Final Year Project (CS)

Namal Remote Server Monitoring System

Final Report

Submitted by: Khizar Farooq

UOB: 15026422

Supervisor: Dr. Adnan Iqbal

Co-Supervisor: Dr. Noman

Abstract

In most national and international organizations there are great datacenters, with many different server types, hosting countless apps. Monitoring the functionality of each and every server is always a major issue. The problem is more serious when the usual number of network engineers works less during late night shifts. In general, when hosting applications on their servers, organizations sign a service level agreement (SLA) with the client. Any lack of commitment by the organization to meet any part of SLA may result in business loss or legal action. So, knowing about the status of the server every second becomes very important for the organizations to take the right measures immediately in case of any problem. In addition, organizations would be curious to aware of these server failures instantly before the client begins to complain about the problem. So, there are two ways to solve these issues. One is to check the server's performance manually again and again after some time slot that is the very imperfect way in this modern world. And, the other one is to check automatically through use powerful computing machine in an efficient way. Even though many organizations use customized software system with limited functionalities to monitor the server behavior, but there exists a need to monitor the different type of multiple servers through one system. The major reason for this project is to develop an efficient fully automated system for large organizations to provide help to network engineer's in server monitoring and optimize their time.

Table of Contents

Chapter 1

1.Introduction

1.1 Purpose

Purpose of writing this document is to give a clear and detailed description of a project named as Namal Remote Server Monitoring System. It will provide, Graphical User Interface, features, constraints under which system will operate, and all functional and non-functional requirements of the project. This document is to help developers, testers, and users to understand this software.

1.2 Project Synopsis

A system which will be able to monitor servers (Web, DNS, DHCP, Proxy and Print server) remotely according to admin configuration. The system will instantly alert on hardware or service failure, to admin or network/system engineer before complaints of users (Mathapati and Aswatha, 2013).

1.3 Motivation for Project

In Pakistan, there are most of the organization are exist that facing a lot of problem during analyzing the current status of servers. The customize software solution currently used in these organizations are not enough efficient to help network engineer's in the monitoring of server status and alert to them in case of issue. Keeping in mind the above-mentioned fact, it is clear that the fully automated server monitoring system can fulfill the organization needs.

1.4 Proposed Methodology

A web-application and Hybrid app will be created which can monitor servers through active monitoring technique. To do this, the first part will be to create customized traffic and send it to servers to measure their given below potential metric(Mathapati and Aswatha, 2013).

- Host Availability on Network
- Application Availability
- Port Availability
- Protocol Availability
- Latency of response
- Packet Loss
- Success and error rates

In the second step, all the responses come from these servers will store in the relational database. In the next step, these responses will be analyzed through a program "analysis agent" on the base of potential metric. In the end, if any type of hardware or service issue happens, the system will generate an alert and send it to network/system engineers through technologies like SMS, E-Mail etc.

1.5 Document Structure

The second chapter is about the background and overview of usual monitoring techniques and tools practically working in organizations relevant to this system. In this chapter, we will be discussing drawbacks of these tools and also comparing them with this system, which provokes the reason of existence of this system. We'll also try to learn their limitations to get the best out of this system. The third chapter is about system analysis which provides a brief summary of this system's (Namal Remote servers monitoring System) functional requirements, non-functional requirements. The fourth chapter is relevant to system design perspective which covers these four actions-architectural designs, data structure design, interface design, procedural design. Architecture design helps to understand the whole system including how it will work. Data structure design describes how the data will actually store and maintain at the backend. Interface design elaborates the frontend view of this system to which a user interacts in order to use it. The fifth chapter covers all aspects of the actual implementation of this system including user interface implementation, java application, analysis agent and alerts. The sixth chapter is relevant to testing using a different testing approach including usability testing, Functionality testing, and unit testing. The seventh chapter is about the conclusion and future work.

Chapter 2

2.Literature Review

2.1 Monitoring Techniques

There are different techniques to monitor the network. The two usual techniques are given below. Both have their values. These techniques can be used at the same time with one another.

- Active Monitoring
- Passive Monitoring

2.1.1 Active Monitoring

This technique depends on the ability of the network environment to inject test packets to measure the network service status. As such, it does make extra artificial traffic. This traffic collects a smaller amount of data specific to the problem. This technique gives the flexibility to control explicitly generation of packets for measurement scenarios. This involves control on the nature of traffic generation, the timing, frequency scheduling, packet size, and types to emulate various application, the path and function were chosen to be monitored. This technique helps to test what you want when you need it (Stanford Linear Accelerator Center, 2001) (Toit, 2016) (Cottrell, 2001)(Andreozzi, Ciuffoletti and Ghiselli, 2019).

On the other direction, the limitation of this technique is that it is more resources intensive than passive monitoring (Stanford Linear Accelerator Center, 2001) (Toit, 2016).

2.1.2 Passive Monitoring

According to this technique, don't need to generate any artificial traffic and inject it into the network. It monitors real user historic traffic and makes predictive analysis easier. This technique requires specialized tools such as sniffer or aCxMon to measure traffic. This technique helps to collect a large volume of data that can be mined for a wide variety of information. It works like an "observational study", making it helpful for analysis of large data volumes (Stanford Linear Accelerator Center, 2001) (Toit, 2016) (Andreozzi, Ciuffoletti and Ghiselli, 2019) (Cottrell, 2001).

On the other dimension, Passive monitoring is limited to measuring traffic on the network the device is connected to (Stanford Linear Accelerator Center, 2001) (Toit, 2016).

2.2 Existing Monitoring Tools

In this part of the chapter, I have analyzed existing monitoring tools.

The first part of this analysis was to find top currently available tools in the market relevant to this system and how they are working. Moreover, how they can help this system. These tools are listed below.

- Nagios
- Zabbix
- Ganglia

2.2.1 Nagios

Nagios is one of the most popular and powerful surveillance tools. It is an open source company monitoring package based on Unix with a web-based front-end developed by “Ethan Galstad”. It gives their customers quick awareness of organization’s about mission-critical infrastructure. The first time, it alerts to their users when the thing is going wrong and the second time, it alerts to them when the problem has been removed. It reduces future issues before they affect end-users and customers. Alerts can be delivered via SMS, email or custom script (Nagios, 2019).

How does it work?

It uses active monitoring methodology to monitor things that exist in the network and does it through either two ways. Given below “Figure 2.1” is the architecture diagram of Nagios.

- First one is an agent
- The second one is Native protocol

Agent

Nagios remote plugin executor (NRPE) is running as an agent used by Nagios for communicating with a remote host. Remote host means (Linux, Windows). It is installed on the remote host which users want to monitor from their Nagios server. The agent waits for check request from the Nagios server. After receiving the request, the agent will execute a plugin on the remote host to execute the request and return that response back to the Nagios server(Kind *et al.*, 2010)**.**

Plugins

Plugins are binary program or scripts that are responsible for executing the request of Nagios server. These are hosted on the remote host being monitored.

Native Protocols

Native protocols that Nagios uses are given below.

- Simple network management protocol (SNMP)
- Window management instrumentation (WMI)

Nagios sends requests to SNMP enabled device (Switch) and generate an alert on received information that is (All good, port down, anything else that may go down)(Kind *et al.*, 2010)**.**

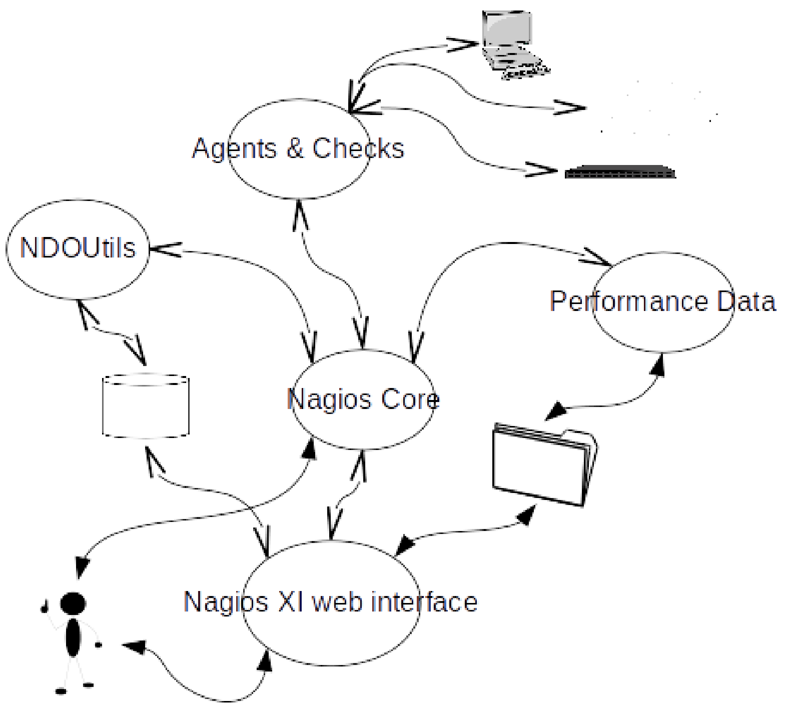


Figure 2.1: Nagios Architecture (Nagios, 2016)

2.2.2 Zabbix

It is an open-source monitoring tool which monitors the network, servers, virtual machines (VM) and cloud service. Zabbix provides monitoring matrices, CPU load and disk space consumption, among other network, uses ('2. What is Zabbix', 2019).

How does it work?

It monitors things in the network through two ways that are given below.

- First one is agent-less. In this technique, it simply verifies the availability and responsiveness of standards services such as SMTP and HTTP without installing any software on the monitoring host(Zabbix SIA, 2019).
- The second one is agent-based also called Zabbix agent that installed on UNIX and windows host to monitor statistics such as CPU load, disk usage and network utilization(Zabbix SIA, 2019).

Given below "Figure 2.2" is the architecture diagram of Zabbix.

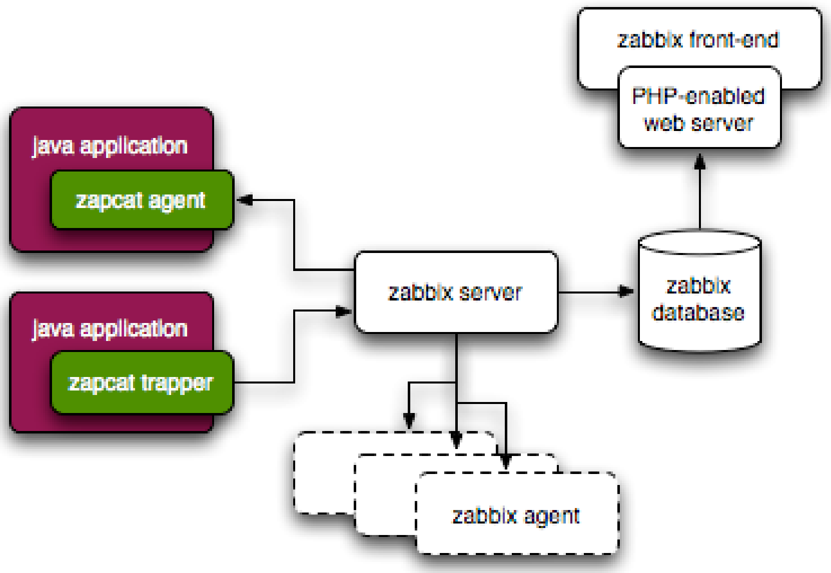


Figure 2.2: Zabbix Architecture (kjkoster, 2019)

2.2.3 Ganglia

It is a scalable distributed monitoring system for high-performance computing systems such as clusters and grid monitoring. Clusters are a combination of a system that works together. It is used to view either live or recorded statistics covering metrics like CPU load average or network utilization for many nodes. It is based on a hierarchical design targeted at federations of clusters. It relies on multicast-based listen/announce protocol to monitor state within clusters and uses a tree of point-to-point connections amongst representative cluster nodes to federate clusters and aggregate their state. It widely used technologies such as (XML) for data representation, XDR for portable data transport and RRD tool for data storage and visualization(Massie, Chu, and Culler, 2004)('Ganglia Distributed monitoring system', 201)**.**

How does it work?

It works through small agents (Gmond) and master agent (Gmated).

Ganglia monitoring daemon (Gmond)

It is a lightweight service that is installed on every machine which users would like to monitor. It has four main responsibilities that are given below.

- Monitor change on the host state
- Announce relevant change
- Listen to the state of another ganglia node via unicast or multicast.
- Answer request for an XML description of the cluster state.

Ganglia meta daemon (Gmated)

It is a service that collects data from other data source called gmated and gmond and stores their state to disk in round-robin database.

The Ganglia web front-end caters to system administrator and users. For example, one can view the CPU utilization over the past hour, day, week, month, or year. The web front-end shows similar graphs for memory usage, disk usage, network statistics, number of running processes, and all other Ganglia metric. Given below "Figure 2.3" is the architecture diagram of Ganglia.

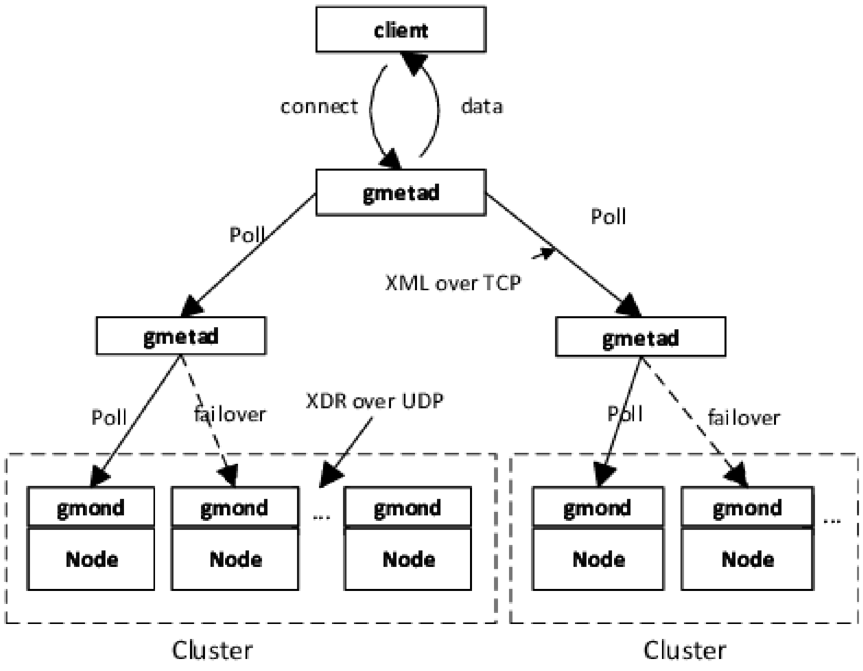


Figure 2.3: Ganglia Architecture (researchgate, 2019)

2.3 Discussion

Here we'll compare the above-mentioned tools with this system. We'll briefly discuss drawbacks of each tool, and compare it with this system. We'll also define why this system needs to exist.

Nagios

The learning phase is required to manage its configuration files and services. Many features such as wizards and interactive dashboard are not available on a free version of Nagios. These are available on Nagios XI, which is too much expensive. Nagios core has a very complicate the graphical interface. Nagios has not the ability to network throughput (bandwidth uses or available) Nagios just monitor the network. It can't manage the network. Mobile web interface is incomplete. Sometimes when you turn off the system and again turn on the database crashes and can cause to make Nagios unstable. Nagios need to tell it what you want to be monitored and how(Mongkolluksamee, Pongpaibool, and Issariyapat, 2010).

