|  |  |  |
| --- | --- | --- |
|  | HOME AUTOMATION USING SINGLE BOARD COMPUTERS |  |

## PROJECT REPORT

***Submitted By***

|  |  |
| --- | --- |
| ARUN R G | 611220104007 |
| CHERAN J | 611220104026 |
| DHANUSSH ADITYA K | 611220104031 |
| GOKUL HARI R | 611220104048 |

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

**IN**

## COMPUTER SCIENCE AND ENGINEERING

**KNOWLEDGE INSTITUTE OF TECHNOLOGY,**

## SALEM-637504

**ANNA UNIVERSITY::CHENNAI 600025**

**MAY 2023**

|  |  |  |
| --- | --- | --- |
|  | HOME AUTOMATION USING SINGLE BOARD COMPUTERS |  |

## PROJECT REPORT

***Submitted By***

|  |  |
| --- | --- |
| ARUN R G | 611220104007 |
| CHERAN J | 611220104026 |
| DHANUSSH ADITYA K | 611220104031 |
| GOKUL HARI R | 611220104048 |

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

**IN**

## COMPUTER SCIENCE AND ENGINEERING

**KNOWLEDGE INSTITUTE OF TECHNOLOGY,**

## SALEM-637504

**ANNA UNIVERSITY::CHENNAI 600025**

**MAY 2023**

**ANNA UNIVERSITY::CHENNAI 600 025**

**BONAFIDE CERTIFICATE**

Certified that this project report titled **“HOME AUTOMATION USING SINGLE BOARD COMPUTERS”** is the bonafide work of **“ARUN R G (611220104007), CHERAN J (611220104026), DHANUSSH ADITHYA K (611220104031), JAYASHREE K (611220104048)”** who carried out the project work under my supervision.

|  |  |
| --- | --- |
| SIGNATURE | SIGNATURE |
| Dr. V. KUMAR M.E., Ph.D., | Mr. T. KARTHIKEYAN B.Tech.,M.S(IT)., |
| HEAD OF THE DEPARTMENT | SUPERVISOR PROFESSOR |
| PROFESSOR | ASSISTANT PROFESSOR |
| Department of Computer Science and Engineering,  Knowledge Institute of Technology,  Kakapalayam,  Salem- 637 504. | Department of Computer Science and Engineering,  Knowledge Institute of Technology,  Kakapalayam,  Salem- 637 504. |
| Submitted for the Project Viva-Voce Examination held on\_\_\_\_\_\_\_\_\_\_\_\_ | |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| INTERNAL EXAMINER | EXTERNAL EXAMINER |

## ACKNOWLEDGEMENT

At the outset, we express our heartfelt gratitude to **GOD,** who has been our strength to bring this project to light.

At this pleasing moment of having successfully completed our project, we wish to convey our sincere thanks and gratitude to our beloved president **Mr. C. Balakrishnan,** who has provided all the facilities to us.

We would like to convey our sincere thanks to our beloved Principal **Dr. PSS. Srinivasan,** for forwarding us to do our project and offering adequate duration in completing our project.

We express our sincere thanks to our Head of the Department **Dr. V. Kumar,** Department of Computer Science and Engineering for fostering the excellent academic climate in the Department.

We express our pronounced sense of thanks with deepest respect and gratitude to our internal guide **Mr. T. Karthikeyan,** Assistant Professor Computer Science and Engineering Department, for his valuable and precious guidance and for having amicable relation.

With deep sense of gratitude, we extend our earnest and sincere thanks to our project coordinators **Mr. P. Sasikumar and Mr. I. Rajesh,** Assistant Professor, Department of Computer Science and Engineering for their guidance and encouragement during this project.

We would also like express our thanks to all the faculty members of our Department, friends and students who helped us directly and indirectly in all aspects of the project work to get completed successfully.

|  |  |  |
| --- | --- | --- |
| **TABLE OF CONTENTS** | | |
| **CHAPTER NO.** | **TITLE**  **ABSTRACT** | **PAGE NO**  I |
|  | **ABSTRACT** | **I** |
|  | **LIST OF FIGURES** | **II** |
|  |  |  |
| **1** | **INTRODUCTION** | 1 |
|  | 1.1 INTRODUCTION | 1 |
| **2** | **LITERATURE SURVEY** | 2 |
| **3** | **SYSTEM ANALYSIS** | 6 |
|  | 3.1 EXISTING SYSTEM | 6 |
|  | 3.2 PROBLEM IDENTIFICATION | 6 |
|  | 3.3 PROPOSED SYSTEM | 6 |
| **4** | **SYSTEM SPECIFICATION** | 7 |
| **5** | **SYSTEM DESIGN** | 8 |
| **6** | **MODULE DESCRIPTION** | 10 |
|  | 6.1 MODULES | 10 |
|  | 6.1.1 Data Preprocessing | 10 |
|  | 6.1.2 Feature Extraction | 10 |
|  | 6.1.3 Clustering | 11 |
|  | 6.1.4 Regression | 11 |
|  | 6.1.5 Accuracy Calculation | 11 |
| **7** | **SYSTEM TESTING** | 13 |
|  | 7.1 TYPES OF TESTING | 13 |
|  | 7.1.1 Unit Testing | 13 |
|  | 7.1.2 Integration Testing | 14 |
|  | 7.1.3 Validation Testing | 14 |
|  | 7.1.4 Output Testing | 15 |
| **8** | **SYSTEM IMPLEMENTATION** | 16 |
| **9** | **CONCLUSION AND FUTURE ENHANCEMENT** | 17 |
|  | 9.1 CONCLUSION | 17 |

9.1 FUTURE ENHANCEMENT 17

1. APPENDIX 18
   1. SOURCE CODE 18
   2. [SCREENSHOTS 23](#_TOC_250000)
2. REFERENCES 26

|  |
| --- |
| ABSTRACT |

## ABSTRACT

While smart devices offer convenience and a range of options, limitations such as installation, cost, variety, and reliability hinder their effectiveness. In contrast, SBCs provide an affordable, customizable, and reliable solution for home automation. SBCs offer an affordable alternative to expensive smart IoT devices, allowing homeowners to retrofit their existing appliances and devices with automation capabilities without the need for costly replacements. The non-IoT sector provides a wider variety of options compared to limited choices for smart devices, enabling homeowners to automate a broader range of devices and tailor the automation experience to their specific needs and preferences. Reliability is a significant concern with smart IoT devices, as a failure in the processing hardware or logic renders the entire device useless. In contrast, SBCs offer robustness and longevity through their modular nature, allowing for easy replacement or upgrade of individual components. This ensures continued functionality and reduces the risk of complete system failure. In conclusion, Single-Board Computers (SBCs) offer an affordable, customizable, and reliable solution for home automation, particularly in homes without pre-installed smart IoT devices or those with budgetary constraints. By overcoming the limitations of smart devices, SBCs empower homeowners to effectively automate their homes, providing sustainable and personalized home automation experience

## LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **NAME OF FIGURE** | **PAGE NO** |
|  |  |  |
|  |  |  |

## LIST OF ABBREVATIONS

|  |  |
| --- | --- |
| ABBREVATION | EXPANSION |
| SBC | SINGLE BOARD COMPUTER |
| DHCP | DYNAMIC HOST CONTROL PROTOCOL |
| BLE | BLUETOOTH LOW ENERGY |
| HTTP | HYPERTEXT TRANSFER PROTOCOL |
| DNS | DOMAIN NAME SYSTEM |
| WS | WEB SOCKET |
| TCP | TRANSMISSION CONTROL PROTOCOL |
| HTTPS | HYPERTEXT TRANSFER PROTOCOL SECURE |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## INTRODUCTION

## CHAPTER – 1

## INTRODUCTION

**1.1 PROJECT OVERVIEW**

Home automation has become increasingly popular as homeowners seek to enhance convenience and control within their living spaces. While the market for smart Internet of Things (IoT) devices is booming, there are significant challenges that need to be addressed. Firstly, not all homes have these smart devices installed, and their high costs can be a deterrent for many households. Additionally, the IoT device sector lacks the variety of categories found in the non-IoT sector. This limitation restricts homeowners to limited options, and they may be stuck with designs that do not meet their specific needs. Furthermore, the reliance on processing hardware and logic in IoT devices poses a risk as the failure of these components renders the entire device useless. Manufacturers are unlikely to replace or repair individual components, making the investment in smart devices potentially futile.

However, amidst these challenges, there is another sector that is also booming in the field of home automation: Single-Board Computers (SBCs). SBCs, such as Arduino and Raspberry Pi, offer a cost-effective alternative to IoT devices. They are often more affordable than their smart counterparts, making them accessible to a wider range of homeowners. Additionally, SBCs provide a wider variety of options in terms of categories and applications, allowing homeowners to have greater flexibility in automating their homes according to their specific preferences. The modular nature of SBCs also ensures reliability and longevity, as individual components can be easily replaced or upgraded in case of failure, reducing the risk of complete system failure.

