

2021

RANIA: A Framework for a Modular, Voice Enabled, GeronTechnology-Centric, Smart-Home System

Emily Francis

West Virginia University, ef00006@mix.wvu.edu

Follow this and additional works at: <https://researchrepository.wvu.edu/etd>



Part of the [Computer and Systems Architecture Commons](#), and the [Digital Communications and Networking Commons](#)

Recommended Citation

Francis, Emily, "RANIA: A Framework for a Modular, Voice Enabled, GeronTechnology-Centric, Smart-Home System" (2021). *Graduate Theses, Dissertations, and Problem Reports*. 10220.

<https://researchrepository.wvu.edu/etd/10220>


This Thesis is protected by copyright and/or related rights. It has been brought to you by the The Research Repository @ WVU with permission from the rights-holder(s). You are free to use this Thesis in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you must obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/ or on the work itself. This Thesis has been accepted for inclusion in WVU Graduate Theses, Dissertations, and Problem Reports collection by an authorized administrator of The Research Repository @ WVU. For more information, please contact researchrepository@mail.wvu.edu.

2021

RANIA: A Framework for a Modular, Voice Enabled, GeronTechnology-Centric, Smart-Home System

Emily Francis

Follow this and additional works at: <https://researchrepository.wvu.edu/etd>

 Part of the [Computer and Systems Architecture Commons](#), and the [Digital Communications and Networking Commons](#)

RANIA: A Framework for a Modular, Voice Enabled, GeronTechnology-Centric, Smart-Home System

Emily Francis

Thesis submitted to the
Benjamin M. Statler College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of

Master of Science
in
Computer Science

Brian Powell, Ph.D
Katerina Goseva-Popstojanova, Ph.D
Layth Sliman, Ph.D
Ramana Reddy, Ph.D, Chair

Lane Department of Computer Science and Electrical Engineering

Morgantown, West Virginia
2021

Keywords: Amazon Alexa, GeronTechnology, REST API, Smart Home

Copyright 2021 Emily Francis

Abstract

RANIA: A Framework for a Modular, Voice Enabled, GeronTechnology-Centric, Smart-Home System

Emily Francis

In the United States, the proportion of senior citizens is expected to rise significantly over the next few decades. This increasing number of senior citizens combined with the increasing demand for at home healthcare workers is putting a strain on the elderly healthcare system.

Smart-home healthcare technology – such as smart medication dispensers, fall detection systems, smart pantry systems, etc. – has the potential to alleviate this strain on the elderly healthcare system. Smart devices can give an individual more autonomized and personalized surveillance of their health and well-being. While these devices are beneficial as standalone devices, they would be more accessible and extensible under a single, modular smart-home system.

For these reasons, the GeronTechnology Lab at West Virginia University is developing the Residents Aware Network for Intelligent Assistance (RANIA) system. RANIA is a modular smarthome system that is designed to integrate numerous types of devices for aiding elderly residents in aging at home.

This thesis presents a generalized framework for integrating RANIA devices within the RANIA smart-home system. The novel aspects of this project are the one-click device connection/disconnection and automated package management for devices connected to the RANIA Hub. Other functionalities outlined include system messaging and Amazon Alexa Echo system integration.

Acknowledgements

A special thanks to Dr. Ramana Reddy for accepting me into the GeronTechnology lab and for advising me as the chair on my committee. More thanks go out to the other advisors on my committee, Dr. Brian Powell, Dr. Katerina Goseva-Popstojanova, and Dr. Layth Sliman for serving on the committee for this project.

Contents

Acknowledgements	iii
List of Figures	vi
List of Tables	vii
1 Introduction	1
1.1 Problem Statement	1
1.2 Project Aims	2
2 Related Work	3
2.1 State of the Art of Smart Homes	3
2.2 General Smart Home Model	3
2.3 Eldercare Smart Homes	5
2.4 Voice Assisted Smart Homes	6
2.4.1 Developing Alexa Skills	7
2.5 Previous RANIA Paper	7
3 Novelty	9
4 Methodology	10
4.1 Overall Architecture of RANIA System	10
4.2 Implementations	13
4.2.1 RANIA Hub	13
4.2.2 RANIA Database	14
4.2.3 RANIA Web Application	14
4.2.4 RANIA App Store Application	20
4.2.5 RANIA Device	22
4.2.6 Amazon Alexa Integration	25
5 Results	28
5.1 Connecting the RANIA Device to the RANIA Hub	28
5.1.1 Using the RANIA Web Server Companion Package	29
5.1.2 Alexa Voice Commands	31
5.2 Disconnecting the RANIA Device From the RANIA Hub	34

6	Conclusions	35
6.1	Project Results Conclusion	35
6.2	Limitations	35
6.3	Future Work	36
	References	37

List of Figures

2.1	General Smart Home Model	4
2.2	Alexa Enabled Smart Home Architecture	6
2.3	Original RANIA Architecture Design	8
4.1	Architecture Diagram	12
4.2	RANIA Database Diagram	14
4.3	RANIA Web Application Directory Diagram	15
4.4	RANIA Web Application GUI Diagram	16
4.5	RANIA App Store Directory Diagram	20
4.6	RANIA Device GUI Diagram	23
5.1	Connect Device Results	29
5.2	Using Companion App Results	30
5.3	Disconnect Device Results	34

List of Tables

4.1	Project Components	13
4.2	Server Specifications	13
4.3	Raspberry Pi 4 Model B Features	22
4.4	Medication Schedule Database Table Description	24
5.1	Requesting the Time for Taking a Medicine	31
5.2	Requesting the Name of a Medicine for a Certain Time	31
5.3	Requesting the Instructions for a Certain Medicine	32
5.4	Adding an Alarm to the Schedule	32
5.5	Adding an Alarm to the Schedule	33
5.6	Requesting the Instructions for a Certain Medicine	33

Chapter 1

Introduction

1.1 Problem Statement

By 2060, at least 25% of the American population is projected to be over 65 years of age. At this age bracket, 90% of people have at least one chronic condition that needs personalized care. Along with the extra demand for customized health care, the Covid-19 pandemic has encouraged the safety of choosing health care options that can be provided in the person's own - such as home health care assistants. The projected Job outlook from 2019-2029 of Home Health and Personal Care Aides is 34% - much faster than the average job growth rate of 4%. However, at the median pay of \$27,000 per year, they are a very costly expense for the average family – who had a median income of \$68,000 in 2019.

These figures indicate that smart-home healthcare technology has the potential to fill a prevalent, growing need in the eldercare system. Smart home systems can coordinate devices that tend to the personalized health care needs of elderly individuals, mitigating the costly expenses of a home health care aid while giving the resident more agency over their own health care.

To contribute to the efforts of smart home health care for the elderly, the gerontechnology lab at West Virginia University has been developing the Residents Aware Network for Intelligent Assistance, RANIA, system. This paper presents a general framework for managing devices that are part of the RANIA system.

1.2 Project Aims

The general goal of this project was to establish a framework for the RANIA smart home system that coordinates RANIA smart devices for a resident. This means that a method is needed to add and remove devices from the RANIA System, a method is needed to view and update the device data. In addition to needing these aforementioned functionalities, this system will be voice-enabled - meaning a voice assistant is needed to retrieve device and data and control the connected devices. Based on these desired functionalities, the following 4 project aims were developed.

1. **Provide a method for connecting/disconnecting a RANIA device from the hub in as few steps possible for the resident**

It must simple for a resident to add or remove devices based on their individual health-care needs.

2. **Provide a method for sending device data between each device and the resident's RANIA hub**

The RANIA hub needs to be able to receive, store, and send the data gathered from the devices that are connected to the resident's RANIA Hub.

3. **Provide a method for allowing the resident to view RANIA Device information on the RANIA Hub** The resident should have access to an interface for their RANIA Devices through their RANIA Hub web application portal.

4. **Provide a method for allowing an Amazon Alexa Echo device to retrieve and update RANIA device data in the RANIA Hub**

The resident should be able to update device data with natural language for more effortless RANIA smart home control.