Zabbix

Difficult to perform the first installation due to some templates may require 3rd party plugin from GitHub. It doesn't offer their clients the same flexibility that they get from open source alternatives. The database can grow very large, very quickly If not tuned properly. Reporting is rather weak(Zabbix *et al.*, 2019).

Ganglia

Updating can be difficult, especially if the libdnet layout changes. Embedding third-party libs in the source violate the principle of the FreeBSD ports. Might cause problems with static binaries (unlikely since we manage with rrd tool). More things require to build from source. Only works if libdnet doesn't change their API. Rename dnet headers so installed ones won't get picked up(Massie, Chun and Culler, 2004)**.**

Namal Remote server monitoring system

The graphical user interface will be human-friendly. The stable database will be guaranteed. Easy to use server monitoring services. Autoconfiguration/learned to monitor. Easy installation unlike Zabbix (No 3rd party plugin required).

Chapter 3

3.System Analysis

3.1 Functional Requirement

Functional requirements are totally depending on the expected output or functioning of the system to be created. This system has multiple users just like a network administrator, and assistant network administrator. Users and his functions are given below.

Network Administrator

- He has access to add a new user of this system and assign permissions to him according to his post level.
- He can easily configure his and other user account setting.
- He can easily add a new server (Web, DHCP, DNS, Proxy, Print) to monitor its services.
- He can configure monitoring setting of the added server.
- He can configure notification setting.
- He can watch the status of the service of servers anytime.
- He can easily communicate with the developer of this system.
- He got alert if any kind of hardware failure or service issue happens.

Assistant Network Administrator

He can only access that feature of this system what network administrator gave him permission to access.

3.2 Non-Functional Requirement

The main attributes of non-functional requirements are performance level, usability, availability, scalability, and maintainability (Barnes-Hoggett, 2018). Given below list explains the non-functional requirements.

- #### Performance

  Response time and the processing time is the performance of any software component. The performance of this system is independent of the internet due to the local host. This system will show information instantly against the action of admin.
- #### Availability

  Availability of this system is based on the availability of the database and analysis agent. In this scenario, if one of these services is not available. In result, this system will not respond to the user.
- #### Scalability

  Scalability of this system is totally independent of the number of system users and the number of servers in which admin is interested to monitor their services. MySQL relational database gives the flexibility to store and manage easily large data in relations. So, this system will not face any difficulty in communicating with the database and showing the status of server services.

- ## Usability

The user interface design of this system is admirable. The system users are trained by the system itself as they move on.

Chapter 4

4.System Design

System design is the initial step in between three activities that are design, code, and test that is mandatory to develop and verify software. Through this, we can easily accurate translate a client view into the finished product. If we have not a strong system design, then we are not sure our final system will be stable or unstable.

Without a strong design, we risk building an unstable system – one that will be difficult to test.

4.1 Architecture Design

Architecture design is the initial part of designing in the development process for any engineered product that is given below in "Figure 4.1". Purpose to make the design of this system at the beginning is to generate a model which represent that entity that will later be built. As I said earlier this part of chapter show detail about all system components and their roles of Namal remote server monitoring system that is the following.

4.1.1 Components

This system has four parts given below.

- Host
- User Interface
- Java Application
- Database
- Analysis agent
- Cell Phone

Host

The host is actually a physical computer where this system will be hosted.

User Interface

It is a front-end of a system for interacting with network admin or any person who have the right to control network. Through UI, these users can easily pause or start monitoring services of those servers in which he interested. Moreover, they can also see all the alert on the web interface and received on their cell phone.

Java Application

It behaves like a server which receives https request from a user through the web interface of Namal remote server monitoring system to run and stop services of added servers according to their configurations. It also stores responses of each server service in MySQL database come from the servers in which admin is interested to monitor.

Database

Relational database “MySQL” will be used as a database at the backend of this system. It will play the role of a storage hub where all responses will save automatically against every request. It will exist where this system will be hosted.

Analysis Agent

It is the most important part of this system which analyzes all responses already saved in the database and generate alerts in the result on bases of hardware failure or service issue happening. It will also exist where this system host.

Cell Phone

It is also a component of this system for receiving an alert of that server in case of its hardware failure or any kind of service issue.

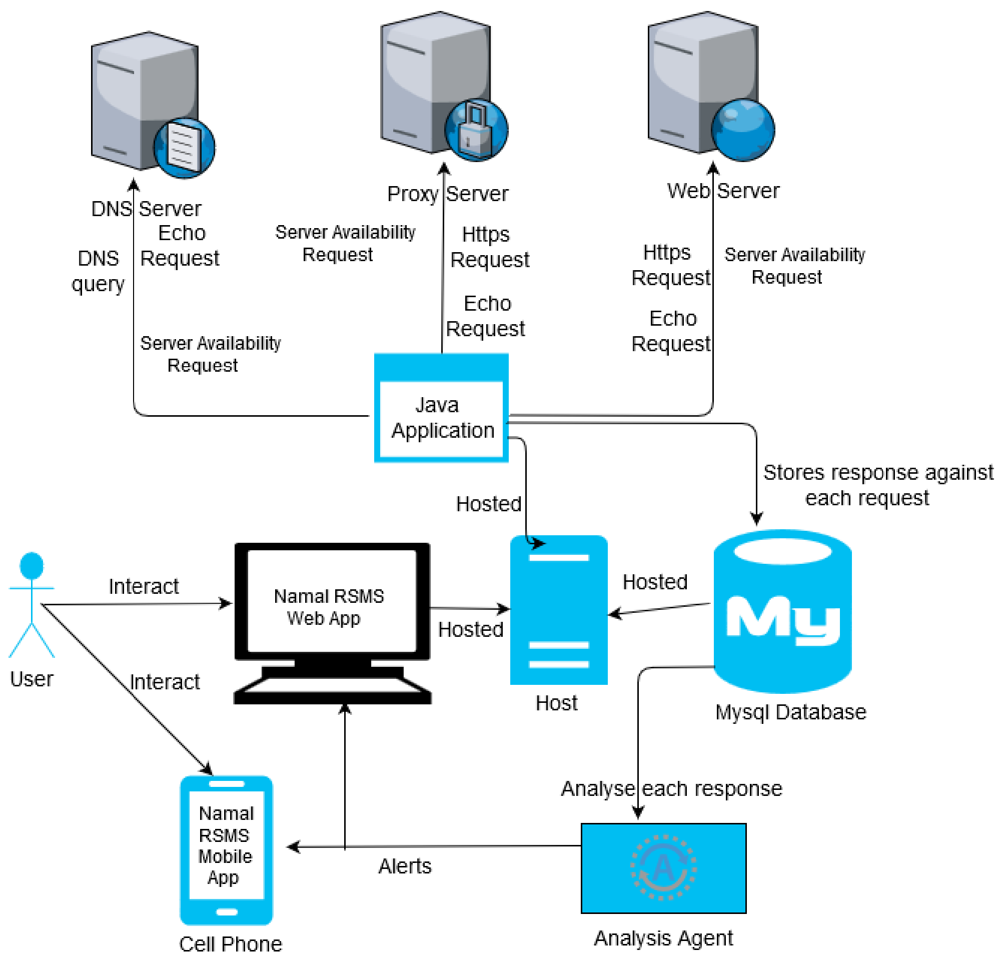


Figure 4.1: Namal remote server monitoring system Architecture

4.2 Data Structure Design

The data structure is a technical format for maintaining and storing data. Array, file, record, table and so on are the general data structure. Every data structure is designed to organize data to perform a specific work so that it can be accessed and worked with in suitable ways. To store and manage data of this system, the MySQL database management system played a big role at the backend. All data is stored in relations. These relations have system user's basic info, monitoring server, and their services info.

Dataflow Diagram

A data flow diagram helps to show the flow of data. It contains inputs, outputs, data stores, and the different subprocess the data moves through. Namal remote server monitoring system has two types of users. So, dataflow diagram with a complete description of this system according to user roles and permission are given below in "Figure 4.2" and "Figure 4.3".

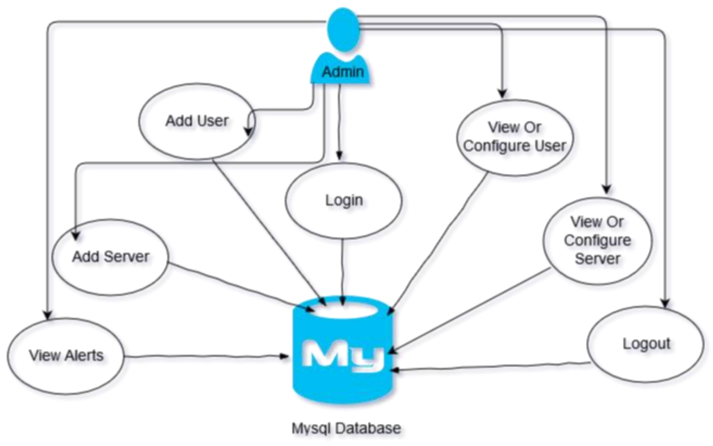


Figure 4.2: Namal remote server monitoring system Admin Level User Data Flow Diagram

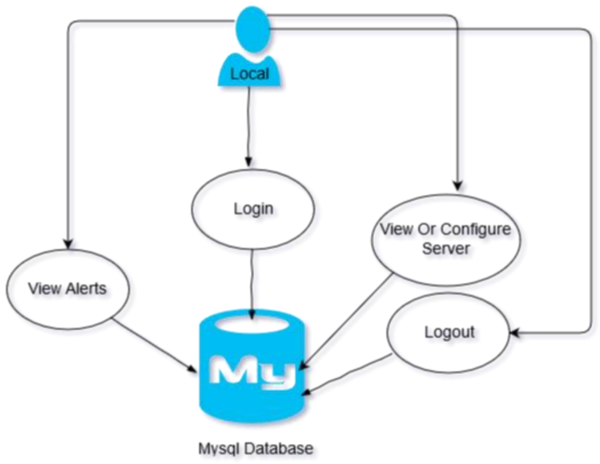


Figure 4.3: Namal remote server monitoring system Local User Data Flow Diagram

Entity Relationship Diagram

Entity relationship diagram of Namal remote server monitoring system is given below in “Figure 4.4”. It specifies what is the name of entities and relations which stores all the information in tabular format and their relations to each other. Entities are the object in the system that we want to store information about. Tables are known as relations in entity relation diagram.

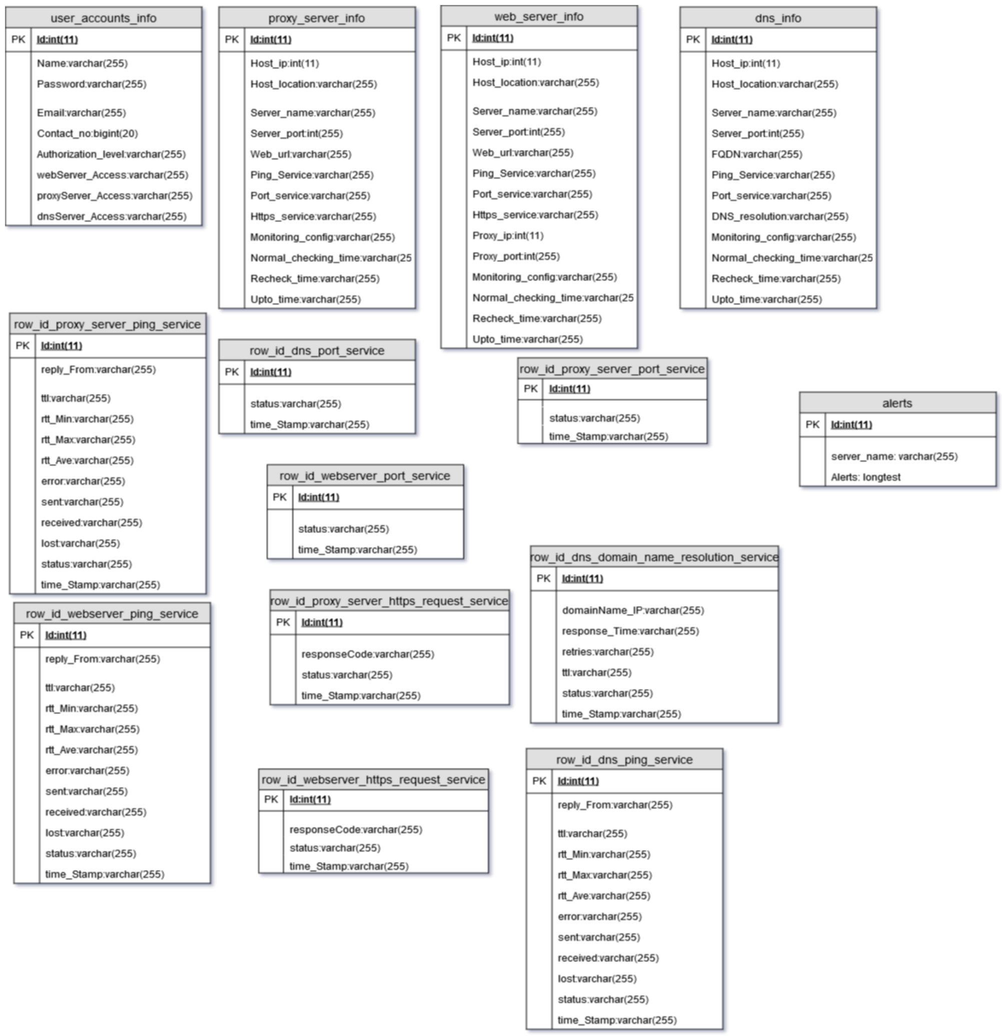


Figure 4.4: Namal remote server monitoring system Entity Relationship Diagram

4.3 User Interface Design

The user interface is the frontend view of the software to which the user interacts in order to use it. It is designed such a way that it is assuming to deliver the user a deep understanding of the software.

In software engineering, interface design of the software has two parts. One part is to design the interface for system user and the other part is to design the interfaces that role is to interact with other systems, which are essential to the functioning of the system that is being designed. Namal remote server monitoring system has also two parts. First part is a user interface to interact with network engineers. Another part is backend design that will play like a middleware to interact with this system or another server in which network engineers is interested to monitor. The user interface of this application is very simple and user-friendly. And, it is very good-looking with use of current modern design techniques. Tools what I have used to design the actual user interface and initial mockup design are briefly explained in the given below section.

Use case Diagram

Use case diagram of this system according to user level or permission are given below in "Figure 4.5" and "Figure 4.6".

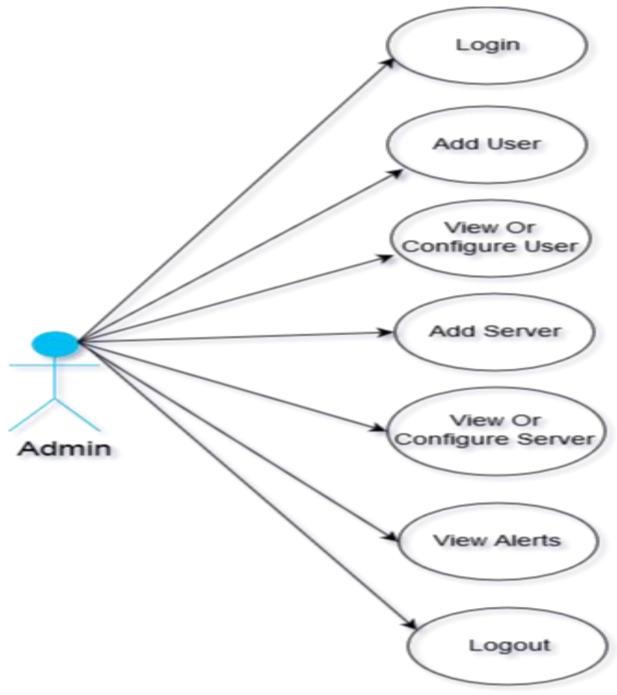


Figure 4.5: Namal remote server monitoring system Admin Level User Use Case Diagram

We can easily understand after saw this picture; admin level user has complete access to this system features.