In this project, we aim to explore the advantages of SBCs over smart IoT devices and demonstrate why they are a superior choice for effective and affordable home automation. By overcoming the limitations of cost, variety, and hardware failures, SBCs empower homeowners to automate their homes effectively while leveraging their existing appliances and devices. Through in-depth analysis and experimentation, we will showcase how SBCs provide personalized and sustainable home automation experiences, offering homeowners the convenience and control they desire within their living spaces.

**1.2 PURPOSE**

The purpose of this project is to address the issues of high cost, limited affordability, and potential hardware failures in smart IoT devices for home automation. The project aims to explore and develop cost-effective solutions that make home automation accessible to a broader range of households. By researching alternative options and evaluating their effectiveness, the project aims to provide practical recommendations and strategies for implementing affordable and reliable smart IoT devices in home automation systems. Ultimately, the project strives to promote the widespread adoption of cost-effective and reliable smart IoT devices, enabling more households to benefit from the advantages of home automation.

**LITERATURE SURVEY**

**CHAPTER - 2**

## LITERACTUTRE SURVEY

**2.1** **THE ROLE OF SINGLE-BOARD COMPUTERS IN AFFORDABLE HOME AUTOMATION (Smith.J,2022)**

**2.1.1** **Problem Identification**

In the study titled the problem statement identified is the high cost and limited affordability of smart IoT devices in home automation. Many households have yet to adopt these devices due to their prohibitive costs. Additionally, the limited variety of options and the risk of hardware failures in smart IoT devices pose significant challenges to their effectiveness.

**2.1.2** **Proposed System**

To address these challenges, the proposed solution is the use of Single-Board Computers (SBCs) as a cost-effective alternative in home automation. SBCs offer a more affordable option, allowing homeowners to retrofit their existing devices and appliances for automation purposes without the need for expensive smart IoT replacements. This solution provides an accessible and customizable approach to home automation, overcoming the limitations of cost, variety, and hardware failures associated with smart IoT devices.

**2.2** **COMPARATIVE ANALYSIS: SINGLE-BOARD COMPUTERS VERSUS SMART/IOT DEVICES IN HOME AUTOMATION (Patel.R,2019)**

**2.2.1**  **PROBLEM IDENTIFICATION**

The increasing popularity of home automation has led to a dominant market presence of smart Internet of Things (IoT) devices. However, not all households have these devices installed, and the high costs associated with smart/IoT devices can be prohibitive for many homeowners. Moreover, the limited variety of options and the risk of hardware failures in smart/IoT devices present significant challenges to their effectiveness. This study aims to address these issues by exploring the potential of Single-Board Computers (SBCs) as a more affordable and customizable alternative for home automation, while also examining their reliability and compatibility with existing appliances and systems.

**2.2.2** **PROPOSED SOLUTION**

The proposed solution is to conduct a comparative analysis of SBCs and smart/IoT devices based on the identified factors. By evaluating the cost, customization options, reliability, and compatibility of both options, the study aims to provide insights and guidance to homeowners in selecting the most suitable solution for their home automation needs. This analysis will assist in understanding the strengths and weaknesses of SBCs and smart/IoT devices and their implications for affordable and effective home automation.

**2.3** **RELIABILITY AND LONGEVITY: SINGLE-BOARD COMPUTERS FOR SUSTAINABLE HOME AUTOMATION (Lee.C,2020)**

**2.3.1** **PROBLEM STATEMENT**

In the field of home automation, the reliability and longevity of automation systems are crucial factors that can significantly impact their effectiveness and sustainability. The current reliance on smart Internet of Things (IoT) devices poses concerns due to the potential failure of their integrated processing hardware or logic, rendering the entire device useless and requiring costly replacements. This limitation raises the need for an alternative solution that offers enhanced reliability and longevity. This study aims to investigate the potential of Single-Board Computers (SBCs) as a more robust and long-lasting option for sustainable home automation, exploring their modular nature that allows for easy replacement or upgrade of individual components to ensure continued functionality and minimize the risk of complete system failure.

**2.3.2** **PROPOSED SOLUTION**

The proposed solution in the study is to leverage Single-Board Computers (SBCs) as a more reliable and long-lasting option for sustainable home automation. By utilizing SBCs, homeowners can benefit from their modular nature, which allows for easy replacement or upgrade of individual components in case of hardware failures. This approach ensures continued functionality of the automation system while minimizing the risk of complete system failure. The study aims to demonstrate how SBCs can provide a viable solution for homeowners seeking a sustainable and reliable home automation setup.

**2.4** **ENHANCING USER EXPERIENCE: SINGLE-BOARD COMPUTERS FOR INTUITIVE HOME AUTOMATION (Wang.L,2018)**

**2.4.1 PROBLEM IDENTIFICATION**

The user experience is a crucial aspect of home automation, as it directly impacts the usability, satisfaction, and adoption of automation systems. The current market dominance of smart Internet of Things (IoT) devices may not always offer an intuitive interface and user-friendly installation guides, leading to complexity and difficulties in setting up and operating home automation systems. This study aims to address this issue by exploring the potential of Single-Board Computers (SBCs) as a solution for enhancing the user experience in home automation. The focus is on how SBCs can provide an intuitive interface and user-friendly installation guides, simplifying the setup and operation of home automation systems. The study aims to highlight the importance of a positive user experience in driving the adoption of SBCs for intuitive and user-friendly home automation.

**2.4.2** **PROPOSED SOLUTION**

The proposed solution in the study is to leverage Single-Board Computers (SBCs) to enhance the user experience in home automation. The study suggests that SBCs can provide an intuitive interface and user-friendly installation guides, simplifying the setup and operation of home automation systems. By utilizing SBCs, homeowners can have a more positive and user-friendly experience when interacting with their automated systems. The study aims to demonstrate how SBCs can be a solution for improving the user experience and driving the adoption of intuitive home automation.

**SYSTEM ANALYSIS**

**CHAPTER – 3**

**SYSTEM ANALYSIS**

* 1. **EXISTING SYSTEM**

The existing system in smart home automation revolves around the utilization of IoT (Internet of Things) devices. These devices, equipped with sensors, connectivity capabilities, and processing power, enable homeowners to automate and control various aspects of their homes. Smart thermostats, lighting systems, security devices, appliances, and entertainment systems are some examples of IoT devices integrated into smart homes. Through mobile applications or voice assistants, homeowners can remotely monitor and manage their homes, accessing features such as temperature control, lighting adjustments, security monitoring, and entertainment systems. The connectivity between IoT devices, a central hub or controller, and the internet facilitates seamless communication and coordination. Wi-Fi, Bluetooth, Zigbee, or Z-Wave are commonly employed communication protocols. By embracing IoT devices, the existing system in smart home automation offers homeowners the convenience of a connected and automated home environment, enhancing comfort, energy efficiency, and security.

## PROBLEM IDENTIFICATION

Smart home automation is the limited accessibility and affordability of smart IoT devices, particularly in middle-class and lower-middle-class households. These households face barriers in adopting smart IoT devices due to the absence of pre-installed devices, higher costs compared to traditional alternatives, limited options in the IoT sector, and the risk of complete device failure without proper manufacturer support. These challenges restrict the widespread implementation of home automation, preventing many homeowners from enjoying the benefits of a connected and automated home environment. There is a need to address these problems and find solutions that make smart home automation more accessible, affordable, and inclusive for middle-class and lower-middle-class households.

