MAKERERE UNIVERSITY
COLLEGE OF ENGINEERING, DESIGN, ART AND TECHNOLOGY
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

SETTING UP OF A HOTSPOT MANAGER FOR MIKROTIK

BY
KAGGULIRE DAVID  13/U/299

A Final Year Project Report Submitted to the Department of Electrical and Computer Engineering in Partial Fulfilment of the Requirements for the award of Bachelor of Science in Computer Engineering of Makerere University.

JUNE 2018
DECLARATION

I, Kagulire David, declare to the best of my knowledge that the work in this report is original and has not been submitted before to any university. It is the result of my effort.

Name: Kagulire David
Registration Number: 13/U/299
Student Number: 213000024
Signature: [Signature]
Date: 18th June 2018

APPROVAL

This report has been submitted with approval from the following supervisor.

Mr. Mark Kagarura
College of Engineering, Design, Art and Technology (CEDAT)
Department of Electrical and Computer Engineering, Makerere University
Makerere University
Signature: [Signature]
Date: 16.08.2018
DEDICATION

I dedicate this Project Report to Mr. Mark Kagarura who with his encouragement, guidance and supervision has led to the completion of this project.
ACKNOWLEDGEMENTS

First of all, I would like to thank the Almighty God for the life and health that He has given me without which this would not be possible.

Special thanks to my supervisors Mr. Mark Kagarura and Mr. Maximus Byamukama. Their unwavering support, wisdom, guidance and supervision has brought me successfully to the completion of the course of this project.

I would also like to thank my research partner, Mr. Ayebale Bright Johnson, whose co-operation, hard work, diligence, selflessness and sacrifice made the research outcome a success. It was a great honor and a blessing to work with you.

Exclusive thanks to Ssemanda Derrick, Atuhaire Jude Innocent, Kukunda Santrinah and all my colleagues for the support given in all forms during this project time and for the last four years. It has been such a blessing to know you and work with you.

Last but not least I would like to thank my siblings Enos, Christine and Esther for their unconditional support.

Finally, I want to thank my parents Eng. Kibuuka Stephen and Ms. Namakula Margaret who have made sure that I attain a great education and supported me endlessly in all that I have set out to do. God bless you.
ABSTRACT

Internet access has become a key issue in how we live and communicate in our communities. This project seeks to design and develop a hotspot manager for MikroTik. Currently the MikroTik hotspot manager can enable a hotspot on a single access point. However there is no way to manage a group of access points from a single location but rather apply configuration settings to each access point one at a time. This presents a challenge to the network administrator as there is time wastage in deploying changes to a single access point. We therefore designed and developed a hotspot manager that could provide hotspot access to devices with wireless capability, control individual access points and also scale a hotspot to various access points. This enables network management of all access points from a single hotspot manager rather than deploying configuration to a single access point at a time. In order to achieve this, we studied about the operation of hotspots or access points from online articles and journals. We then used prototyping tools to prototype the User Interface of the system. We then designed a wrapper Application Programming Interface (API) that accesses the Router Operating System API (Router OS API) which enables functions like firewall, bandwidth management on the MikroTik hardware called RouterBoard. We designed a desktop application that integrates and interfaces with the wrapper API to send commands to the MikroTik router. Within the application, a network administrator can click a button to create a hotspot on a master access point and access points which broadcasts the same Service Set Identifier (SSID) on the slave access points, perform user management and volume based monitoring. This project thus leads to creation of a hotspot manager that eases monitoring, bandwidth control for different users and can easily scale hotspots to multiple routers.
PREFACE

This report entails an overall explanation of the steps followed to obtain the design and implementation of a hotspot manager that provides businesses with an easy to use interface for business owners to easily create a hotspot. It contains five chapters in total and an overview of these chapters is as follows:

Chapter One is the introductory part of this report, explaining an improvement in the hotspot manager for the Mikrotik router and also the justification for the project, the project objectives and summary of the project.

Chapter Two provides the Literature Review information about Hotspot Managers, Mikrotik Routers, Router Operating System and the tools used.

Chapter Three shows how the Project Objectives were achieved that is how information was collected, how decisions were made and how the different parts of the system were implemented that is the Methodology of this project.

Chapter Four contains the Key Results of the tests carried out and the Analysis done of these results.