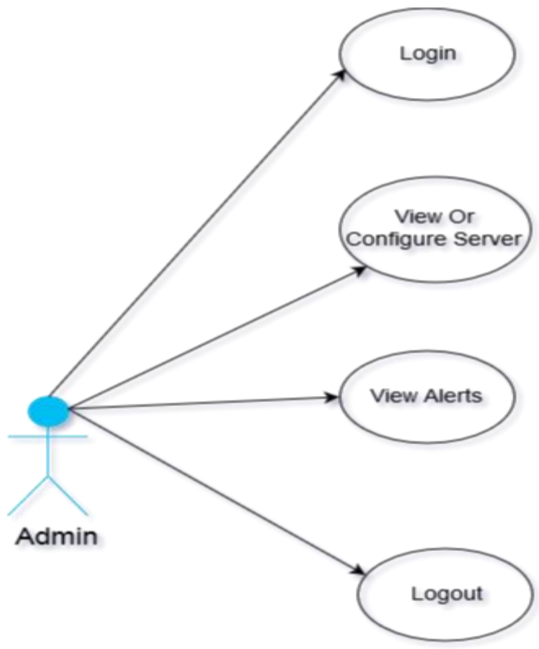


Figure 4.6: Namal remote server monitoring system Local User Use Case Diagram

According to this diagram, it is clear local users has limited access to this system feature what have admin give the right to them. They can only configure server or view alerts against that server what they have access.

4.3.1 Tools and technologies used

The tools and technologies I have used to develop this system are given below.

- Balsamiq
- Sublime
- IntelliJ IDEA
- WampServer
- PHP
- JAVA
- JAVASCRIPT
- BOOTSTRAP
- MYSQL
- Android Studio

Balsamiq

Balsamiq was used to design user interface mockup of this system.

Sublime

Sublime is an editor of source code across platforms. It supports a lot of programming and markup language natively. This editor was used to implement the user interface of this system.

IntelliJ IDEA

Built on Java, it provides a cross-platform approach to building tools for any kind of language, independently of whether this targets the JVM or not. (Jetbrains, 2019).This tool was used to implement the backend functionality part of this system.

WampServer

WampServer is an environment for Windows web development. It enables Apache2, PHP and a MySQL database to create web applications. PhpMyAdmin also enables you to easily manage your databases (WampServer, 2019). As I have said earlier, this system is web-based. So, the role of WampServer is to help in the implementation of database and functionality testing of this system.

PHP

It is a commonly used open source scripting language that is particularly suitable for web development and can be integrated into HTML (PHP, 2019) In this system, php plays a role at the backend to take system configuration and store into a database.

JAVA

It is a programming language and computing platform. This language was used to create packets artificially and inject them into the network environment, to measure the current status of the servers and their services.

JavaScript

JavaScript does not create standalone applications.it is mostly used with HTML. It is an OOP scripting language. Java code needs to be compiled while java scripts code is all text. JavaScript can be coded in any text editor. It was used to write functions which help to take system configuration and send it to PHP to perform an action on it.

BOOTSTRAP

Bootstrap is an open source toolkit for developing with HTML, CSS, and JS. (Bootstrap, 2019). It was used to create a web page of this system.

MYSQL

It is the most popular Open Source structural query language database management system, is developed, distributed, and supported by Oracle Corporation. (Mysql, 2019). It was used to create a relational database of this system and store all its generated data into separate tables to optimize speed for reading and write.

Android Studio

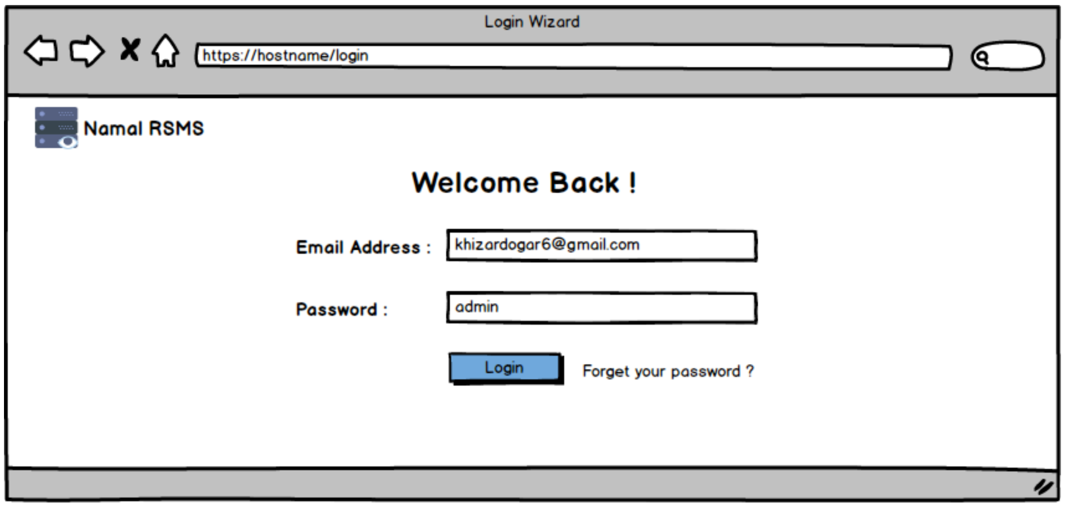
It is the official Integrated development environment for the development of mobile applications. It is used to develop a hybrid mobile application of this system. XML is used in android studio to develop the front end of the application. It also provides the facility of building the front end with drag and drop. Java is used to implement the back end of the application.

4.3.2 Prototype

Purpose of this part to show all the workflow of the system with the help of a prototype. Initially, its design was paper prototyped and then its mockups were designed on the software application. And, it is not a copy of any other system interface. It is totally based on self-imagination and creation.

Login

Given below screen in "Figure 4.7" is opened first on the start of the application. It has two input field email address, password and one button names login and one text view forget the password. On clicked login button, if the user is authorized then this screen redirects to the index screen. On clicked forget password, a popup appears on user screen which asks the user to enter login email address to send verification code on this email address. After sent verification code, this screen redirects to another screen names update the password.



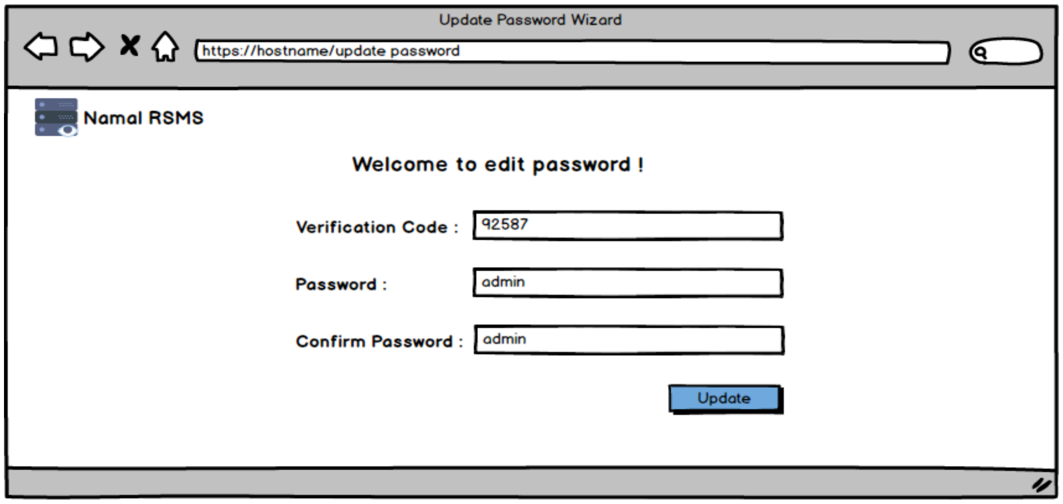
Email Address : khizardogar6@gmail.com

Password : admin

Figure 4.7: Namal remote server monitoring system Login Wizard

Update Password

Given below screen in “Figure 4.8” is display in front of users if they are admin level user and interested to update the password. This screen has three input field verification code, new password and re-enters new password which assists the system-user to update their password. If the verification code is right then the user is eligible to change password otherwise this screen redirect to the login screen.



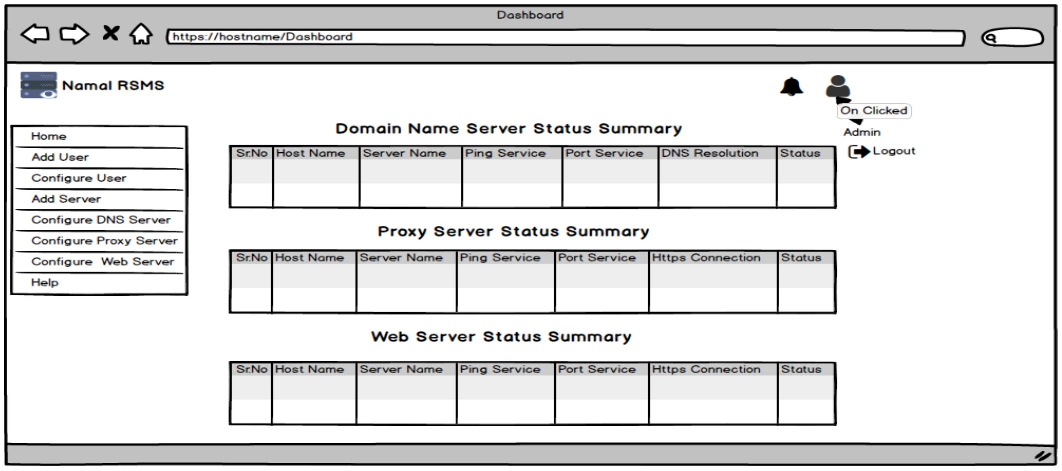
Welcome to edit password !

Verification Code :	92587
Password :	admin
Confirm Password :	admin

Figure 4.8: Namal remote server monitoring system Update Password Wizard

Dashboard

Given below “Figure 4.9” is the dashboard of the Namal remote server monitoring system. This screen helps users to interact with features of this system according to their roles and permissions. If the user has admin rights, then he/she can access all features of this system including add user, view or configure user, add a server, view status of servers or configure servers, view alerts against each server and log out.



Domain Name Server Status Summary

SrNo	Host Name	Server Name	Ping Service	Port Service	DNS Resolution	Status

Proxy Server Status Summary

SrNo	Host Name	Server Name	Ping Service	Port Service	Https Connection	Status

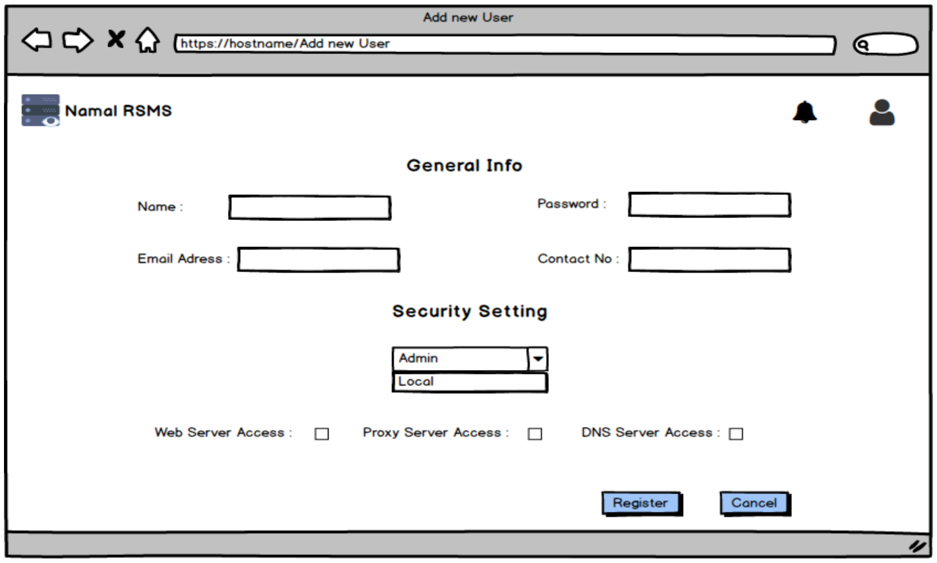
Web Server Status Summary

SrNo	Host Name	Server Name	Ping Service	Port Service	Https Connection	Status

Figure 4.9: Namal remote server monitoring system admin Level User Dashboard

Add New User

Given below screen in "Figure 4.10" is loaded on user screen after clicked add new user button from the side menu bar. This screen assists admin user to add a new user of this system. A new user can be an admin or can be local.



General Info

Name :		Password :	
Email Adress :		Contact No :	

Security Setting

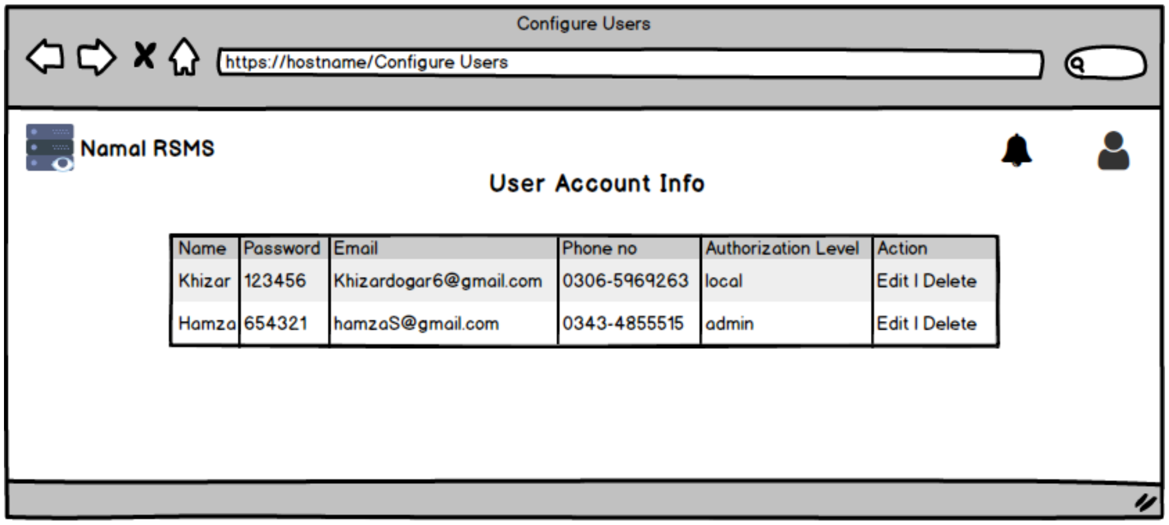
Admin
Local

Web Server Access : ☐ Proxy Server Access : ☐ DNS Server Access : ☐

Figure 4.10: Namal remote server monitoring system Add New User Wizard

Configure User

Given below "Figure 4.11" is the configure user screen that loaded in front of the user after clicked configure user button from the side menu bar. This screen shows a list of users to admin which assist him to configure any user info through click on the edit button. Moreover, admin can also remove a user by just click on the delete button.