## PROPOSED SYSTEM

The proposed system aims to address the challenges faced in the installation of smart IoT devices in middle-class and lower-middle-class homes by utilizing Single Board Computers (SBCs) such as Arduino and Raspberry Pi. These SBCs offer a cost-effective alternative and provide enhanced functionality and customization options. The system maps the electric circuits of regular household appliances to the SBCs, enabling users to control and monitor their operations through programmed instructions. Wireless communication protocols facilitate seamless connectivity between the SBCs and a central hub or gateway, allowing efficient data exchange and centralized control. By adopting SBCs, the system overcomes limitations such as the absence of pre-installed smart devices, high costs, limited variety, and potential hardware failures. Overall, the proposed system harnesses the capabilities of SBCs to create an accessible and inclusive home automation solution

# **SYSTEM SPECIFICATION**

## CHAPTER - 4

## SYSTEM SPECIFICATION

**4.1 HARDWARE REQUIREMENTS**

Computing Devices : SBC Board

RAM : 32 KB

Hard Disk : 256 KB

Editing Devices : PC/Computer

## 4.2 SOFTWARE REQUIREMENTS

Operating System : Windows/Linux/Mac

Language : Python

Editor : VS Code

Simulation Tool : Packet Tracer

Version : 8.2.1

## 4.3 TOOL REQUIREMENTS

Operating System : Windows 10 / 8.1 / 8

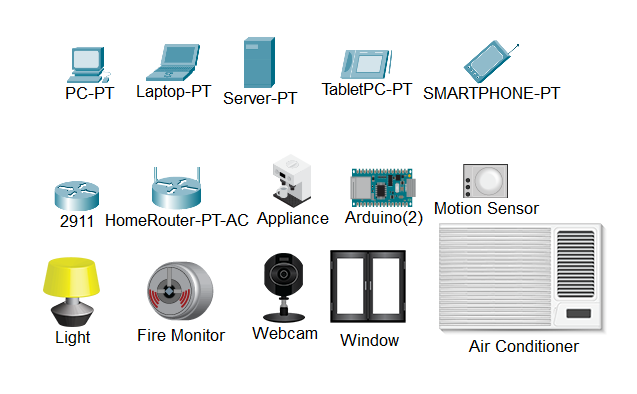
OS RAM : 8 GB

OS Disk Space : 1 TB

Processor : Intel Core i5

## 4.4 DEVICES REQUIREMENTS

|  |  |
| --- | --- |
| DEVICE | QUANTITY |
| **SBC Board** | **5** |
| **Laptop** | **2** |
| **PC** | **1** |
| **Server** | **1** |
| **Home Router** | **1** |
| **Router** | **2** |
| **Motion Sensor** | **6** |
| **Light** | **3** |
| **Smart Phone** | **1** |
| **Air Conditioner** | **1** |
| **Web Cam** | **2** |
| **Fire Monitor** | **1** |
| **Smoke Sensor** | **1** |
| **Tablet** | **1** |



**Fig. 4.1 DEVICES USED**

# MODUES

# **CHAPTER - 5**

# **MODULE DESCRIPTION**

# **5.1 MODULES**

## Module 1: Packet Tracer

## Module 2: Single Boarded Computer (SBC)

## Module 3: HTTP Server

## Module 4: IoE Client

## Module 5: IoE Server

## Module.6: DNS

## Module 7: DHCP

## Module 8: Router

## Module 9: Home Router

## Module 10: Home Appliances

## Module 11: Sensors

## Module 12: Remote Access

## Module 13: NAT

## Module 14: RIP

## Module 15: Simulation Environment

**5.1.1 Packet Tracer**

Packet Tracer is a versatile network simulation tool widely used for designing, configuring, and simulating IoT environments. It provides a user-friendly interface that allows users to create virtual networks, connect devices, and simulate network behaviors. With its Python and JS APIs, Packet Tracer enables users to emulate real-world Single Board Computer (SBC) programming languages, facilitating the development and testing of IoT applications in a simulated environment.

**5.1.2 Single Boarded Computer (SBC)**

Single Board Computers (SBCs) are compact computing devices that integrate all the components of a regular computer onto a single circuit board. In this project, an SBC with specifications similar to the Arduino Nano 33 IoT is utilized. It offers 6 digital pins, 4 analog pins, and supports WiFi, BLE, TCP, and HTTP server capabilities. This SBC serves as the core platform for implementing home automation solutions, providing extensive connectivity options and programmability.

**5.1.3 HTTP Server**

The HTTP server is a critical component of the SBC's functionality. It enables the SBC to serve HTTP requests from client devices connected to the local network. By accessing the SBC's local IP address through a web browser, users can interact with the HTTP server to configure the SBC, manage connected devices, and monitor system status. The HTTP server acts as the gateway for controlling and accessing the automation features provided by the SBC.

**5.1.4 IoE Client**

The SBC operates as an Internet of Things (IoT) client, representing the central control unit for the connected IoT devices in the home automation setup. As an IoE client, the SBC enables communication and coordination between the various IoT devices and the central IoE server. It facilitates the transmission of commands and data between the user interface and the physical devices, ensuring seamless integration and control of the IoT ecosystem.

**5.1.5 IoE Server**

The IoE server functions as the central hub for managing and controlling the registered IoT devices. It provides users with a web-based interface or a dedicated application to monitor and control the devices in real-time. The IoE server allows users to set conditions, define automation rules, and receive status updates from the connected devices. It acts as the backbone of the home automation system, facilitating intelligent decision-making and efficient device management.

**5.1.6 DNS**

The Domain Name System (DNS) is a critical network service that translates domain names into IP addresses. In the context of this project, the DNS is responsible for maintaining a record mapping of the IoT server's domain name to its corresponding IP address. This ensures that users can access the IoT server using its domain name, simplifying the process of connecting to and controlling the home automation system.

**5.1.7 DHCP**

The Dynamic Host Configuration Protocol (DHCP) is a network protocol used to automatically assign IP addresses to devices within a network. In this project, the home router acts as a DHCP server, dynamically assigning IP addresses to devices connected to the local network. This simplifies network setup and configuration, allowing devices to obtain IP addresses automatically without manual intervention.

**5.1.8 Router**

Routers are networking devices that facilitate the transfer of data packets between different networks. In this project, the router plays a crucial role in connecting the local network to the internet, enabling communication with external networks and services. The router routes network traffic, manages IP addresses, and provides network security features to ensure reliable and secure data transmission.

**5.1.9 Home Router**

The home router represents a typical router device commonly found in residential settings. It serves as the primary gateway for connecting devices within the home network to the internet. Along with routing capabilities, the home router incorporates features such as switching and WLAN connectivity, allowing devices to communicate with each other and access the internet seamlessly.

**5.1.10 Home Appliances**

Home appliances refer to the existing devices in a home that do not have built-in IoT capabilities. These appliances can be integrated into the home automation system by connecting them to the SBC. Users can access the SBC's HTTP server to configure and control these appliances, leveraging the SBC's presets and functionalities. The SBC acts as an intermediary, providing the necessary signals and instructions to automate the operation of the appliances.

**5.1.11 Sensors**

Sensors play a crucial role in gathering data from the physical environment. In this project, sensors without built-in IoT functionalities are connected to the SBC. The SBC receives signals from these sensors and transmits the data to the IoE server for further processing and analysis. By integrating sensors into the home automation system, users can monitor and respond to environmental conditions effectively.

**5.1.12 Remote Access**

Remote access allows users to control and monitor their home automation system from a remote location. While direct control of the SBC is restricted for privacy and security reasons, users can access the IoE server remotely. This enables them to check the status of connected devices, receive notifications, and manage automation settings conveniently, ensuring seamless control even when away from home.

**5.1.13 NAT**

Network Address Translation (NAT) is a technique used to map private IP addresses to public IP addresses. In this project, NAT is employed in the ISP router to enable external access to the IoE server. By mapping the internal IP address of the IoE server to a public IP address, NAT allows users to access and interact with the home automation system from outside the local network securely.

**5.1.14 RIP**

The Routing Information Protocol (RIP) is a dynamic routing protocol used by routers to exchange information about network paths and determine the most efficient routes for data transmission. In this project, RIP may be configured on the routers to ensure proper routing of packets within the network, maximizing network performance and minimizing packet loss.

**5.1.15 Simulation Environment**

The simulation environment provided by Packet Tracer allows users to simulate and emulate network behaviors accurately. It offers simulation modes that enable the inspection of packet traversal and network interactions. Additionally, Packet Tracer supports the transpilation of Python code to JavaScript, which helps replicate the programming languages used in real-world SBCs. This simulation environment allows for comprehensive testing and validation of the home automation system's functionality before its actual implementation.

# **SYSTEM DESIGN**

**CHAPTER - 6**

**SYSTEM DESIGN**

**6.1 SYSTEM DESIGN OVERVIEW**

The home automation system consists of various components that work together to enable seamless control and automation of devices within a household. The system includes Single Board Computers (SBCs), IoT devices, an IoE server, routers, DNS, DHCP, and user devices. The design aims to provide a user-friendly interface for controlling and monitoring the connected devices while ensuring reliable communication and security.

**6.2 SYSTEM COMPONENTS**

**6.2.1 Single Board Computers (SBCs)**

* Utilizes an SBC, such as the Arduino Nano 33 IoT, with digital and analog pins, WiFi, BLE, and TCP/HTTP server capabilities.
* Serves as the central control unit for the home automation system.
* Connects to various devices and sensors within the household.

**6.2.2 IoT Devices**

* Includes home appliances and sensors without built-in IoT capabilities.
* Connected to the SBCs for integration into the home automation system.
* Controlled and monitored through the SBC's HTTP server.

**6.2.3 IoE Server**

* Acts as the central management system for the connected IoT devices.
* Provides a web-based interface or application for users to control and monitor devices.
* Allows users to set conditions, define automation rules, and receive status updates and notifications.

