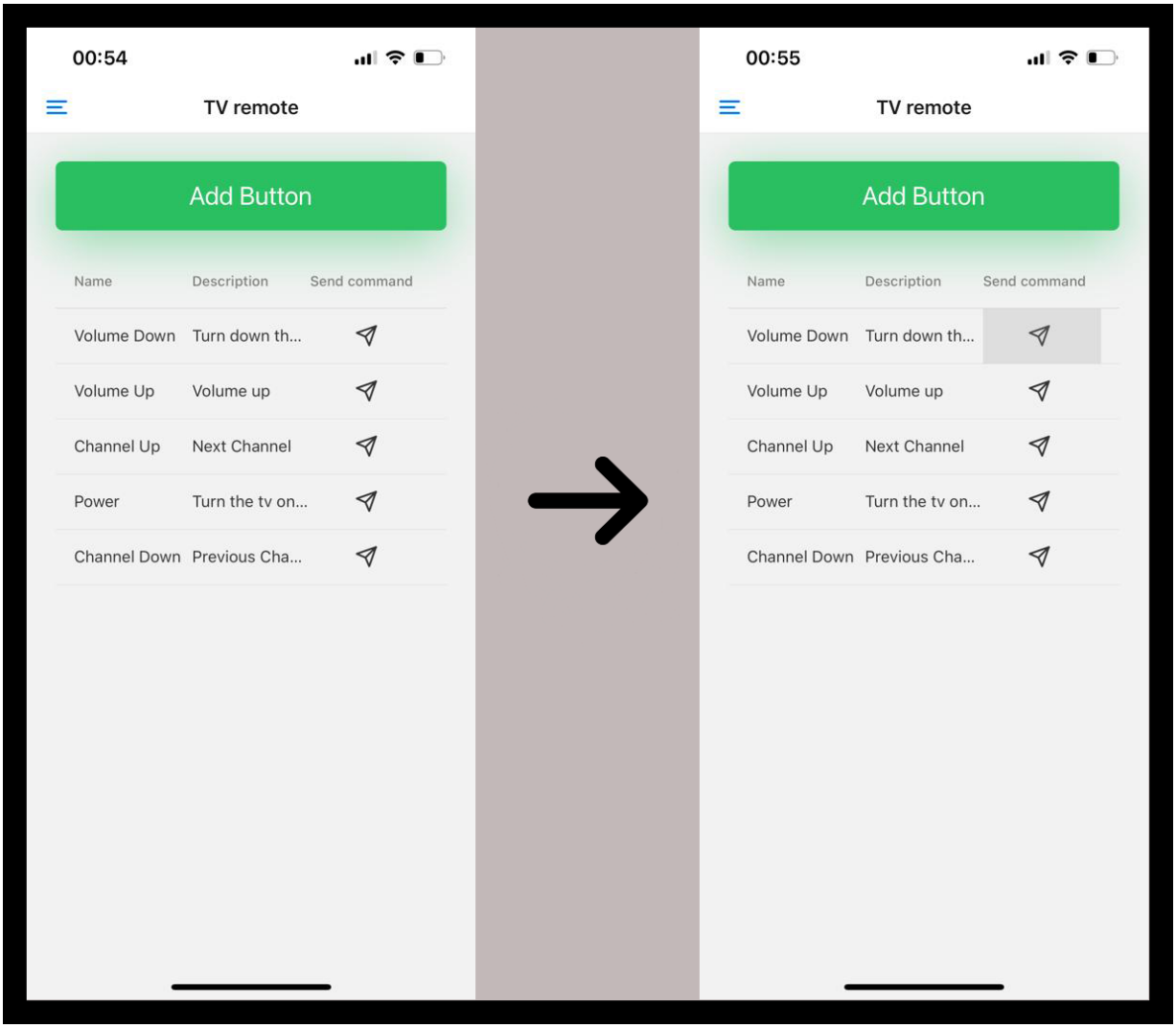

Figure 4.11: Remotes Screen

By clicking on a row a modal will be open with the details of that remote in figure 4.12. On the initial state of the modal the text boxes are in read only mode due to the fact that the user can also use clicking on a button to see its description and he shouldn't be able to send update requests by mistake. Only after pressing on the "Edit" button, the text boxes become editable and he can send the update request.