Name	Password	Email	Phone no	Authorization Level	Action
Khizar	123456	Khizardogar6@gmail.com	0306-5969263	local	Edit I Delete
Hamza	654321	hamzaS@gmail.com	0343-4855515	admin	Edit I Delete

Figure 4.11: Namal remote server monitoring system Configure Users Wizard

Add New Server

Given below screen in "Figure 4.12" is loaded on user screen after clicked add new server button. This screen has a total of three images of servers named web, proxy, DNS which assist the user to choose what server he/she wants to add to monitor its services.

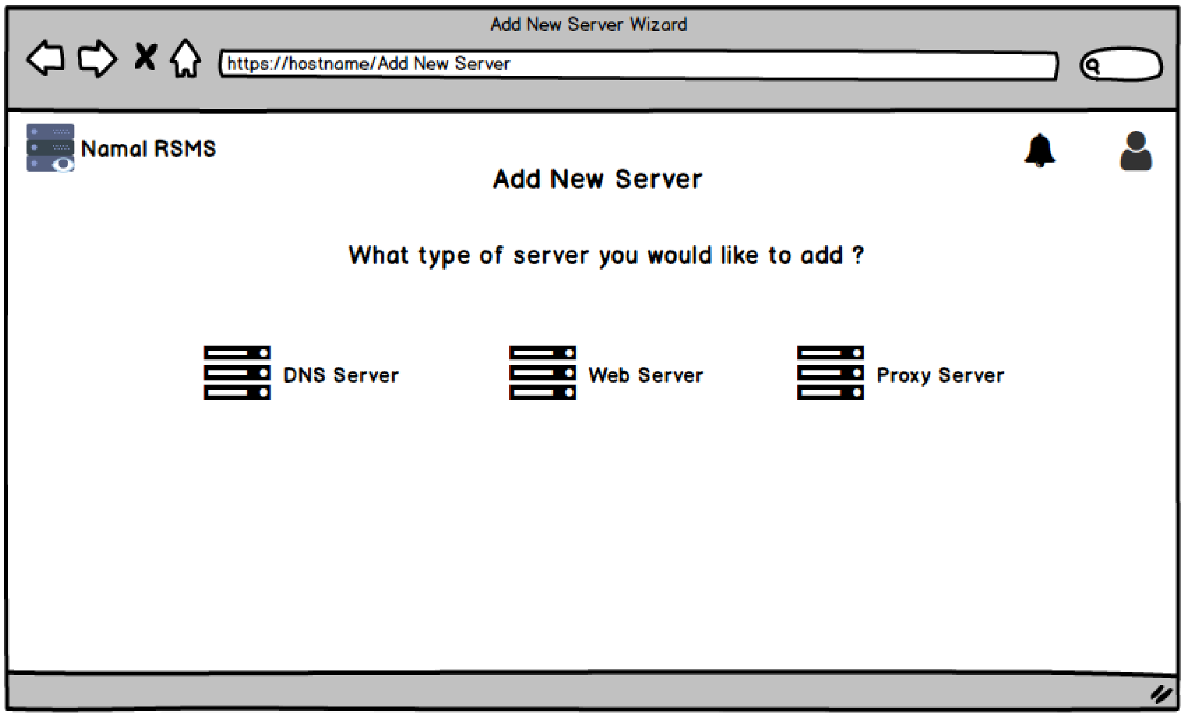
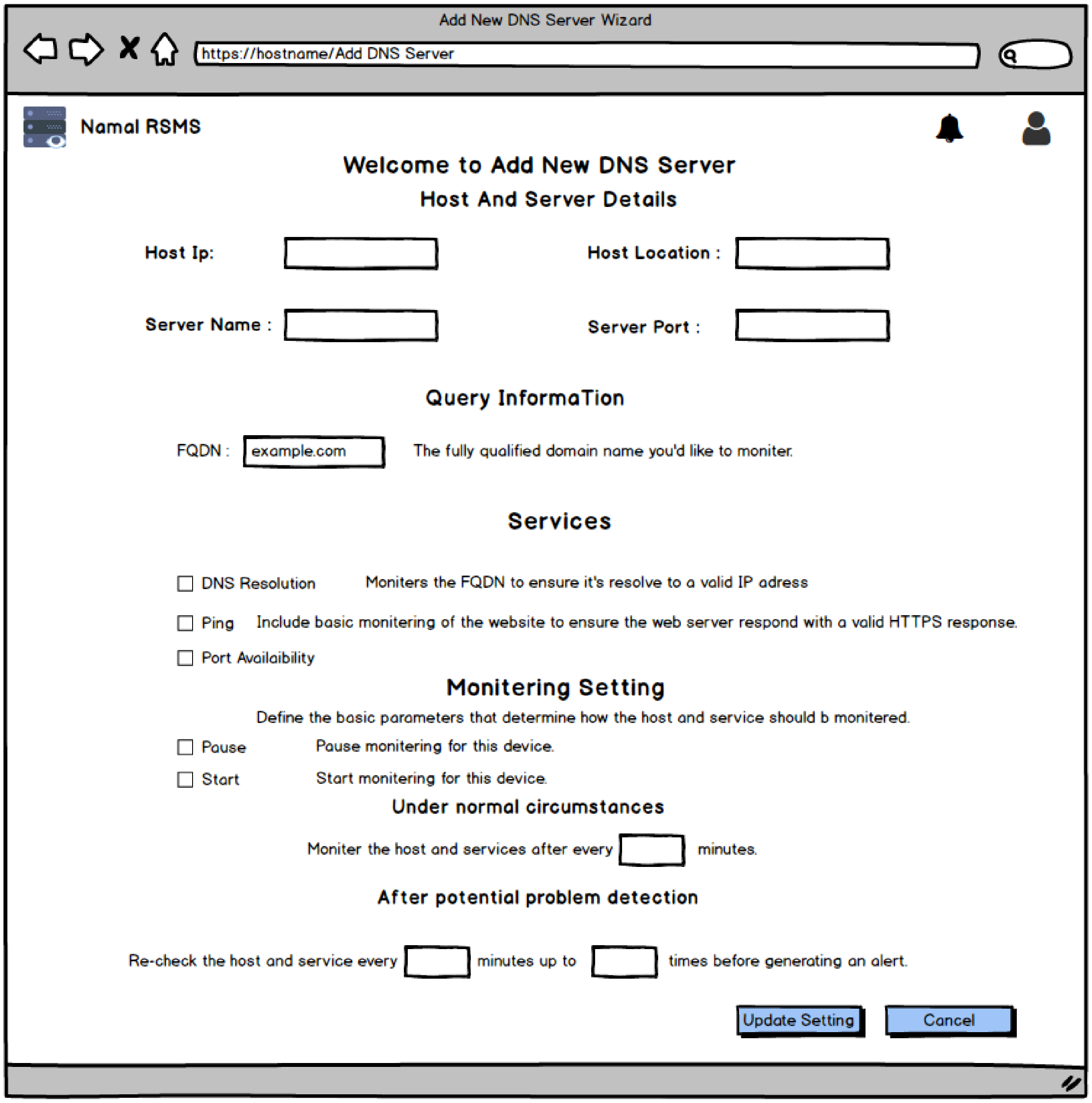


Figure 4.12: Namal remote server monitoring system Add New Server Wizard

Add New Domain name Server

Given below screen in “Figure 4.13” is shows after clicked add new DNS server image from Add new server screen. This wizard has eight input field, three checkbox and 2 radio buttons which assist the user to give complete information of domain name server that is mandatory for check status of it according to admin configuration.



Add New DNS Server Wizard

https://hostname/Add DNS Server

Namal RSMS

Welcome to Add New DNS Server

Host And Server Details

Host Ip:

Host Location :

Server Name :

Server Port :

Query InformaTion

FQDN : example.com The fully qualified domain name you'd like to moniter.

Services

☐ DNS Resolution Moniters the FQDN to ensure it's resolve to a valid IP adress

☐ Ping Include basic monitering of the website to ensure the web server respond with a valid HTTPS response.

☐ Port Availaibility

Monitering Setting

Define the basic parameters that determine how the host and service should b monitered.

☐ Pause Pause monitering for this device.

☐ Start Start monitering for this device.

Under normal circumstances

Moniter the host and services after every ☐ minutes.

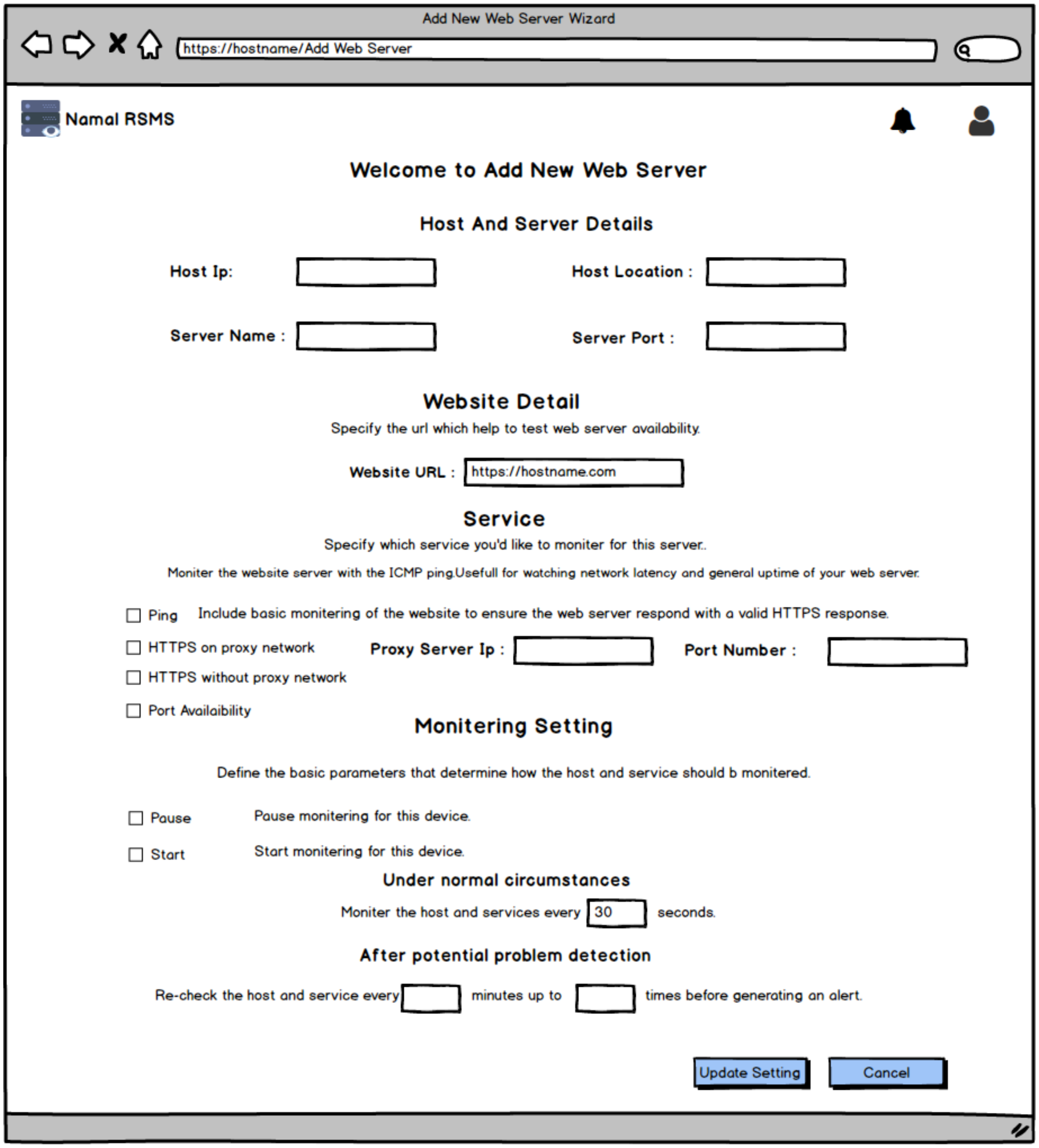
After potential problem detection

Re-check the host and service every ☐ minutes up to ☐ times before generating an alert.

Figure 4.13: Namal remote server monitoring system Add New DNS Wizard

Add New Web Server

Given below screen in “Figure 4.14” is shows after clicked add new web server image from Add new server screen. This wizard has ten input field, three checkbox and 2 radio buttons which assist the user to give complete information of web server that helps to monitor server.


Add New Web Server Wizard

https://hostname/Add Web Server

Namal RSMS

Welcome to Add New Web Server

Host And Server Details

Host Ip:

Host Location :

Server Name :

Server Port :

Website Detail

Specify the url which help to test web server availability.

Website URL : https://hostname.com

Service

Specify which service you'd like to moniter for this server.

Moniter the website server with the ICMP ping.Usefull for watching network latency and general uptime of your web server.

☐ Ping Include basic monitering of the website to ensure the web server respond with a valid HTTPS response.

☐ HTTPS on proxy network Proxy Server Ip : Port Number :

☐ HTTPS without proxy network

☐ Port Availaibility

Monitering Setting

Define the basic parameters that determine how the host and service should b monitered.

☐ Pause Pause monitering for this device.

☐ Start Start monitering for this device.

Under normal circumstances

Moniter the host and services every 30 seconds.

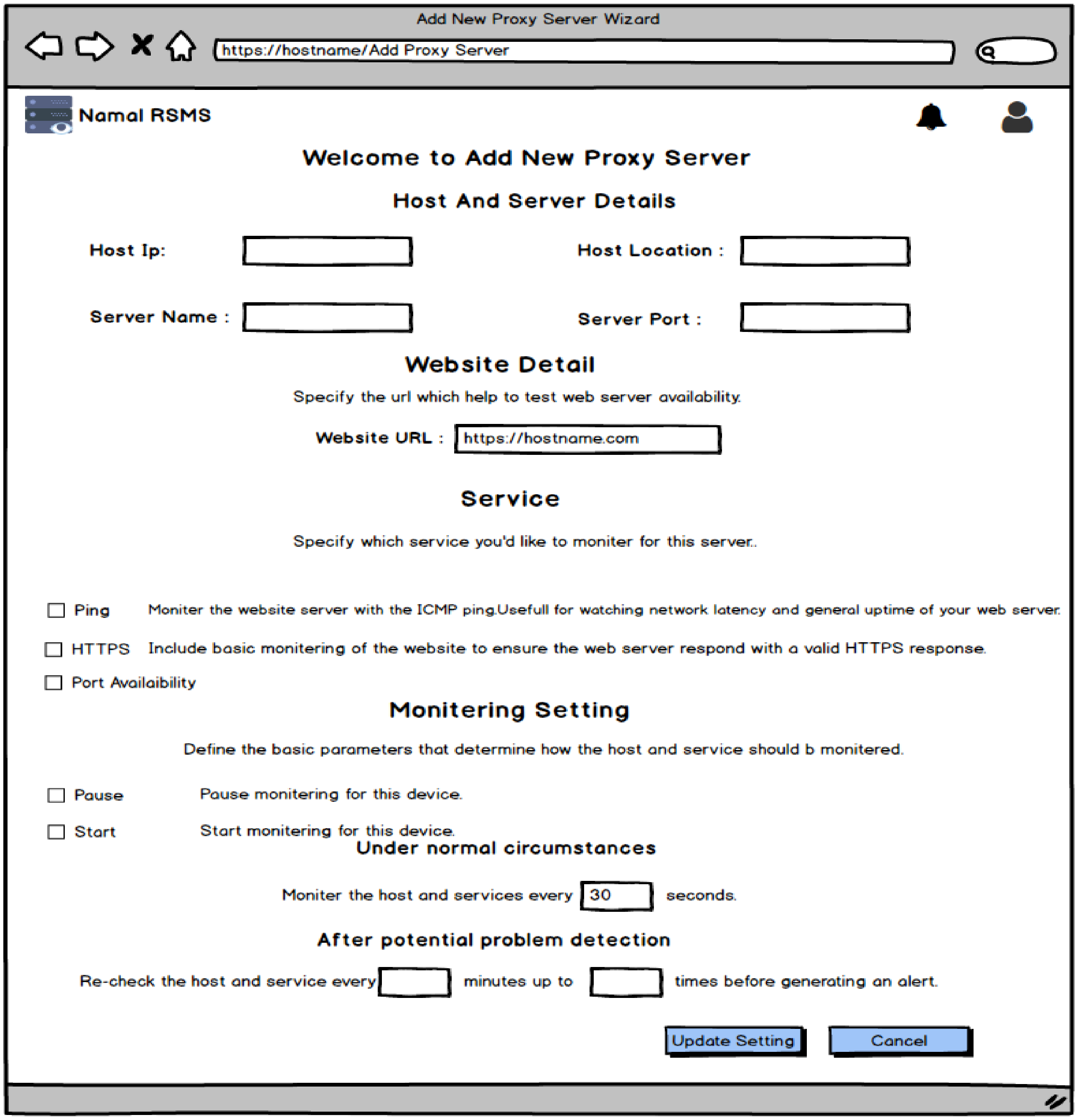
After potential problem detection

Re-check the host and service every minutes up to times before generating an alert.

