MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE

NATIONAL AVIATION UNIVERSITY

Department of Electronics, Robotics, Monitoring and

IoT Technologies

ADMIT TO THE DEFENSE

Head of the graduate department

_______________ Volodymyr Shutko

«_____» ____________ 20__.

DIPLOMA

(EXPLANATORY NOTE)

GRADUATE OF A BACHELOR'S DEGREE

SPECIALTY 171 «ELECTRONICS»

Subject: Temperature control in the apartment from a smartphone.

Performer: Hanzin Andrii.

Supervisor: sr. lecturer, Mykola Bidnyi.

Standard controller: _________ ___________________

Kyiv – 2023

NATIONAL AVIATION UNIVERSITY

Educational and scientific institute ________________________________.

Department of Electronics, Robotics, Monitoring and IoT Technologies.

Specialty 171 «Electronics».

APPROVE

Head of the graduate department

__________ (V. Shutko)

«____» __________ 20__.

OBJECTIVE

to defense a diploma work

Hanzin Andrii

1. Subject «Temperature control in the apartment from a smartphone» approved by the order of the rector dated 14.12.2017 № 594/st.

2. Deadline: from ________ to ________

3. Output data: temperature measurement data, microcontrollers.

4. Content of the explanatory note: analytical review of temperature measurement systems. Comparison of the relevance of temperature control systems. Comparison of microcontrollers. Modeling of an electronic temperature control system in the apartment from a smartphone.

5. List of mandatory illustrative material: pictures, tables, code.

6. Calendar plan-schedule

№	Subject	Deadline	Supervisor Signature
1.	Processing of materials for the topic of the diploma thesis. Internet resources, textbooks.	24.02.2023 – 03.03.2023	
2.	Review of existing monitoring systems, algorithms for the Internet of Things, and automation systems.	03.03.2023 – 17.03.2023	
3.	Analysis of microcontrollers. Selection of the main components.	17.03.2023 – 31.03.2023	
4.	Modulation and development of the user-system interaction block diagram.	31.03.2023 – 14.04.2023	
5.	Creation of a user interface.	14.04.2023 – 21.04.2023	
6.	Writing the source code of the mobile application program.	21.04.2023 – 20.05.2023	
7.	Formatting the explanatory note.	20.05.2023 – 31.5.2023	

7. Consultation on specific section(s):

Section title	Consultant (position, full name)	Date, signature	
		Assigner	Accepter

8. Issue assignment date: «____» _________ 20 __ р.

Diploma supervisor (project): ________ _______________

The task was accepted: _____________ Hanzin Andrii

Abstract

Explanatory note to the diploma " Temperature control in the apartment from a smartphone ": 24 pic., 2 figures, 6 tables, 15 references.

Object of study: Microcontroller-based temperature control in the apartment from a smartphone.

Objective of the work: To develop an apartment temperature control from a smartphone system.

Research methods: comparative analysis, processing of literature sources, prototype modeling.

TEMPERATURE COBTROL, ELECTRONIC SYSTEM, MICROCONTROLLER, SENSORS, MOBILE APPLICATION, ARDUINO NANO, ESP8266, DHT11, ATMEGA328P

CONTENTS

Introduction 7
CHAPTER 1. OVERVIEW OF EXISTING MONITORING SYSTEMS 8
1.1 A temperature control program for an apartment 8
1.2 The principle of operation and features of the program. 8
1.3 Advantages of Automatic Temperature Control 9
1.4 Features of Smartphone Control 10
1.5 Cons of temperature control software 10
1.6 Existing Analogues 10
1.6.1 Nest Learning Thermostat 11
1.6.2 Ecobee Smart Thermostat 12
1.6.3 Honeywell Home Wi-Fi Smart Color Thermostat 14
1.6.4 Tado Smart Thermostat 15
1.7 Comparison of existing analogs with my system 16
1.8 Conclusions to Chapter 1 19
CHAPTER 2. THEORY AND REVIEW OF PROJECT IMPLEMENTATION PLATFORMS 20
2.1 Raspberry Pi 20
2.2 STM32 23
2.3 Arduino 26
2.4 The result of the comparison 29
2.5 Selection of the Arduino platform based on the ATmega328P microcontroller 29
2.6 Software (IDE) 31
2.7 Advantages and disadvantages of Arduino UNO 32
2.8 Temperature sensor DS18B20 34
2.9 Resistor for DS18B20 37
2.10 Choosing the ESP8266 WIFI Module 37
2.11 Power supply 40
2.12 Conclusions to Chapter 2 41
CHAPTER 3. DEVELOPMENT OF THE PROGRAM ALGORITHM AND PROGRAMMING 42
3.1. Block Schema 42
3.2. Design 44

3.3 Selecting an environment for creating a user interface 47
3.4. User interface design and coding 49
3.5 Conclusions to Chapter 3 64
Conclusion 65

INTRODUCTION

The modern world is constantly being influenced by the latest technologies that change our lifestyle and contribute to increased comfort. One of the most important aspects of our daily lives is controlling the temperature in our rooms, especially in our homes. In order to provide an optimal indoor climate and convenience for users, new developments and technologies are emerging that allow temperature control via smartphones.

This thesis project is dedicated to the study and development of a smartphone-based apartment temperature control system. The goal of this project is to create a convenient and effective tool for users that will allow them to monitor and regulate the temperature parameters in their living quarters using a mobile device.

The thesis project will examine the main aspects and technologies related to temperature control in residential premises, as well as research and compare existing systems that are already used to control the temperature in homes.

In addition, the thesis project will develop a smartphone-based temperature control system. Modern technologies, such as wireless communications and the Internet of Things, will be used to ensure the convenience and speed of temperature control in residential premises.

Remote monitoring systems can help businesses save time and money by reducing the need for manual temperature control by on-site personnel.

The described system for controlling the temperature in an apartment using a smartphone is a relevant and promising research topic. It opens up new opportunities for

CHAPTER 1. OVERVIEW OF EXISTING MONITORING SYSTEMS

1.1 A temperature control program for an apartment

A temperature control program for a hospital, private home, or apartment is capable of providing comfortable living conditions for residents and employees, taking into account their particular needs. In short, such an electronic system provides centralized temperature control for all existing temperature control systems in the residence and their individual components.

The foundation of the module is the software. It regulates the working parameters of the equipment to heat or cool the air in the rooms, taking into account the preferences of the homeowners. The temperature control program, optimized for an apartment or house, is capable of controlling all devices that affect the temperature, including air conditioners and split systems, warm floors, boilers, and other heating devices.

1.2 The principle of operation and features of the program.

A temperature control program is one or more devices that maintain a desired air temperature. The control unit compares the air readings in the room with the specified parameters. If there is a deviation in the readings, it sends a new command to adjust them. If the temperature in the room drops below the specified norm, the heating element will start heating the air to a comfortable temperature. Different temperature modes can be set in different rooms.

The program automatically controls the room temperature according to the set upper and lower limits. Human intervention is necessary to check the equipment's operation and set new values.

To operate the system, a heating element must be installed to heat the air to the desired temperature. Then it will be evenly distributed throughout the room perimeter. The control unit is designed with a simple scheme for dividing airflows. Temperature readings are adjusted using.

A temperature control program is a complex of devices that affect the temperature of the apartment. It can control the air conditioner, radiators, boilers, underfloor heating system, and other equipment.

The main functions of the apartment temperature control system may include:

- Air temperature control. the program can measure the temperature in the apartment and automatically adjust the operation of heating and air conditioning to provide a comfortable temperature.
- Time programming. the program can be programmed to turn on and off heating and air conditioning at specific times to save energy and provide comfortable conditions according to the residents' schedule.
- Remote control. the program can be connected to the internet, allowing users to control the system's operation using a mobile application on a smartphone or tablet.
- Touch control. some systems may have a touch display or remote display, allowing users to see real-time information about the temperature and other parameters and control the system's operation.

1.3 Advantages of Automatic Temperature Control

- Comfort. Automatic temperature control allows for comfortable living conditions in the home. The temperature control and monitoring program can also provide comfortable conditions for people with specific needs. For example, people with certain illnesses who require constant temperature control can use this program to ensure stable temperature in the room.
- Energy savings. An automatic temperature control program can save on energy costs by regulating the temperature in the home, optimizing the heating system based on the number of people in the house and the time of day.
- Increased safety. Some automatic temperature control systems can be linked to security systems, such as smoke detectors, which can automatically turn off the heating system in case of a fire.
- Convenience. With automatic temperature control, you can set the system to automatic mode and forget about it.

- Remote control options. The program can be installed on a smartphone for automatic temperature control and remote management. This allows you to adjust the temperature in the apartment from anywhere and at any time.

1.4 Features of Smartphone Control

The temperature control program can be integrated into the "Smart Home" system and controlled remotely. To do this, you need to install special software on your mobile phone or tablet that allows you to monitor the temperature in different parts of the apartment and turn equipment on/off from anywhere. The main requirement is the presence of the Internet in the apartment and on the mobile device. The program transmits data over the Internet to the device, even if it is not connected to the home network. In addition, if the program cannot maintain the specified parameters or if there is a equipment failure, a notification will be sent to the mobile phone. The "Smart Home" system also provides a high level of safety, as the risk of overheating individual elements is minimal. If the system experiences a heavy load, it will automatically turn off the device.

1.5 Cons of temperature control software

Temperature control software has its drawbacks, which make their operation and installation more complicated. It requires time to learn the instructions and software. Additionally, installing and setting up temperature control software can be quite expensive, especially if specialized devices and equipment are used. Temperature control software may only be limited to controlling the temperature in a room and may not provide other important functions such as ventilation, air purification, and humidity control.

1.6 Existing Analogues

Analogies are a powerful tool to help us understand complex concepts and processes by comparing them to more familiar things or phenomena. In this section, I will look at various analogs and their functions to help you understand the advantages and disadvantages of existing systems. They can help increase understanding, avoid mistakes, and build new knowledge. I will compare the main functions and initial parameters with my system and provide its advantages compared to others.

1.6.1 Nest Learning Thermostat

The Nest Learning Thermostat is a home thermostat that can be controlled by your smartphone. This thermostat learns your life schedule and adjusts the room temperature according to your habits. The Nest Learning Thermostat has a very stylish and modern design with a round shape and smooth surface, giving it the look of true modern technology

Pic. 1.6.1 Nest Learning Thermostat

The main function of the thermostat is to automatically adjust the temperature depending on the presence of people in the room and the outside weather conditions. In addition, the Nest Learning Thermostat has an Airwave feature that can reduce energy costs by reducing the use of air conditioning during high humidity.

Another important feature of this thermostat is its remote control via smartphone. You can control the thermostat even if you're not at home. The Nest Learning Thermostat is also compatible with various smart device manufacturers, such as Amazon Alexa, Google Assistant, and others.

The main benefits of the Nest Learning Thermostat include automatic temperature control based on your schedule, Airwave function to reduce energy costs, and the ability to control remotely with your smartphone. However, it's important to note that the Nest Learning Thermostat may be on the pricey side compared to other thermostats, and that may not be enough for some users.

1.6.2 Ecobee Smart Thermostat

The Ecobee Smart Thermostat is another smart thermostat that can be controlled by a smartphone. It allows you to customize the heating and cooling schedule in your home to save money on energy consumption. The Ecobee Smart Thermostat also has built-in sensors that measure humidity and motion sensors, allowing the device to adjust the temperature in your home even more efficiently.

Pic. 1.6.2.1 Ecobee Smart Thermostat

One of the main features of the Ecobee Smart Thermostat is the ability to connect to various smart home devices through support for Wi-Fi, Bluetooth, Zigbee, etc. This allows you to control the thermostat from anywhere with internet access. Ecobee Smart Thermostat also has a built-in Amazon Alexa voice assistant that allows you to control the device by voice.

Another of the main advantages of Ecobee Smart Thermostat is the ability to customize the temperature in individual areas of the house. This is achieved with the help of additional temperature sensors that can be placed in different rooms. Thus, the thermostat will be able to control the temperature in each room separately, which provides a comfortable living environment.

However, the Ecobee Smart Thermostat has a relatively high price compared to other thermostats on the market, which may be out of reach for many users. It can also be difficult to set up additional temperature sensors to control the temperature in separate areas of the house, which requires some knowledge and experience.

1.6.3 Honeywell Home Wi-Fi Smart Color Thermostat

The Honeywell Home Wi-Fi Smart Color Thermostat is another fairly popular thermostat with a smartphone app for control. It has a high-definition color display and the ability to change the temperature shown on the display.

Pic. 1.6.3.1 Honeywell Home Wi-Fi Smart Color Thermostat

This thermostat has Wi-Fi support, which allows you to connect it to the app on your smartphone and control it from anywhere. In addition, it supports voice control, which allows you to turn the thermostat on and off by voice.

A special feature of Honeywell Home Wi-Fi Smart Color Thermostat is its ability to determine the optimal humidity level in the apartment. This can be useful if you have health problems associated with dry air.

In addition, the thermostat can automatically switch from heating to cooling when the temperature changes. It also has the ability to be programmed to set it to run on a schedule.

While the Honeywell Home Wi-Fi Smart Color Thermostat has many features and capabilities, it can be a little more complicated to use than some of the other thermostats on this list. It can also be a bit more expensive than some of the more basic models. However, its rich features and humidity tracking capabilities make it a good choice for those who want extra control over the conditions in their home.

1.6.4 Tado Smart Thermostat

The Tado Smart Thermostat is another very popular thermostat that can be controlled by a smartphone. It has the ability to automatically adjust the temperature in the room depending on the weather conditions outside, and it also has the option of working in automatic mode, where it determines when the heating should be turned on and off.

Pic. 1.6.4 Tado Smart Thermostat

The thermostat can control both heating and air conditioning, allowing you to adjust the temperature at any time, even when you're not at home. Thanks to the geolocation feature, the thermostat can automatically turn on when you return home and also turn off when you leave.