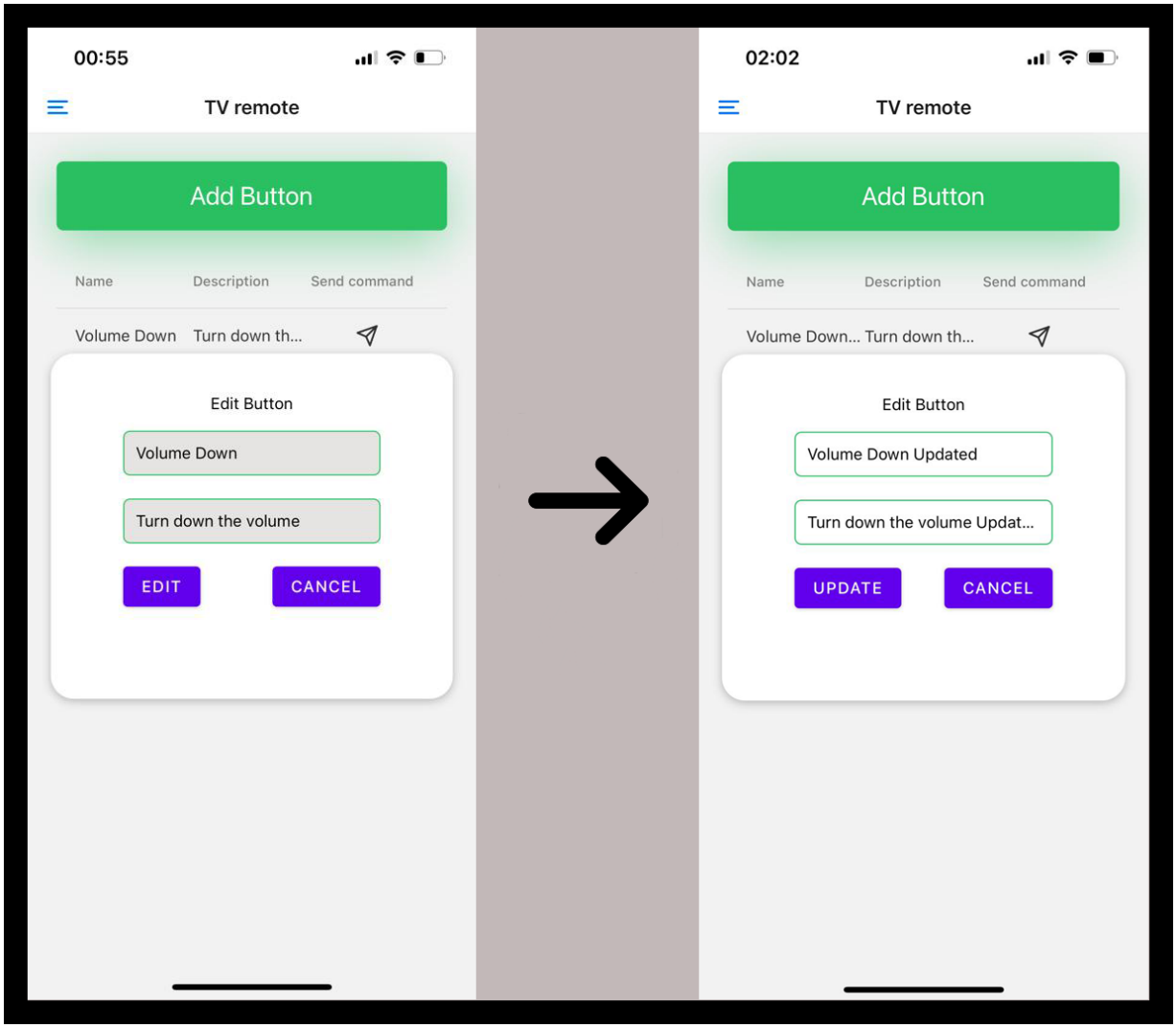

Figure 4.12: Info/Edit Button

By hitting the ”Add Button”, you will be shown to the modal seen in figure 4.13. After entering the necessary information (Remote Name and Remote Description), click the ”Add” button to register the button. When you hit the Add button, a request is made to the raspberry pi flask server to begin reading infrared. When the KY-022 Infrared receiver begins reading, use your control remote to push the button where you want the infrared command to be recorded. Only once the infrared command has been successfully recorded will the new remote with the read pulses be entered into the database.