Chapter 2

Related Work

2.1 State of the Art of Smart Homes

There are multiple approaches when developing a smart home system, but the typical approach is to incorporate sensors within the home to collect multiple different types of data. These sensors would gather data and forward it to a central computer that receives and analyses it. In [1], the authors identified 3 main categories of smart home system implementations. Smart homes that provide survey the home to ensure the resident's safety from emergencies, smart homes that collect and analyze multimedia such as photos and videos, and smart homes that aim to collect data related to a resident's state of health and wellbeing. There are also different subdisciplines of smart-home healthcare, including eldercare; where this project falls under.

2.2 General Smart Home Model

In [2] through a literature review, the authors identified 5 parts to a smart home; **Smart Objects**, Smart Object **Hubs**, the **Cloud**, **Third Party Applications**, and **Security**. This is visualized in Figure 2.1, and described below.

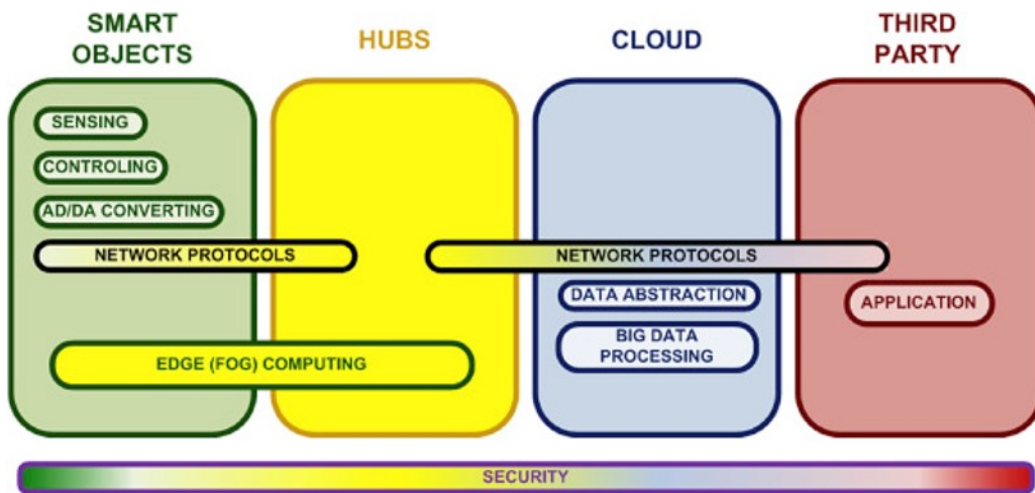


Figure 2.1: General Smart Home Model

Smart Objects: These are the devices that are within the home that monitor the state of the home with their built-in sensors. They can also perform actions through any actuators they might have. The data that the devices acquire are sent to the Hub, either with or without any data processing in the device.

Hubs: The Hub is an edge level system that collects the data from the devices and processes /interprets the data before sending it to the Cloud. It is typically physically located within within the resident’s home and is responsible for interpreting the data that is collected from the devices and deciding if any action needs to be taken.

Cloud: The Cloud is located remotely and stores all the data received by the Hub. It is responsible for data abstraction and any big data processing that might be necessary for the system.

Third Party Applications: Third party applications can be employed for the processes of scheduling, regulating, and balancing loads.

Security: Security measures such as authorization and data encryption need to be in place for all levels of a smart home system.

2.3 Eldercare Smart Homes

In [3], the authors identify 6 major services that smart homes for eldercare typically provide to help assist and monitor a resident.

Physiological Monitoring:

This entails the the monitoring a person's biological processes. This includes monitoring a person's vital signs, such as blood pressure, sugar level, respiration, body temperature, etc. The data gathered from these kinds of sensors can be sent to a hub which would then determine if medical attention is needed for the resident.

Functional Monitoring:

This entails assessing the functionality of the human body. Examples include monitoring a person's emotional state, food intake amounts, manner of walking, and other daily activities. With this type of information, the hub can assess if the resident's mood is changing, if they are eating properly, if they have fallen and need tended to, or if any of their other daily routine activities are changing.

Safety Monitoring and Assistance:

This entails monitoring for any things that could cause the resident direct harm, such as fires in the home or gas leaks. When these should happen, the system would be able to react by either notifying the authorities or potentially cutting of the source of the issue.

Security Monitoring and Assistance:

This entails monitoring for and responding to any suspicious activity that may be occurring either in or around the resident's home.

Social Interaction Monitoring and Assistance:

This entails providing convenient means for the resident to be able to communicate with friends and family as well as monitoring aspects such as the frequency and duration of these interactions.

Cognitive and Sensory Assistance:

These kinds of devices help the resident with day-to-day tasks such as taking medication or locating the items which the resident has lost.

2.4 Voice Assisted Smart Homes

Voice assisted smart homes are expected to increase in development by 1000% between 2018 and 2023, likely in part due to the easy and natural interactivity that vocal smart-home assistants provide for a resident. Because of the intuitive interactivity that a voice assistant provides a resident, the inclusion of a voice assistant in a smart home for elderly residents is essential.

Integrating a voice assistant in a smart home is typically implemented through Representational State Transfer architecture. The smart-home hub would need to have paths available for the voice assistant to use in order to access smart-home system information. Once these paths are available, the voice assistant can access them through a communication protocol, which is typically HTTP, HTTPS or MQTT.

One of the most popular voice assistants that enable voice control for smart homes is Amazon Alexa, which makes up roughly 70% of smart speakers in the Western market as of 2021. For this project, Amazon Alexa is implemented for interacting with the RANIA Smart-Home System. [4]. In [5] and [6], the authors utilize the architecture shown in 2.2. These projects utilize a Raspberry Pi 3 model B, Alexa Voice Services, and Ngrok.

- **Raspberry Pi** Connects physically to led strips to control them and provides the paths for the Alexa Voice Services to access.
- **Alexa Voice Services** An Alexa skill provides voice interfacing for the resident to control the led strips in the system. The skill interprets what the user wants to accomplish and then the skill accesses the Raspberry Pi through Ngrok services.
- **Ngrok** Ngrok is a service that provides secure communication tunnels between the internet and different devices that are on a user's local network.

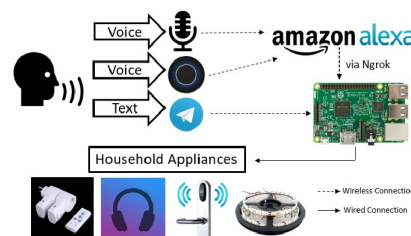


Figure 1. Block Diagram of the System.

Figure 2.2: Alexa Enabled Smart Home Architecture

2.4.1 Developing Alexa Skills

According to Amazon's Alexa Skills documentation, there are 7 key concepts in developing an Interaction Model for an Alexa Skill: Wake Word, Launch Word, Invocation Name, Utterances, Prompts, Intents, Slot Values.

1. **Wake Word:** The wake word indicates for Alexa to start listening to the speaker. Amazon Echo devices can be awakened by saying the predefined wake word. This is either "Alexa", "Echo" or "Computer".
2. **Launch Word:** The launch word signals to Alexa that a skill invocation should follow, such as "open" or "ask". Launch words are used to invoke a specific skill, with or without a request from the user.
3. **Invocation Name:** The invocation name is the name of a skill the user wants to use. Skill names need to be two words in length and cannot include reserved words such as "Amazon" or "Alexa".
4. **Intent:** An intent represents an action that fulfills a user's request.
5. **Utterance:** Utterances are statements used to invoke intents and must be predefined by the developer in the interaction model. Multiple utterances may be defined to invoke an intent.
6. **Prompt:** Prompts are requests for information that Alexa asks the user for.
7. **Slot Value:** A slot provides a way to allow an intent to accept different potential values in an utterance.