Chapter Five contains the Recommendations and Conclusions.
# Table of Contents

CHAPTER ONE: INTRODUCTION .................................................................................. 1

1.1 PROJECT BACKGROUND .................................................................................. 1

1.2 PROBLEM STATEMENT ................................................................................... 1

1.3 JUSTIFICATION ............................................................................................... 2

1.4 PROJECT OBJECTIVES ................................................................................... 2

1.4.1 GENERAL OBJECTIVE ................................................................................ 2

1.4.2 SPECIFIC OBJECTIVES: ............................................................................... 2

1.5 SCOPE ............................................................................................................... 3

1.6 SUMMARY OF THE PROJECT ......................................................................... 3

CHAPTER TWO: LITERATURE REVIEW .................................................................. 5

2.1 INTRODUCTION ............................................................................................... 5

2.2 HOTSPOT MANAGER ..................................................................................... 5

2.2.1 MikroTik hAPLite ......................................................................................... 6

2.3 ACCESS LISTS .................................................................................................. 6

2.4 API ................................................................................................................... 6

2.4.1 State Management ....................................................................................... 7

2.4.2 D3JS ............................................................................................................. 13

2.5 IDE (INTEGRATED DEVELOPMENT ENVIRONMENT) ........................................ 13

2.7 OPERATING SYSTEMS ................................................................................... 14

2.8 REQUIREMENTS SPECIFICATION .................................................................. 14

2.10 USE CASE ..................................................................................................... 14

2.11 NETWORK TOPOLOGY ................................................................................. 14

2.11.1 Centralized architecture .......................................................................... 14

2.11.2 Point to point networks ............................................................................ 15

CHAPTER THREE: METHODOLOGY .................................................................... 17
3.1 SYSTEM DESIGN .......................................................................................................................... 17
  3.1.1 Block Diagram ......................................................................................................................... 17
  3.1.2 Inputs and Outputs of Router Operating System ...................................................................... 18
  3.1.3 Data Flow Diagram .................................................................................................................. 18
3.3 HARDWARE ARCHITECTURE ...................................................................................................... 20
3.4 SCALING THE HOTSPOT ........................................................................................................... 20
3.5 USE CASE DIAGRAM .................................................................................................................. 21
  3.5.1 Hotspot Use Case for Administrator ...................................................................................... 22
  3.5.2 Hotspot Use Case for Hotspot Visitor .................................................................................... 24
3.6 STAGES OF DEVELOPMENT OF SYSTEM .............................................................................. 26
  3.6.1 Requirements Specification ..................................................................................................... 26
  3.6.2 Prototyping .............................................................................................................................. 26
  3.6.2. Designing and Writing Code ................................................................................................. 27
3.7 PHYSICAL LEVEL DIAGRAM (BASED ON TCP/IP NETWORK MODEL) ..................................... 34

CHAPTER FOUR: KEY RESULTS ........................................................................................................... 35
  4.1 Hotspot Creation .......................................................................................................................... 35

CHAPTER FIVE: CONCLUSIONS AND RECOMMENDATIONS ......................................................... 41
  5.1 CONCLUSIONS ............................................................................................................................ 41
  5.2 CHALLENGES FACED ................................................................................................................. 41
  5.3 RECOMMENDATIONS ................................................................................................................... 41
REFERENCES ........................................................................................................................................ 42
LIST OF TABLES

Table 3.5.1.6: Use Case Scale Hotspot ................................................................. 24
Table 3.5.1.7: Use Case View Usage Statistics ......................................................... 24
Table 3.5.1.8: Use Case Login .................................................................................. 24
Table 3.5.1.9: Use Case Use System ........................................................................ 25
Table 3.5.1.10: Use Case Logout ............................................................................. 25
LIST OF FIGURES