**6.2.4 Routers**

* Consist of the ISP-provided router and a home router.
* The ISP router connects the local network to the internet and supports NAT for external access.
* The home router provides WLAN, switching, and other capabilities for the local network.

**6.2.5 DNS**

* Maintains a record mapping of the IoT server's domain name to its IP address.
* Enables users to access the IoT server using a domain name instead of an IP address.

**6.2.6 DHCP**

* The home router acts as the DHCP server, dynamically assigning IP addresses to devices within the local network.
* Simplifies the network setup process by automating IP address assignment.

**6.2.7 User Devices**

* Includes smartphones, tablets, or computers used by users to access and control the home automation system.
* Connect to the system via the IoE server's web-based interface or dedicated application.

**6.3 COMMUNICATION FLOW**

1. The SBCs establish communication with the connected IoT devices, receiving data from sensors and controlling appliances.
2. The SBCs transmit the collected data and device status updates to the IoE server for processing and storage.
3. Users access the IoE server through their devices using a web-based interface or application.
4. Users interact with the IoE server to monitor device status, set automation rules, and receive notifications.
5. The IoE server sends commands and instructions to the SBCs to control the connected IoT devices.
6. The SBCs communicate with the devices, triggering actions based on the received commands or automation rules.

**6.4 SECURITY CONSIDERATIONS**

* Remote access to the SBCs is restricted to ensure privacy and security.
* External access to the home automation system is facilitated through NAT in the ISP router.
* Proper authentication and encryption measures are implemented to secure communication between devices and the IoE server.
* Regular software updates and security patches are applied to all system components to address vulnerabilities.

This system design provides a foundation for an efficient and user-friendly home automation system. It allows users to seamlessly control and monitor their devices, create automation rules, and receive real-time status updates, enhancing convenience and improving the overall home automation experience.

# **SYSTEM TESTING**

# **CHAPTER - 7**

## SYSTEM TESTING

**7.1 UNIT TESTING**

* Test the functionality of individual components, such as SBCs, IoT devices, and sensors, in isolation.
* Verify that each component performs its intended operations correctly.
* Test the SBC's HTTP server by sending HTTP requests and ensuring proper responses.
* Validate the communication between the SBCs and connected IoT devices, ensuring data exchange and control commands function as expected.

**7.2 INTEGRATION TESTING**

* Test the integration between various system components to ensure seamless communication and interoperability.
* Verify that the SBCs can successfully communicate with the IoE server and exchange data.
* Validate that the IoE server can receive and process data from multiple SBCs and update the device status accordingly.
* Test the integration between the SBCs and IoT devices, ensuring proper control and monitoring functionality.

**7.3 SYSTEM TESTING**

* Perform end-to-end testing of the entire home automation system as a whole.
* Test the system's functionality by simulating real-world scenarios and user interactions.
* Validate that user devices can successfully connect to the IoE server and interact with the home automation system.
* Test the system's response to various user commands, automation rules, and device status updates.
* Verify the system's ability to handle concurrent connections, ensuring scalability and performance.

**7.4 ACCEPTANCE TESTING**

* Involve end-users or stakeholders to validate and verify that the system meets their requirements and expectations.
* Conduct usability testing to ensure the user interface is intuitive and user-friendly.
* Test the system against predefined use cases and scenarios to ensure it behaves as intended.
* Validate that the system meets security and privacy requirements, ensuring data –––protection and access control.

Gather feedback from users and stakeholders to identify any areas for improvement or additional features.

**SYSTEM IMPLEMENTATION**

## CHAPTER – 8

## SYSTEM IMPLEMENTATION

**8.1 CONFIGURING SBC**

The Instruction Logic for the SBC have to be set up before this project can continue. The logic contains IoEClient setup, Server Call to Setup server, and initializes all connected devices if the configuration contains them, it also reads and write states to and from server and components.

**8.1.1 SBC INSTRUCTION LOGIC**

from json import loads,dumps

from gpio import \*

from time import \*

from file import \*

from ioeclient import \*

from HttpServer import initHttp

from physical import \*

from networking import localIP

pinSlots = {"A0":A0, "A1":A1,"A2":A2, "A3":A3}

ConnectedDevices = {}

ip = localIP()

class Device:

def \_\_init\_\_(self, device):

global pinSlots

self.name = device['config']['name']

self.defaultState = device['defaultState']

self.pin = device['pin'] if device['pin'] not in pinSlots else pinSlots[device['pin']]

self.config = device['config']

self.currentState = None

self.readable = device['readable']

self.writable = device['config']['controllable']

self.read ,self.write = getReadWrite(self.config["type"])

self.setState(self.defaultState)

if(self.readable):

add\_event\_detect(self.pin, lambda : self.getState())

self.getState()

def getState(self):

if not self.readable:

return

self.currentState = self.read(self.pin)

reportState()

def setState(self, value):

if not self.writable:

return

self.currentState = value

if self.config["type"] == "bool":

self.write(self.pin, HIGH if value!="0" else LOW)

else:

self.write(self.pin,value)

def init():

loadConfig()

setupIoE()

def main():

init()

initHttp(onConfigUpdate)

while True:

checkForIPChange()

reportState()

sleep(100)

def setupIoE():

global ConnectedDevices

global ip

IoEClient.setup({

"type":ip,

"states": [i.config for i in ConnectedDevices.values()]

})

IoEClient.onStateSet(setDeviceState)

def setDeviceState(device, state):

ConnectedDevices[device].setState(state)

reportState()

def reportState():

try:

states = ",".join([str(ConnectedDevices[i].currentState) for i in ConnectedDevices])

print(states)

IoEClient.reportStates(states)

except Exception as e:

print(e)

def checkForIPChange():

global ip

if ip!=localIP():

ip = localIP()

setupIoE()

def loadConfig():

global ConnectedDevices

global SBC\_Name

json = ""

jsonFile =open("/config.json","r")

line = jsonFile.readline()

while(line!=''):

json+=line.strip('\n')

line = jsonFile.readline()

jsonFile.close()

json = loads(json)

setDeviceProperty(getName(),"name",json['deviceID'])

for i in json['devices'].values():

device = Device(i)

ConnectedDevices[device.name] = device

def onConfigUpdate(configJSON):

global ConnectedDevices

configFile = open("/config.json","w")

for i in configJSON:

configFile.write(i)

configFile.close()

ConnectedDevices = {}

init()

def getReadWrite(type):

if(type == 'bool'):

return (digitalRead, digitalWrite)

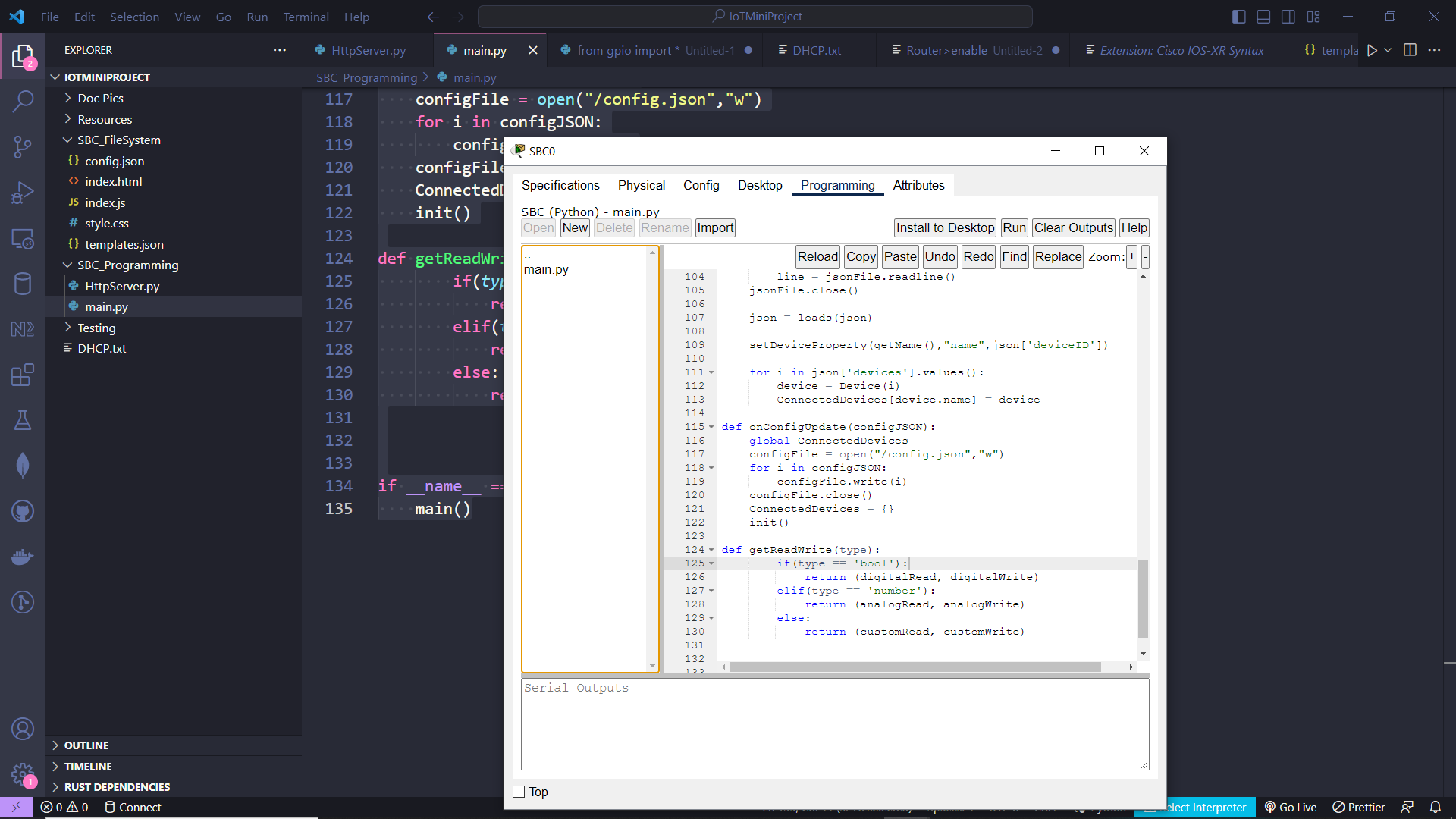
elif(type == 'number'):