2.5 Previous RANIA Paper

This project was created and implemented under the Gerontechnology lab at West Virginia University. The Natural Language Interfacing framework is part of the Resident Aware Network for Intelligent Assistance (RANIA) initiative. RANIA is a smart home environment WVU's Gerontechnology lab is developing for elderly individuals to help them live more comfortably at home and give their loved ones peace of mind. In the original published paper about the RANIA system, the system is outlined in Figure 1. In this paper,

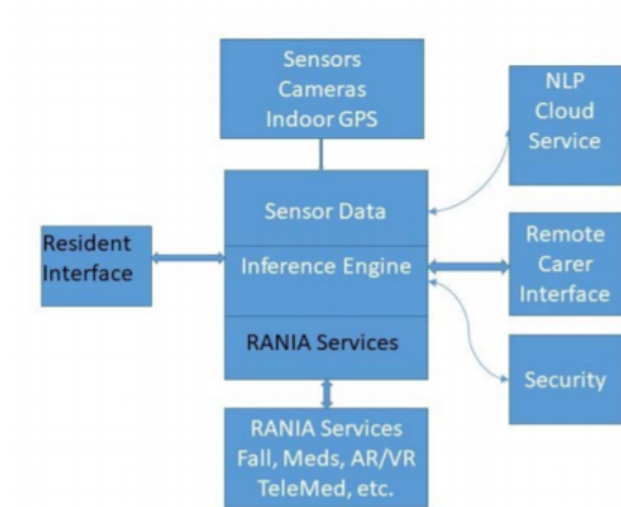


Figure 2.3: Original RANIA Architecture Design

Chapter 3

Novelty

The main novel aspect of this project is the development of a modular architecture for managing the applications and devices that are part of a resident's smart home system. Residents using the RANIA system are able to log into their local servers and add/remove the applications corresponding to their RANIA devices at their discretion. The state of the art of smart home frameworks does not provide the means for a resident to add and remove devices and applications based on their home care needs. Applications created for use as part of the RANIA system are hosted on a web server that acts as an application store for the resident. With the framework introduced in this project, the resident is able to install new device application packages to their servers, update packages on their server, and uninstall packages they do not need anymore.

Chapter 4

Methodology

4.1 Overall Architecture of RANIA System

The architecture of the RANIA system presented in this paper is visualized in Fig 4.1. For quick reference, a table with all of the project components used in implementation is included in table 4.1. While this paper outlines how an external caregiver is included in the system, this thesis does not allow for external caregiver access.

RANIA System Components:

1. RANIA Hub

The RANIA Hub hosts the database to store data sent from the devices as well as the RANIA Web Application that provides the resident interfacing for accessing the device data. Ideally this would be located locally within the home of the resident, but due to network security constraints within the GeronTechnology lab, the RANIA Hub needed to be implemented remotely. The specifications of the RANIA Hub are briefly summarized in table 4.2 below.

2. RANIA Devices

The RANIA Smart Home devices collect data gathered from the resident or the resident's environment and sends the data to the RANIA server. The devices may or may not have a screen themselves and each device will have a corresponding package that the resident can use to view the device data on the RANIA Server application GUI. this project demonstrates a mock RANIA device to keep track of medication alarms

created on a Raspberry Pi 4 Model B. The specifications for the Raspberry Pi can be found in the table below.

3. **RANIA Web Application**

The RANIA Web Application provides a portal for the resident to view RANIA Devices that are connected to their account. It provides a Graphical User Interface for the user to view the data collected from their devices. It also provides a REST API to accept and respond to HTTPS requests from the resident's local Amazon Alexa ECHO device and from the resident's devices that are connected to the account. The API also is connected to the database that is hosted on the RANIA Hub and can interact with the database via the API.

4. **Amazon Alexa Echo**

The Amazon Alexa Echo provides a way to access RANIA device data on the RANIA server and update the device data, via an Alexa skill that is provided by the developers of the RANIA system. The resident can invoke the skill to request or update data related to a RANIA device. After a skill intent is realized by the skill, the skill sends an HTTPS request to the database on the RANIA Hub to request or update the information per the user's request. When a new RANIA device is created, the skill will need to be updated to provide voice access to the new device.

5. **RANIA Application Server**

The RANIA Application Server hosts the RANIA App Store Application that is set up on a subdomain of raniahouse.com. For the purposes of this project the RANIA Application Server and the RANIA Hub are the same physical remote Linux server. In a production environment, the RANIA Application Server and the RANIA Hub would be two separate physical servers, with the application server being global access for all RANIA residents.

6. **RANIA App Store Application**

The RANIA App Store Application hosts device packages that the RANIA devices request to be downloaded on the resident's RANIA Web Application. These packages can be requested via an API path on the back-end of the application. The RANIA Application Server and the RANIA App Store Application will be managed by the

developers of the RANIA system. When new RANIA devices are developed, the device companion package will need to be uploaded to the RANIA App Store Application.

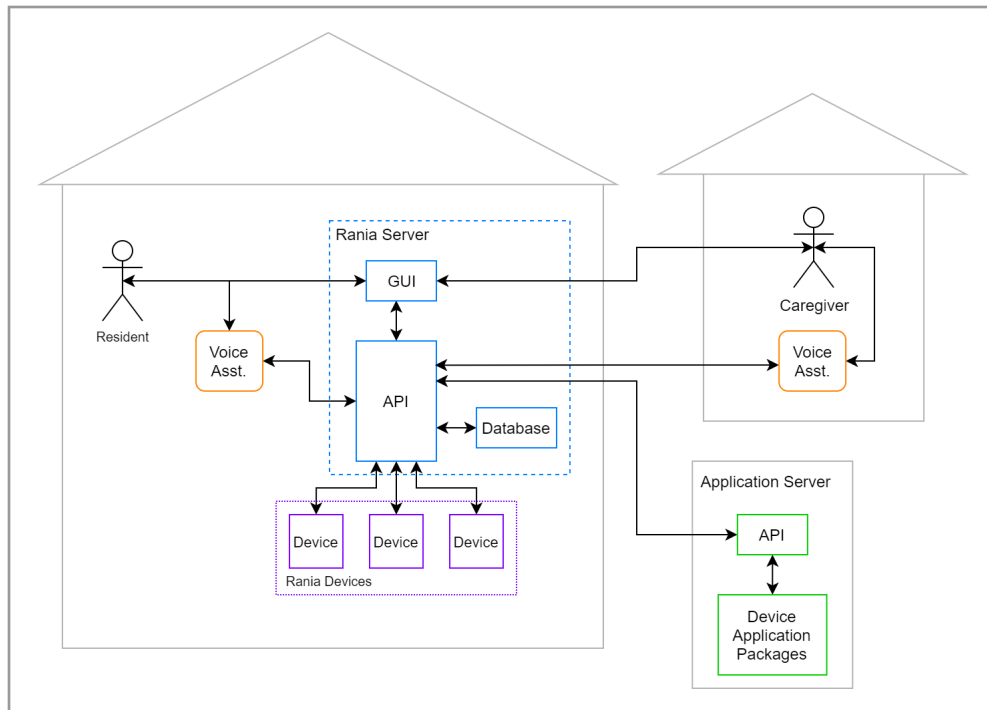


Figure 4.1: Architecture Diagram

4.2 Implementations

This section describes how each part of the RANIA System Components have been implemented. For quick reference of the components used in this project, refer to table 4.1.

Table 4.1: Project Components

RANIA Hub	Remote Linux Server
RANIA Web Application	Node.js Express Server Application
RANIA App Store Application	Node.js Express Server Application
RANIA Database	MySQL Database
Example Device	Raspberry Pi Medication Alarms
Voice Assistant	Amazon Alexa
Communication Protocol	HTTPS Requests

4.2.1 RANIA Hub

The RANIA Hub is established on a remote Linux server that hosts the RANIA Web Application, RANIA App Store Application, and the RANIA Database. Ideally the RANIA Hub would be located physically within the resident's home, but due to network security restraints within the lab that this thesis was produced in, the RANIA Hub could not be established locally in the lab, and had to be remote. The specifications of the remote server used in this thesis can be found for quick reference in Table 4.2. The processor is allotted 1 vCore, has 1024 Megabytes of RAM, 25 Gigabyte SSD, runs on Ubuntu 21.04 x64 operating system, and is located in New Jersey.