Figure 2-1: MikroTik hAPLite 941-2nD ................................................................. 6
Figure 2-2: Flow chart for API process ................................................................. 7
Figure 2-3: Components interacting ..................................................................... 8
Figure 2-4: Component Interaction ....................................................................... 9
Figure 2-5: Data flow diagram for Interaction between multiple components .......... 9
Figure 2-6: Sending data between Components ..................................................... 10
Figure 2-7: Dataflow diagram for the store ........................................................... 11
Figure 2-8: Data flow in Redux architecture .......................................................... 11
Figure 2-9: Block diagram of Store ....................................................................... 12
Figure 2-10: Action structure ............................................................................... 12
Figure 2-11: Block diagram of reducer ................................................................. 13
Figure 2-0-12: Star topology .................................................................................. 15
Figure 2-13: Point to point connection between devices ....................................... 16
Figure 3-1: Block diagram showing methodology ............................................... 17
Figure 3-2: Inputs and Outputs to Router Operating Systems ............................... 18
Figure 3-3: Data flow diagram ............................................................................. 19
Figure 3-4: Block Diagram for Access Point Architecture .................................... 20
Figure 3-5: CAPsMAN Setup ................................................................................ 21
Figure 3-6: Use Case Diagram for Administrator .................................................. 21
Figure 3-7: Use Case for Hotspot Visitor ............................................................... 22
Figure 3-8: Stages of development of system ....................................................... 26
Figure 3-9: Prototyping with Figma ....................................................................... 27
Figure 3-10: API for RouterOS ............................................................................ 27
Figure 3-11: Implementing Redux using State Reducers ....................................... 28
Figure 3-12: Exporting API module ..................................................................... 29
Figure 3-13: Code for API Wrapper .............................................................................. 30
Figure 3-14: Include API in Electron ........................................................................... 31
Figure 3-15: API Usage in Electron ............................................................................. 32
Figure 3-16: Code for blocking User ........................................................................... 33
Figure 3-17: Hotspot Manager Level based on TCP/IP network model ......................... 34
Figure 4-1: Network Administrator Login .................................................................... 35
Figure 4-2: Setting SSID for hotspot .......................................................................... 36
Figure 4-3: Inputting Password for hotspot ................................................................. 36
Figure 4-4: Clicking button to set up hotspot ............................................................... 37
Figure 4-5: Viewing Connected Users ......................................................................... 38
Figure 4-6: Viewing Network Traffic ......................................................................... 39
Figure 4-7: Viewing User Traffic ................................................................................. 40
Figure 4-8: Limiting bandwidth for a user .................................................................... 40
# LIST OF ACRONYMS

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Full Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP</td>
<td>Access Point</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>APP</td>
<td>Mobile Application</td>
</tr>
<tr>
<td>CEDAT</td>
<td>College of Engineering, Design and Art</td>
</tr>
<tr>
<td>D3</td>
<td>Data Driven Documents</td>
</tr>
<tr>
<td>EAP</td>
<td>Extensible Authentication Protocol</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>MAC</td>
<td>Media Access Control</td>
</tr>
<tr>
<td>ISP</td>
<td>Internet Service Provider</td>
</tr>
<tr>
<td>JS</td>
<td>JavaScript</td>
</tr>
<tr>
<td>OS</td>
<td>Operating System</td>
</tr>
<tr>
<td>SSID</td>
<td>Service Set Identifier</td>
</tr>
<tr>
<td>UI</td>
<td>User Interface</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless Local Area Network</td>
</tr>
<tr>
<td>WPA</td>
<td>Wi-Fi Protected Access</td>
</tr>
<tr>
<td>WPS</td>
<td>Wi-Fi Protected Setup</td>
</tr>
</tbody>
</table>
CHAPTER ONE: INTRODUCTION

1.1 PROJECT BACKGROUND

Network access has become an essential part of how we live and interact in our communities. It influences the way we work, communicate, study and share content amongst our circles. As of December 2017, 54.4% of the world population use the Internet while Wi-Fi a wireless technology is used by over 700 million people worldwide[1][2]. MikroTik has also become one of the most widely and extensive provider for network hardware and software solutions globally being used by Information Technology and Services at 14%, Telecommunications at 13% and Computer Software Industry at 8%[3].

The ability to provide a wireless Internet access not only cheaply improves the extent of reach of network services but also makes it faster to deploy these services across communities[4]. It also improves quality of doing research for different individuals such as students, business people to improve on quality of products, medicine in delivering better health care etc [5]. For wireless networks service provision, managing access to network resources, bandwidth and number of simultaneous users among other network related resources is key.

1.2 PROBLEM STATEMENT

Currently the Mikrotik hotspot manager allows a user to connect to one hotspot created however a problem arises when the hotspot service is scaled up by broadcasting the same SSID on different Mikrotik nodes, so that the hotspot can be accessed from different nodes. It thus becomes extremely difficult to manage because each access point (AP) within the cluster of nodes is an IP-addressable device requiring management, monitoring, and control. Deployment of more APs typically doubles the number of network infrastructure devices that require management. This presents a significant additional burden to the network administration resources and is often a hurdle to adoption of wireless technologies,
particularly because the configuration of each access point is nearly identical to the next. This near-sameness often leads to misconfiguration and improper operation of the WLAN.