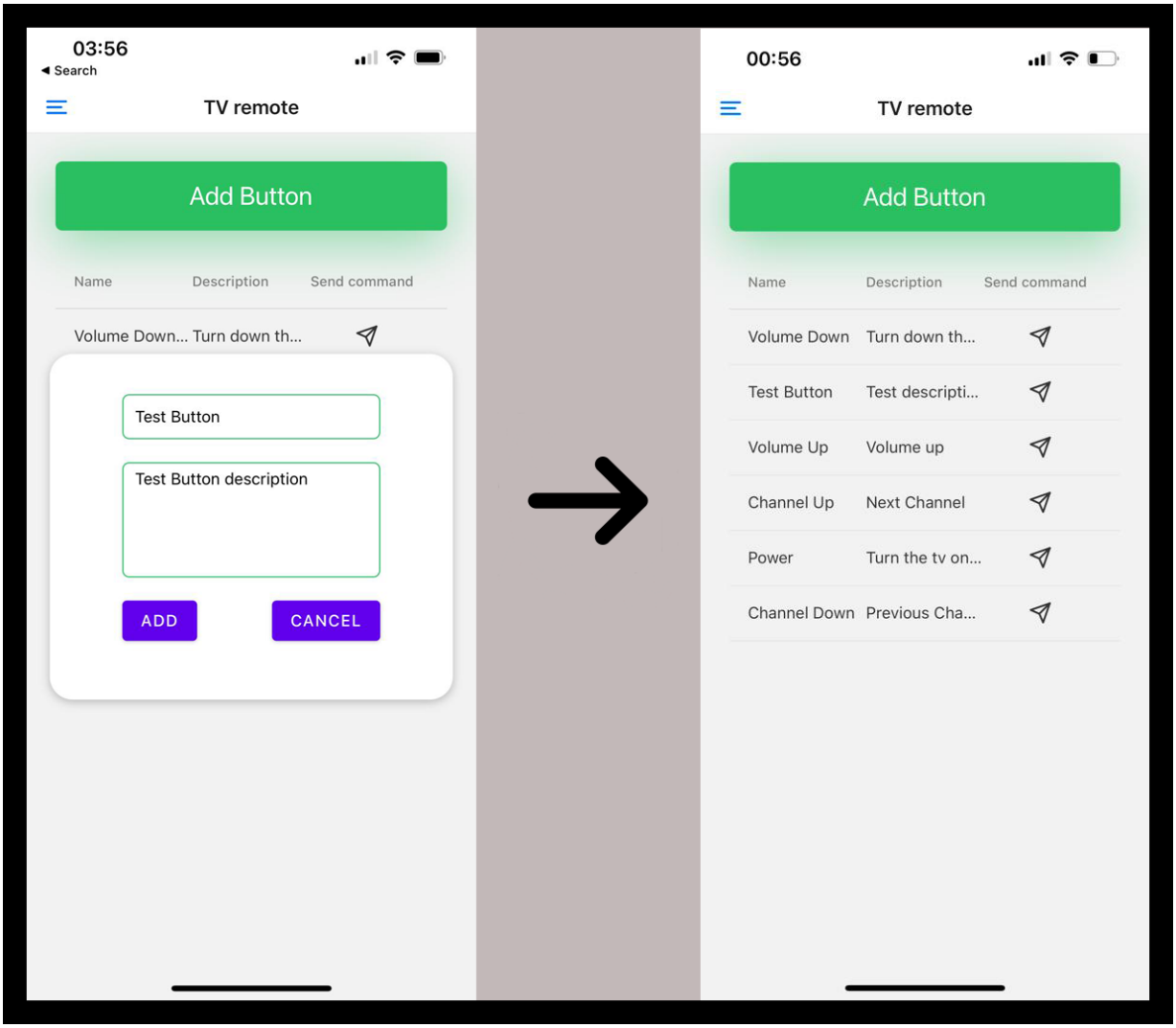

Figure 4.13: Add new button

Chapter 5

Conclusions

This chapter provides the outcomes of the previously stated project's design and execution, as well as the personal contributions made to its completion.