Table 4.2: Server Specifications

Processor	1 vCore
Memory	1024MB of RAM 25 GB SSD
Operating System	Ubuntu 21.04 x64
Location	New Jersey

4.2.2 RANIA Database

The RANIA Database stores RANIA user information and data sent from RANIA Devices. When there are no RANIA devices connected to a resident's RANIA Hub the RANIA Database contains a table for RANIA resident information, caretaker contact information, and messages. When a RANIA device is connected to the RANIA Hub, any relevant tables that the RANIA device needs are added to the RANIA Database. When a device is removed from the RANIA Hub, any tables relevant to the RANIA device being disconnected will be removed from the RANIA database.

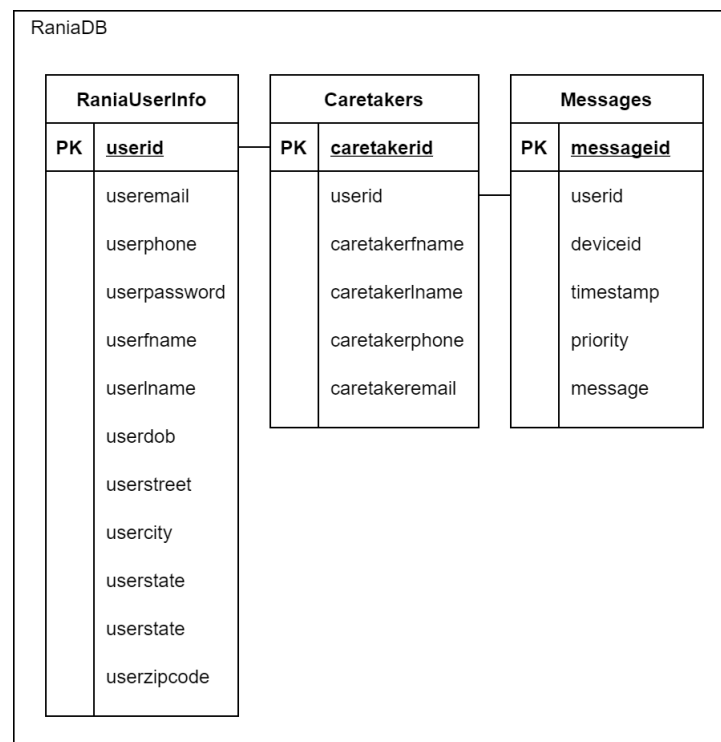


Figure 4.2: RANIA Database Diagram

4.2.3 RANIA Web Application

Application Layout The RANIA Web Application is a Node.js web application with an Express REST API and simple front end. The layout of the RANIA Web Application is provided in Figure 4.3. The root directory contains the index.js files that contains all defined API paths that are located in the API Routes Folder in the Web App Resources Directory, the Node Modules that are required for the application to run, a Web App Resources folder, and a Devices Folder that contains all of the device companion packages that have been

connected to the application. The Web App Resources Folder contains 4 folders, one for the RANIA Web Application's API Routes, one for Images, one for Pages and one for Scripts. Each Device folder in the Devices folder has a folder for the companion package's Images, Folders, and Scripts.

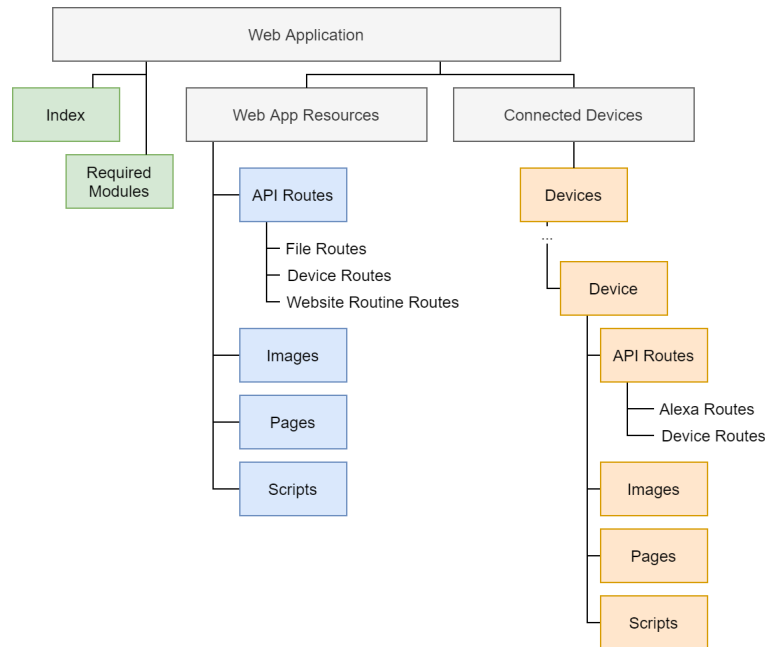


Figure 4.3: RANIA Web Application Directory Diagram

GUI Layout The GUI layout of the RANIA Web application is available for quick reference in Figure 4.4. The Landing Page greets the resident when they access the domain associated with their RANIA account through a web browser. The resident can then log in to their account to access their personal information and device data. On a successful login, the resident is redirected to the Devices Page, which shows a table of all devices connected to the RANIA Web Server and allows the user to open the device companion packages. From the **Devices Page**, the resident can either move to the Messages Page, Settings Page, or they can log out. The **Messages Page** displays messages that are stored in the RANIA Hub's Database. From the Message Page, the resident can either move back to the Devices Page, move to the Settings Page, or log out. The **Settings Page** is the home page for the settings pages, where the user can access and update personal information that is stored by the RANIA Hub's Database. From the Settings Page, the user can move to the Personal Information Page, the Contacts Page, the Devices Page, or log out. The **Personal Information Page** displays to the resident and allows them to update their personal information

such as their name, phone number, email, and home address. From the Personal Information Page, the resident can either move to the Contacts page, the Settings Page, the Devices Page, or log out. The **Contacts Page** displays to the resident and allows them to add, update, and remove contact information, such as a contact's name, phone number, and email. From the Contacts Page, the resident can either move to the Contacts page, the Settings Page, the Devices Page, or log out.

The user's home page of the RANIA Web Application contains a table that shows the resident each device that is connected to the RANIA Hub. This table is populated by reading through each device name in the devices directory and inserting a button with the device name into it's own table cell. This button is given a reference link to the `"/api/open-app"` path, and passes it the name of the device application that needs to open up when clicked. On clicking on the button, the application is opened.

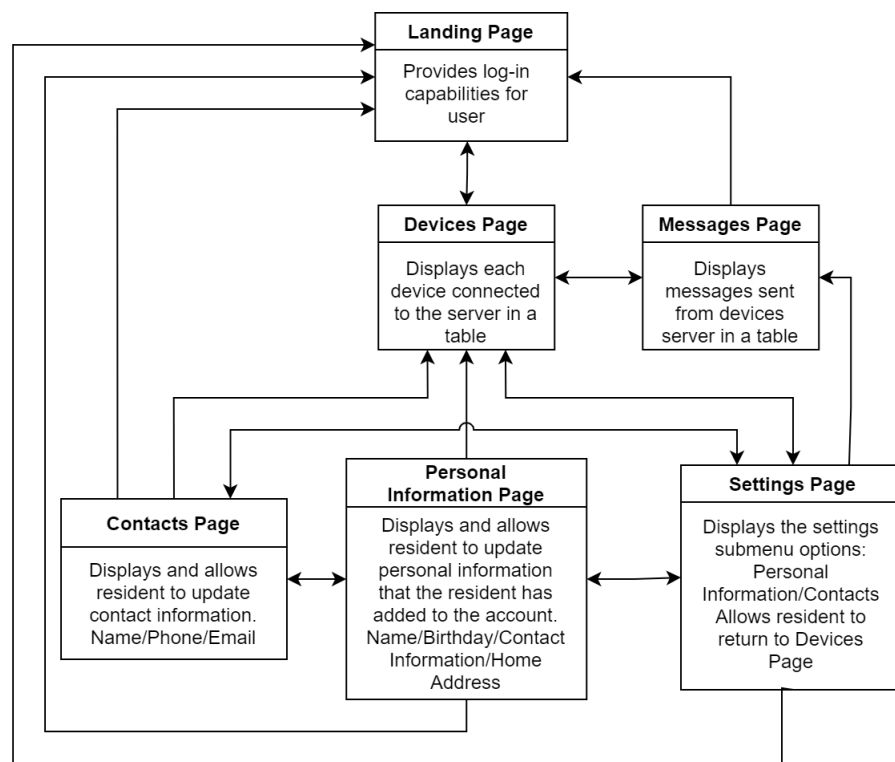


Figure 4.4: RANIA Web Application GUI Diagram

API

Sending device data between the RANIA devices and the RANIA Hub There is one general API path for which queries can be conducted with. It accepts a query which it then uses to query the database. The result from the query is then sent back to the source that called the query function. The HTTPS request should contain JSON in body that is in the format { query: "Query to Execute"}.