1.3 JUSTIFICATION

There are millions of devices running MikroTik’s RouterOS today, some are used in universities and other institutions for both business critical and non-critical functionalities as wireless access points[6]. Scaling up the hotspot service would mean that a user would always be connected not only to a network but always to the best service which implies better signal reception for his devices that connect wirelessly to networks. In order to provide better bandwidth, services-on-demand to end-users and relieve carrier infrastructure of some traffic, a hotspot manager is required for a dynamic monitoring model which allows volume-based monitoring and visualization, bandwidth limiting, load averaging by dynamically limiting the number of simultaneous connected users to ensure that hotspot services are reliably delivered to the end client. At CEDAT, a solution such as this can enable scaling the hotspot whenever other internet access is slowed down due to many users for example at conferences at the CEDAT main hall.

1.4 PROJECT OBJECTIVES

1.4.1 GENERAL OBJECTIVE

To design a hotspot manager for MikroTik that can expand a hotspot service to more than one router.

1.4.2 SPECIFIC OBJECTIVES:

To design a hotspot management API for the MikroTik router.
To build a hotspot manager for a MikroTik router that is able to setup one or more hotspots, carry out volume-based monitoring and scale hotspot to multiple routers.
To install and test the hotspot manager on different MikroTik devices.
1.5 SCOPE

The scope of the project was about managing access points and as such to design a hotspot manager that is

- Able to setup one or more hotspots (if device supports multiple) on a device.
- Able to carry out volume-based monitoring of network traffic.
- Able to scale up the hotspot to multiple routers.

1.6 SUMMARY OF THE PROJECT

To achieve the main and specific objectives of this project – the methods used, summarized chronologically, are as follows:

![Figure 1-1: Mind map for Hotspot Manager](image)

The mind map for the hotspot manager breaks down the project into features that taken into consideration to fully achieve the purpose of the project.

The hotspot manager makes it basis off Router OS which sits on the MikroTik routerboard RB941-2ND hAPLite equipment. In order to access Router OS, a wrapper API is developed around RouterOS API in Node JS. This wrapper API exposes functionality that can be called upon by the Desktop application when issuing commands to the routerboard. The Desktop application is prototyped using a prototyping tool called Figma and developed using a JavaScript Framework called Angular JS.
The expectations of the project include Network traffic monitoring, Bandwidth limiting, a scaled hotspot service and a view for connected users.
CHAPTER TWO: LITERATURE REVIEW

2.1 Introduction

Hotspot access has changed how individuals access the Internet. This chapter includes a literature review on features that apply to a hotspot manager and a hotspot Management System that basically provides Internet access to people through an access point.

2.2 Hotspot Manager

A hotspot is a physical location where people can access the Internet, typically using Wi-Fi, via a wireless local area network (WLAN) with a router connected to the Internet service provider (ISP). A hotspot is usually referred to as an access point which can be understood as a device that creates a wireless network for other devices to connect to the Internet or network[7].

A hot spot enables connectivity between a device with wireless capability and the Internet. In general Wi-Fi access provided by any device guarantees to the term a hot spot[8], [9]. Due to the nature of urbanization today, computing technology has helped interconnect many users through computing devices that can be accessed anywhere thus ubiquitous computing. However due to this, a need to access Internet from everywhere or ubiquitously becomes key in our modern world since the Internet facilitates most of the work done by different people and systems such as e-learning, e-commerce, research, instant messaging and social media platforms and many others[10].

A hotspot manager is a solution used to centrally manage and operate several Wi-Fi hotspots independently from the hotspot location and the Wi-Fi equipment used[11]. This implies that one can manage one or more several Wi-Fi hotspots through a hotspot manager. A hotspot manager offers authentication that can be implemented using an EAP (Extensible Authentication Protocol). EAP provides a framework with many authentication features for access control in local area networks such as Wireless(802.11)[12].
2.2.1 MikroTik hAPLite

The MikroTik hAPLite is a home Access Point that supports WPS a wireless network security standard that makes connections between routers and wireless devices faster and easier, it runs RouterOS[13].

2.3 Access Lists

An access list is a list used by an access point to restrict allowed connections from other devices, and to control connection parameters. It works in a way that access list rules are checked sequentially, disabled rules are always ignored, only the first matching rule is applied, if there are no matching rules for the remote connection then the default values from the wireless interface configuration are used.

For example to limit the bandwidth of a certain user, we can use a property called client-tx-limit which asks a client to limit rate of data transmission and the value of the limit is in bits per second i.e. kilobits per second, megabits per second, gigabits per second. This property is supported by RouterOS clients[14].

2.4 API

An API is a set of routines or functions or protocols used for building software applications. In a way an API offers access to certain functions in a program that are needed to achieve a certain purpose in another application. In computing, an API can receive a request from a
client and then pass it on to the server for processing. It then gets a response from the server which is then sent to the client who made initial request in figure 2.