In addition, the Tado Smart Thermostat has an indoor air quality tracking function that allows you to control humidity and other parameters. Additionally, the thermostat has the ability to work with windows, which allows you to automatically turn off the heating when the window is open.

One of the main advantages of the Tado Smart Thermostat is its ability to use smart technology to maximize energy efficiency, which allows you to save on heating. In addition, its simple interface makes it easy to control the temperature anytime and anywhere.

However, the Tado Smart Thermostat can be a bit expensive compared to other thermostats that have similar features. Also, additional features such as air quality tracking and window operation may not be necessary for some users.

1.7 Comparison of existing analogs with my system

After researching several temperature control system analogs, I can conclude that each of them has its own advantages and disadvantages compared to my system.

For example, the Nest Learning Thermostat stands out from other analogs because of its ability to learn and memorize user habits, as well as its ability to automatically turn off when no one is home. It also has an attractive design and is quite energy efficient. However, it can be considered quite expensive and may not work with some heating systems.

The Ecobee Smart Thermostat, on the other hand, offers integration with voice assistants and has a fairly large and user-friendly display that allows the user to control the temperature with ease. It also offers support for multi-zone control, allowing different areas of the home to have different settings. The disadvantage may be the unpleasant design and relatively high price.

The Honeywell Home Wi-Fi Smart Color Thermostat can be considered one of the most affordable options among its peers. It has quite extensive customization options and can be connected to remote control from a smartphone. Its design can attract users with its ability to change the background colors of the display. However, it can be somewhat difficult to use and has limited multi-zone control capabilities.

Finally, the Tado Smart Thermostat can be considered one of the most "smart" of the bunch, as it uses intelligent algorithms to learn user habits and optimize the heating system's performance. However, unlike my system, Tado requires a subscription to their service to use some of the features, which can be an additional burden on the user's budget.

In general, my system has several advantages over its counterparts. First of all, it is a low price, which makes it more accessible to a wide range of users. In addition, it has a simple and intuitive interface, which makes it very easy to use. My system also has the ability to manually control the temperature, which allows users to quickly change the temperature without having to use additional features.

In addition, my system has the ability to connect to any temperature sensors and heating control devices via WiFi, which allows for maximum usability. Also, the system automatically adjusts the temperature within the user-defined parameters, which reduces energy consumption and reduces heating bills.

So, my system provides a simple and convenient way to control the temperature in an apartment, which allows users to save time and money. It is an effective and affordable solution to control the temperature in apartment.

In order to compare the above options, a table will be created. In it, we will list the main parameters of the systems. To do this, let's build Table 1.

Table 1 - Comparison of systems

	Price	Complexity of use	Energy efficiency	Ease of implementation	Reliability	
My system	1	1	1	1	1	5
Nest Learning	0	1	1	1`	1	4

Thermostat						
Ecobee Smart Thermostat	0	0	1	0	1	2
Honeywell Home Wi-Fi Smart Color Thermostat	0	1	1	0	1	3
Tado Smart Thermostat	0	0	1	0	1	2

Let's calculate the generalized quality factor, which is calculated using the following formula:

$$K = \frac{\mathrm{Ei}}{\mathrm{En}}$$

A generalized criterion for the quality of the first system:

$$K = \frac{5}{5} = 1$$

A generalized criterion for the quality of the second system:

$$K = \frac{4}{5} = 0.8$$

A generalized criterion for the quality of the third system:

$$K = \frac{2}{5} = 0.4$$

Generalized quality criterion of the fourth system:

$$K = \frac{3}{5} = 0.6$$

Generalized quality criterion of the fifth system:

$$K = \frac{2}{5} = 0.4$$

As you can see, the quality criterion for the first system is higher than for other analog systems. Based on these calculations, we can conclude that my circuit has all the necessary qualities for the task we have set. Let's use these parameters to build an electrical schematic diagram of the system under development.

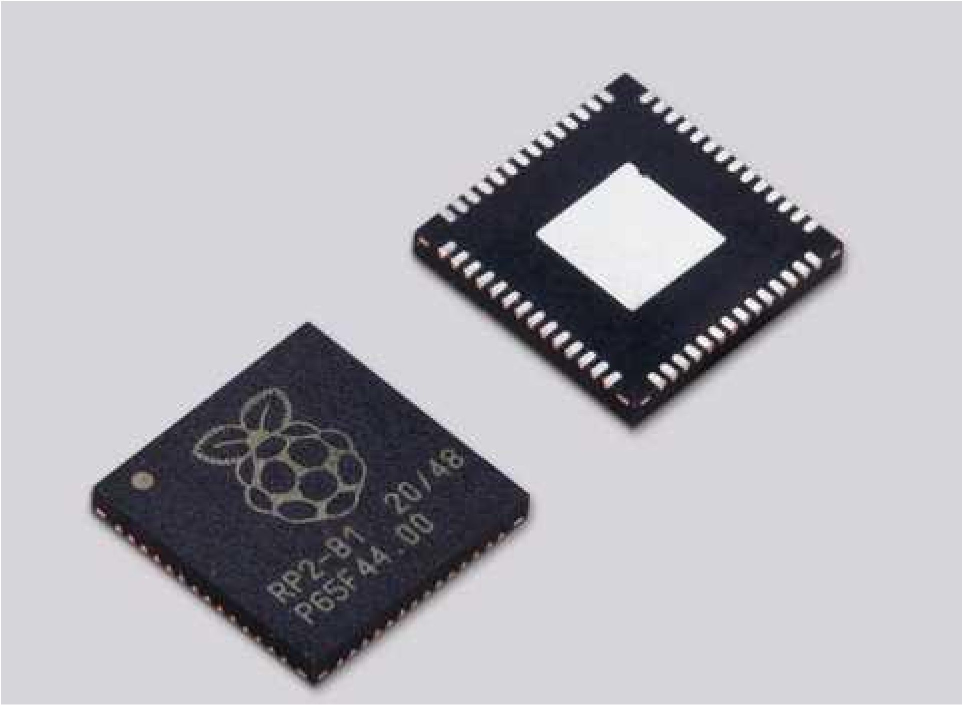
1.8 Conclusions to Chapter 1

In Chapter 1, an overview of the state of the problem was made on the basis of literature on the topic of the thesis. In this thesis, temperature control system was designed. climate control system. The problems and disadvantages of various systems that can control the temperature in an apartment were discussed.

CHAPTER 2. THEORY AND REVIEW OF PROJECT IMPLEMENTATION PLATFORMS

2.1 Raspberry Pi

Raspberry Pi is a single-board computer that has garnered widespread acclaim for its impact on programming and electronic device development. It is a compact board, resembling the size of a credit card, that boasts an ARM processor, RAM, and an array of ports for seamless peripheral connectivity.



Pic. 2.1.1 Raspberry Pi

Originally introduced by the Raspberry Pi Foundation in the United Kingdom, Raspberry Pi has gained immense popularity since its inception in 2012. It supports various operating systems, such as Raspbian (a Linux-based system) and Windows 10 IoT Core, catering to a diverse range of user preferences and project requirements.

One notable aspect of Raspberry Pi is its expandability and compatibility with a plethora of peripheral devices. With USB, HDMI, Ethernet, and audio ports, users can effortlessly integrate a multitude of accessories into their setups, whether they're building a media center, gaming console, or smart home system. This microcontroller is actively used in Internet of Things (IoT), robotics, web server and hosting, process automation and much more.

Because of its ease of use and accessibility, the Raspberry Pi has attracted significant attention in education and in the DIY community. He is the perfect platform for nurturing curiosity, empowering learners of all ages to explore technology, electronics, and programming, and fostering the development of valuable skills.

That said, compared to Arduino and STM32, the Raspberry Pi has higher power consumption, making it less suitable for battery-powered applications. The Raspberry Pi does not have the same accuracy and low latency as the STM32, which can be important in mission-critical systems.

The Raspberry Pi is a powerful and flexible development platform that can be used for a variety of projects. Its compact form, affordability, and boundless possibilities make it an invaluable asset for beginners, enthusiasts, and educators seeking to unlock their potential in these dynamic fields. The advantages are high performance, versatility, and a large community of developers. Disadvantages include higher power consumption, lower accuracy, and lower latency compared to other platforms such as the STM32.

Table 2 - Comparison of the main models of Raspberry Pi microcontrollers

Microcontroller Model	**Raspberry Pi Pico**	**Compute Module 4**	**Raspberry Pi Zero**	**Raspberry Pi 3 Model A+**	**Raspberry Pi 2 Model B**
Processor	RP2040	Broadcom BCM2711	BCM2835	BCM2837	BCM2836

Number of Cores	2	4	1	4	4
Processor Frequency	133 MHz	1.5 GHz	1 GHz	1.4 GHz	900 MHz
Number of GPIO Pins	26	244	40	40	40
Ethernet	No	Yes	No	Yes	Yes
Wi-Fi	No	Yes	No	Yes	No
Bluetooth	No	Yes	No	Yes	No
USB	No	2x USB 2.0	1x micro-USB	1x USB 2.0	4x USB 2.0
HDMI	No	Yes	No	No	1x HDMI

The Raspberry Pi Pico is a new microcontroller that stands out for its compact size, low cost, and use of the RP2040 processor. It has 26 GPIO pins and supports many interfaces such as UART, I2C and SPI. However, the Raspberry Pi Pico lacks built-in Ethernet, Wi-Fi, and Bluetooth, which limits its wireless connectivity.

Compute Module 4 is designed for use in industrial applications and has a significant number of GPIO pins - 244. It is based on Broadcom's BCM2711 processor and supports Ethernet, Wi-Fi and Bluetooth. The Compute Module 4 also has two USB ports, providing additional connectivity for external devices. However, because of its focus on industrial use, it can be more difficult to use for casual enthusiast projects and requires additional hardware to connect and program it.

The Raspberry Pi Zero is the least powerful and affordable model, which is suitable for projects with limited resources. It has 40 GPIO pins and a BCM2835 processor. However, the Raspberry Pi Zero does not have Ethernet, Wi-Fi, or Bluetooth capabilities, which may limit its use for wireless projects or networking applications.

The Raspberry Pi 3 Model A+ and Raspberry Pi 3 Model B+ are both based on the BCM2837 processor and offer improved performance over previous models. They both have 40 GPIO pins and support Ethernet, Wi-Fi, and Bluetooth for more networking and wireless connectivity. The Raspberry Pi 3 Model B+ also has additional USB ports, making it easier to connect external devices. However, the Raspberry Pi 3 Model A+ and Raspberry Pi 3 Model B+ may have a higher cost than other models.

The Raspberry Pi 2 Model B is also based on the BCM2836 processor and has 40 GPIO pins. It offers good performance and some connectivity such as Ethernet and four USB ports. However, unlike newer models, the Raspberry Pi 2 Model B lacks built-in Wi-Fi and Bluetooth, which can be a drawback in certain projects where wireless connectivity is required.

2.2 STM32

STM32 is a family of microcontrollers manufactured by STMicroelectronics. These microcontrollers boast a diverse set of features and capabilities to satisfy a wide range of applications. They are powerful and flexible development platforms with high performance, low latency and high accuracy, making them the ideal choice for mission-critical systems.

Pic. 2.2 STM32

STM32 uses the ARM Cortex-M core, which provides fast and efficient task execution. They have many inputs/outputs, built-in modules for working with various peripheral devices, as well as different memory and speed options.

One of the main advantages of STM32 is its performance and low latency. It can process data faster than Raspberry Pi and Arduino, making it ideal for use in critical systems where quick response to external events is critical.

However, STM32 can be more complex to use than Arduino, as it requires deeper knowledge of programming and electronics. It can also be more expensive to use than Arduino, as it requires additional equipment for programming and debugging.

Overall, STM32 is a powerful and flexible development platform with high performance, low latency, and high accuracy. It can be used to create critical systems where quick response to external events is critical. However, it can be more complex to use and more expensive to use than Arduino. The STM32 family of microcontrollers includes many different models, each with its own technical specifications.

The STM32 family of microcontrollers includes many different models, each with different specifications. Here are the general characteristics of the STM32:

Table 3 - Comparison of the main models of STM32 microcontrollers

Model	**STM32F0**	**STM32F1**	**STM32F4**	**STM32F7**	**STM32L4**
Core	Cortex-M0	Cortex-M3	Cortex-M4F	Cortex-M7	Cortex-M4F
Processor Speed	Up to 48 MHz	Up to 72 MHz	Up to 180 MHz	Up to 216 MHz	Up to 120 MHz
Flash Memory	Up to 256 KB	Up to 1 MB	Up to 2 MB	Up to 2 MB	Up to 2 MB
RAM	Up to 32 KB	Up to 96 KB	Up to 192 KB	Up to 512 KB	Up to 640 KB
GPIO Pins	16-128	20-144	32-176	100-216	32-144
Analog Inputs	Yes	Yes	Yes	Yes	Yes
Communication	I2C, SPI, UART	I2C, SPI, UART	I2C, SPI, UART	I2C, SPI, UART	I2C, SPI, UART
Ethernet	No	No	Up to 1x Ethernet	Up to 1x Ethernet	No
USB	Up to 2x USB 2.0	Up to 2x USB 2.0	Up to 3x USB 2.0	Up to 2x USB 2.0, 1x USB HS	Up to 2x USB 2.0
CAN	No	Up to 2	Up to 2	Up to 2	Up to 2
I2C	Up to 1	Up to 2	Up to 3	Up to 4	Up to 3

The STM32F0 are low-cost microcontrollers based on the Cortex-M0 core. They offer low cost and sufficient performance for simple applications. However, the Flash and

RAM capacity is limited and the functionality of the I/O ports (GPIOs) and communication buses is limited compared to other models.