Figure 4.14: Namal remote server monitoring system Add New Web Server Wizard

Add New Proxy Server

Given below screen in “Figure 4.15” is shows after clicked add new proxy server image from Add new server screen. This wizard has eight input field, three checkbox and 2 radio buttons which assist the user to give complete information of proxy server that is mandatory for monitoring.



Add New Proxy Server Wizard

https://hostname/Add Proxy Server

Namal RSMS

Welcome to Add New Proxy Server

Host And Server Details

Host Ip:

Host Location :

Server Name :

Server Port :

Website Detail

Specify the url which help to test web server availability.

Website URL : https://hostname.com

Service

Specify which service you'd like to moniter for this server..

☐ Ping Moniter the website server with the ICMP ping.Usefull for watching network latency and general uptime of your web server.

☐ HTTPS Include basic monitering of the website to ensure the web server respond with a valid HTTPS response.

☐ Port Availaibility

Monitering Setting

Define the basic parameters that determine how the host and service should b monitered.

☐ Pause Pause monitering for this device.

☐ Start Start monitering for this device.

Under normal circumstances

Moniter the host and services every 30 seconds.

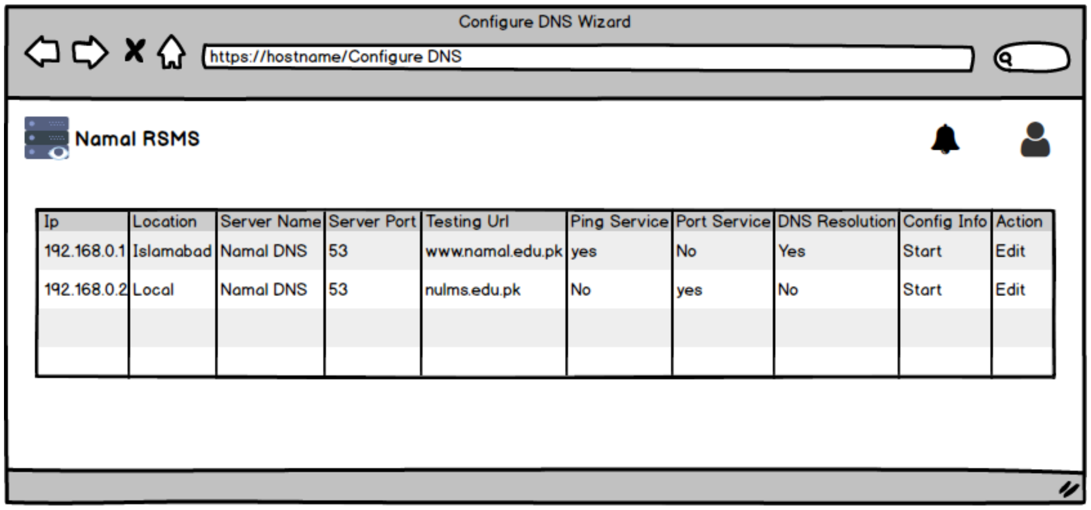
After potential problem detection

Re-check the host and service every ___ minutes up to ___ times before generating an alert.

Figure 4.15: Namal remote server monitoring system Add New Proxy Server Wizard

Configure DNS

Given below screen in “Figure 4.16” load after clicked configure DNS button. It helps the user to view the list of the added domain name server and configure them by just click on the edit button that is displayed on right side of each row. Each row represents one domain name server.

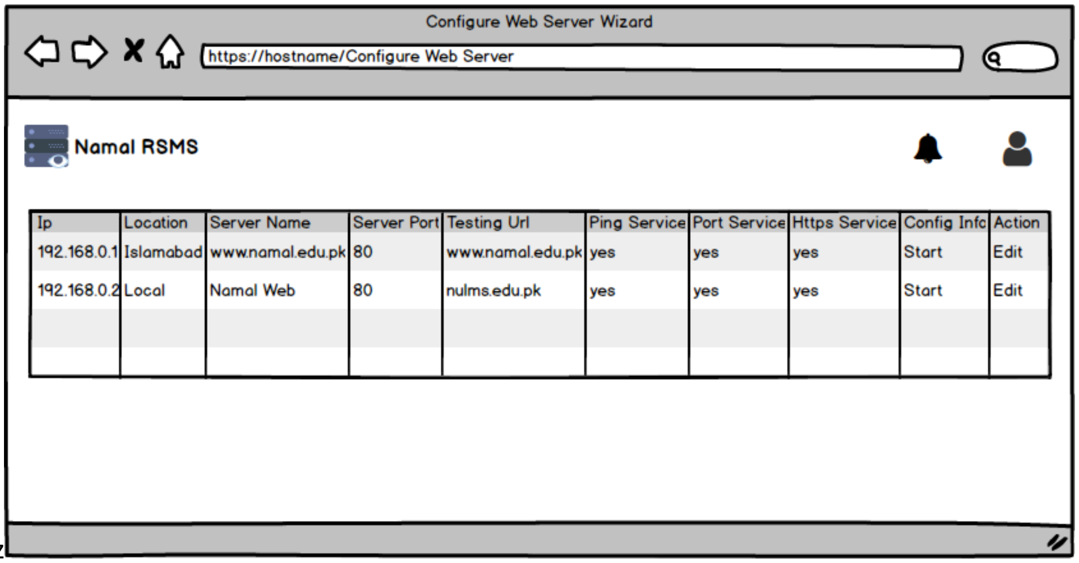


Ip	Location	Server Name	Server Port	Testing Url	Ping Service	Port Service	DNS Resolution	Config Info	Action
192.168.0.1	Islamabad	Namal DNS	53	www.namal.edu.pk	yes	No	Yes	Start	Edit
192.168.0.2	Local	Namal DNS	53	nulms.edu.pk	No	yes	No	Start	Edit

Figure 4.16: Added Domain Name Server List

Configure Web Server

Given below screen in “Figure 4.17” load after clicked configure Web server button. It helps the user to view the list of the added web server and configure them by just click on the edit button that is displayed on right side of each row. Each row represents one web server.


Configure Web Server Wizard
https://hostname/Configure Web Server
Namal RSMS

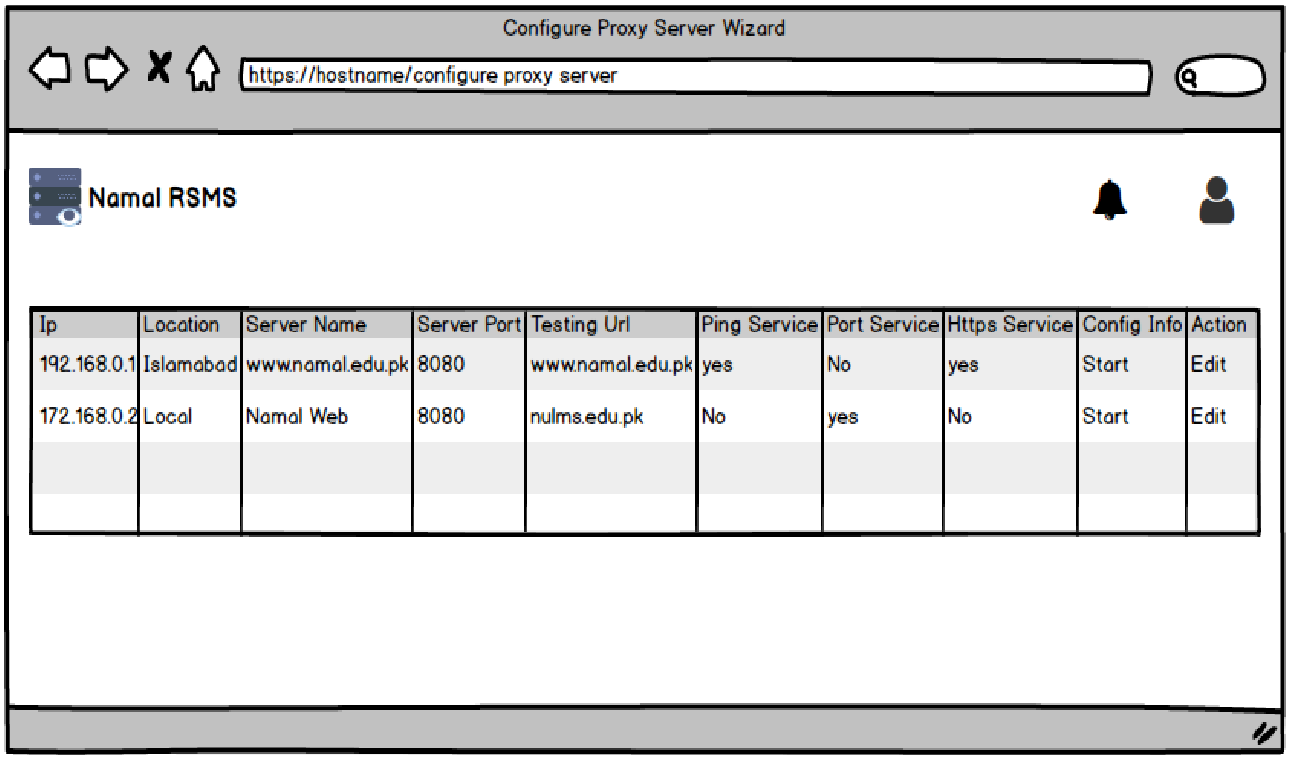
Ip	Location	Server Name	Server Port	Testing Url	Ping Service	Port Service	Https Service	Config Info	Action
192.168.0.1	Islamabad	www.namal.edu.pk	80	www.namal.edu.pk	yes	yes	yes	Start	Edit
192.168.0.2	Local	Namal Web	80	nulms.edu.pk	yes	yes	yes	Start	Edit

z

Figure 4.17: Added Web Servers List

Configure Proxy Server

Given below screen in “Figure 4.18” load after clicked configure proxy server button. It helps the user to view a list of the added proxy server and configure them by just click on the edit button that is displayed on right side of each row. Each row represents one proxy server.

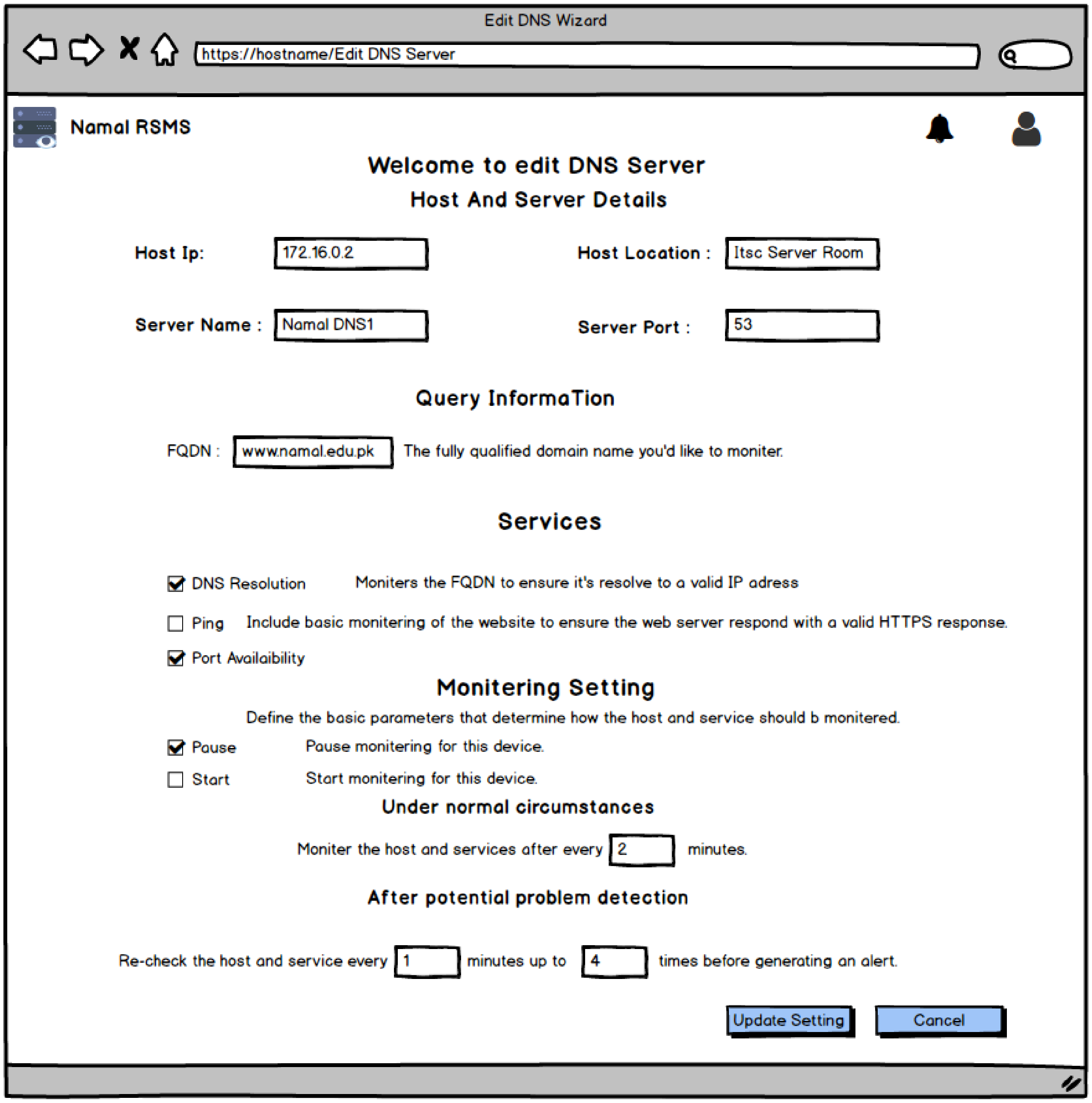

Configure Proxy Server Wizard
https://hostname/configure proxy server
Namal RSMS

Ip	Location	Server Name	Server Port	Testing Url	Ping Service	Port Service	Https Service	Config Info	Action
192.168.0.1	Islamabad	www.namal.edu.pk	8080	www.namal.edu.pk	yes	No	yes	Start	Edit
172.168.0.2	Local	Namal Web	8080	nulms.edu.pk	No	yes	No	Start	Edit

Figure 4.18: Added Proxy Servers List

Edit Domain Name Server

This screen load in front of the user after clicked Edit button of DNS. It assists the user to edit domain server info and monitor server services according to his/her define configuration. Given below “Figure 4.19” is Edit domain name server screen.