return (analogRead, analogWrite)

else:

return (customRead, customWrite)

if \_\_name\_\_ == "\_\_main\_\_":

 main()

**Fig 8.1 Configuring SBC Logic**

**8.2 CONFIGURING HTTP SERVER**

The HTTP Server can configured using Packet Tracer's http package. We create an HTTP server and set it to listen to port 80. The Server serves files available in SBC file System and sanitizes the configuration acquired from GET Request send by HTTP Client device

**8.2.2 HTTP LOGIC**

from http import \*

def urlDecode(url):

url = url.replace("%22",'"')

url = url.replace("%7B",'{')

url = url.replace("%7D",'}')

url = url.replace("%20",' ')

return url

def initHttp(callback):

HTTPServer.start(80)

HTTPServer.route('/\*',sendFile)

HTTPServer.route('/setConfig/\*', lambda u,r : setConfig(u,r,callback))

def sendFile(url,response):

print("Requested : "+url)

if(url == "/"):

response.sendFile("/index.html")

else:

response.sendFile(url)

def setConfig(url,response,callback):

callback(urlDecode(url.strip("/setConfig/")))

def call(string):

print(string)

if \_\_name\_\_ == "\_\_main\_\_":

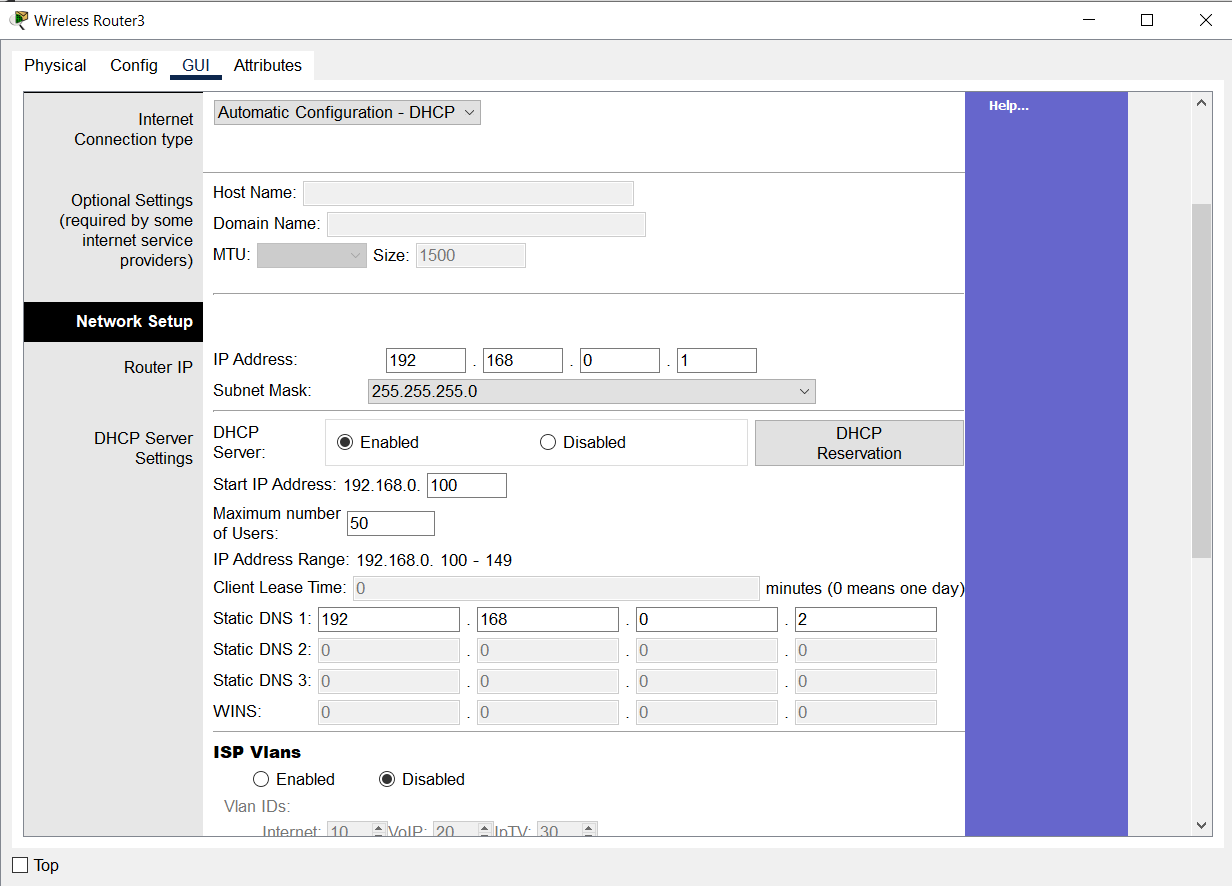
from time import \*

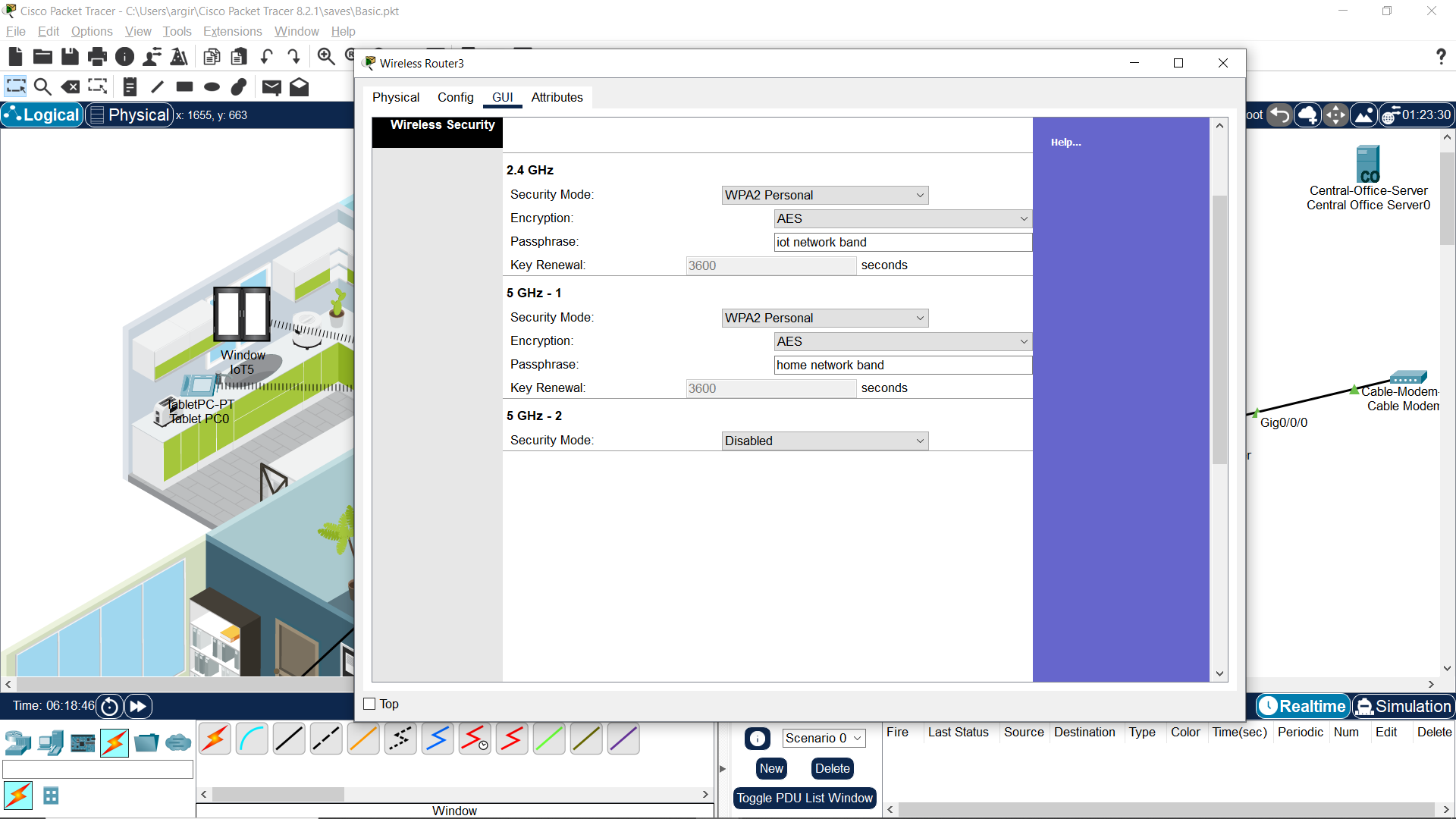
initHttp(call)