Algorithm 1: Database Interaction

Input: HTTPS request: *req*

Output: Result of Query: *res*

- 1 $Q = \text{extract query from } req$ $res = \text{Query database with } Q$;
 - 2 $\text{Send}(res)$;
-

Display Data From Smart Home Devices Each device that is connected to the RANIA server has a file package that corresponds to it. This package is responsible for providing GUI functionality for the user to view/update device data when logged in to their RANIA Server Application. For this project, all packages need to have a specific layout to integrate with the system properly, visualized in (). Within the main folder, there must be a folder for application pages, a folder for images, and a folder for scripts.

Algorithm 2: Load Application Data

Input: Package Name: *Name*

Output: Result of Query: *res*

- 1 $device = Name$;
 - 2 $path = \text{rootDirectory} + "/\text{devices}/" + device + "/\text{Pages}/" + device + "-\text{index.html}"$;
 - 3 $html = \text{readFile}(path)$;
 - 4 $\text{send}(html)$;
-

Connect Device The process for connecting a RANIA device to the RANIA Hub is a one step process for the resident. The driver function accepts the name of the package to be added to the resident's RANIA Web Server. The function then sends an HTTPS request with the requested package name to the RANIA Application store. The API path that receives this HTTPS request locates the directory that corresponds to the package name and gathers the layout and contents of the package into a dictionary to be sent back to the requesting device.

If the request is successful, the requesting device forwards the dictionary via HTTPS request to the resident's RANIA Hub Web Application. The API path that receives this request parses through the package dictionary and copies it to the devices folder in the resident's RANIA Hub Web Application.

Disconnect Device

Disconnecting the RANIA device is also a simple one step process for the resident. Disconnected the device can either be initiated through the device the resident wants to disconnect from the Hub or through the Devices page in the RANIA Web Server Application. The API path that accepts the request to disconnect the device from the RANIA Server Hub removes the device package directory from the RANIA Server Hub and drops the relevant tables for the device from the database.

Algorithm 3: Install RANIA Device Web Application Package

Input: HTTPS request: *req*

Output: Result of : *res*

```

1 pkg = Extract package from req body;
2 name = Extract name from pkg;
3 folders = ["Routes", "Images", "Pages", "Scripts"];
4 pkgPath = "./devices";
5 if directory pkgPath does not exist then
6   createDirectory(pkgPath+name);
7   for each in folders do
8     newSubFolderPath = pkgPath+name+ "/" +each;
9     createDirectory(newSubFolderPath);
10    files = Extract 'each' from pkg;
11    fileKeys = Get keys from Files;
12    for key in fileKeys do
13      writeFile(newSubFolderPath+ '/' +key, files[key])
14    end
15  end
16  createDB = readFile(pkgPath+name+ '/Scripts/create.txt');
17  queryDatabase(createDB)
18 else
19   res = "package exists";
20 end
21 Send(res);

```

Algorithm 4: Disconnect RANIA Device From RANIA Hub**Input:** HTTPS request: *req***Output:** Result of Query: *res*

```

1 devicePackage = request.body['package'];
2 var packagePath = "./devices/" + devicePackage;
3 deleteDatabase = readFile(packagePath);
4 query(deleteDatabase);
5 removeDirectoryRecursively(packagePath);

```

4.2.4 RANIA App Store Application

Layout The RANIA App Store hosts all of the RANIA device application packages that are able to be downloaded by the resident. The layout of the RANIA App Store Application is given in Figure 4.5. The App Packages folder contains all of the folders for the RANIA Device Web Application packages. When the resident requests to connect a device to the RANIA Hub, the RANIA Hub requests to download one of the "Device" folders in the App Packages folder.

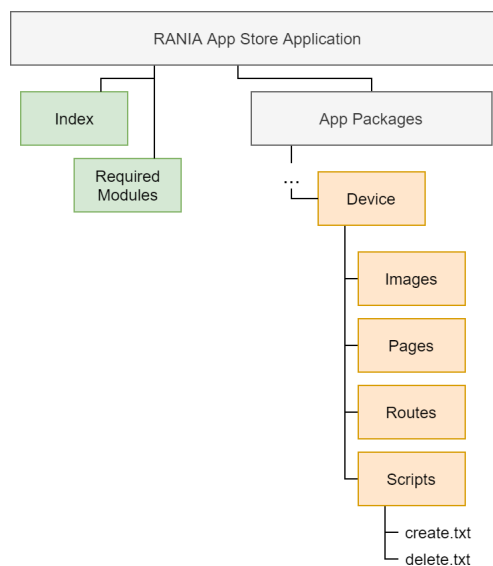


Figure 4.5: RANIA App Store Directory Diagram

API

Connect Device The process for connecting a RANIA device to the RANIA Hub is a one step process for the resident. The driver function accepts the name of the package to be added to the resident’s RANIA Web Server. The function then sends an HTTPS request with the requested package name to the RANIA Application store. The API path that receives this HTTPS request locates the directory that corresponds to the package name and gathers the layout and contents of the package into a dictionary to be sent back to the requesting device.

If the request is successful, the requesting device forwards the dictionary via HTTPS request to the resident’s RANIA Hub Web Application. The API path that receives this request parses through the package dictionary and copies it to the devices folder in the resident’s RANIA Hub Web Application.

Algorithm 5: Get Layout and Contents of RANIA Device Package

Input: HTTPS request: *req*

Output: Device Package: *res*

```

1 pkgName = Extract package name from req body;
2 pkgData = {};
3 folders = ["Routes", "Images", "Pages", "Scripts"];
4 pkgData["Name"] = pkgName;
5 for each folder in folders do
6   | filesDict = {};
7   | pkgData[folder] = files;
8   | path = currentDirectory + "/" ;
9   | for each file in directory of path do
10  |   | data = readFile(path + "/" + file);
11  |   | filesDict[file] = data;
12  | end
13 end
14 Send(res);

```

4.2.5 RANIA Device

The example RANIA Device presented in this thesis is produced on a Raspberry Pi 4 Model B. The features and of the Raspberry Pi 4 Model B are summarized in Table 4.3. The program that serves as the device was made using Python with Tkinter providing the GUI.

Table 4.3: Raspberry Pi 4 Model B Features

Processor	1.5 GHz 64-bit Quad Core ARM Cortex-A72
Connectivity	On-board 802.11ac Wi-Fi Full Gigabit Ethernet Bluetooth 5
Ports	(2) USB 2.0 (2) USB 3.0 (2) Micro HDMI USB-C Power
Memory	(2) 8 GB of RAM 64 GB MicroSD Card
Operating System	Raspbian

GUI The GUI of the Example Medication Schedule RANIA Device is a simple one page display, visualized in Figure 4.6. There is a large Text Area in the top portion of the screen to display the data retrieved from the RANIA Database. Below the Text Area are 3 buttons. One button that when clicked, calls the RANIA Web Server API to retrieve the Medication Schedule data in the RANIA Database. One button that when clicked, connects the device to the RANIA Hub. The final button, when clicked, disconnects the device from the RANIA Hub. Below the buttons, is text that displays whether the device is connected or disconnected from the RANIA Hub.