STM32F1 are mid-level models based on the Cortex-M3 core. They offer more Flash and RAM, as well as a wider range of GPIO and communication bus functionality. However, their performance is limited to 72 MHz, which may not be enough for some applications with high performance requirements.

STM32F4 are high performance microcontrollers based on the Cortex-M4F core. They offer high processor frequency, large Flash and RAM capacities, and rich GPIO and communication bus functionality. The STM32F4 also supports an Ethernet interface for network connectivity. However, their cost may be higher and power consumption may be higher compared to some other models.

The STM32F7 are high performance microcontrollers based on the Cortex-M7 core. They offer even higher processor frequencies, more Flash and RAM, and an expanded set of GPIO and communication bus functionality. The STM32F7 also supports an Ethernet interface and has multiple USB ports. However, their cost can be higher and power consumption can be higher, making them less suitable for resource-constrained applications.

STM32L4s are low-power microcontrollers based on the Cortex-M4F core. They offer good performance while consuming less power. They have large Flash and RAM capacities and a variety of GPIO and communication bus functionality. The STM32L4 also supports analog inputs and has a fair number of USB ports. However, their processor frequency is limited to 120 MHz, which may not be sufficient for some high-performance applications.

2.3 Arduino

Atmel AVR and ARM microcontrollers provide the basis for hardware and software projects developed on an open platform known as Arduino. Different tasks can be accomplished using the various models and variants of Arduino.

Easy programming and usage are key benefits of using an Arduino. Novice users can create simple projects within minutes using Arduino’s simple and intuitive programming language. Furthermore, a substantial community of both users and developers share experiences while providing assistance to other users.

Using Arduino is also very cost-effective and inexpensive. It can be purchased at a very low cost without needing extra hardware for programming and debugging. An ideal choice for projects with limited funds.

Nevertheless, the Arduino lacks the advanced capabilities and versatility of the Raspberry Pi or STM32. Data processing at the same speed is not possible for it and it is equipped with lesser built-in modules to operate various peripherals.

As a whole, the Arduino is a superb selection for amateur developers and for undertakings with restricted finances. It's not hard to utilize and it's reasonably priced. Simple projects not requiring fast data processing can also be created using it.

Table 4 - Comparison of the main models of Arduino microcontrollers

Microcontroller	**Arduino Uno**	**Arduino Mega**	**Arduino Leonardo**	**Arduino Due**	**Arduino Nano**
Clock Speed	16 MHz	16 MHz	16 MHz	84 MHz	16 MHz
Flash Memory	32 KB	256 KB	32 KB	512 KB	32 KB
SRAM	2 KB	8 KB	2.5 KB	96 KB	2 KB
EEPROM	1 KB	4 KB	1 KB	N/A	1 KB
Digital I/O Pins	14	54	20	54	14
Analog Input Pins	6	16	12	12	8
UART	1	4	1	4	1
SPI	1	1	1	1	1
I2C	1	1	1	2	1
USB	Yes	Yes	Yes	Native	Yes

Arduino Uno is one of the most popular Arduino models. It is equipped with the ATmega328P microcontroller, which runs at 16 MHz. Arduino Uno has enough flash memory (32 KB) that you can load and run small programs. It also has 14 digital inputs/outputs, 6 analog inputs and supports basic interfaces such as UART, SPI and I2C. The Arduino Uno is easy to use, affordable and well suited for novice users. However, its limited resources may be a problem for more complex projects that require more memory or more I/O.

The Arduino Mega is equipped with the ATmega2560 microcontroller and has significantly more resources than the Arduino Uno. With its 256 KB flash memory, 54 digital I/O and 16 analog inputs, the Arduino Mega provides more space for developing complex projects. It also supports basic communication interfaces such as UART, SPI and I2C. However, the Arduino Mega is larger and more expensive than the Arduino Uno, which can be a disadvantage for some projects.

The Arduino Leonardo uses the ATmega32u4 microcontroller, which has built-in USB support. This makes it easy to use for developing projects that require a direct connection to a computer. The Arduino Leonardo also has 20 digital I/O and 12 analog inputs, which gives enough flexibility for most projects. However, the Arduino Leonardo has less flash memory and SRAM compared to the Arduino Mega, which can limit the possibilities for complex programs.

The Arduino Due has a 32-bit SAM3X8E microcontroller based on the ARM Cortex-M3 core, making it more powerful and faster than previous models. With a clock speed of 84 MHz, 512 KB of flash memory, and 96 KB of SRAM, the Arduino Due provides great resources for developing complex projects. It also supports a wider range of communication interfaces, including UART, SPI, I2C, USB, and Ethernet. Arduino Due is ideal for projects that require high performance, such as robotics or signal processing. However, keep in mind that Arduino Due has a higher cost compared to simpler Arduino models, and its use may require more advanced programming knowledge.

Arduino Nano is a compact and budget-friendly Arduino model. It uses the ATmega328P microcontroller, similar to the Arduino Uno, and has a basic set of features. The Arduino Nano has 32KB of flash memory, 2KB of SRAM, and a limited number of digital and analog I/Os. However, because of its compact size, the Arduino Nano is ideal for space-saving projects. It is also more affordable in terms of price, making it popular with novice developers and students.

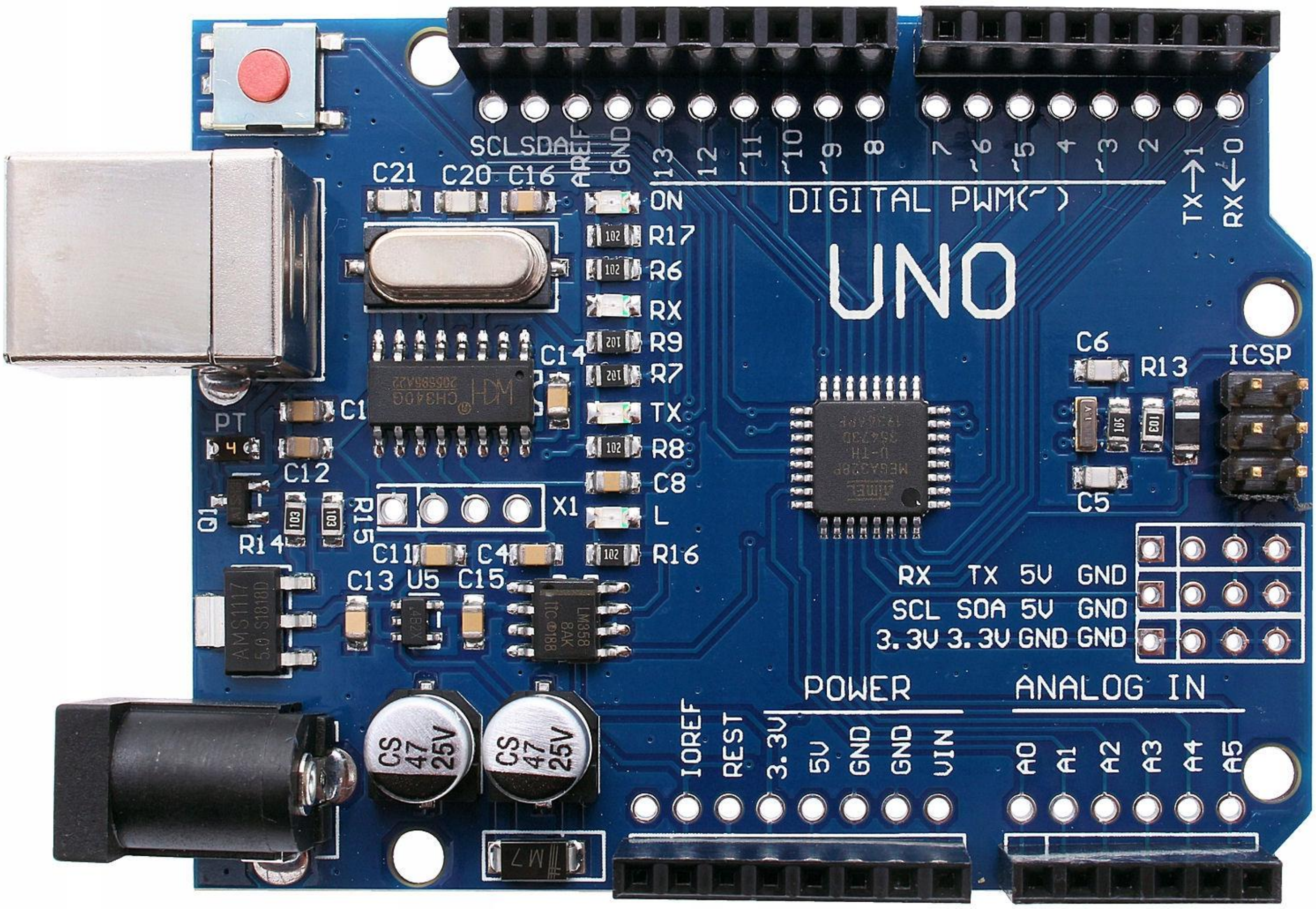
2.4 The result of the comparison

The choice between these platforms depends on the specific project and its requirements. If the project requires running an operating system, networking, and higher processing power, then the Raspberry Pi may be the best choice. If the project requires high performance, lots of peripherals, and low power consumption, the STM32 may be a better choice. If the project requires a simple and affordable platform with which to get started quickly, the Arduino may be a better choice. However, Arduino has a simple programming language, extensive community support, availability, and a wide variety of peripherals, making it a great choice for beginners and for developing simple electronic devices.

2.5 Selection of the Arduino platform based on the ATmega328P microcontroller

The Arduino UNO platform, based on the ATmega328P microcontroller, was chosen for the project implementation.

Arduino UNO is one of the most popular Arduino platforms for developing electronic devices. Thanks to its wide support and active developer community, you can find many ready-made libraries and programs for Arduino UNO that help to expand its capabilities. It has an easy-to-use design and an affordable price, making it the perfect choice for a project like my system to control the temperature in apartment.



Pic. 2.5.1 Arduino UNO

Arduino can interact with physical devices through various types of sensors and input/output devices such as buttons, temperature sensors, gyroscopes, accelerometers, displays, motors, and more. Devices can be connected to the Arduino board using different interfaces such as digital and analog inputs/outputs, communication buses (such as I2C, SPI, UART), external interrupts, and more.

It is easy to use, has a user-friendly IDE, and is cost-effective. In addition, Arduino has a large community of users that provides extensive documentation, examples, and libraries. A wide range of boards and models are available with different characteristics and functionalities, allowing you to choose the most suitable board for your project. Additionally, Arduino is based on open-source code, which allows users to modify and improve the code, create their own libraries, and applications.

Technical characteristics of Arduino UNO:

- Microcontroller: ATmega328P

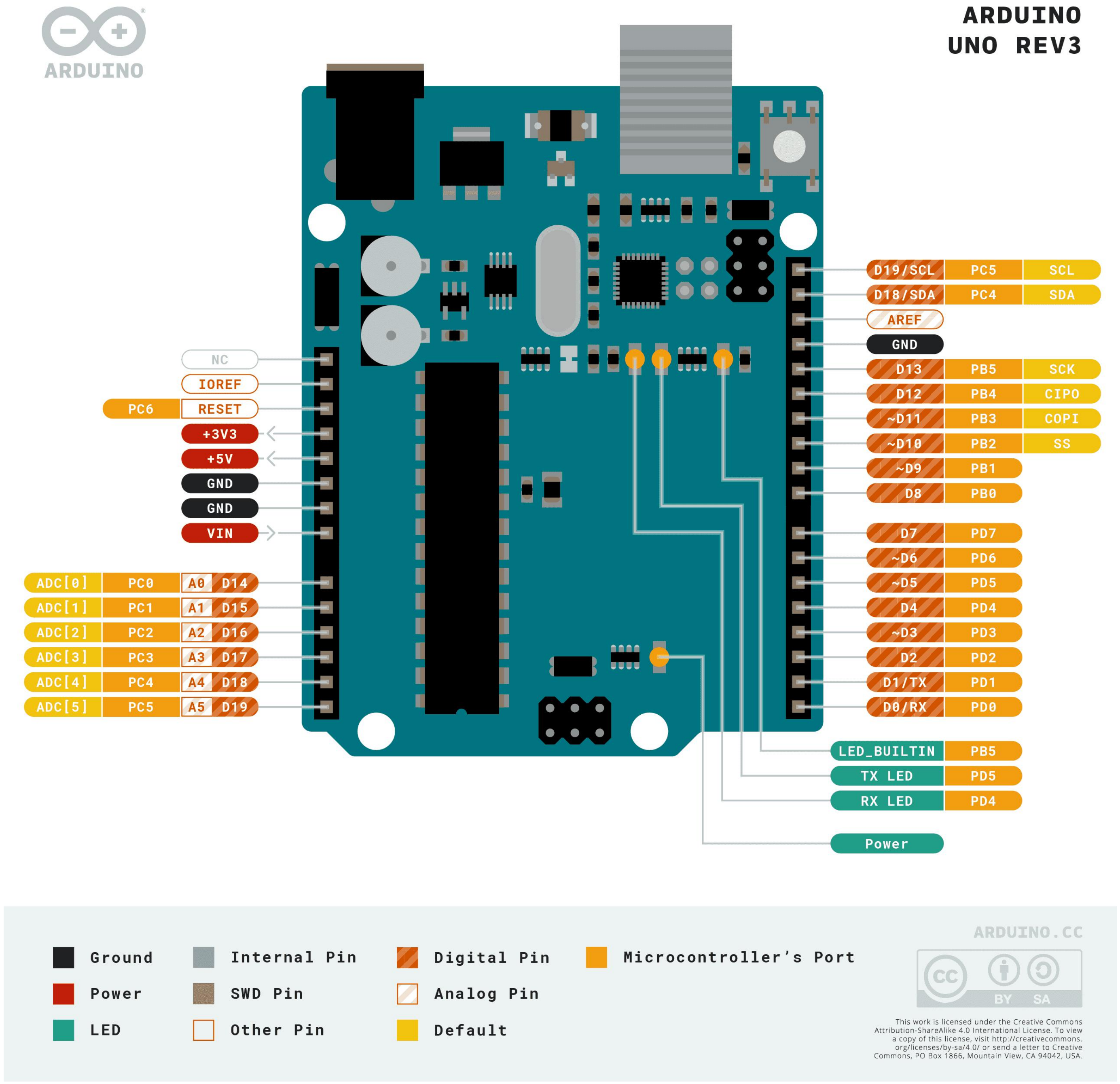
- Processor Frequency: 16 MHz
- Power supply voltage: 5V
- Input voltage (recommended): 7-12 B
- Digital inputs/outputs: 14 (of which 6 can be used as PWM outputs)
- Analog inputs: 6
- USB Streaming Interface: Type B
- FLASH memory: 32 KB
- RAM (SRAM): 2 KB
- EEPROM: 1 KB

In addition, Arduino Uno has many additional features, such as a built-in voltage regulator, the ability to program through the USB port, and extensive connectivity of additional modules and sensors. All this makes it a popular choice for a wide range of projects, from simple electronic games to smart homes and robotics.