Welcome to edit DNS Server

Host And Server Details

Host Ip: 172.16.0.2

Host Location : Itsc Server Room

Server Name : Namal DNS1

Server Port : 53

Query InformaTion

FQDN : www.namal.edu.pk The fully qualified domain name you'd like to moniter.

Services

- [✓] DNS Resolution — Moniters the FQDN to ensure it's resolve to a valid IP adress
- [ ] Ping — Include basic monitering of the website to ensure the web server respond with a valid HTTPS response.
- [✓] Port Availaibility

Monitering Setting

Define the basic parameters that determine how the host and service should b monitered.

- [✓] Pause — Pause monitering for this device.
- [ ] Start — Start monitering for this device.

Under normal circumstances

Moniter the host and services after every 2 minutes.

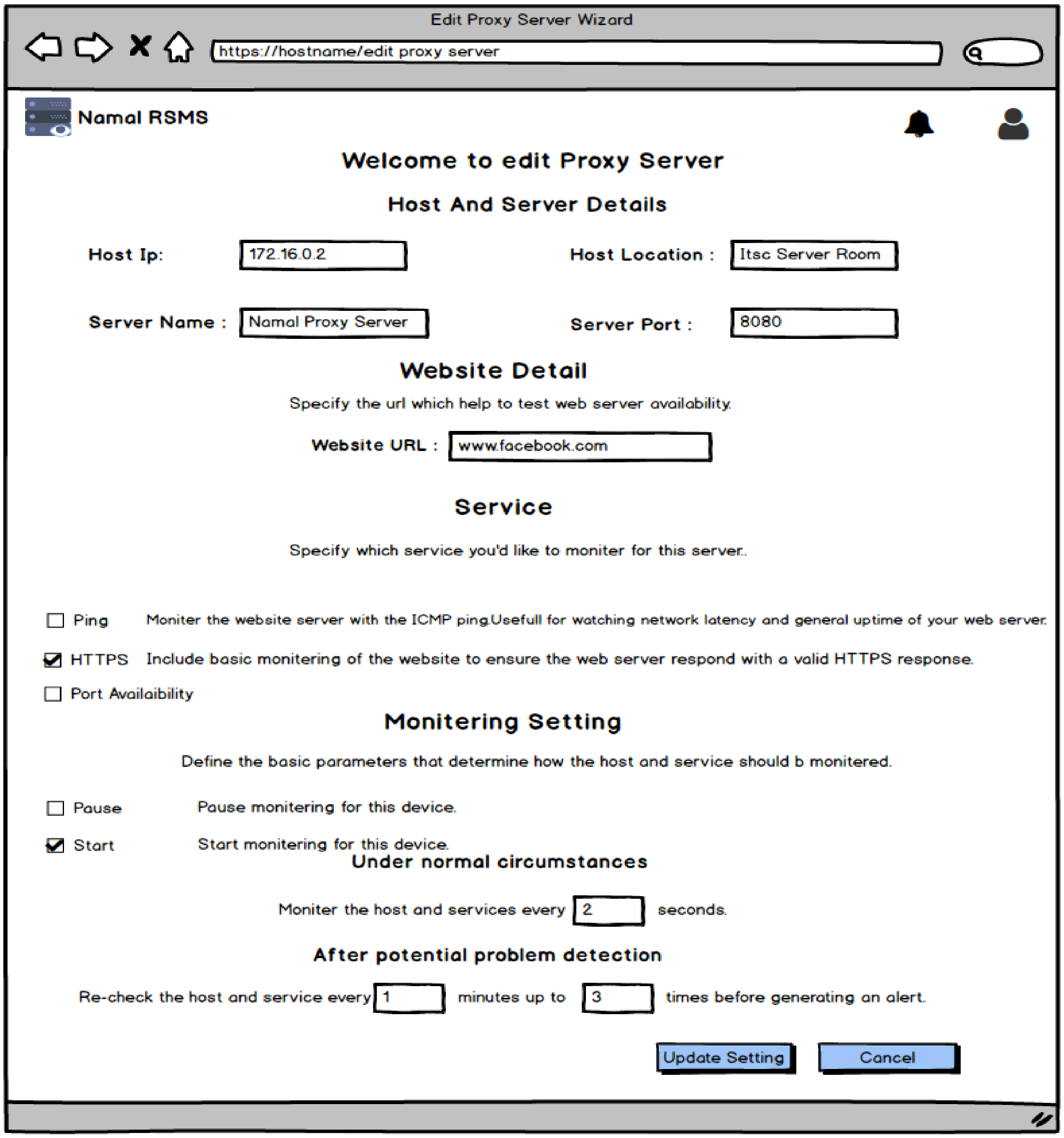
After potential problem detection

Re-check the host and service every 1 minutes up to 4 times before generating an alert.

Figure 4.19: Edit Domain Name Server Wizard

Edit Proxy Server

This screen load in front of the user after clicked Edit button of the proxy server. It assists the user to edit proxy server info and monitor server services according to this configuration. Given below “Figure 4.20” is the edit proxy server screen.


Edit Proxy Server Wizard

https://hostname/edit proxy server

Namal RSMS

Welcome to edit Proxy Server

Host And Server Details

Host Ip: 172.16.0.2

Host Location : Itsc Server Room

Server Name : Namal Proxy Server

Server Port : 8080

Website Detail

Specify the url which help to test web server availability.

Website URL : www.facebook.com

Service

Specify which service you'd like to moniter for this server..

- [ ] Ping Moniter the website server with the ICMP ping.Usefull for watching network latency and general uptime of your web server.
- [x] HTTPS Include basic monitering of the website to ensure the web server respond with a valid HTTPS response.
- [ ] Port Availaibility

Monitering Setting

Define the basic parameters that determine how the host and service should b monitered.

- [ ] Pause Pause monitering for this device.
- [x] Start Start monitering for this device.

Under normal circumstances

Moniter the host and services every 2 seconds.

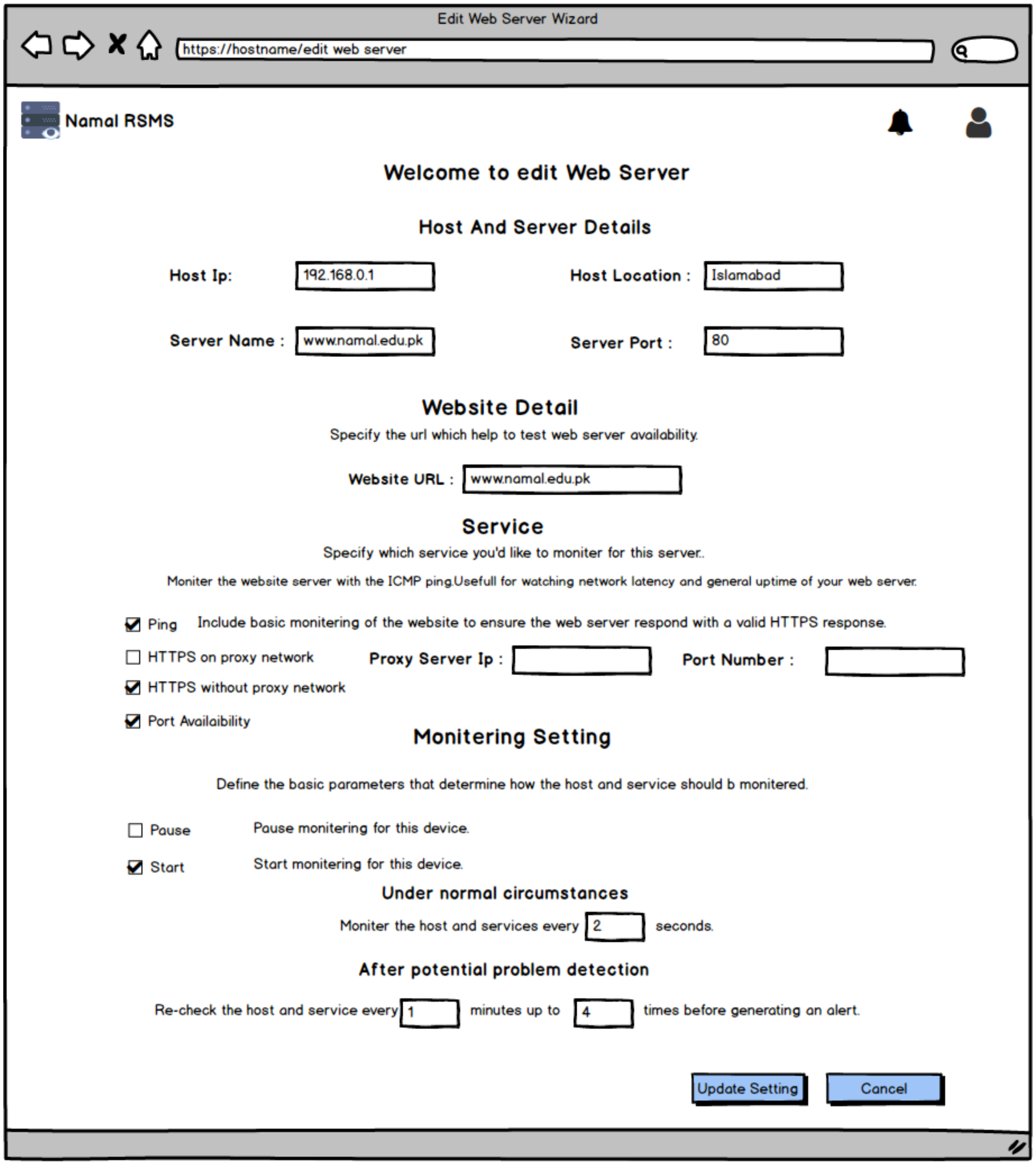
After potential problem detection

Re-check the host and service every 1 minutes up to 3 times before generating an alert.

Figure 4.20: Edit Proxy Server Wizard

Edit Web Server

This screen load in front of the user after clicked Edit button of the web server. It assists the user to edit web server info and monitor server services according to this configuration. Given below “Figure 4.21” is the edit web server screen.



Welcome to edit Web Server

Host And Server Details

Host Ip: 192.168.0.1
Host Location : Islamabad
Server Name : www.namal.edu.pk
Server Port : 80

Website Detail

Specify the url which help to test web server availability.

Website URL : www.namal.edu.pk

Service

Specify which service you'd like to moniter for this server..

Moniter the website server with the ICMP ping.Usefull for watching network latency and general uptime of your web server.

- [✓] Ping — Include basic monitering of the website to ensure the web server respond with a valid HTTPS response.
- [ ] HTTPS on proxy network — Proxy Server Ip : — Port Number :
- [✓] HTTPS without proxy network
- [✓] Port Availaibility

Monitering Setting

Define the basic parameters that determine how the host and service should b monitered.

- [ ] Pause — Pause monitering for this device.
- [✓] Start — Start monitering for this device.

Under normal circumstances

Moniter the host and services every 2 seconds.

After potential problem detection

Re-check the host and service every 1 minutes up to 4 times before generating an alert.

Figure 4.21: Edit Web Server Wizard

4.4 Procedural Design

Procedural design of this system is given below in “Figure 4.22” which represent the architecture as a set of interacting process that passes data from one to another.

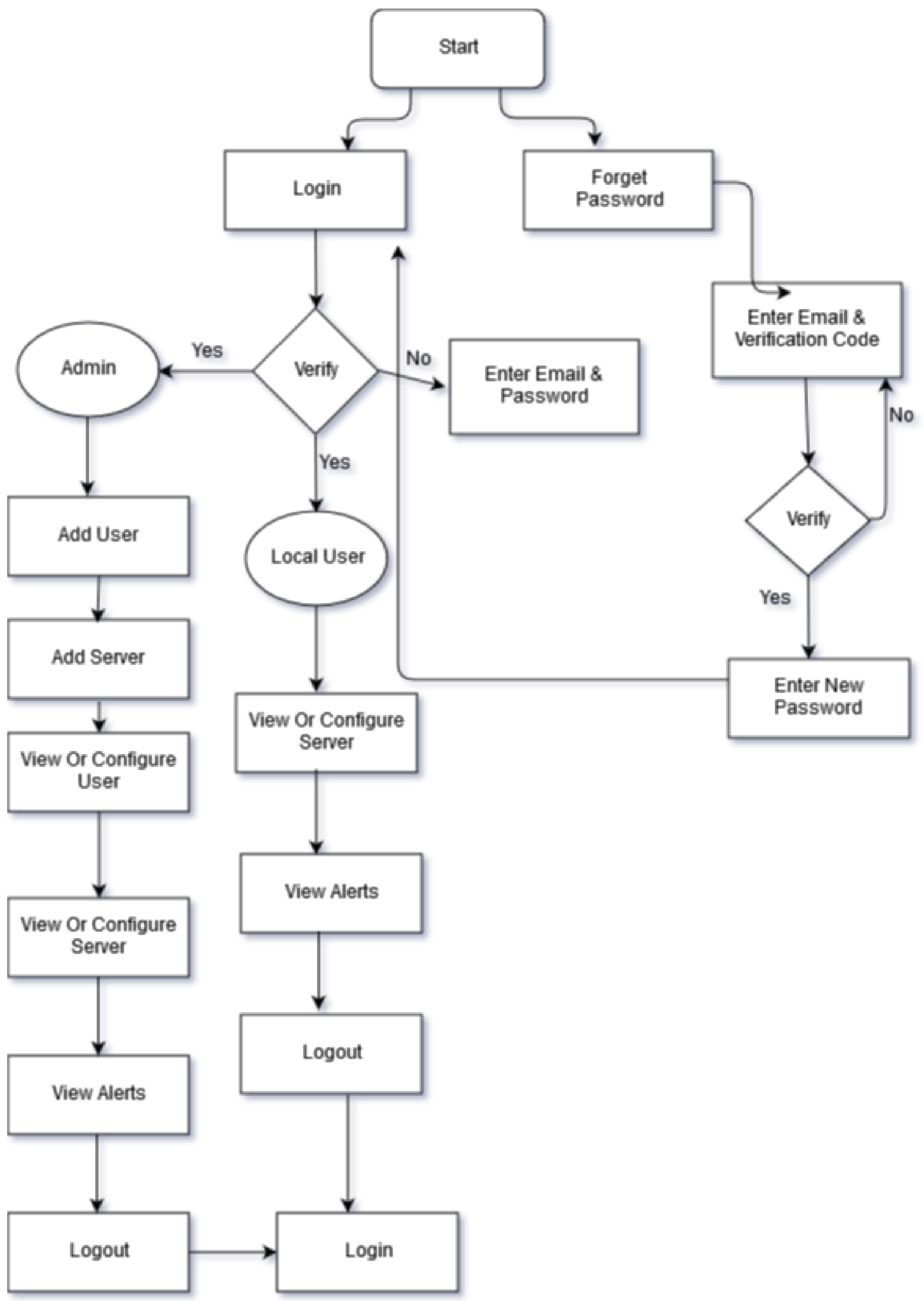


Figure 4.22: Namal remote server monitoring system Procedure Design

Chapter 5

5.System Implementation

5.1 Development Environment

Throughout the development of this system, the Local Development Environment has been created with local development servers of a different kind. WAMP Servers are used to run the website which involves the frontend and its backend and access database. Java application act like a server and client to monitor these servers' services through creating artificial traffic.