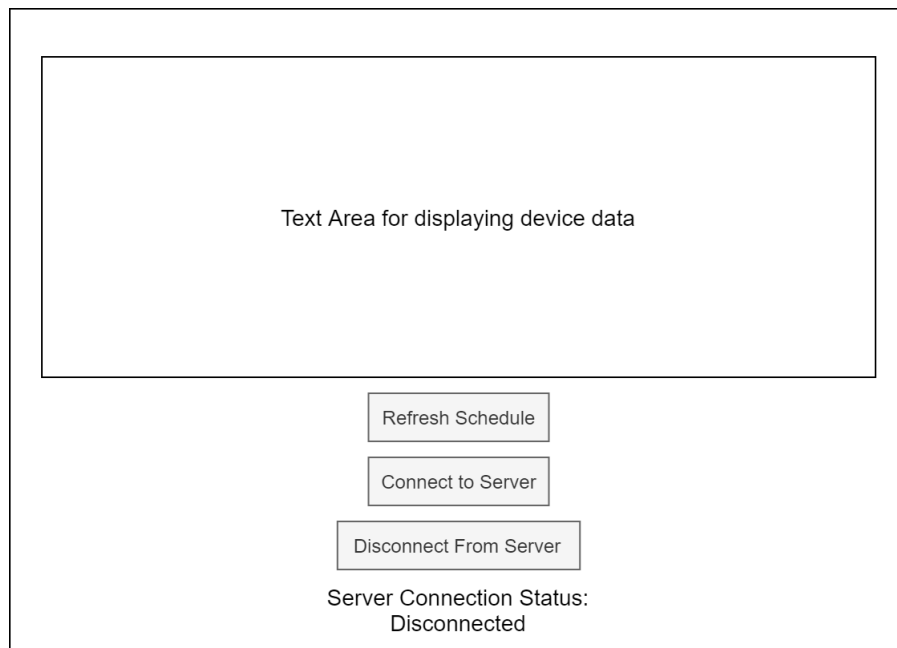


Figure 4.6: RANIA Device GUI Diagram

Button Functions

Refresh Schedule Refreshing the schedule sends a request to the RANIA Server Application to get the data from the corresponding device table. Once the data is received, the text area on the screen will be updated with the new information.

Connect to Server The process for connecting a RANIA device to the RANIA Hub is a one step process for the resident. The driver function accepts the name of the package to be added to the resident's RANIA Web Server. The function then sends an HTTPS request with the requested package name to the RANIA Application store. The API path that receives this HTTPS request locates the directory that corresponds to the package name and gathers the layout and contents of the package into a dictionary to be sent back to the requesting device.

If the request is successful, the requesting device forwards the dictionary via HTTPS request to the resident's RANIA Hub Web Application. The API path that receives this request parses through the package dictionary and copies it to the devices folder in the resident's RANIA Hub Web Application.

Algorithm 6: Initiate Connecting RANIA Device to RANIA Hub

Input: Package Name: *Name***Output:** Result of Query: *res*

```

1 result = getLayout(Name);
2 result2 = installPackage(result);

```

Disconnect From Server The process of disconnecting the device is a one step process for the resident. When the resident clicks the Disconnect From Server button, the device sends a request to the RANIA Server Application to remove the companion package from the Application and the corresponding database table.

Database Table Layout The table for the example RANIA device Medication Schedule has 6 columns in the table. For quick reference, the Medication Shedule Table description is viewable in Table 4.4. The *alarmid* column is the primary key and auto-increments when a new alarm is added to the table. The *userid* column corresponds to the userid that is stored in the *raniauserinfo* table in the RANIA Database. The *time* column stores, in military time, the time that the resident needs to take a certain medication. the *medication* column stores a single medicine name that the resident needs to take. The *instructions* column stores any special instructions a resident might have when taking a certain medication, e.g. before breakfast, with food, at bedtime, etc. When the resident connects the Medication Schedule RANIA Device to their RANIA Hub, this table will be added to the RANIA Database. When the resident disconnects their RANIA Device from the RANIA Hub, this table will be removed from the RANIA Database.

Table 4.4: Medication Schedule Database Table Description

Field	Type	Null	Key	Default	Extra
<i>alarmid</i>	int	NO	PRI	NULL	auto-increment
<i>userid</i>	varchar(45)	YES		NULL	
<i>time</i>	varchar(45)	YES		NULL	
<i>medication</i>	varchar(45)	YES		NULL	
<i>instructions</i>	varchar(45)	YES		NULL	
<i>active</i>	varchar(45)	YES		NULL	

4.2.6 Amazon Alexa Integration

In order to give the resident the ability to vocally interact with the RANIA System, an Amazon Alexa Skill was developed to access device data stored in the RANIA Database. Skill intents and sample utterances were developed based on the Medication Schedule's database table layout, meaning when more RANIA devices are developed there will need to be more intent handlers and sample utterances defined. The interaction model for the Medication dispenser is based on situational design, where the basis for an interaction is a potential request from a user. The situations that this system has been developed to handle are divided into 3 categories: for **requesting device data**, for **requesting control of the device**, and **requesting to log a message in the RANIA System**. To invoke the RANIA skill, the resident first needs to say "Alexa, open RANIA house". After being greeted, the resident can say any of the sample utterances defined below, and invoke the intent.

Each created intent handler follows the following basic steps:

1. Extract utterance slot value(s) - Optional step, some utterances don't include slots
2. Form JSON object with slot value
3. Execute HTTPS request to RANIA API with object
4. Parse the response from the request
5. Form Alexa spoken response
6. Speak response

Requesting Device Data Intent handlers were created to access and update data from the Medication Schedule's database table in the RANIA Database. The resident can request what time they need to take a certain medication, what medication they need to take at a certain time, the instructions for taking a certain medication.

Requesting the Time for Taking a Medicine This intent is invoked when the resident wants to know what time of the day they need to take a certain medication.

Sample Utterance: When do I need to take {*Medicine*}

Slot Name: *Medication*

Slot Type: AMAZON.SearchQuery

Requesting the Name of a Medication for a Certain Time This intent is invoked when the resident wants to know what medication they need to take at a certain time.

Sample Utterance: What medicine do I need to take at {*Time*}

Slot Name: *Time*

Slot Type: AMAZON.TIME

Requesting the Instructions for Taking a Medication This intent is invoked when the resident wants to know what instructions are stored with a certain medication.

Sample Utterance: How do I take {*Medication*}

Slot Name: *Medication*

Slot Type: AMAZON.SearchQuery

Requesting Control of the Machine Intent handlers were created to give the user control of the database table. The user can add and remove medication times to/from the Medication Schedule database table. These intents are structured slightly differently than the intents for requesting device information.

Adding an Alarm to the Schedule This intent is invoked when the resident wants to add a medication time to the Medication Schedule.

Sample Utterance: Add an alarm to my medication dispenser

Slot Names: *Time, Medication, Instructions*

Slot Type: AMAZON.TIME, AMAZON.SearchQuery, AMAZON.SearchQuery

Removing an Alarm From the Schedule This intent is invoked when the resident wants to remove a medication time to the Medication Schedule.

Sample Utterance: Add an alarm to my medication dispenser

Slot Names: *Medication, Time*

Slot Type: AMAZON.SearchQuery, AMAZON.Time

Requesting to Log a Message in the RANIA System An intent handler was created so that the user can log a message in the messages table in the RANIA Database.

Log a Message in the Database This intent is invoked when the resident wants to log a message in the messages table in the RANIA database.

Sample Utterance: Tell RANIA {*Message*}

Slot Name: *Message*

Slot Type: AMAZON.SearchQuery

Chapter 5

Results

This section evaluates the implementation of the 3 project goals by running through each scenario described in the methodology section. Several scenarios are run through in the RANIA system and the results are recorded. An example user was added to the raniauserinfo table in the RANIA Database for testing the results.

5.1 Connecting the RANIA Device to the RANIA Hub

Before the Medication Schedule device was connected to the RANIA Hub, the web app companion package was uploaded to the App Packages folder in the RANIA App Store Application and the RANIA Web Application was logged into as the example user. After the package was uploaded, the example device was turned on and the Connect to Server button was clicked. After clicking the Connect to Server button, the Connection status changed to "Connected" and upon refreshing the user home page in the RANIA Web application, the companion app button was viewable on the user home page. Refer to Figure 5.1 for visual representation.