2.6 Software (IDE)

The software used to program the Arduino is represented by the Arduino IDE. The IDE is a Java program that runs on many different platforms, including such well-known systems as PC, Mac, and Linux. It was developed for beginners who are not familiar with all the intricacies of programming. It includes an editor, compiler, and loader. The IDE provides code libraries for using peripherals, serial ports and various types of screens. Programs for Arduino are called "sketches".

Most Arduino boards are connected to a computer using a 27 USB cable. This connection allows you to download sketches to your Arduino board.



Pic. 2.6.1 Arduino UNO PINOUT

2.7 Advantages and disadvantages of Arduino UNO

Advantages:

- Ease of use: The Arduino Uno stands out for its simplicity and user-friendly features. Its interface and programming language are designed to be easily understood, even by individuals with no prior experience in microcontroller programming.

Furthermore, a vibrant community of developers constantly creates libraries and development tools to facilitate its usage.

- Price: The Arduino Uno offers a cost-effective solution, making it accessible to a wide range of users. With its competitive pricing, it ranks among the most economical microcontrollers available, enabling individuals on a tight budget to venture into microcontroller projects without substantial financial investments.
- Expandability: The Arduino Uno boasts a vast selection of extensions and modules, rendering it suitable for a diverse range of projects. Its abundant variety of inputs and outputs allows for seamless integration with various sensors and devices, expanding its potential applications.
- Programmability: The Arduino Uno supports the widely used C and C++ programming languages, endowing it with remarkable flexibility and power. Additionally, it embraces an array of libraries and development tools, streamlining the process of creating programs with minimal complexity.
- Reliability: The Arduino Uno demonstrates exceptional reliability and stability during operation. It incorporates built-in protection mechanisms against undesirable events such as short circuits, ensuring safe usage across numerous scenarios

Disadvantages:

- Limited Resources: The Arduino Uno has limited resources, such as RAM and EEPROM. This can be a problem for more complex projects that require more memory.
- Low performance: Arduino Uno has low performance compared to more powerful microcontrollers. This can limit the capabilities of projects that require higher performance.
- Limited support: Although Arduino is a popular platform, it has some support limitations. Specifically, its documentation does not always contain all the information you need, and sometimes it can take you a long time to find the answer to your question. Also, Arduino is focused on beginners and intermediate users, so

if you have a project that requires a higher level of professionalism, you may need to use a different platform.

- Difficulty building large projects: Using Arduino for large and complex projects can require a lot of extra effort to assemble and control components. In order to use Arduino for large projects, you must be prepared for the fact that you may need to write your own code as well as purchase additional hardware.
- Low speed: The Arduino is not a fast platform. Processing speed is limited by processor frequency and other characteristics of the platform. This means that if you need to process large amounts of data or work with fast-changing conditions, you may need to use a more powerful platform.
- Limited memory: The Arduino UNO has only 32 KB flash memory and 2 KB RAM. This can limit your programming and project design capabilities. If you need more memory for your project, you may need to use a different platform with higher specifications.

2.8 Temperature sensor DS18B20

One of the central components of the system is the temperature sensor. To determine the most suitable option to start with, you need to compare the characteristics of the possible options and understand the advantages and disadvantages of each.

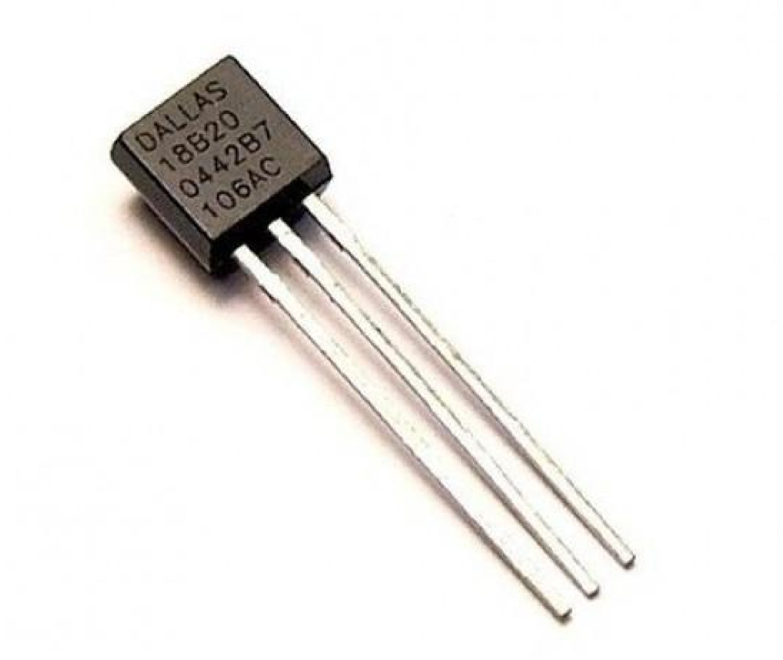
Table 5 - Comparison of the main models of Temperature Sensor

Temperature Sensor	**Interface Type**	**Measurement Accuracy**	**Measurable Temperature Range**	**Output Signal**	**Price**
DS18B20	1-Wire	±0.5°C	-55°C to +125°C	Digital	$2-5
DHT22	1-Wire	±0.5°C (temp.)	-40°C to +80°C	Digital	$5-10
LM35	Analog	±0.5°C	-55°C to +150°C	Analog	$1-2
MAX31850	1-Wire	±2°C	-55°C to +125°C	Digital	$5-

					10
PT100	Analog	Varies	-200°C to +850°C	Analog	$5-15

After comparing the characteristics of the DS18B20 counterparts, the following conclusions can be made. The DS18B20 has a temperature accuracy of ±0.5°C, the same as the DHT22 (AM2302). However, the DS18B20 has the widest temperature measurement range, from -55°C to +125°C, making it suitable for a variety of applications. The DS18B20, DHT22 (AM2302), and MAX31850 support a 1-Wire interface that is easy to use. The LM35 and PT100 use an analog interface. The DS18B20, DHT22 (AM2302), and MAX31850 provide a digital output signal, while the LM35 and PT100 provide an analog signal. The DS18B20 is priced around $2 to $5, making it the most affordable option compared to other counterparts that have higher prices starting at $5 and up to $15. Based on these factors, the DS18B20 can be considered an excellent choice among temperature sensor counterparts. It offers high measurement accuracy, a wide temperature range, a user-friendly 1-Wire interface, and affordability. Because of its advantages, the DS18B20 was chosen to implement my system

The DS18B20 temperature sensor is an inexpensive and versatile digital temperature sensor that can be used in a variety of projects, including my thesis. It is based on the OneWire protocol, which allows multiple devices to be connected to the same pin on the microcontroller, making it easy to work with the sensors in systems where temperature monitoring at different points is required.



Pic. 2.8 Temperature sensor DS18B20

The DS18B20 sensor has high temperature measurement accuracy, as well as a wide measurement range from -55°C to +125°C, making it convenient for use in various projects, such as temperature control in rooms or temperature monitoring in industrial systems.

The sensor also has a simple connection interface and does not require external calibration. It also has built-in reverse polarity and short circuit protection, ensuring the sensor's safety and protection against damage during connection.

Additionally, the DS18B20 sensor can operate in either controller pin or external power supply mode, allowing it to be used in various conditions.

In conclusion, the DS18B20 temperature sensor is a reliable and versatile tool for measuring temperature in various projects, including diploma work. It has high measurement accuracy, a simple connection interface, and does not require external calibration, making it convenient to work with.

2.9 Resistor for DS18B20

To connect the data line of the DS18B20 sensor, I selected a 4.7 kΩ resistor. Proper functionality requires a pull-up resistor on the data line. The maximum length of the single-wire bus is determined by the value of the pull-up resistor. Longer bus lengths are achievable by decreasing resistor values.

A 4.7 kΩ pull-up resistor value is commonly used with the DS18B20. The balance between bus length and power consumption is favorable. A bus length of up to 100 meters can have reliable communication with minimal power consumption ensured.

Selecting a higher resistor value will reduce power consumption but limit the maximum bus length. Selecting a lower resistor value will increase the bus length, but consumes more power and may result in signal integrity issues.

For this reason, a 4.7 kΩ resistor was selected for the DS18B20 as it provides a good balance between bus length and power consumption, making it suitable for most applications.

2.10 Choosing the ESP8266 WIFI Module

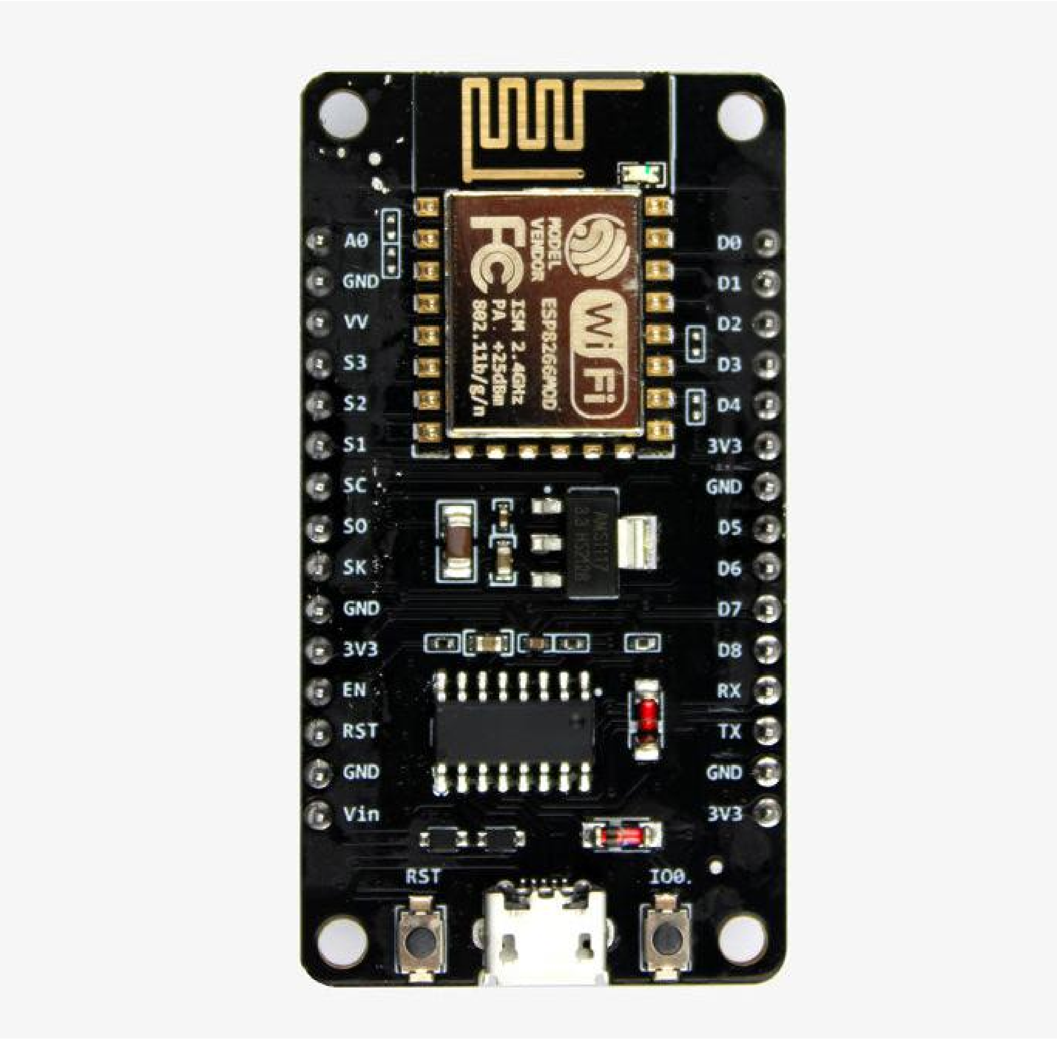
There are many different wifi modules on the market. To begin with, I decided to compare the possible options to determine the most suitable one for my system. A comparison of the characteristics is presented in table 6.