5.1 Obtained results

As I stated in the introduction chapter of this paper, my thesis main purpose is to offer to the user the possibility of controlling his infrared devices over the internet. Also, this application is meant to remove the need of having many remotes for many devices in your home. Reaching the end of the implementation process and after all the validations and verification, I can state that the goal of the application was accomplished.

If compared to other kinds of remote controls, I believe mine is pretty unique since the user can quickly register any remote and button without understanding anything about infrared or programming. I was able to construct a practical, user-friendly mobile application with the intention of scaling it and improving it in the future.

5.2 Personal Contributions

The personal contributions brought in the implementation of this project are:

- Design and implementation of the hardware specifications
- Design and implementation of an user friendly mobile application
- Design and implementation of servers which are able to take multiple requests at a time

5.3 Future improvements

The project presented in this thesis, was designed in the perspective of developing further functionalities using the same technologies which are more performant when it comes to sending and receiving infrared. The results obtained when testing the project where good but not good enough for a project that would be taken into production.

Some of the improvements would be:

- For the time being, the application is restricted to controlling the gadgets in the room where the raspberry pi is located. The hassle of needing to relocate the main board in the room with the item that you want to operate is significant. Having numerous raspberry pis is not a good idea since they are fairly costly, and the project cost would skyrocket. A good idea would be to set up Arduino boards with infrared transmitters, which is far less costly than the Raspberry Pi, all linked to the Raspberry Pi main board through the internet and setting up your remotes such that the raspberry pi can communicate to the arduino which is in the room with the device that you want to control.

- As I explained in subsection In section 4.2.2 the raspberry pi 3b had a small CPU and depending on what runs on it sending a steady 38 kHz signal isn't always successful even with that pigpio library. Some remotes have even bigger frequencies and that would be quite the challenge for the raspberry pi 3b to send. In 2019 Raspberry Pi 4 Model B was released which has 1.8 GHz CPU and a lot more RAM memory (up to 8 GB). I had a change to test my project on this board and the results were much better on this board.

- Introducing cloud services for hosting the database and the .Net server

- Many brand do have universal remotes codes which can be found on the internet. If the user already has a remote which has an universal code he should be able to import those into his remote profile without having to register each button.

Bibliography

[1] X. Han, "Infrared remote control design based on single chip microcomputer," in *2015 IEEE International Conference on Computer and Communications (ICCC)*, 2015, pp. 245–249.

[2] J. Bourne, "2022 global e-waste already weighs more than four burj khalifa," 2022. [Online]. Available: https://www.unlocknetzero.co.uk/news/2022-global-e-waste-already-weighs-more-than-four-burj-khalifa

[3] P. Pinaki. (2015, Feb) Wireless infrared remote controller for multiple home appliances. [Online]. Available: https://www.academia.edu/7142885/Wireless_Infrared_Remote_Controller_for_Multiple_Home_\Appliances_144

[4] R. A. Serway and J. W. Jewett, *Principles of physics*. Saunders College Pub. Fort Worth, TX, 1998, vol. 1.

[5] M. Atkins and M. Boer, "Chapter 2 - flow visualization," in *Application of Thermo-Fluidic Measurement Techniques*, T. Kim, T. J. Lu, and S. J. Song, Eds. Butterworth-Heinemann, 2016, pp. 15–59. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128097311000022

[6] M. J. Haque and M. Muntjir, "Night vision technology: an overview," *International Journal of Computer Applications*, vol. 167, no. 13, pp. 37–38, 2017.

[7] W. Herschel, "Experiments on the refrangibility of the invisible rays of the sun. by william herschel, ll. d. f. r. s." *Philosophical Transactions of the Royal Society of London*, vol. 90, pp. 284–292, 1800. [Online]. Available: http://www.jstor.org/stable/107057

[8] E. Becquerel, *La lumiere, SES causes et ses effets*. Firmin Didot, 1867. [Online]. Available: https://books.google.com/books?id=SyWP1zBJiv0C&pg=PA141

[9] T. J. Seebeck, *Magnetische polarisation der metalle und erze durch temperatur-differenz*. W. Engelmann, 1895, no. 70.