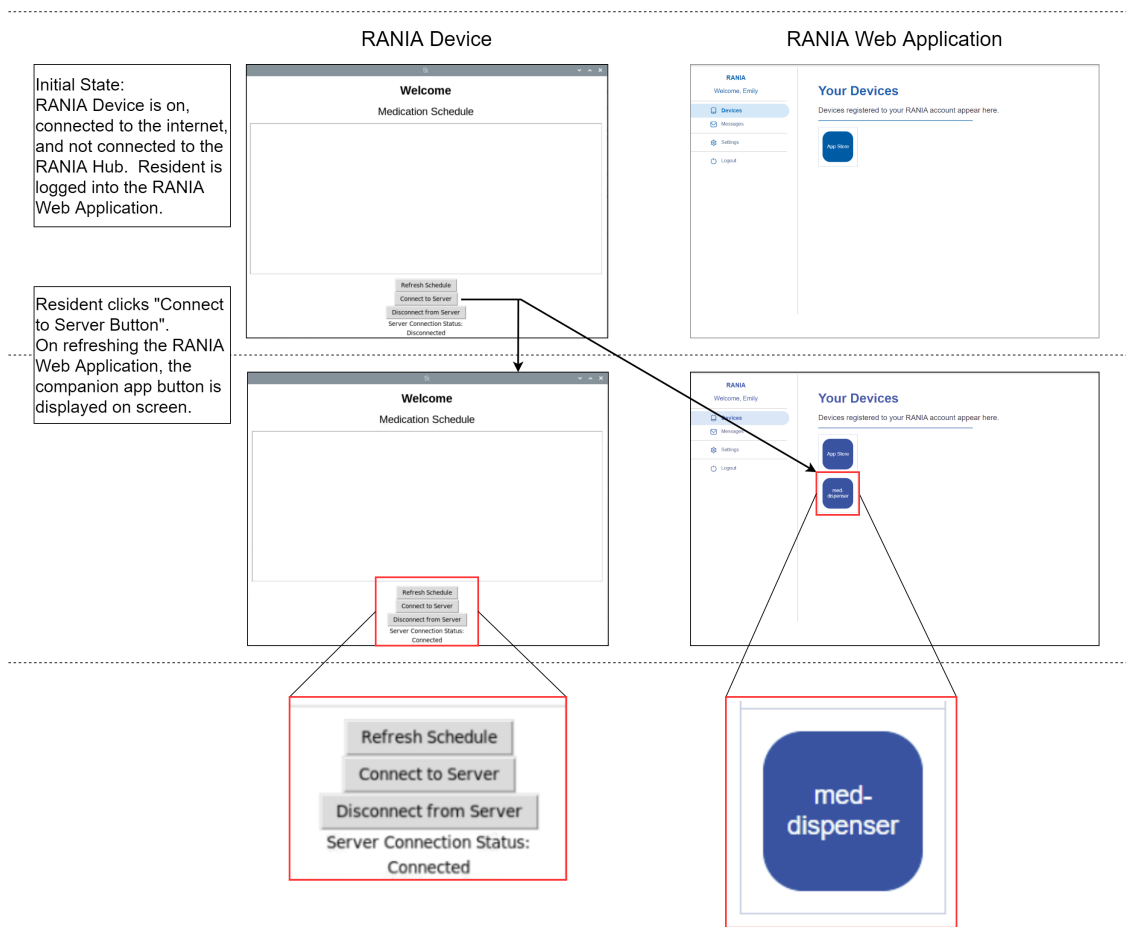


Figure 5.1: Connect Device Results

5.1.1 Using the RANIA Web Server Companion Package

After the device companion package was added to the resident’s RANIA Hub, the companion package was opened to test its functionality. Opening the device’s companion package brought up the correct index page, and displayed an empty table that pulls from the device’s database table. Clicking on the "Send Test Data" button and refreshing the page adds the test data to the database and the package loads the data onto the page of the companion app.

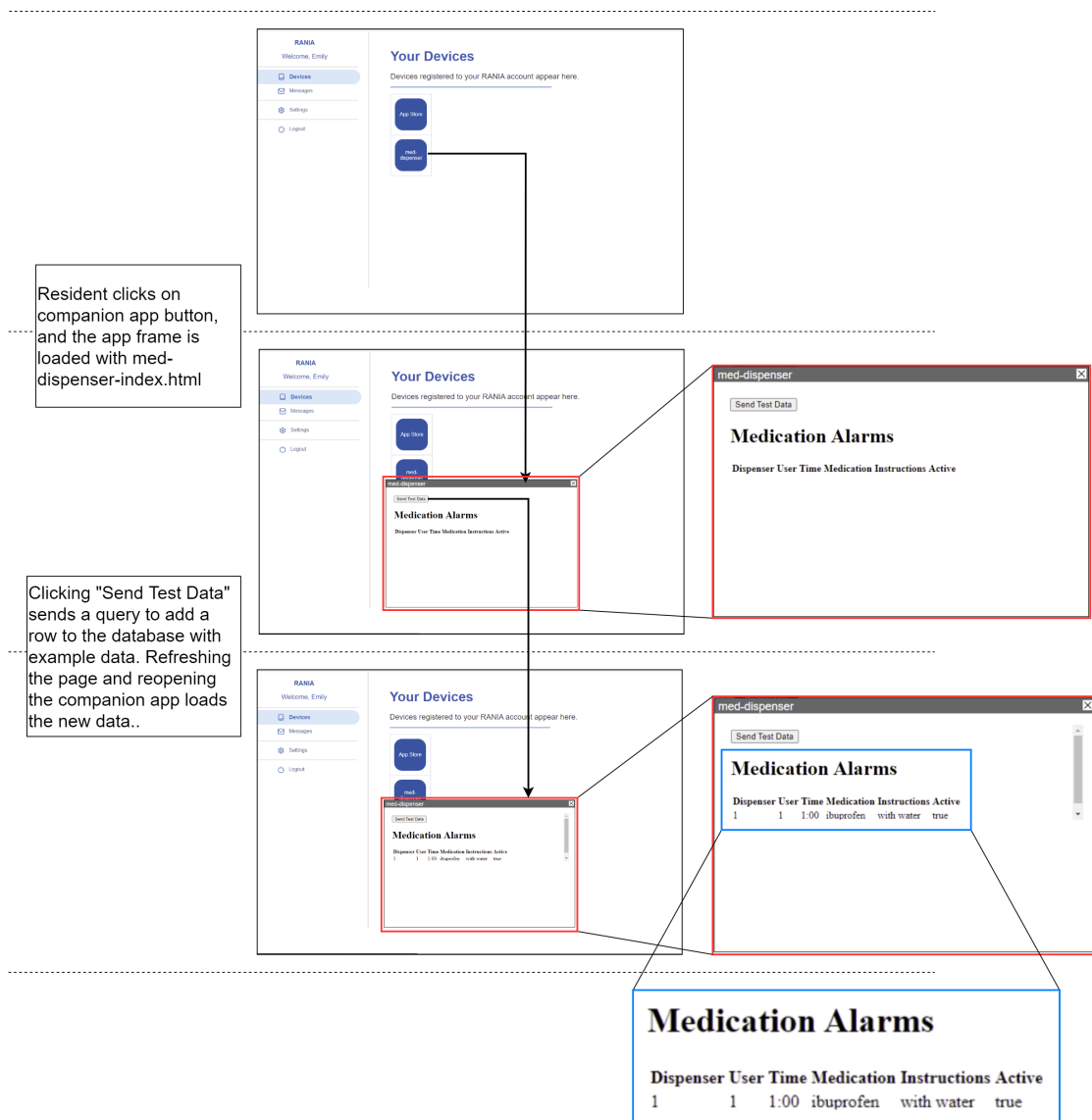


Figure 5.2: Using Companion App Results

5.1.2 Alexa Voice Commands

Each intent defined in the methodology section was run through and the results were recorded below. At the time of testing all of the intents, the medication schedule device contained a single entry; The medication is ibuprofen, the time is 1:00, and the instructions are to take with water.

Requesting Device Data

Requesting the Time for Taking a Medicine The expected value of time for Alexa to give is 1:00 when requesting the time for ibuprofen and the actual given time is 1:00. This intent is executed in 2 steps for the user.