![Flow chart for API process](image)

**Figure 2-2: Flow chart for API process**

In a way an API allows abstraction of functionality between one system and another. A wrapper API is a library that exposes a third party API in a more friendly way. It give us a simpler set of methods that we can call from within our code.

2.4.1 State Management

State management means to preserve the state of a control like a button, webpage, object or data and user in an application[15]. In web applications written in Angular JS or React, state management is required especially when modules used are complex. It is normally achieved using a software platform called Redux. State management involves managing the state being shared by various components and modules in an application. State can include server responses and cached data. UI state is also increasingly complex since it involves managing routes, pagination, selected tabs, toggles, alerts and error messages. State
management also involves managing user input such as form submissions, filters and search queries[16][17]. In Angular each component refers to the basic building block of a UI in an angular application. A component defines a view in a UI which is a set of screen elements.[18]

State management comes with benefits such as

- Chain UI layout-specific actions with other events such as saving data server side.
- Control the UI from any point of the application without worrying about how components are related or how to inject a specific service.

2.4.1.1 State

State includes the state of the UI and state of variables in code. In angular, each component has its own state and a component has no idea about other component’s states unless data is flowing between the two components. Figure 3 illustrates this concept.

![Component Interaction Diagram](image)

Figure 2-3: Components interacting

Figure above shows components interacting where one login a new page or view or component is loaded that shows successfully logged in. This new component has a different state from the component before and therefore state management is required.

Data flow diagram for Component interaction in Angular between main (app component) and child component.
In simple applications, it is easy to pass information however, information flow becomes complicated with complex app architecture.

In order to pass information between two components Create Hotspot and Block Users. The data flow diagram would look as follows.
2.4.1.2 Core principles of State Management in Redux architecture

*Single store:* the state of the application is stored in a single store. A store basically holds the whole state tree of an application. The store is responsible for providing data to the components.

*Read-only state:* state should be read-only or immutable. To change the state, new action should be emitted.

*Pure function reducers:* Reducers are pure functions. A pure function is a function in JavaScript which given always the same input will return the same output i.e. its execution does not depend on the state of the system. Reducers take the current state and an action and return next state. Reducers should not take any other parameters outside the function.

The Redux architecture strictly follows a unidirectional flow of data which implies that all data in the application follows the same lifecycle pattern making the logic of the application more predictable and thus easier to understand.
To explain the data flow, a component dispatches an action. An action is simply a plain object describing what has happened for example

```javascript
{ type: 'LIKE_ARTICLE', articleId: 42 }
{ type: 'FETCH_USER_SUCCESS', response: { id: 3, name: 'Mary' } }
```

We can say that Mary liked article 42 is an action.

After component dispatching action, Redux store calls the defined reducer function given and passes to arguments to the reducer i.e. the current state tree and the action. The root reducer then may combine the output of multiple reducers into a single state tree. The Redux store saves the complete state tree returned by the root reducer. This state tree is the new state of the application.
@ngrx/store

@ngrx/store is a state management extension for Angular applications inspired by Redux. It imports all core concepts from Redux[19].

a) Store

Store is the core structure where actions, reducers and state are located. Store receives actions as inputs, and transmit actions to reducers. Reducers produce new state depending on the action and emit new the state. Store holds the new state until it changes with another action.

![Block diagram of Store](image)

*Figure 2-9: Block diagram of Store*

b) Action

An action consists of two parts, a type and payload. Type is required to make reducer enable to distinguish the actions. Therefore type must be unique for each action.

![Action structure](image)

*Figure 2-10: Action structure*

c) Reducer
Reducer is the core element in Store and it is only responsible for changing the state. A reducer takes action and current state as input parameters and return new state according to the function inside it. This is shown in the diagram below *(State is immutable in Redux architecture, so reducer does not mutate the state but produces a new state.)*

![Reducer Diagram](image)

*Figure 2-11: Block diagram of reducer*

2.4.2 D3JS

D3JS is a framework developed using JavaScript for mapping data to a chart(s). D3 stands for Data Driven Documents.

2.5 IDE (Integrated Development Environment)

An Integrated Development Environment is a set of information system development concepts, techniques and computer-aided tools[20]. An IDE is a software application used by programmers for code development. We used Visual Studio Code as an IDE.
2.7 Operating Systems

An operating system is a set of programs that controls how a hardware device or system operates. It is a technical layer of software for driving software components [21].

2.8 Requirements Specification

Requirements specification includes phase of requirements engineering which is the process of determining user expectations for a product. The features called requirements are quantifiable, relevant and detailed.