while True:

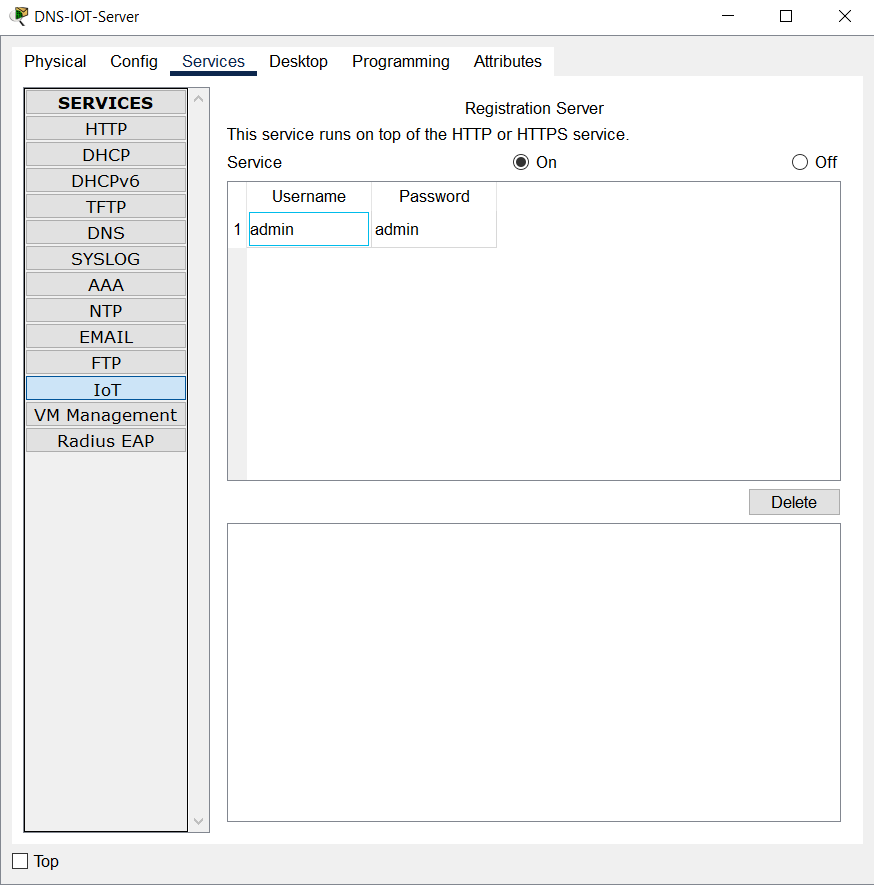
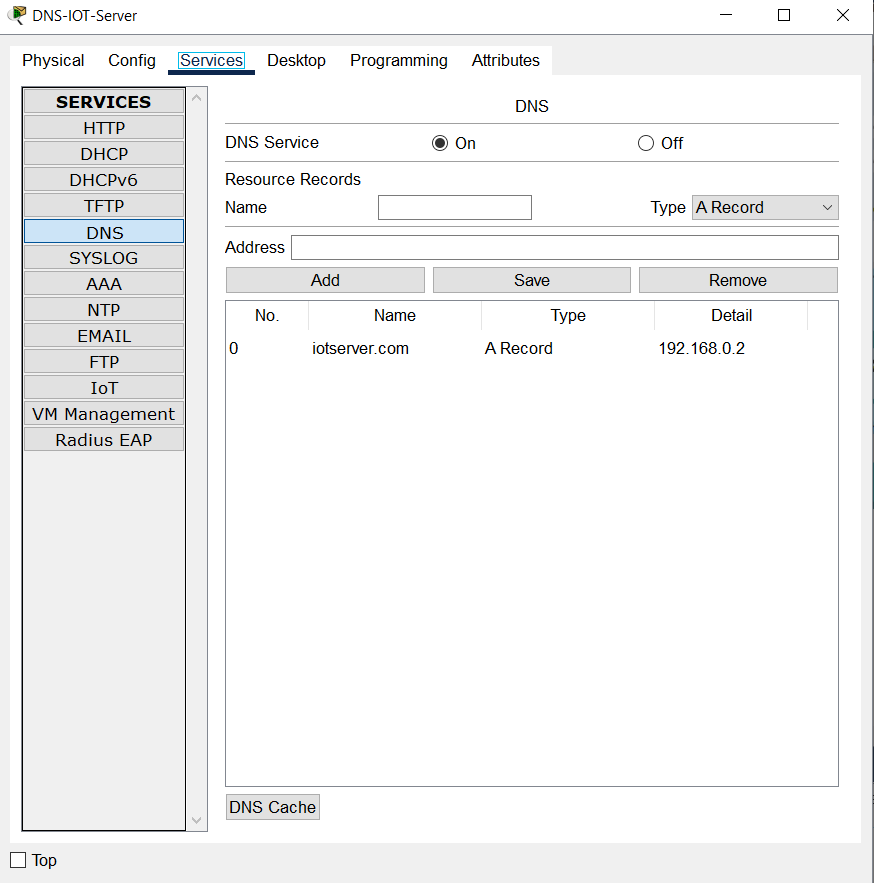
sleep(100)