Table 5.1: Requesting the Time for Taking a Medicine

Resident	Alexa
Open RANIA House	
	How can I help you with your RANIA devices?
When do I need to take ibuprofen?	
	1:00 (1 o'clock)

Requesting the Name of a Medicine for a Certain Time The expected name of medication for Alexa to give is ibuprofen when requesting the medication to take at 1:00 and the actual given medication is ibuprofen. This intent is executed in 2 steps for the user.

Table 5.2: Requesting the Name of a Medicine for a Certain Time

Resident	Alexa
Open RANIA House	
	How can I help you with your RANIA devices?
What medicine do I need to take at 1 o'clock	
	ibuprofen

Requesting the Instructions for a Certain Medication The expected instructions for Alexa to give when requesting the instructions for taking ibuprofen is "with water" and the actual given medication is "with water". This intent is executed in 2 steps for the user.

Table 5.3: Requesting the Instructions for a Certain Medicine

Resident	Alexa
Open RANIA House	
	How can I help you with your RANIA devices?
How do I take Ibuprofen?	
	with water

Requesting control of the Device

Adding an Alarm to the Medication Schedule Adding an alarm to the medication schedule is completed in 5 steps for the user. Before executing, the database table for the medication schedule was empty. The alarm expected to be added after executing the intent has the medication declofan, time 14:00, instructions "take with dinner". After executing the intent for adding an alarm, the correct values are added to the table.

Table 5.4: Adding an Alarm to the Schedule

Resident	Alexa
Open RANIA House	
	How can I help you with your RANIA devices?
Add an alarm to my medication dispenser	
	What time do you want your alarm to go off?
6pm	
	What medications do you want to add to your alarm?
declofan	
	Do you have any special instructions for taking medications with this alarm?
take with dinner	
	ok

Removing an Alarm From the Medication Schedule Removing an alarm from the medication schedule is completed in 4 steps for the resident. Before executing the intent, the medication schedule had one alarm with the medication declofan, time 14:00, and instructions "take with dinner". After executing the intent, the alarm schedule is empty.

Table 5.5: Adding an Alarm to the Schedule

Resident	Alexa
Open RANIA House	
	How can I help you with your RANIA devices?
Remove an alarm from my medication dispenser	
	What time is the alarm you want to remove?
6pm	
	What medication do you want to remove for that time?
declofan	
	ok

Logging a Message

Logging a message to the RANIA Hub is completed in 2 steps for the resident.

Table 5.6: Requesting the Instructions for a Certain Medicine

Resident	Alexa
Open RANIA House	
	How can I help you with your RANIA devices?
Tell RANIA I am getting low on medication	
	ok

5.2 Disconnecting the RANIA Device From the RANIA Hub

Clicking the "Disconnect Device" button on the medication removes the corresponding companion package from the RANIA Hub and the corresponding database table from the RANIA Hub.

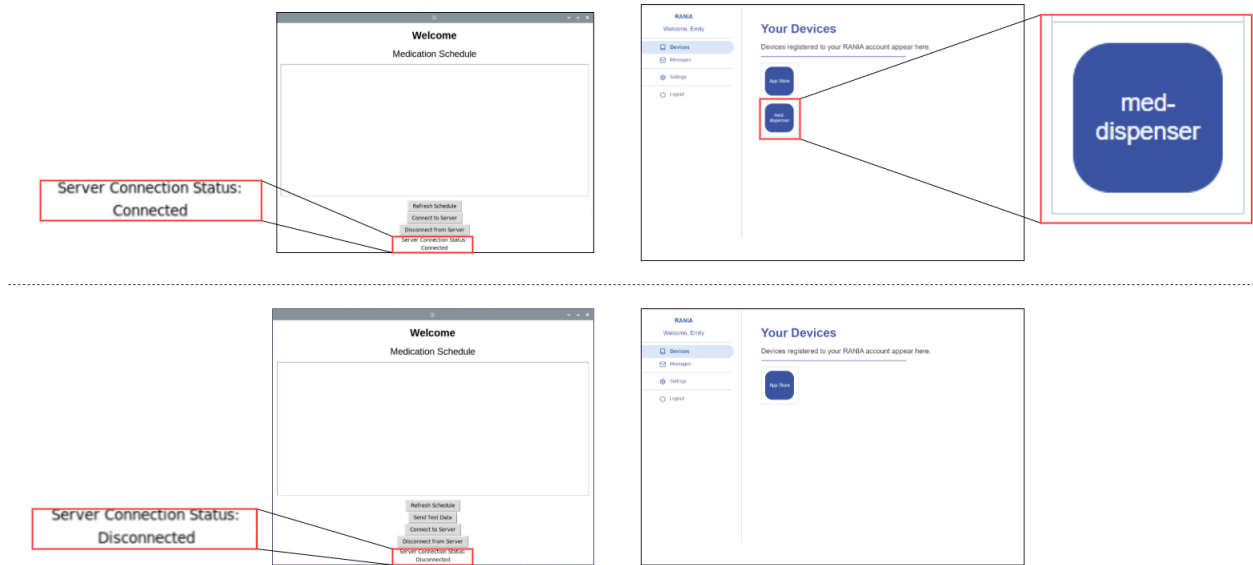


Figure 5.3: Disconnect Device Results

Chapter 6

Conclusions

6.1 Project Results Conclusion

This paper presents a modular, voice-enabled framework for managing a smart-home system. Methods for connecting and disconnecting devices to/from a smart home hub were presented. Methods for a package-based system for viewing device data was presented. Methods for allowing access and control of the devices were presented.

6.2 Limitations

Time Constraints This thesis was developed over the course of 6 months, and as such was limited to that time frame. A significant portion of time in this development was figuring out how to work around network security restrictions.

Security Constraints Due to network security constraints within the lab that this thesis was developed in, the RANIA Hub was unable to be hosted locally. The original plan was to have the RANIA Hub hosted on an on-site server - as it should be - but due to network security concerns, the RANIA Hub was implemented on a remote server.

Device Limitations While theoretically, this framework should be compatible with any device that supports HTTPS communications, it was tested only on a Raspberry Pi Model

B. It was not tested with any other micro-controllers.

6.3 Future Work

The most significant realm of future development in the RANIA system is in security. This system will be storing sensitive data. Because of this, a system of verification and validation will need to be implemented within the system. This verification and validation will need to be present on both the RANIA Hub and each device developed as part of the RANIA System. The RANIA Hub will need to make sure only authorized users are able to access the residents data stored in the hub, and the devices need to verify that they are operating correctly. Some devices will be very sensitive such as smart medicine dispensers, which need to verify that it is dispensing the correct amount amount of medication at the correct time. Extra security for the devices is also of utmost importance, especially in the case of a smart medication dispenser, where tampering with the dispensed dosage could potentially lead to fatal overdoses or underdoses. External caregiver access to the system is also planned for the system, so that a resident's loved one can check up on them when necessary. The caregiver will be able to access the resident's data either through a Graphical User Interface or through Amazon's Alexa voice interface.

References

- [1] L. C. D. Silva, C. Morikawa, and I. M. Petra, “State of the art of smart homes,” *Engineering Applications of Artificial Intelligence*, vol. 25, no. 7, p. 1313–1321, 2012.
- [2] B. L. R. Stojkoska and K. V. Trivodaliev, “A review of internet of things for smart home: Challenges and solutions,” *Journal of Cleaner Production*, vol. 140, p. 1454–1464, 2017.
- [3] K. Maswadi, N. B. A. Ghani, and S. B. Hamid, “Systematic literature review of smart home monitoring technologies based on iot for the elderly,” *IEEE Access*, vol. 8, p. 92244–92261, 2020.
- [4] C. Jimenez, E. Saavedra, G. D. Campo, and A. Santamaria, “Alexa-based voice assistant for smart home applications,” *IEEE Potentials*, vol. 40, no. 4, p. 31–38, 2021.
- [5] P. Mtshali and F. Khubisa, “A smart home appliance control system for physically disabled people,” *2019 Conference on Information Communications Technology and Society (ICTAS)*, 2019.
- [6] C. Z. Yue and S. Ping, “Voice activated smart home design and implementation,” *2017 2nd International Conference on Frontiers of Sensors Technologies (ICFST)*, 2017.