2.10 Use Case

Use case diagrams are used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actor). Each use case provides some observable and valuable result to the actors or other stake holders of the system.

2.11 Network Topology

A topology determines the way in which different devices of the network are arranged and how they communicate with each other. The topology can either be physical or logical. Physical topology refers to the physical layout of the system i.e. how devices are actually interconnected either through cables or wireless communication links. Logical topology refers to how data flows through the network from one node to another without worrying about physical interconnection of the devices[22].

2.11.1 Centralized architecture

In centralized architectures, there is one central server to which many clients connect. The fact that there is only one server in the system, means that the system is easily manageable, all devices are connected to a central controller often referred to as the master. The peripheral nodes are called slaves. Securing a centralized system is very easy, there is only
one host that needs to be protected. An infrastructure that uses an access point operates mainly in a star topology.

![Star topology diagram]

Figure 2-0-12: Star topology

2.11.2 Point to point networks

This is a topology in which two devices are directly connected with each other[22].
Figure 2-13: Point to point connection between devices
CHAPTER THREE: METHODOLOGY

3.1 System Design

3.1.1 Block Diagram

![Block Diagram](image)

*Figure 3-1: Block diagram showing methodology*

The system consists of three major sections the input, RouterOS API and the output. The input consists of a client graphical user interface cross-platform desktop application that is responsible for issuing user (network administrator) commands through and configurations through a press of a button. These commands and configurations are translated into RouterOS API functions through a wrapper API that sits on top of the RouterOS API and these functions call the RouterOS API.

The RouterOS API then resolves these function calls into raw string RouterOS commands that it sends to the connected AP nodes (RouterBoards - for single antenna boards) through a socket connection made through the RouterOS Client instantiated at the time of the login.
to the RouterBoard. Each RouterOS Client manages a single RouterBoard and therefore there as many RouterOS Clients as the number of RouterBoards. These raw commands (the output) in turn access the several menus that RouterOS exposes to configure, manage or monitor each connected RouterBoard to achieve the desired functionality.

Through printing data at the different RouterOS menus, the hotspot manager can query the RouterBoards for the data associated with those menus which data in turn can be interpreted and presented as visually to a network administrator in form of graphs, tables or any other visually understandable interpretation.

3.1.2 Inputs and Outputs of Router Operating System

![Figure 3-2: Inputs and Outputs to Router Operating Systems](image)

Currently, inputs to router OS are raw commands. This leads to time wastage inputting raw commands to a system by network administrators. The system we designed enables one to simply click a button so as to create a hotspot rather than write many commands. For example to configure a wireless interface on a MikroTik router. These would be the sequence of steps.

3.1.3 Data Flow Diagram

The Data flow diagram in figure 1 shows the flow of data in the entire system for operation of the hotspot manager.
Figure 3-3: Data flow diagram
3.3 Hardware Architecture

In the hardware architecture, a WLAN controller acts as a central controller that controls all access points connected to it wirelessly. These Access Points provide network access to other computing devices such as computers, phones, printers, tablets and so on.

3.4 Scaling the Hotspot

In order to have centralized management that is scalable to many Access Point, we have set CAPsMAN (Controlled Access Point system Manager) on the MikroTik routerboard and other Controlled Access Points (CAPs) connected to the router. This ensures instead of configuring each router one by one, with CAPsMAN changes are specified and all CAPs receive the configuration changes[23]. CAPsMAN offers a centralized registration table that allows you to easily monitor devices and create Access Lists for devices in the network from a single device.
3.5 Use Case Diagram

Figure 3-5: CAPsMAN Setup

Figure 3-6: Use Case Diagram for Administrator
3.5.1 Hotspot Use Case for Administrator

Table 3.5.1.1: Use Case Add hotspot

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Add Hotspot</th>
</tr>
</thead>
<tbody>
<tr>
<td>Participating Actor</td>
<td>Initiated by Administrator</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>1. The Administrator signs into the system.</td>
</tr>
<tr>
<td>Flow of Events</td>
<td>2. The Administrator clicks on a button to add a hotspot.</td>
</tr>
<tr>
<td></td>
<td>3. He then adds an SSID for the hotspot and password.</td>
</tr>
<tr>
<td></td>
<td>4. He then finally clicks the save button to save the hotspot that has been added.</td>
</tr>
<tr>
<td>Exit Condition</td>
<td>5. The hotspot is added to a table of hotspots created</td>
</tr>
</tbody>
</table>