**8.3 CONFIGURING HOME ROUTER**

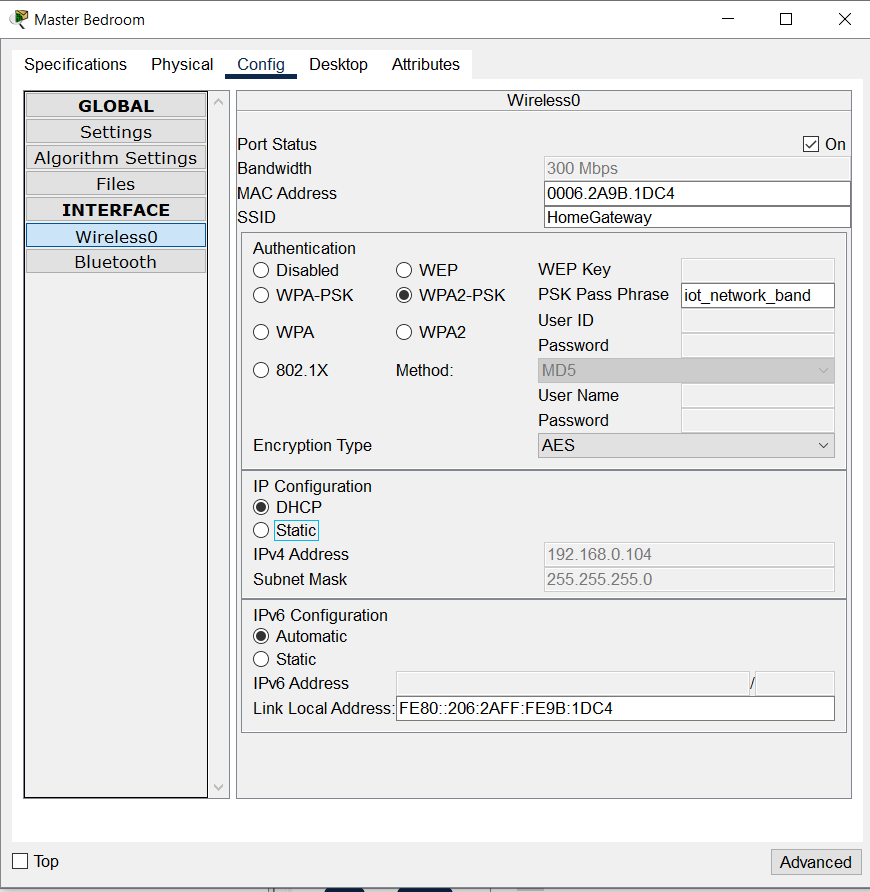


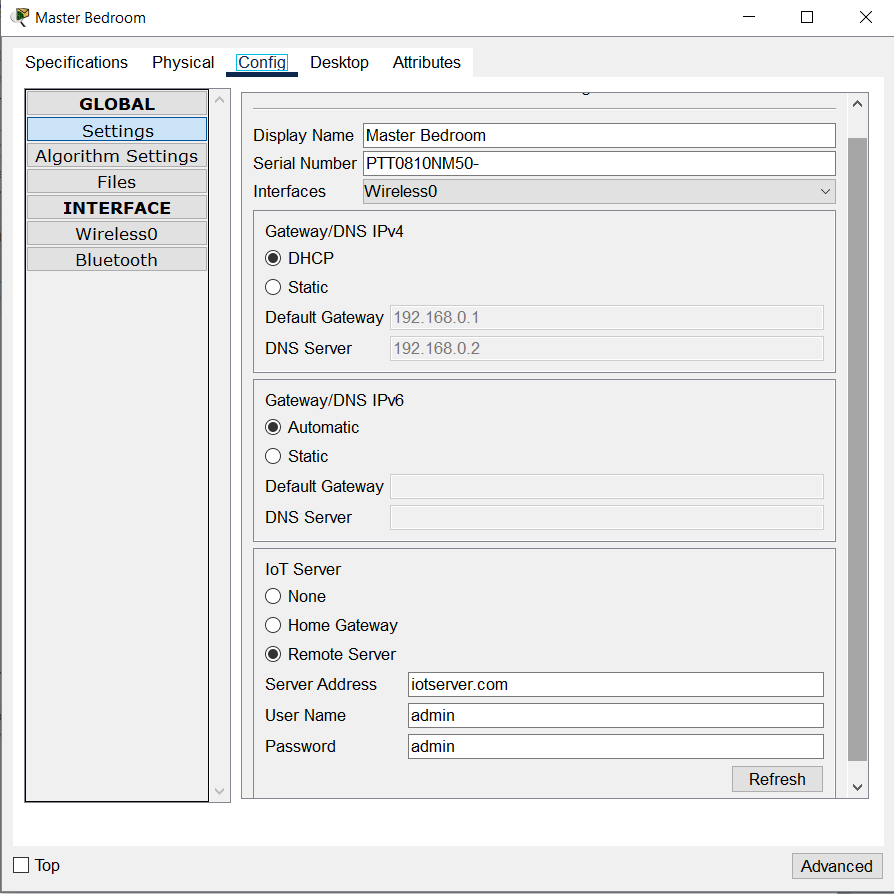
****

**8.4 CONFIGURE DNS AND IOE SERVER**

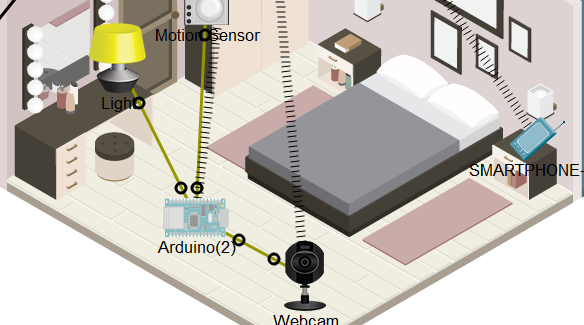


**8.5 CONECTING SBC TO ROUTER**

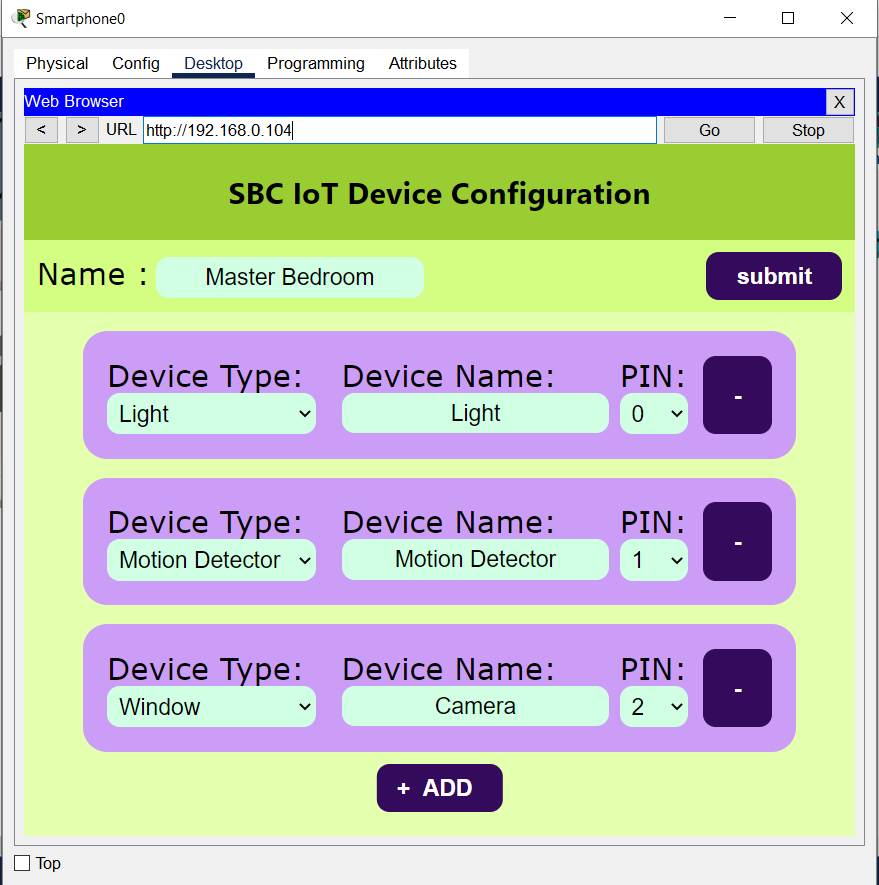


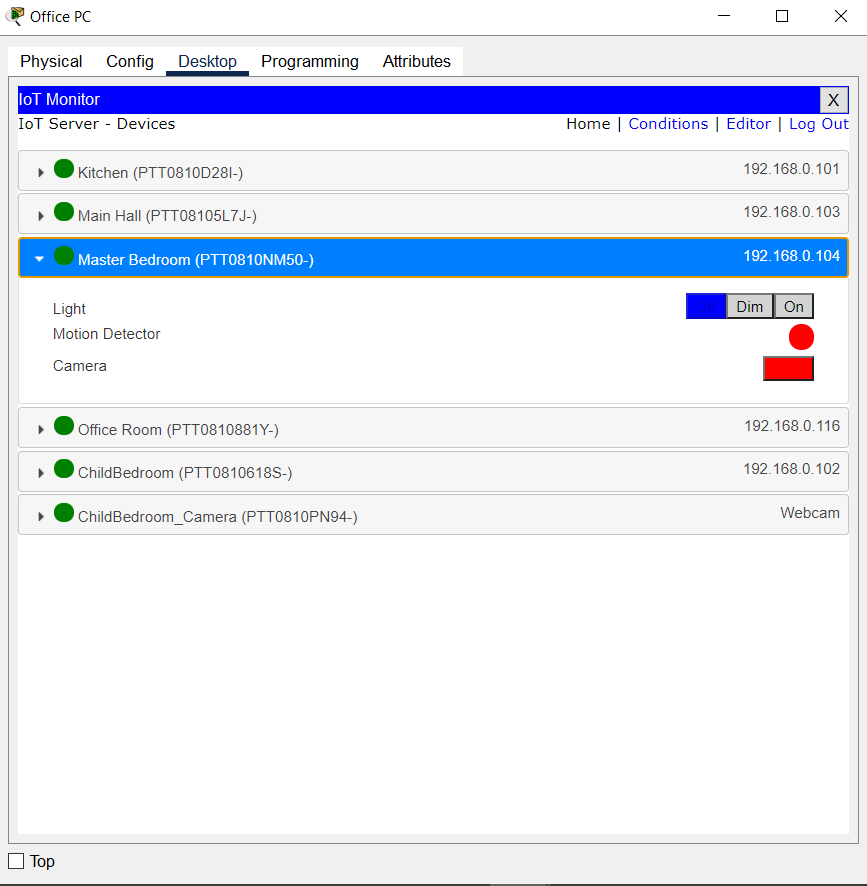


**8.6 CONNECTING DEVICES**



**8.7 CONFIGURING DEVICES**





**8.8 ADDING CONDITIONS**

# 

# **8.9 CONFIGURING NAT**

# On ISP Router CLI :

# ISPR#conf t

# ISPR(config)#int gig0/0/0

# ISPR(config-if)#ip nat inside

# ISPR(config-if)#exit

# ISPR(config)#int gig0/0/1

# ISPR(config-if)#ip nat outside

# **8.10 CONFIGURING RIP**

# On ISP Router CLI:

# ISPR#conf t

# ISPR(config)#router rip

# ISPR(config-router)# version 2

# ISPR(config-router)# network 192.168.0.0 192.168.30.0

# ISPR(config-router)#end

# CONCLUSION AND FUTURE ENHANCEMENT



## CHAPTER – 10

**CONCLUSION AND FUTURE ENHANCEMENT**

* 1. **CONCLUSION**

In conclusion, our project aims to address the challenges faced in home automation due to the high cost, limited variety, and potential hardware failures of smart IoT devices. We propose Single-Board Computers (SBCs) as a cost-effective and reliable alternative to IoT devices, such as Arduino and Raspberry Pi, which offer homeowners greater flexibility and control over their home automation systems. Through comprehensive analysis and experimentation, we will demonstrate the advantages of SBCs in terms of affordability, versatility, and modularity. By promoting the widespread adoption of SBCs, we strive to make home automation accessible to a broader range of households, empowering homeowners to create personalized and sustainable automation experiences. Our project seeks to enhance convenience, control, and efficiency within living spaces, revolutionizing the way people interact with their homes.

## FUTURE ENHANCEMENT

Furthermore, the future enhancement can be made by using deep learning algorithm which may give better solution. For future work, use XGBoost for heart disease prediction in children and compare if better accuracy can be achieved. If features are properly managed, then there will be significant performance in the classification of heart disease prediction. In future studies, the outcomes of our proposed methods will serve as the standard performance results on heart disease.

APPENDIX



## APPENDIX

**A.1 SOURCE CODE**

**A.1.1 HTML SOURCE CODE**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>IoT Configuration</title>

<link rel="stylesheet" href="/style.css">

<script defer src="/index.js">

</script>

</head>

<body>

<header>

<h2>SBC IoT Device Configuration</h2>

</header>

<div id="info">

<div>

<label for="deviceName">Name :</label> <input id="deviceName" type="text" value="">

</div>

<button onclick="submit()">submit</button>

</div>

<div id="templateForm" hidden>

<form>

<div>

<label for="deviceType">Device Type:</label>

<select id="deviceType" name="deviceType"></select>

</div>

<div>

<label for="name">Device Name:</label>

<input type="text" name="name">

</div>

<div>

<label for="pin">PIN: </label>

<select id="pin" name="pin" onclick="getAvailable(this)" onchange="unallocate(this)" onfocus="this.\_\_data\_previous = this.value"></select>

</div>

<button type="button" id="delete" onclick="remove(this.parentElement)">-</button>

</form>

</div>

<div id="forms">

</div>

<button id="addButton" onclick="addEmpty()" disabled>ADD</button>

</body>

</html>

**A1. CSS SOURCE CODE**

html, body{

margin:0px;

padding:0px;

font-family: system-ui;

background-color: hsl(80, 100%, 84%);

}

header{

background-color: hsl(80, 61%, 50%);

padding: 0.1rem;

color:black;

text-align: center;

font-weight:bolder;

}

button{

padding: 0.5rem 1.5rem 0.5rem 1.5rem;

background-color: hsl(271, 81%, 20%);

border-radius: 10px;

border: 0px;

display: block;

font-size: large;

font-weight: bold;

color: hsl(0, 0%, 100%);

cursor: pointer;

}

button:disabled{

cursor: crosshair;

}

#addButton{

margin-top: 10px;

position: relative;

padding-left: 1.0rem;

left: 50%;

transform: translateX(-50%);

}

#addButton::before{

content: '+';

position: relative;

font-size: large;

padding-right: 10px;

}

#info{

display: flex;

justify-content: space-between;