Table 6 - Comparison of the main Wi-Fi modules

Wi-Fi Module	**Chipset**	**Data Transfer Speed**	**Number of I/O Pins**	**Bluetooth Support**	**Programming Complexity**	**Price**
ESP8266	ESP8266	72 Mbps	17	No	Low	$3-5
ESP32	ESP32	150 Mbps	36	Yes	Medium	$5-10
CC3000	TI	50 Mbps	8	No	High	$20-30
RN-XV WiFly	Roving	1-2 Mbps	14	No	Low	$20-30
CC3200	TI	80 Mbps	27	Yes	Medium	$20-30

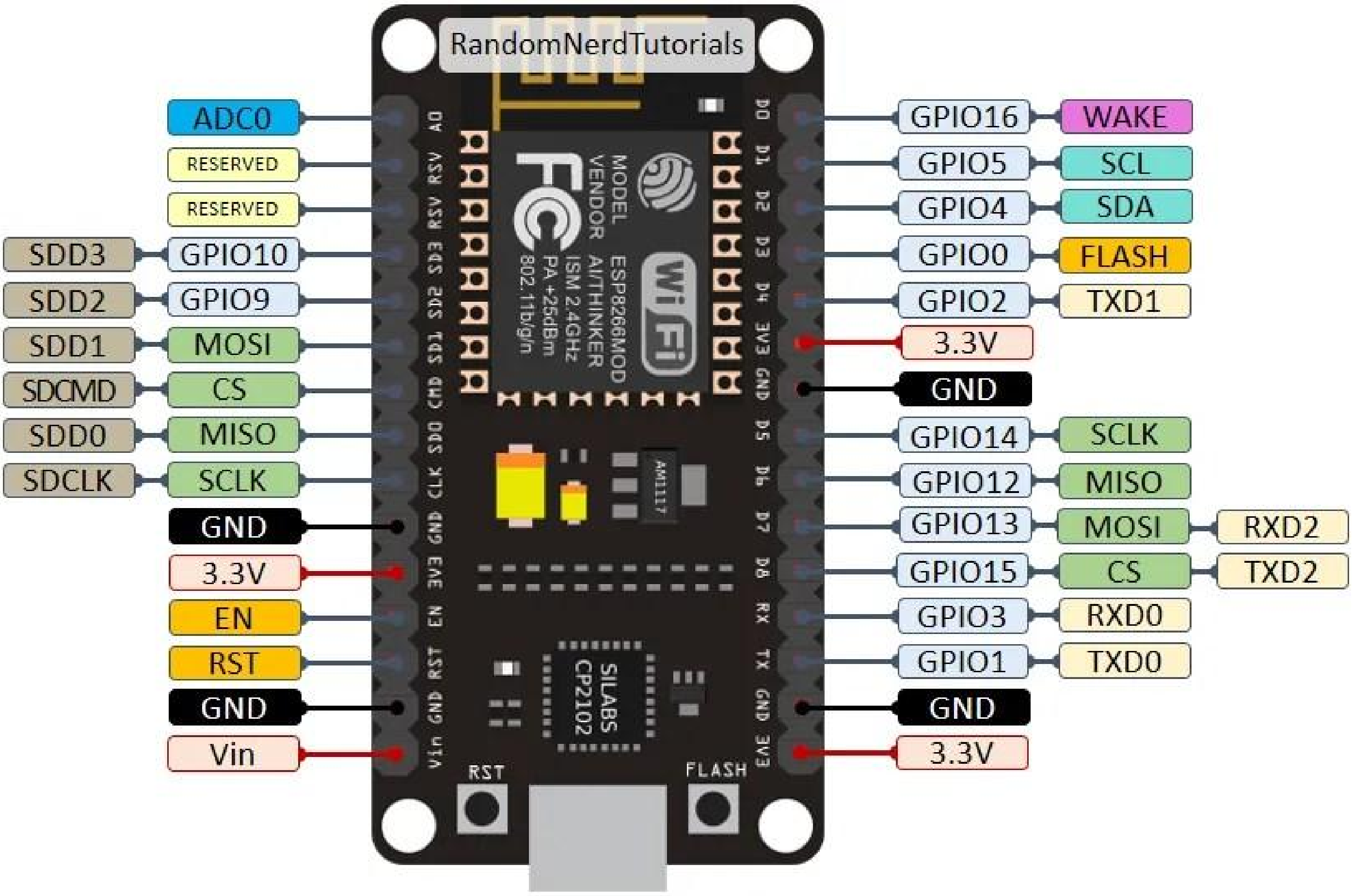
As you can see from the table, each module has its own unique features and is suitable for different types of projects. The ESP8266 and ESP32 are two of the most popular modules, offering high data rates and a large number of I/Os. The CC3000 and CC3200 have a more complex architecture and higher performance, but require more complex programming. The RN-XV WiFly is a low performance module but is easy to use. From this comparison I have chosen for my system the ESP8266.

The ESP8266 is a small and inexpensive Wi-Fi module which is widely used in various IoT applications. It was created by the Chinese company Espressif Systems and became popular because of its high performance and affordability.



Pic. 2.10.1 ESP8266 WIFI Module

The ESP8266 module offers extensive functionality beyond just Wi-Fi, encompassing GPIO pin operation, PWM, ADC, UART, I2C, and SPI. This versatility enables its utilization in diverse projects like smart homes, industrial process automation, and device monitoring and control.



Pic. 2.10.2

An outstanding benefit of the ESP8266 module is its affordability. It stands as one of the most economical Wi-Fi modules available, making it a cost-effective option for a range of student projects.

Additionally, the ESP8266 boasts a thriving developer community that offers complimentary documentation, code samples, and libraries, facilitating the development and design of various IoT applications.

The ESP8266 module does come with certain limitations. For instance, it possesses a limited number of GPIO pins and modest power capabilities. Moreover, its performance may not meet the demands of more intricate projects.

Nonetheless, when considering the bigger picture, the ESP8266 module remains an exceptional choice for students aiming to create straightforward IoT applications, encompassing tasks such as smart home implementation, device control, and monitoring of diverse parameters.

The main characteristics of the ESP8266 module are:

80 MHz operating frequency;
Tensilica L106 32-bit embedded processor;
32 Kbytes of RAM and 1 MB of flash memory;
Support for Wi-Fi 802.11 b/g/n;
Support for TCP/IP protocols;
Various interfaces including UART, I2C, SPI.

2.11 Power supply

In order to ensure the proper operation of the components in the setup, I decided to utilize a 5V DC power supply. The reason behind this choice is that the Arduino Uno can be powered either through the USB port or the DC jack, with a voltage range of 7-12V. Additionally, the ESP8266 and DS18B20 can draw power directly from the 5V pin on the Arduino.

Opting for a 5V DC power supply guarantees that all the elements within the configuration receive the necessary voltage to function optimally. It is crucial to ascertain that the power supply possesses adequate current capacity to support the smooth operation of all the components.

2.12 Conclusions to Chapter 2

In Chapter 2:

1. Compared Microcontrollers.

2. The Arduino platform based on the ATmega328P microcontroller was chosen;

3. The humidity and temperature sensor DS18B20 is selected;

4. Microcontroller ESP8266 for wireless data transmission;

5. Power supply;

The device will be placed on the Arduino UNO board consisting of a DS18B20 temperature and humidity sensor and an ESP8266 microcontroller on which the Internet of Things system and wireless access to the to the device via WiFi.

CHAPTER 3. DEVELOPMENT OF THE PROGRAM ALGORITHM AND PROGRAMMING

3.1. Block Schema

The block schema (Fig. 3.1.1) shows the block schema of the temperature control system, which includes an Arduino platform based on the ATmega328P microcontroller, a DS18B20 temperature sensor, a resistor, an ESP8266 microcontroller for wireless data transmission and a USB output on the on the Arduino board itself, to which the power supply will be connected. Creating a block schema is crucial before implementing a project, as it helps visualize the system's architecture and understand each stage involved. This visualization promotes modularization and efficient programming, streamlining the development process. The block schema provides insights into component interconnections and dependencies, allowing a comprehensive understanding of the system's functionality. Such understanding is valuable for optimizing module performance and ensuring seamless integration into the overall system.

Additionally, the block schema helps identify potential bottlenecks or areas requiring further refinement during project development. By visually analyzing the system's structure, limitations or design flaws can be pinpointed early on, enabling proactive addressing and enhancing overall performance.

To summarize, the block schema acts as a fundamental blueprint for the temperature control system, visually representing its components and interconnections. It plays a vital role in comprehending the system's architecture, facilitating modularization, programming, and optimization.

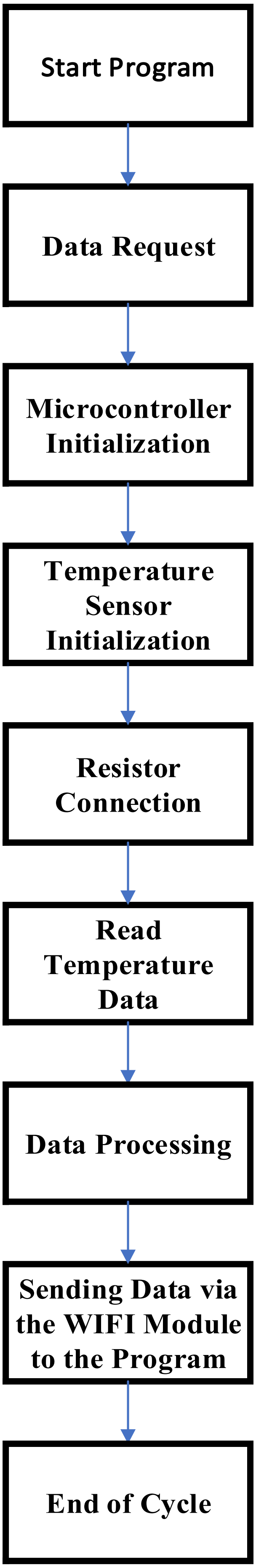

Start Program
Data Request
Microcontroller Initialization
Temperature Sensor Initialization
Resistor Connection
Read Temperature Data
Data Processing
Sending Data via the WIFI Module to the Program
End of Cycle

Figure 3.1.1. Block Schema

Scheme of the program operation:

1. Launch the application on your smartphone.
2. Data request.
3. The Arduino UNO reads the data from the temperature sensor DS18B20.
4. Arduino UNO sends the temperature data to the ESP8266 Wi-Fi module.
5. The ESP8266 Wi-Fi module sends temperature data to the mobile app.
6. The mobile app displays the current temperature and allows you to set the temperature or set a range of values for automatic temperature control.
7. If the current temperature exceeds the set limits, the mobile app sends a command to the Arduino UNO to turn the temperature control device on or off.

3.2. Design

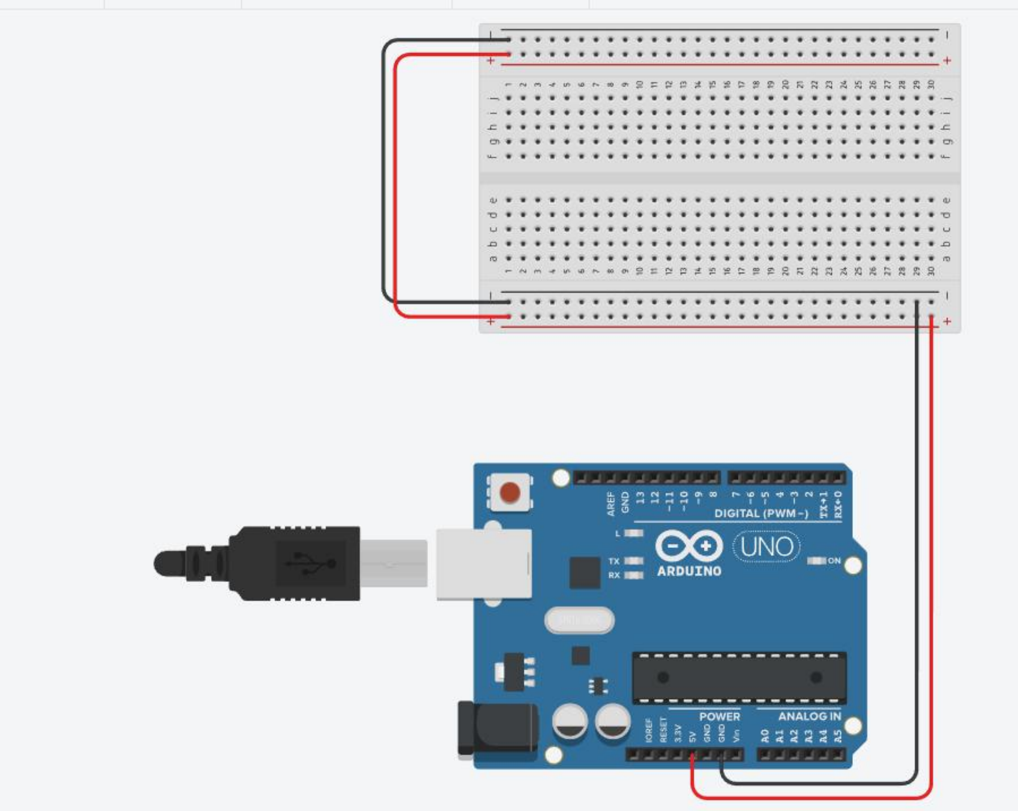
I decided to use Tinkercad to modify my circuit. It has a simple interface and extensive functionality that allows you to create complex 3D models in a short time. This platform is a free online tool, which makes it accessible to anyone, regardless of budget. In the field of 3D modeling, Tinkercad is an easy and intuitive tool for designing circuits.

Pic 3.2.1

In summary, Tinkercad is an affordable and functional tool for creating a 3D model, so it is a good option for my project.

First, I need to connect the breadboard. To ensure that all components of the circuit are powered from one source, I connected the positive (+) power rail of the breadboard to the 5V pin using a red jumper wire, and the negative (-) rail using a black jumper wire to the GND pin of the Arduino Uno board.

Thus, I have connected the power and ground of the breadboard to the Arduino Uno (Fig. 123).