Table 3.5.1.2: Use Case Disable Hotspot

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Disable Hotspot</th>
</tr>
</thead>
<tbody>
<tr>
<td>Participating Actor</td>
<td>Initiated by Administrator</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>1. Administrator signs into the system.</td>
</tr>
<tr>
<td>Flow of Events</td>
<td>2. The Administrator clicks on a disable button to disable hotspot</td>
</tr>
<tr>
<td>Exit Condition</td>
<td>3. The hotspot is removed from table of hotspots.</td>
</tr>
</tbody>
</table>
### Table 3.5.1.3: Use Case Block user

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Block User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Participating Actor</td>
<td>Initiated by Administrator</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>1. Administrator is signed into hotspot manager.</td>
</tr>
<tr>
<td>Flow of Events</td>
<td>2. Administrator checks registration table for users.</td>
</tr>
<tr>
<td></td>
<td>3. Administrator then clicks particular user.</td>
</tr>
<tr>
<td></td>
<td>4. He then clicks a button to block the user.</td>
</tr>
<tr>
<td>Exit Condition</td>
<td>5. User is blocked from the hotspot.</td>
</tr>
</tbody>
</table>

### Table 3.5.1.4: Use Case Blacklist User

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Blacklist User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Participating Actor</td>
<td>Initiated by Administrator</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>1. Administrator is signed into hotspot manager.</td>
</tr>
<tr>
<td>Flow of Events</td>
<td>2. Administrator checks registration table for users.</td>
</tr>
<tr>
<td></td>
<td>3. Administrator then clicks particular user.</td>
</tr>
<tr>
<td></td>
<td>4. He then clicks a button to blacklist a user.</td>
</tr>
<tr>
<td>Exit Condition</td>
<td>5. User is put into a blacklist user table and cannot transact anymore on the hotspot service.</td>
</tr>
</tbody>
</table>

### Table 3.5.1.5: Use Case Limit Bandwidth

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Limit Bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Participating Actor</td>
<td>Initiated by Administrator</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>1. Administrator is signed into hotspot manager.</td>
</tr>
<tr>
<td>Flow of Events</td>
<td>2. Administrator checks registration table for users.</td>
</tr>
<tr>
<td></td>
<td>3. Administrator then clicks particular user.</td>
</tr>
</tbody>
</table>

23
4. He then clicks a button to limit bandwidth of a user.

Exit Condition

5. User speeds reduce to bandwidth limit.

Table 3.5.1.6: Use Case Scale Hotspot

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Scale Hotspot</th>
</tr>
</thead>
<tbody>
<tr>
<td>Participating Actor</td>
<td>Initiated by Administrator</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>1. Administrator is signed into hotspot manager.</td>
</tr>
<tr>
<td>Flow of Events</td>
<td>2. Administrator clicks on a button to scale a hotspot.</td>
</tr>
<tr>
<td></td>
<td>3. Administrator then clicks button to add routers to which hotspot is scaled.</td>
</tr>
<tr>
<td>Exit Condition</td>
<td>4. Hotspot is scaled to other routers</td>
</tr>
</tbody>
</table>

Table 3.5.1.7: Use Case View Usage Statistics

<table>
<thead>
<tr>
<th>Use case name</th>
<th>View Usage Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Participating Actor</td>
<td>Initiated by Administrator</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>1. Administrator is signed into hotspot manager.</td>
</tr>
<tr>
<td>Flow of Events</td>
<td>2. Administrator clicks on a particular hotspot to scale a hotspot.</td>
</tr>
<tr>
<td></td>
<td>3. He then clicks on a button to view the graph of users or networks.</td>
</tr>
<tr>
<td>Exit Condition</td>
<td>4. A graph of network statistics is shown.</td>
</tr>
</tbody>
</table>

3.5.2 Hotspot Use Case for Hotspot Visitor

Table 3.5.2.1: Use Case Login

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Login</th>
</tr>
</thead>
<tbody>
<tr>
<td>Participating Actor</td>
<td>Initiated by Hotspot Visitor</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>1. Hotspot Visitor starts his laptop and searches for available hotspots.</td>
</tr>
</tbody>
</table>
### Flow of Events

| 2. | Hotspot Visitor enters his login credentials in input form presented and hits button to submit data. |
| 3. | AP intercepts requests, interprets it in the hotspot manager. |
| 4. | The hotspot manager checks if login credentials and returns message Connected if credentials are correct and failure message if credentials are incorrect. |