align-items: center;

background-color: hsl(80, 98%, 75%);

padding: 10px;

}

form{

display: flex;

justify-content: space-evenly;

align-content: center;

width: 80%;

margin:auto;

margin-top:15px;

padding: 1em;

font-size: larger;

border-radius: 1em;

background-color: hsl(271, 83%, 79%);

}

label{

font-size: x-large;

font-family: Verdana;

}

select, input[type='text'] {

font-family: Helvetica;

align-self: center;

font-size: large;

padding: 0.3em;

text-align: center;

background-color: hsl(143, 100%, 91%);

border: 0px;

border-radius: 10px;

}

**A1. JS SOURCE CODE**

var config;

var templates;

var pins = {'0':false,'1':false,'2':false,'3':false,'4':false,'5':false,'A0':false,'A1':false,'A2':false,'A3':false}

var templateForm = document.querySelector("#templateForm form")

var types = [];

var deviceName = document.querySelector("#deviceName");

var forms = document.querySelector("#forms");

for(let i in pins){

let option = document.createElement('option');

option.value = i;

option.innerText = i;

templateForm.querySelector("#pin").appendChild(option);

}

async function init(){

await fetch('/config.json')

.then( response => response.json())

.then( json => config = json);

await fetch('/templates.json')

.then(response => response.json())

.then(json => templates = json);

for(let i in templates){

types.push(i)

let option = document.createElement('option');

option.value = i;

option.innerText = i;

templateForm.querySelector("select#deviceType").appendChild(option);

}

deviceName.value=config["deviceID"];

let devices = config.devices;

for(let i in devices){

add(devices[i].deviceType,devices[i].config.name,devices[i].pin)

}

document.querySelector("#addButton").disabled = false;

}

function add(type,name,pin){

let form = templateForm.cloneNode(true);

form.elements["deviceType"].value = type;

form.elements["name"].value = name;

form.elements["pin"].value = pin;

pins[pin]=true;

forms.appendChild(form);

}

function addEmpty(){

let available = [];

for(let i in pins){

if(pins[i] != true) available.push(i);

}

if(available.length>0) add(types[0],"",available[0]);

else return;

}

function getAvailable(select){

for(let i of select.childNodes){

i.hidden = pins[i.value]

}

}

function unallocate(select){

if(!select.\_\_data\_previous || select.\_\_data\_previous === select.value)

return;

pins[select.\_\_data\_previous]=false;

pins[select.value]=true;

select.\_\_data\_previous = select.value;

}

function remove(device){

pins[device.elements.pin.value]=false;

device.remove();

}

function submit(){

let devices = {}

for(let form of forms.querySelectorAll("form")){

let name = form.elements.name.value;

let deviceType = form.elements.deviceType.value;

let pin = form.elements.pin.value;

if(!name || !deviceType || !pin){

alert("No Empty/Repeated Data Permitted!!");

return;

}

let device = JSON.parse(JSON.stringify(templates[deviceType]));

device.config.name = name;

device.pin = pin;

device.deviceType = deviceType;

devices[name] = device;

}

let newConfig = JSON.stringify({

"deviceID" : deviceName.value,

"devices" : devices

});

fetch("/setConfig/"+newConfig);

}

init()

REFERENCES



## REFERENCES

1. Jacob B. Hjelmborg, James W. Vaupel, Nancy L. Pederson, Mac McGue, Karre Christensen, Marku Koskenvuo, Axel Skytthe [2019] “Genetic influence on human lifespan and longevity”.
2. Oliver Berry, Campbell Davies, Jessica Farley, Simon Jarman [2019] “A Genomic Predictor of lifespan in vertebrates Benjamin Mayne”.
3. Linda Mary, John Ashima Sharma, Siddhant Gujarathi [2019], “Detector and Predictor System for lifeaspan using Naives Bayes and Decision Tree Algorithm”.
4. Barros, Rodrigo C, Basgalupp, Carvalho, Freitas [2012], “A Survey of Evolutionary Algorithms for Decision-Tree Induction”, *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 42.
5. The Prediction of heart disease using Naïve bayes classifier International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056 Volume: 06 Issue: 03 | Mar 2019.
6. “Developing Machine Learning Algorithms for the Prediction of Early Death in Elderly Cancer Patients: Usability Study. JMIR Cancer”[2019]
7. S.R.Bhagya Shree, H.S.Seshadri, [2019],” Prediction using Naïve Bayesian Classifier”, *Neural Computing & Applications*.

26