5.2 Features Implementation

Login

Given below Login Forum in "Figure 5.1" takes the credentials of the registered users and when submitted, the information is sent to the server for the authenticity of this user. When the server receives information, it compares these credentials with all the credentials that are already saved in the database to finding it. If the user is authenticated, the user is navigated to the particular profile or dashboard. If the user is not authenticated, the user is redirected to this screen.

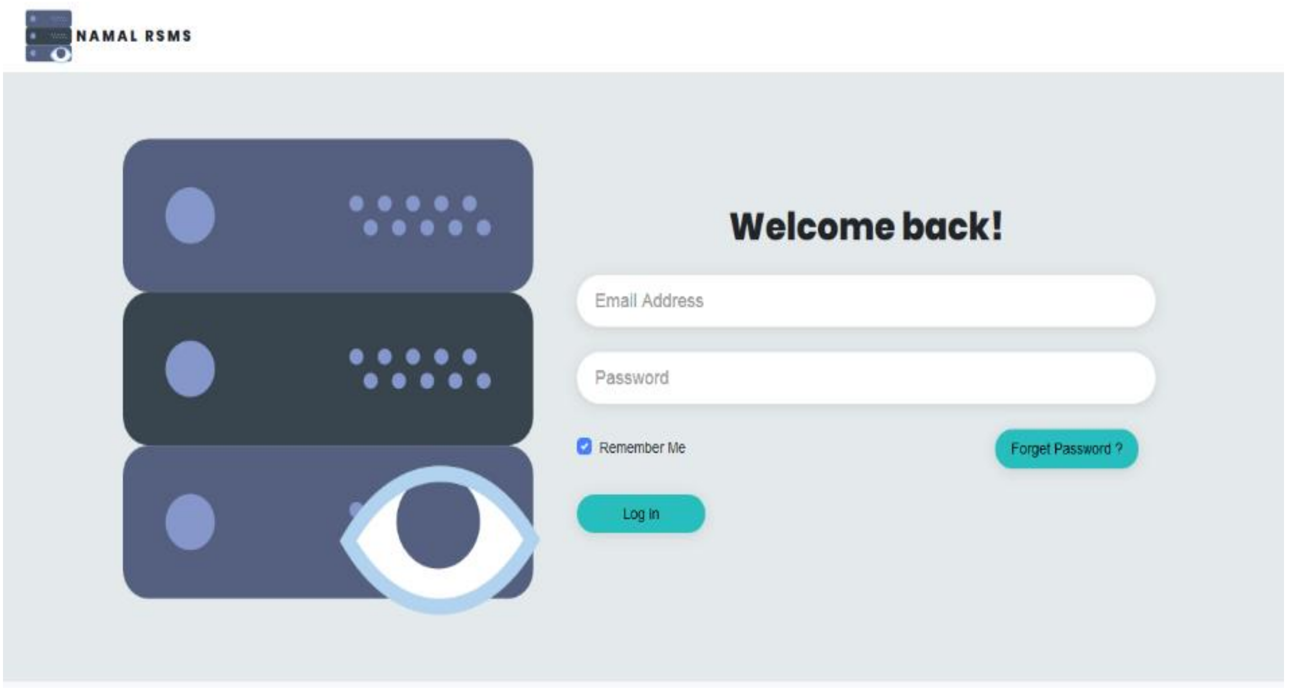
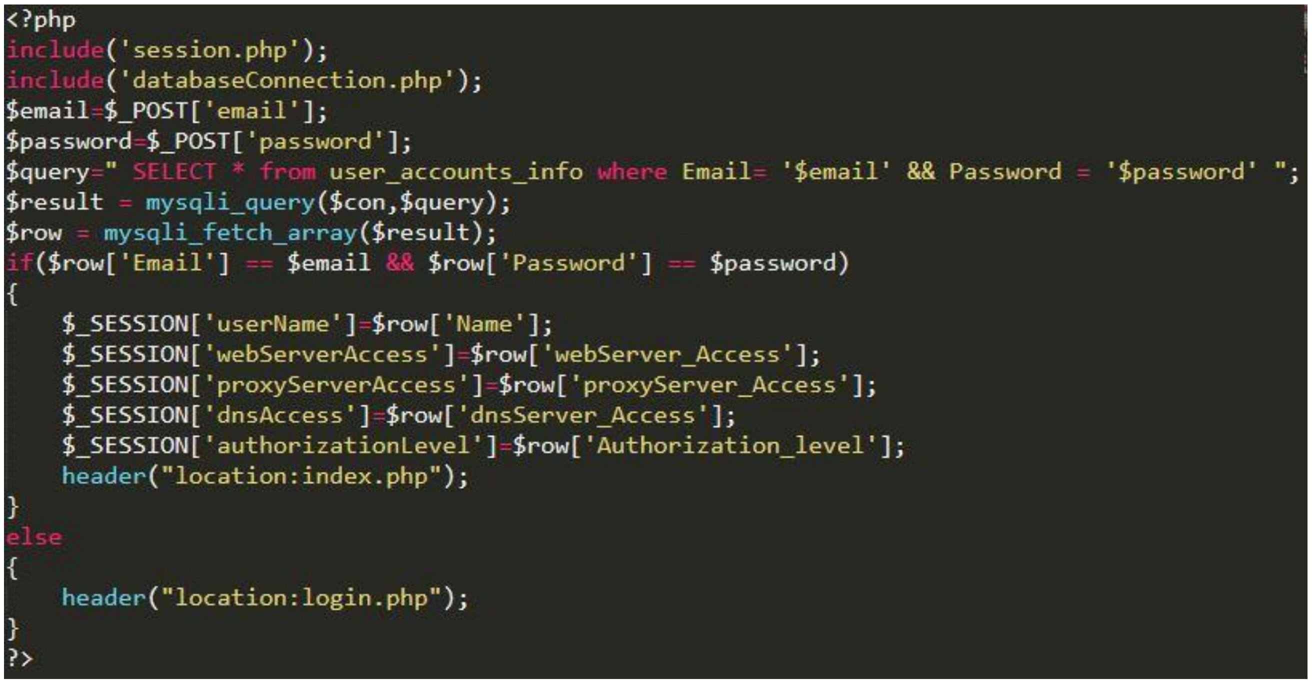


Figure 5.1: Login Forum

Given below “Figure 5.4” is the code to authenticate the user.



```php
<?php
include('session.php');
include('databaseConnection.php');
$email=$_POST['email'];
$password=$_POST['password'];
$query=" SELECT * from user_accounts_info where Email= '$email' && Password = '$password' ";
$result = mysqli_query($con,$query);
$row = mysqli_fetch_array($result);
if($row['Email'] == $email && $row['Password'] == $password)
{
    $_SESSION['userName']=$row['Name'];
    $_SESSION['webServerAccess']=$row['webServer_Access'];
    $_SESSION['proxyServerAccess']=$row['proxyServer_Access'];
    $_SESSION['dnsAccess']=$row['dnsServer_Access'];
    $_SESSION['authorizationLevel']=$row['Authorization_level'];
    header("location:index.php");
}
else
{
    header("location:login.php");
}
?>
```

Figure 5.2: Login Forum code snippet

Update Password

Given below update password forum in “Figure 5.3” takes the required info including verification code, new Password of the registered users and when submitted, the information is sent to the server to update user password. When the server receives information, it updates the password of this user after verifying the verification code that the user has entered. After the update, the user is navigated to the login screen to re-enter email address and password.

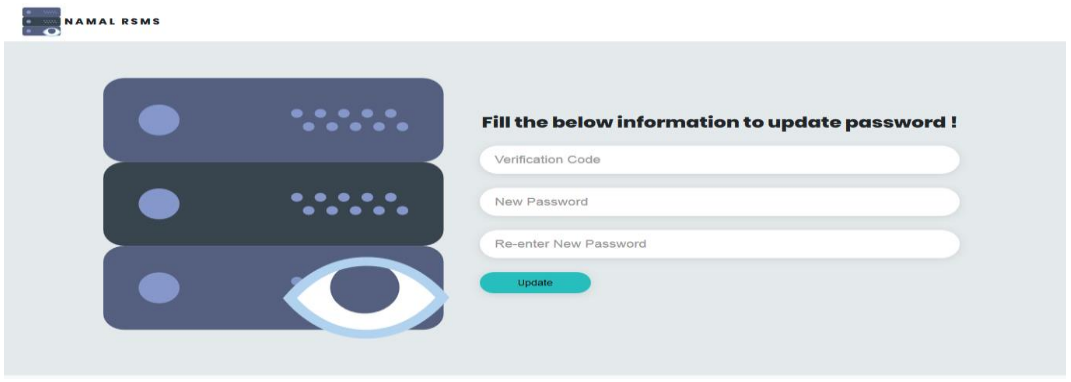
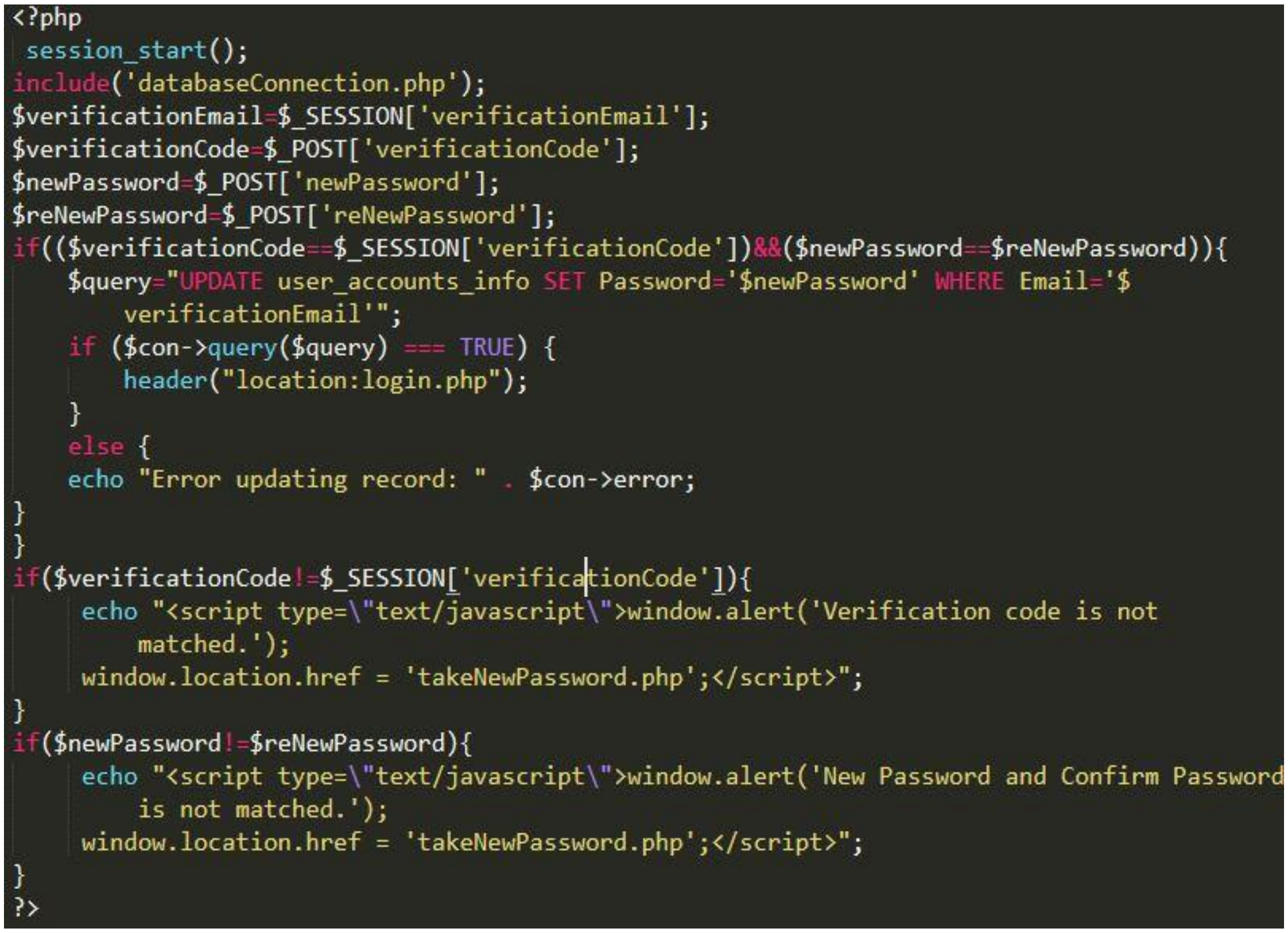


Figure 5.3: Update Password Forum

Given below “Figure 5.4” is the code to verify user given verification code and navigate to the login screen.

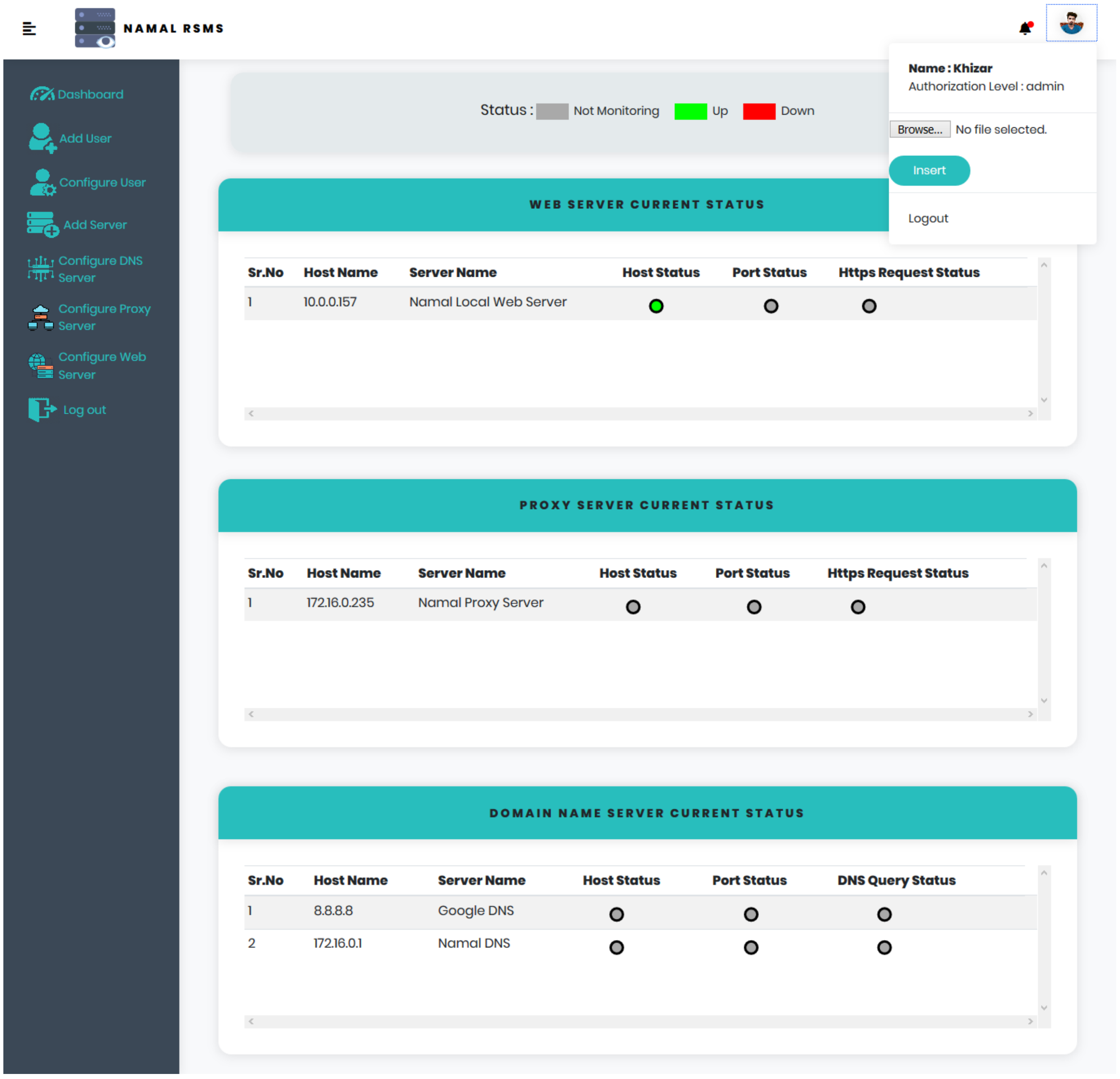


```php
<?php
 session_start();
include('databaseConnection.php');
$verificationEmail=$_SESSION['verificationEmail'];
$verificationCode=$_POST['verificationCode'];
$newPassword=$_POST['newPassword'];
$reNewPassword=$_POST['reNewPassword'];
if(($verificationCode==$_SESSION['verificationCode'])&&($newPassword==$reNewPassword)){
    $query="UPDATE user_accounts_info SET Password='$newPassword' WHERE Email='$
        verificationEmail'";
    if ($con->query($query) === TRUE) {
        header("location:login.php");
    }
    else {
    echo "Error updating record: " . $con->error;
}
}
if($verificationCode!=$_SESSION['verificationCode']){
     echo "<script type=\"text/javascript\">window.alert('Verification code is not
         matched.');
     window.location.href = 'takeNewPassword.php';</script>";
}
if($newPassword!=$reNewPassword){
     echo "<script type=\"text/javascript\">window.alert('New Password and Confirm Password
         is not matched.');
     window.location.href = 'takeNewPassword.php';</script>";
}
?>
```