### Exit Condition

| 5. | The hotspot visitor receives either a success or failure message. |

---

**Table 3.5.2.2: Use Case Use System**

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Use system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Participating Actor</td>
<td>Initiated by Hotspot Visitor</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>1. Hotspot Visitor is successfully authenticated by AP.</td>
</tr>
<tr>
<td>Flow of Events</td>
<td>2. AP authenticates visitor by checking input parameters and matching those given by administrator if so it allows the query to pass.</td>
</tr>
<tr>
<td></td>
<td>3. AP adds visitor to a registration table.</td>
</tr>
<tr>
<td></td>
<td>4. AP then forwards browsed site to visitor.</td>
</tr>
<tr>
<td>Exit Condition</td>
<td>5. The hotspot visitor can browse a site.</td>
</tr>
</tbody>
</table>

---

**Table 3.5.2.3: Use Case Logout**

<table>
<thead>
<tr>
<th>Use case name</th>
<th>Logout</th>
</tr>
</thead>
<tbody>
<tr>
<td>Participating Actor</td>
<td>Initiated by Hotspot Visitor</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>1. Hotspot Visitor is successfully logged onto hotspot.</td>
</tr>
<tr>
<td>Flow of Events</td>
<td>2. Hotspot Visitor clicks button to disconnect or logout of hotspot.</td>
</tr>
<tr>
<td></td>
<td>3. AP then removes the visitor from the registration table.</td>
</tr>
</tbody>
</table>
3.6 Stages of Development of System

Each stage indicates phases of the development life cycle for the system.

3.6.1 Requirements Specification

Requirement specification involves specifying the system requirements and functionality of the system. At this stage, we looked at different specifications and required components such as the MikroTik router, topology to use for network design and architecture to use. We chose the Redux architecture that follows a unidirectional flow of data.

3.6.2 Prototyping

For prototyping, we used a tool called Figma to come up with a mockup design that we followed to design the UI of the hotspot manager as shown in Figure 22.
3.6.2. Designing and Writing Code

3.6.2.1 Designing API

Figure 3-9: Prototyping with Figma

Figure 3-10: API for RouterOS
We designed the API using Visual StudioCode creating functions for the Router OS. The functions included creating network, updating Network.

3.6.2.2 Designing State Reducer based on Redux Architecture

![Image of code editing in Visual StudioCode](image)

*Figure 3-11: Implementing Redux using State Reducers*
3.6.2.3 Exporting API module

We wrote code that can export the created API as a module to be used by other applications. By using a function module.exports, all the API functionality can be exported to be used as a single module in another application.

Figure 3.12: Exporting API module
3.6.2.4 Designing Code for API Wrapper

Figure 3-13: Code for API Wrapper

We wrote code for the API Wrapper to extend functionality for use for the hotspot manager. The functions included getting creating connection, creating Network, enabling Network, disabling Network, get available Networks, check connected users, check hotspot status, block user, unblock user, limit bandwidth.
3.6.2.5 Including API in Electron

We designed the API and wrote code to interface it with Electron a JavaScript Framework which builds the JavaScript code for a desktop application.
3.6.2.6 API Usage in Electron

We designed the API and wrote code to interface it with Electron a JavaScript Framework which builds the JavaScript code for a desktop application.

Figure 3-15: API Usage in Electron
3.6.2.7 Designing code for blocking User

We wrote code for blocking a user using access lists which ensure that a user can be blocked in case an access rule is applied. An access rule is imposed onto the users IP address which blocks the user from accessing network access.
3.7 Physical Level Diagram (based on TCP/IP network model)

Figure 3-17: Hotspot Manager Level based on TCP/IP network model
CHAPTER FOUR: KEY RESULTS

4.1 Hotspot Creation

Figure 4-1: Network Administrator Login

A hotspot administrator adds his account name and password for him to be able to use the hotspot manager. If the credentials pass authentication, the administrator accesses another screen with buttons.
In order to create a hotspot, the administrator can input an SSID which is basically the name to identify the network, the password and then a hotspot is created.
Figure 4-4: Clicking button to set up hotspot
A view for connected users by MAC address is also shown in form of a registration table where details of the user are shown. The view indicates the upload speed in kilobytes per second, download speeds in kilobytes per second and also indicates the bandwidth limit on a user.
As can be seen a graph of bandwidth against the time showing network traffic is plotted by the application indicating bandwidth usage per second.
Figure 4-7: Viewing User Traffic

Figure 4-8: Limiting bandwidth for a user
5.1 Conclusions
We designed a hotspot manager can do bandwidth management, volume based network monitoring and scale a hotspot to multiple routers or access points.

5.2 Challenges faced
Access to equipment was a challenge.

5.3 Recommendations
We recommend that this hotspot manager is adopted for use at CEDAT for extending network access to more users especially when available access has been scaled down due to many users.
REFERENCES