[10] E. S. Barr, "The infrared pioneers—i. sir william herschel," *Infrared Physics*, vol. 1, no. 1, pp. 1–IN6, 1961.

[11] ——, "The infrared pioneers—ii. macedonio melloni," *Infrared physics*, vol. 2, no. 2, pp. 67–74, 1962.

[12] A. Rogalski, "History of infrared detectors," *Opto-Electronics Review*, vol. 20, no. 3, pp. 279–308, 2012.

[13] E. S. Barr, "The infrared pioneers—iii. samuel pierpont langley," *Infrared physics*, vol. 3, no. 4, pp. 195–206, 1963.

[14] B. S. Beschloss. (2013) Object of interest: Remote control. [Online]. Available: https://www.newyorker.com/tech/annals-of-technology/object-of-interest-remote-control

[15] Infrared waves. [Online]. Available: https://science.nasa.gov/ems/07_infraredwaves

[16] F. J. Low, G. H. Rieke, and R. D. Gehrz, "The beginning of modern infrared astronomy," *Annu. Rev. Astron. Astrophys.*, vol. 45, pp. 43–75, 2007.

[17] E. Ring and K. Ammer, "Infrared thermal imaging in medicine," *Physiological measurement*, vol. 33, no. 3, p. R33, 2012.

[18] C. Hildebrandt, K. Zeilberger, E. F. J. Ring, and C. Raschner, *The application of medical infrared thermography in sports medicine.* INTECH Open Access Publisher, 2012.

[19] "What are HTTP Headers? Definition + Examples - Seobility Wiki," [Online; accessed 2022-06-11].

[20] "What is a REST API? Definition and Principles - Seobility Wiki," [Online; accessed 2022-06-12].

[21] "What is JavaScript? - Learn web development | MDN," mar 23 2022, [Online; accessed 2022-06-12].

[22] B. Eisenman, *Learning react native: Building native mobile apps with JavaScript.* " O'Reilly Media, Inc.", 2015.

[23] "Reactjs components - geeksforgeeks," Mar 2018. [Online]. Available: https://www.geeksforgeeks.org/reactjs-components/

[24] "Virtual dom and internals." [Online]. Available: https://reactjs.org/docs/faq-internals.html

[25] A. Chiarelli, "What is .net? an overview of the platform," Oct 2021. [Online]. Available: https://auth0.com/blog/what-is-dotnet-platform-overview/

[26] "What is python? executive summary," 2022. [Online]. Available: https://www.python.org/doc/essays/blurb/

[27] M. Makai, "Flask - full stack python," 2012. [Online]. Available: https://www.fullstackpython.com/flask.html

[28] "ngrok and cross-platform development." [Online]. Available: https://www.pubnub.com/learn/glossary/what-is-ngrok/

[29] CamSoper, "Blink an led | microsoft docs," Mar 2022. [Online]. Available: https://docs.microsoft.com/en-us/dotnet/iot/tutorials/blink-led

[30] ArduinoModules, "Ky-022 infrared receiver module - arduinomodulesinfo," Oct 2016. [Online]. Available: https://arduinomodules.info/ky-022-infrared-receiver-module/

[31] ELPROCUS, "2n2222a transistor: Pin configuration, circuit, working its applications," Jul 2021. [Online]. Available: https://www.elprocus.com/2n2222a-transistor/

[32] B. Schwind, "Sending infrared commands from a raspberry pi without lirc," May 2016. [Online]. Available: https://blog.bschwind.com/2016/05/29/sending-infrared-commands-from-a-raspberry-pi-without-lirc/

[33] "Nec infrared transmission protocol." [Online]. Available: https://techdocs.altium.com/display/FPGA/NEC+Infrared+Transmission+Protocol

[34] lirc.org, "What is lirc?" [Online]. Available: https://www.lirc.org/

[35] joan2937, "Github - joan2937/pigpio: pigpio is a c library for the raspberry which allows control of the general purpose input outputs (gpio)." Mar 2021. [Online]. Available: https://github.com/joan2937/pigpio

[36] B. Schwind, "Github - bschwind/ir-slinger: A small c library for sending infrared packets on the raspberry pi," Oct 2019. [Online]. Available: https://github.com/bschwind/ir-slinger

List of Figures