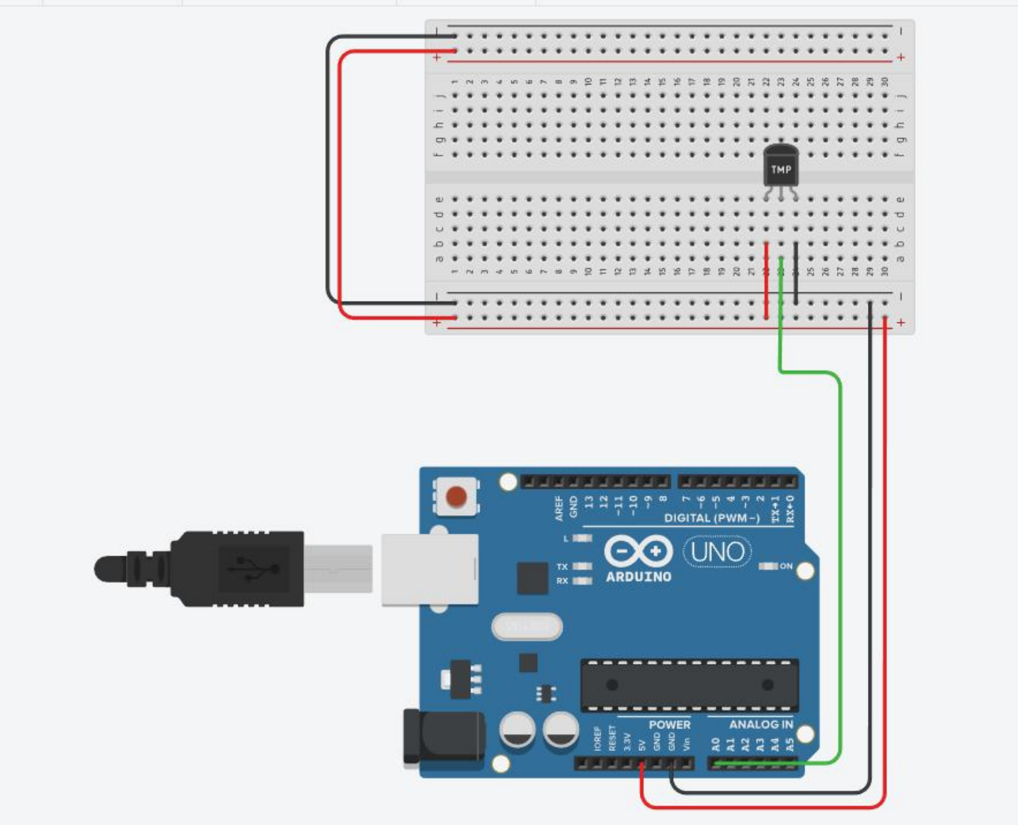


Pic 3.2.2

Next, I will connect the DS18B20 temperature sensor. This sensor has the following pins:

1. VCC, which I connected to the positive (+) rail of the breadboard using a red jumper wire.
2. GND, which I connected to the negative (-) rail of the breadboard with a black jumper wire.
3. DQ (Data input/output), which was connected to the digital output of the Arduino Uno using a green jumper wire.

Thus, the Arduino Uno will now be able to read the data from the DS18B20 sensor and get the current temperature value (Fig. 123).



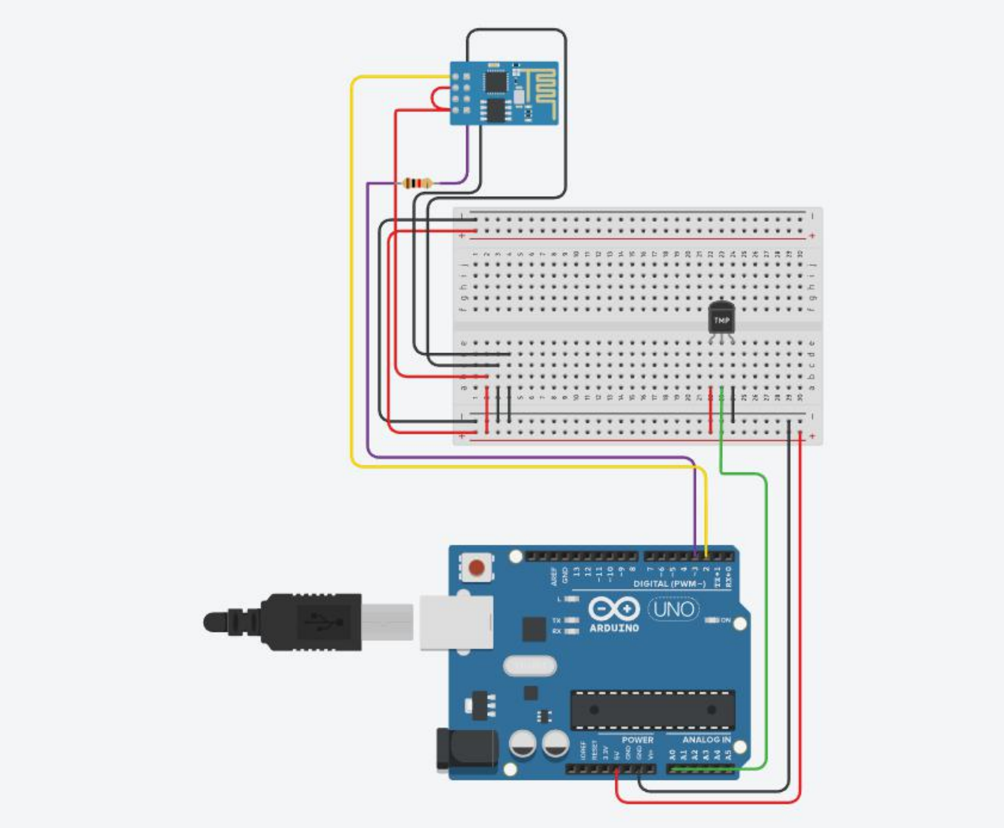
Pic. 3.2.3

Finally, I'll connect the ESP8266 Wi-Fi module, which will allow me to transfer data to the smartphone application.

To do this, I need to connect the following pins of the ESP8266 module:

1. VCC, which I connected to the positive (+) rail of the breadboard using a red jumper wire.
2. GND was connected to the negative (-) rail of the breadboard using a black jumper wire.
3. GPIO0 was connected to the negative (-) rail with a black jumper wire.
4. RX, which I connected to the D3 pin on the Arduino Uno with a yellow jumper wire.
5. TX connected to pin D4 of the Arduino Uno board using a purple jumper wire.

So, the Wi-Fi module was connected to my circuit and now it will be able to perform its function (Fig. 123).



Pic. 3.2.4

3.3 Selecting an environment for creating a user interface

To choose the development of a mobile application, you need to analyze the market. It is possible to develop a mobile application based on two operating systems - Android or IOS. According to statistics, approximately 30% of people use Apple devices, and the remaining 70% use Android-based products. Given the statistics, the development of a mobile application will be focused on the Android platform.

For this work, we have chosen the specialized environment Android Studio.

Android Studio is an integrated development environment (IDE) for developing mobile applications on the Android operating system. The tool was developed by Google and is based on the popular open source development tool IntelliJ IDEA. Android Studio is the official development environment for the Android platform and is the most widely used development environment for creating applications for the Android platform.

The environment is available for download on Windows, MacOS, and Linux operating systems. The program is a replacement for the older version of Eclipse Android Development Tools (E-ADT).

Pic. 3.3.1

Android Studio was introduced on May 16, 2013 at the Google I/O conference. The first stable build was released in December 2014, starting with version 1.0.

Android Studio has undergone significant changes and updates in recent years, including many new versions with new features and performance improvements. The new features include built-in tools for developing various types of applications, including Android apps, Android Wear apps, and Android TV apps, as well as support for Kotlin, the programming language that became the official Android development language in 2019.

In addition to Kotlin, Android Studio supports Java, C++, and JavaScript. Android Studio's interface is intuitive and easy to use, allowing you to focus on developing your application rather than learning a complex IDE.

One of the advantages of Android Studio is that it is free to use and also has a large community of developers that provides support to other developers. In addition, Android Studio has many tools to help developers develop applications, such as an auto-completion system, built-in code templates, and tools for automated testing and performance data collection.

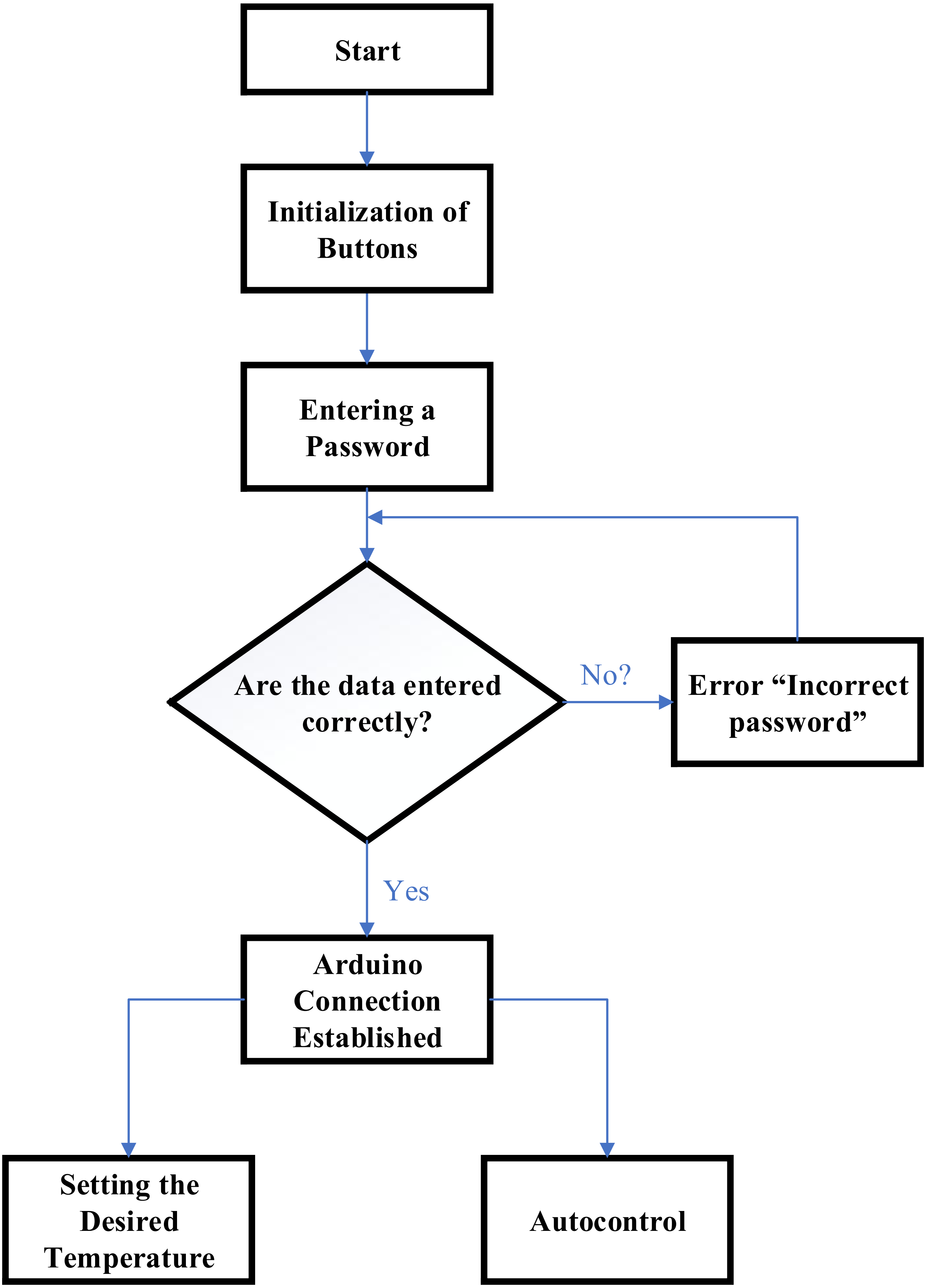
The disadvantages of Android Studio are that it can require significant computer resources, which can affect development performance. In addition, Android Studio can have a large number of settings, which can be difficult for beginners in mobile application development.

A specific feature of Android Studio is the lack of an option to disable the auto-save function.

To sum up, Android Studio is a powerful tool for developing mobile applications on the Android platform. It provides developers with many tools to create high-quality applications and has a large community of developers to provide support. Although it may have some drawbacks, its advantages far outweigh its disadvantages.

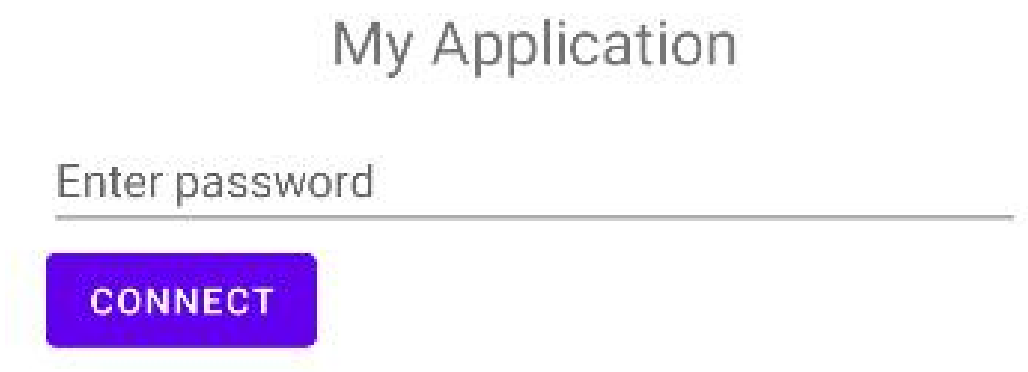
3.4. User interface design and coding

The next step is to develop the user interface. It is important that the interface is simple and intuitive. This is necessary to ensure that a user of any level of knowledge and experience in using gadgets can quickly navigate, because comfort is an important component of any product.


Start
Initialization of Buttons
Entering a Password
Are the data entered correctly?
No?
Error “Incorrect password”
Yes
Arduino Connection Established
Setting the Desired Temperature
Autocontrol

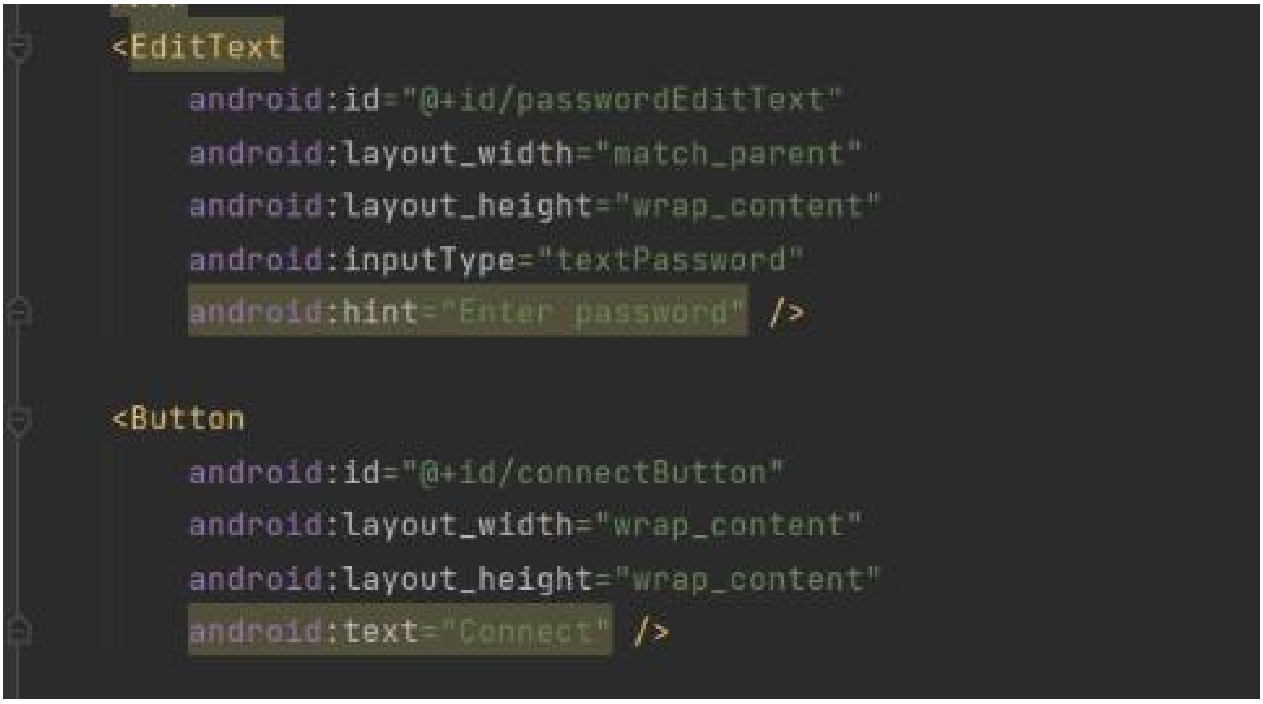
Pic 3.4.1.

The interface will consist of three blocks. The first block is used to connect the application to the system. The menu of this block will consist of the "Enter password" field and the "Connect" button.



Pic 3.4.2.

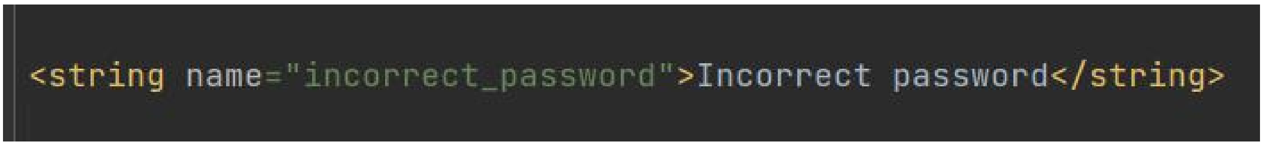
In the password field, the user will have to enter the password correctly, and then press the confirmation button to connect to the system from the Arduino via the ESP8266 Wi-Fi module and start transmitting data to the program. After that, the user will be able to set the desired temperature.



```
<EditText
    android:id="@+id/passwordEditText"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:inputType="textPassword"
    android:hint="Enter password" />

<Button
    android:id="@+id/connectButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Connect" />
```

Pic 3.4.3.

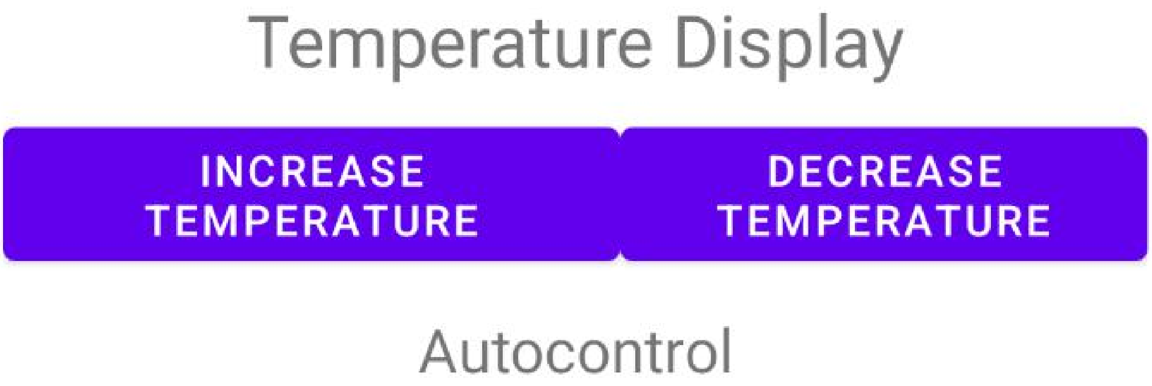
If the password is entered incorrectly, the program will send the message "Incorrect password" and the user will not be able to access the application functions until the correct password is entered.



```
<string name="incorrect_password">Incorrect password</string>
```

Pic 3.4.4.

The second one displays the set temperature and two buttons that allow you to increase or decrease the set temperature by 1 degree in either direction with one click.



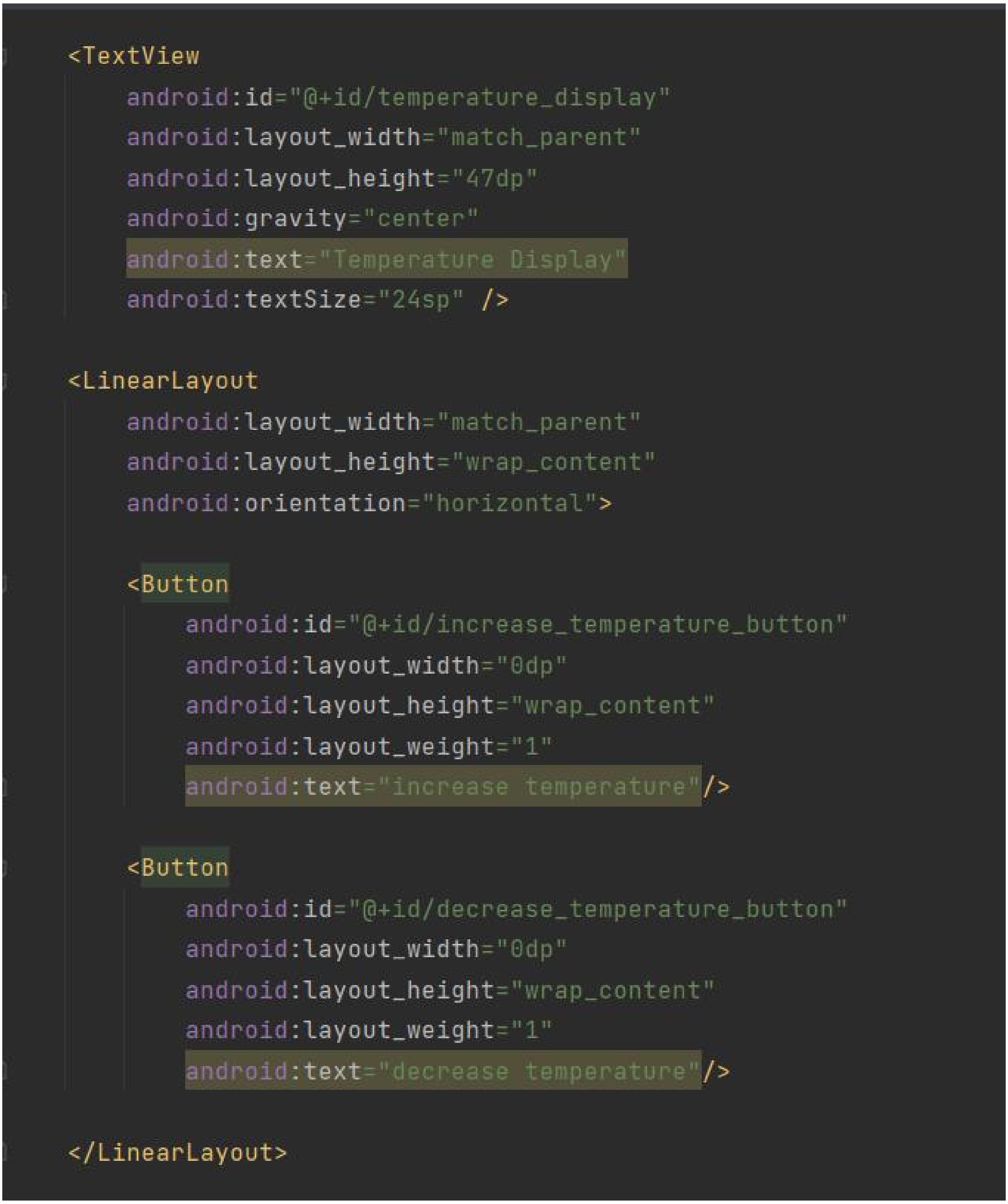
Pic 3.4.5.

The Temperature display field is responsible for displaying the set temperature in the apartment.

The Increase temperature button is responsible for increasing (+1 degree) the set temperature.

The Decrease temperature button is responsible for decreasing (-1 degree) the set temperature.

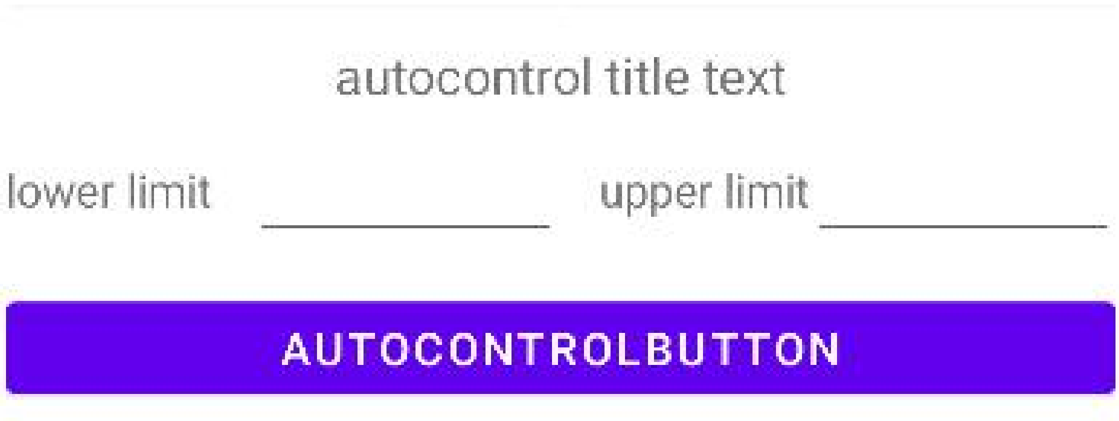
Thus, the user will be able to control the temperature in the apartment manually and set the desired value to meet their needs.



```
<TextView
    android:id="@+id/temperature_display"
    android:layout_width="match_parent"
    android:layout_height="47dp"
    android:gravity="center"
    android:text="Temperature Display"
    android:textSize="24sp" />

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">

    <Button
        android:id="@+id/increase_temperature_button"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="increase temperature"/>

    <Button
        android:id="@+id/decrease_temperature_button"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="decrease temperature"/>

</LinearLayout>
```

Pic 3.4.6.

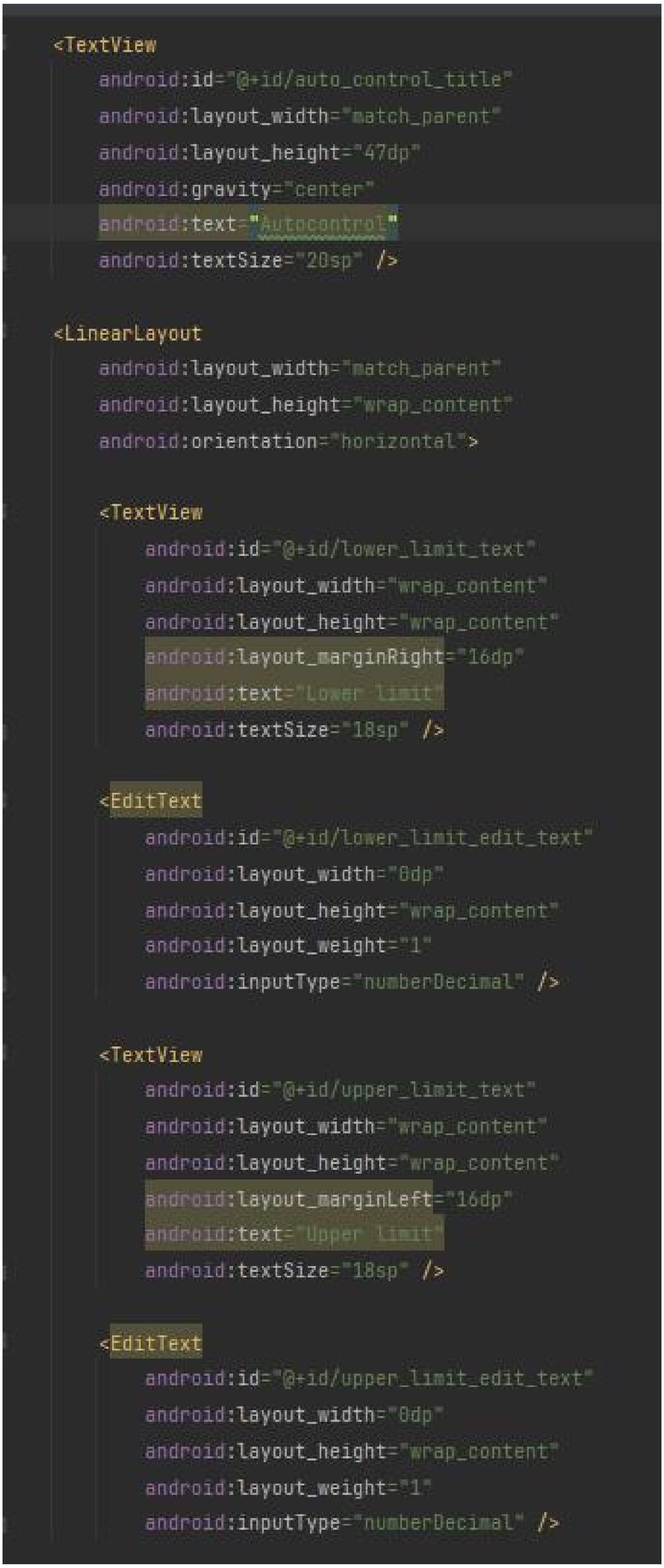
The third block will implement the function of auto-control of the temperature within the specified limits. The menu of this block will consist of several items and a button. The user will have to set the desired temperature limits within which the program will maintain the maximum and minimum values in degrees Celsius. That is, for example, if you set the maximum temperature to 25 degrees Celsius and the minimum to 20 degrees, then if the temperature drops to 19 degrees, the system will consider this value and increase it to 20 degrees. The program will work in the same way if the temperature exceeds the maximum set limit of 25 degrees and cooling starts.



Pic 3.4.7.

Lower limit - in this field, enter the minimum value of the permissible temperature in the apartment.

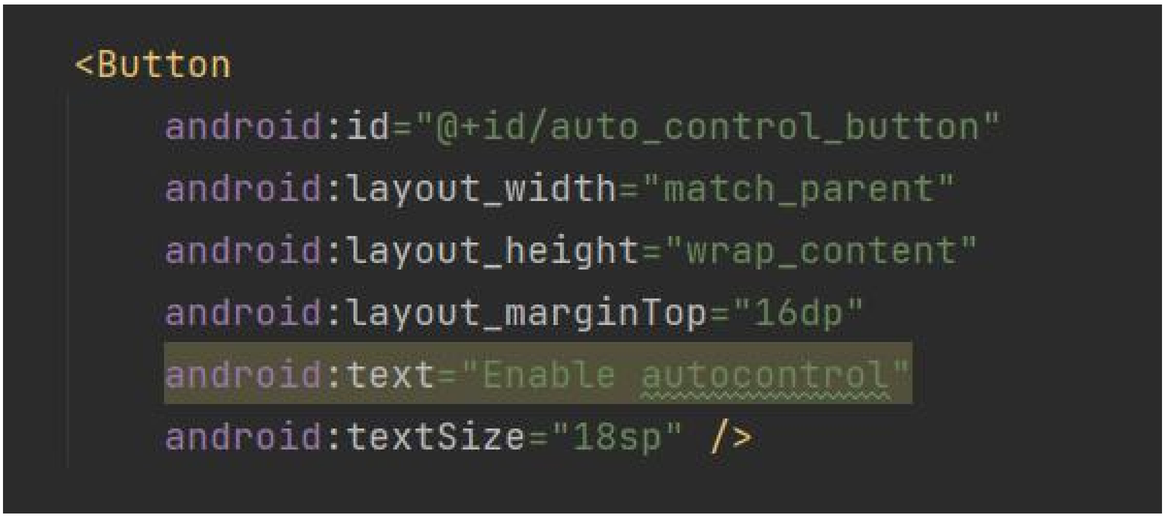
Upper limit - enter the maximum value of the permissible temperature in the apartment in this field.



```
<TextView
    android:id="@+id/auto_control_title"
    android:layout_width="match_parent"
    android:layout_height="47dp"
    android:gravity="center"
    android:text="Autocontrol"
    android:textSize="20sp" />

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">

    <TextView
        android:id="@+id/lower_limit_text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginRight="16dp"
        android:text="Lower limit"
        android:textSize="18sp" />

    <EditText
        android:id="@+id/lower_limit_edit_text"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:inputType="numberDecimal" />

    <TextView
        android:id="@+id/upper_limit_text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="16dp"
        android:text="Upper limit"
        android:textSize="18sp" />

    <EditText
        android:id="@+id/upper_limit_edit_text"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:inputType="numberDecimal" />
```

Pic 3.4.8.

The ENABLE AUTOCONTROL button is responsible for enabling the auto-control function in the program.



```
<Button
    android:id="@+id/auto_control_button"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="16dp"
    android:text="Enable autocontrol"
    android:textSize="18sp" />
```

Pic 3.4.9.

So, if the user wants to keep the temperature within a certain range rather than at a single value, for example, 22 degrees, the auto-control function will be able to satisfy their needs.

Next, I need to implement the code for connecting an Arduino via the ESP8266 Wi-Fi module to my temperature control program.

- connectWiFi(): This method establishes a connection to a Wi-Fi network using the provided SSID (network name) and password. It creates an instance of WiFiClient and connects to the specified host (IP address) and port.
- disconnectWiFi(): This method disconnects from the Wi-Fi network by sending the "AT+CWQAP" command to the ESP8266 module.
- sendSerialData(String data, long delay): This method sends data to Arduino UNO through the serial port connected to the ESP8266 module. It takes a data string and a delay (in milliseconds) between sending data.

- readSerialData(): This method reads data from Arduino UNO through the serial port connected to the ESP8266 module. It checks for available data in the input stream and reads it if available, then returns the read data as a string.
- setUpperLimit(int value): This method sets the upper temperature limit on Arduino UNO. It sends the "U" command with the specified value to Arduino to set the upper limit.
- setLowerLimit(int value): This method sets the lower temperature limit on Arduino UNO. It sends the "L" command with the specified value to Arduino to set the lower limit.
- setManualControl(boolean value): This method controls the manual temperature control mode on Arduino UNO. If the value is true, it sends the "M1" command to enable manual control, otherwise sends the "M0" command to disable manual control.
- increaseTemperature(): This method increases the temperature manually on Arduino UNO. It sends the "I" command to increase the temperature.
- decreaseTemperature(): This method decreases the temperature manually on Arduino UNO. It sends the "D" command to decrease the temperature.

These methods allow the Android application to control Arduino UNO through the ESP8266 Wi-Fi module. They send commands and data to Arduino UNO and read data from it over Wi-Fi.

The code of the file temperature_control.ino:

```
#include <OneWire.h>
#include <DallasTemperature.h>
#include <ESP8266WiFi.h>

// Wi-Fi settings
const char* ssid = "your_SSID";
const char* password = "your_PASSWORD";
const char* host = "192.168.1.100";
const int port = 8888;
WiFiClient client;

// Temperature sensor settings
#define ONE_WIRE_BUS 2
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
float currentTemp = 0.0;

// Setpoint temperature settings
float upperLimit = 25.0;
float lowerLimit = 20.0;
bool autoControl = false;
bool manualControl = false;

void setup() {
  // Start serial communication
  Serial.begin(9600);

  // Connect to Wi-Fi
  WiFi.begin(ssid, password);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(1000);
  }
  Serial.println();
  Serial.println("Connected to Wi-Fi");

  // Start temperature sensor
  sensors.begin();
}

void loop() {
  // Read temperature
  sensors.requestTemperatures();
  currentTemp = sensors.getTempCByIndex(0);
  Serial.print("Temperature: ");
  Serial.println(currentTemp);

  // Check if temperature is within limits
  if (autoControl && !manualControl) {
    if (currentTemp < lowerLimit) {
      // Turn on heater
      Serial.println("Heater ON");
    } else if (currentTemp > upperLimit) {
      // Turn off heater
      Serial.println("Heater OFF");
    }
  }

  // Check for incoming data
  if (client.available()) {
    char c = client.read();
    if (c == 'u') {
      // Receive upper limit
      upperLimit = receiveFloat();
      Serial.print("Upper limit set to ");
      Serial.println(upperLimit);
    } else if (c == 'l') {
      // Receive lower limit
      lowerLimit = receiveFloat();
      Serial.print("Lower limit set to ");
      Serial.println(lowerLimit);
    } else if (c == 'a') {
      // Enable/disable auto control
      autoControl = !autoControl;
      manualControl = false;
      Serial.print("Auto control: ");
      Serial.println(autoControl);
    } else if (c == '+') {
      // Increase temperature
      manualControl = true;
      currentTemp += 0.5;
      Serial.print("Temperature set to ");
      Serial.println(currentTemp);
    } else if (c == '-') {
      // Decrease temperature
      manualControl = true;
      currentTemp -= 0.5;
      Serial.print("Temperature set to ");
      Serial.println(currentTemp);
    } else if (c == 's') {
      // Save settings to EEPROM
      saveSettings();
    } else if (c == 'd') {
      // Disconnect Wi-Fi
public void disconnectWiFi() {
    sendSerialData("AT+CWQAP\r\n", 1000);
}
```

```
// Send data to Arduino UNO
private void sendSerialData(String data, long delay) {
    try {
        outputStream.write(data.getBytes());
        Thread.sleep(delay);
    } catch (IOException | InterruptedException e) {
        e.printStackTrace();
    }
}

// Read data from Arduino UNO
public String readSerialData() {
    String data = null;
    try {
        int byteCount = inputStream.available();
        if (byteCount > 0) {
            byte[] rawBytes = new byte[byteCount];
            inputStream.read(rawBytes);
            data = new String(rawBytes, StandardCharsets.UTF_8);
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
    return data;
}

// Set upper temperature limit
public void setUpperLimit(int value) {
    sendSerialData("U" + value + "\n", 1000);
}

// Set lower temperature limit
public void setLowerLimit(int value) {
   sendSerialData("L" + value + "\n", 1000);
}

// Set manual temperature control
public void setManualControl(boolean value) {
   String data = value ? "M1\n" : "M0\n";
   sendSerialData(data, 1000);
}

// Increase temperature manually
public void increaseTemperature() {
   sendSerialData("I\n", 1000);
}

// Decrease temperature manually
public void decreaseTemperature() {
   sendSerialData("D\n", 1000);
}
```

3.5 Conclusions to Chapter 3

In Chapter 3:

1. The device construction scheme was developed;
2. Interaction with the DS18B20 sensor is established;
3. The device is connected to the output devices;
4. Implemented the IoT temperature control algorithm;

CONCLUSION

In this thesis project, we developed a smartphone-based apartment temperature control system that allows users to conveniently monitor and adjust the temperature parameters in their living quarters using a mobile device.

As part of the project, a comparative analysis of existing home temperature control systems on the market was conducted. This allowed us to identify their advantages and disadvantages and take these aspects into account when developing our own system.

We also compared microcontrollers and chose the Arduino UNO, which meets the project requirements and has the necessary functionality to implement the temperature control system. A DS18B20 sensor was used to read the temperature reading, and an ESP8266 was used for wireless communication between the smartphone and the system.

Thus, the project successfully achieved its goals. A prototype temperature control system was developed that provides convenience and efficiency for users. The components used - Arduino UNO, DS18B20, and ESP8266 - allowed us to achieve the project goal and ensure the functionality of the system.

This thesis project contributes to the field of modern technology and the Internet of Things by expanding the possibilities of controlling the temperature in homes using smartphones. The introduction of such systems will bring many benefits to improve the quality of life, ensure energy efficiency, and provide convenience in everyday routines. This means that users will be able to enjoy a comfortable indoor climate, minimize energy consumption and reduce their environmental impact.

REFERENCES

1.Arduino to AVR C: Getting Started" by Elliot Williams

2.The AVR Microcontroller and Embedded Systems: Using Assembly and C" by Muhammad Ali Mazidi, Sarmad Naimi, and Sepehr Naimi

3.Atmel AVR Microcontroller Primer: Programming and Interfacing" by Steven Barrett

4.Programming Arduino: Getting Started with Sketches" by Simon Monk

5.Arduino WiFi Projects" by Agus Kurniawan

6.Internet of Things with Arduino Blueprints" by Pradeeka Seneviratne

7.Building Arduino Projects for the Internet of Things: Experiments with Real-World Applications" by Adeel Javed

8.Programming the ESP8266 with Arduino IDE" by Marco Schwartz

9.The Rise of the Smart Homes: Integrating Technology into Everyday Life
https://www.nogentech.org/smart-homes-integrating-technology-into-everyday-life/

10.ATmega328P Datasheet:
https://ww1.microchip.com/downloads/en/DeviceDoc/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061A.pdf

11.Arduino Arduino UNO microcontroller:
https://docs.arduino.cc/hardware/uno-rev3

12.ATmega328P Complete Manual:

https://www.arduino.cc/en/Guide

13.Arduino Playground:
https://playground.arduino.cc/Main/ArduinoBoardUno/

14.STM32 Datasheet:
https://www.st.com/en/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus.html#resource

15.Documentation, manuals and example projects for working with the ESP8266 module:
https://www.espressif.com/en/products/socs/esp8266

16.The Arduino forum section is dedicated to questions and projects related to the use of the ESP8266 module with the Arduino platform:
https://forum.arduino.cc/c/esp8266/