Figure 5.4: Update Password Forum Code Snippet

Dashboard

When the user logins into the system, the first page, Dashboard, would be shown where user can view the current status of servers and their services. Users can use that feature of this system what they have access. Given below “Figure 5.5” is the admin level user dashboard in which clearly, we can view three type of colors and their unique meanings which helps the user to view the current status of servers.



Name : Khizar
Authorization Level : admin
No file selected.

Status : ▨ Not Monitoring 🟩 Up 🟥 Down

WEB SERVER CURRENT STATUS

Sr.No	Host Name	Server Name	Host Status	Port Status	Https Request Status
1	10.0.0.157	Namal Local Web Server	●	●	●

PROXY SERVER CURRENT STATUS

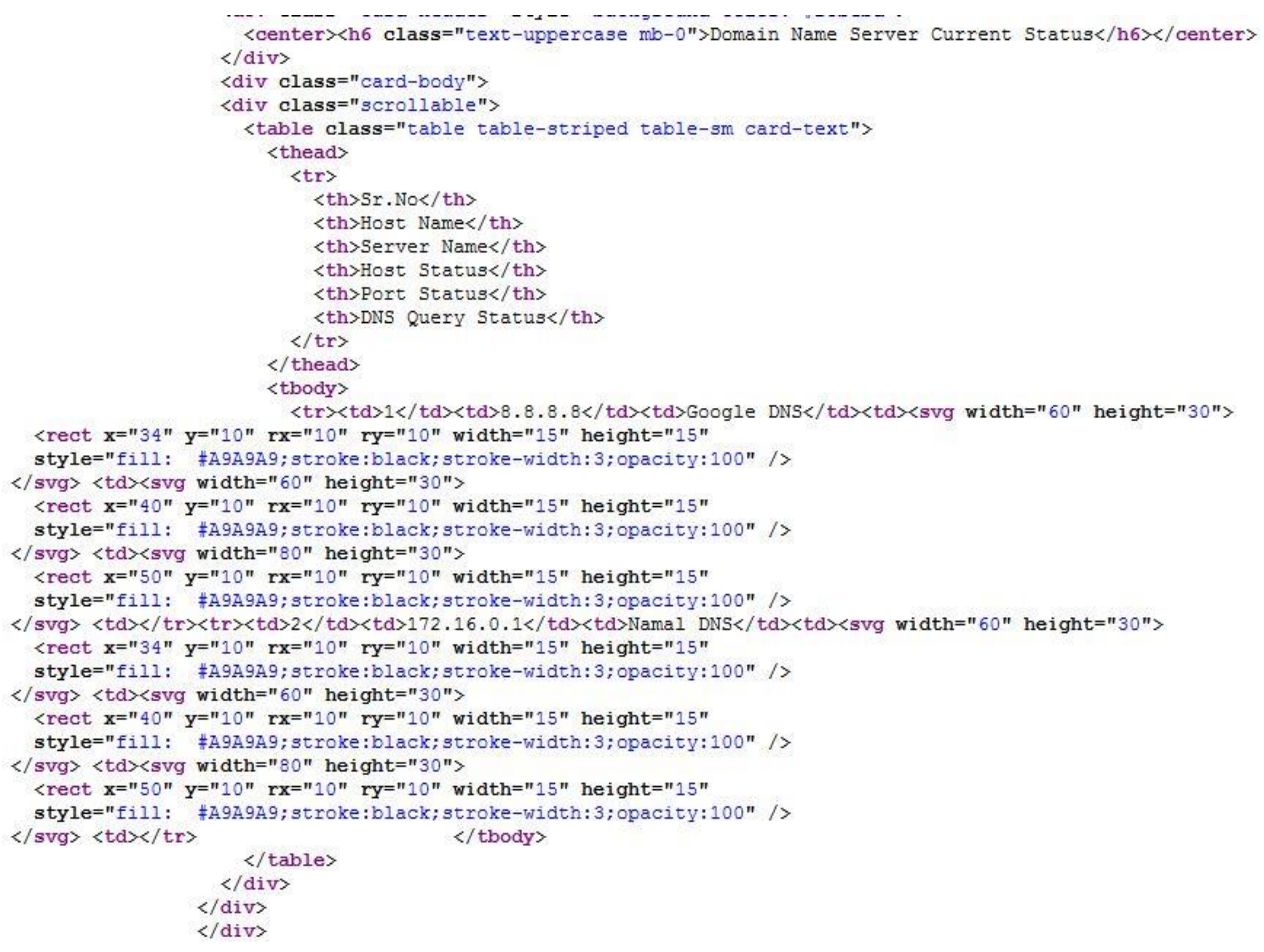
Sr.No	Host Name	Server Name	Host Status	Port Status	Https Request Status
1	172.16.0.235	Namal Proxy Server	●	●	●

DOMAIN NAME SERVER CURRENT STATUS

Sr.No	Host Name	Server Name	Host Status	Port Status	DNS Query Status
1	8.8.8.8	Google DNS	●	●	●
2	172.16.0.1	Namal DNS	●	●	●

Figure 5.5: Dashboard

Given below "Figure 5.6" is the code to show logged in user dashboard.

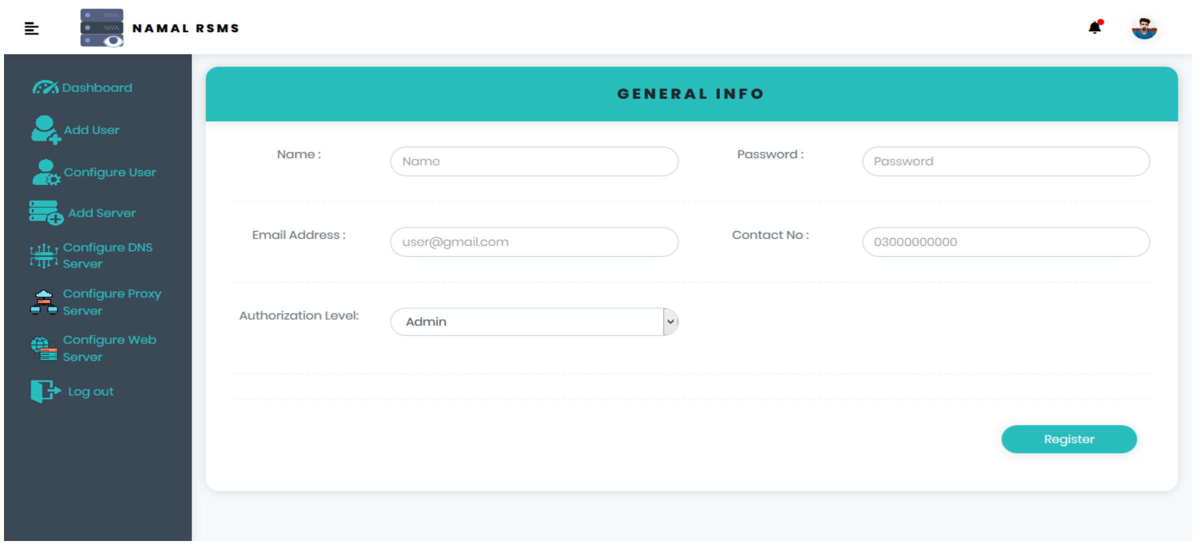


```
                      <center><h6 class="text-uppercase mb-0">Domain Name Server Current Status</h6></center>
                    </div>
                    <div class="card-body">
                    <div class="scrollable">
                      <table class="table table-striped table-sm card-text">
                        <thead>
                          <tr>
                            <th>Sr.No</th>
                            <th>Host Name</th>
                            <th>Server Name</th>
                            <th>Host Status</th>
                            <th>Port Status</th>
                            <th>DNS Query Status</th>
                          </tr>
                        </thead>
                        <tbody>
                          <tr><td>1</td><td>8.8.8.8</td><td>Google DNS</td><td><svg width="60" height="30">
  <rect x="34" y="10" rx="10" ry="10" width="15" height="15"
  style="fill:  #A9A9A9;stroke:black;stroke-width:3;opacity:100" />
</svg> <td><svg width="60" height="30">
  <rect x="40" y="10" rx="10" ry="10" width="15" height="15"
  style="fill:  #A9A9A9;stroke:black;stroke-width:3;opacity:100" />
</svg> <td><svg width="80" height="30">
  <rect x="50" y="10" rx="10" ry="10" width="15" height="15"
  style="fill:  #A9A9A9;stroke:black;stroke-width:3;opacity:100" />
</svg> <td></tr><tr><td>2</td><td>172.16.0.1</td><td>Namal DNS</td><td><svg width="60" height="30">
  <rect x="34" y="10" rx="10" ry="10" width="15" height="15"
  style="fill:  #A9A9A9;stroke:black;stroke-width:3;opacity:100" />
</svg> <td><svg width="60" height="30">
  <rect x="40" y="10" rx="10" ry="10" width="15" height="15"
  style="fill:  #A9A9A9;stroke:black;stroke-width:3;opacity:100" />
</svg> <td><svg width="80" height="30">
  <rect x="50" y="10" rx="10" ry="10" width="15" height="15"
  style="fill:  #A9A9A9;stroke:black;stroke-width:3;opacity:100" />
</svg> <td></tr>                      </tbody>
                      </table>
                    </div>
                  </div>
                  </div>
```

Figure 5.6: Dashboard code snippet

Add New User

Given below screen in "Figure 5.7" appears on user screen after clicked add new user button. This screen one drops down which assist the user to choose authorization level of user. If the user clicked admin, then added user has complete access of this system. If the user clicked local, then added user has access of that server which admin has given it.

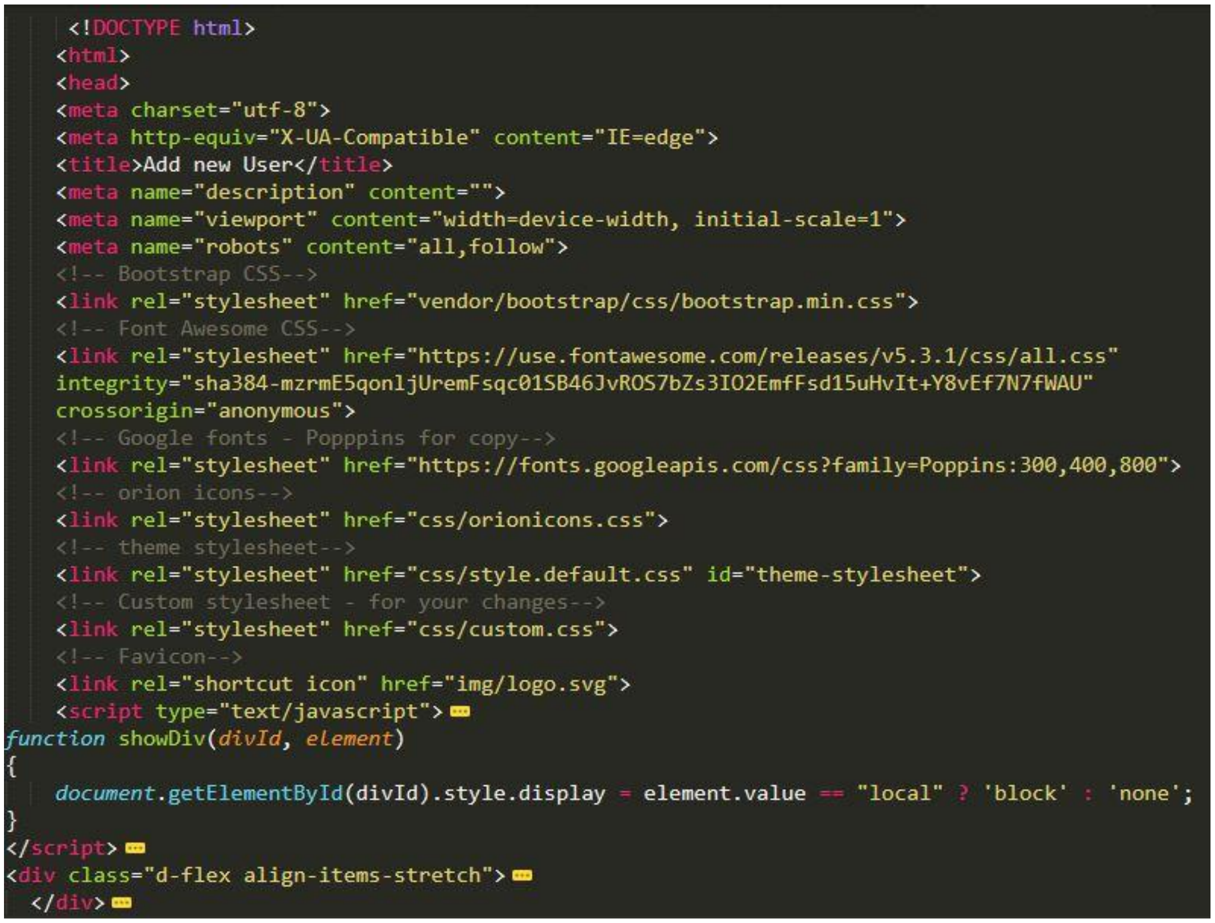


GENERAL INFO

Name :	Name	Password :	Password
Email Address :	user@gmail.com	Contact No :	03000000000
Authorization Level:	Admin		

Figure 5.7: Add new User Forum

Given below “Figure 5.8” is the code to add new user and store added user info into a database.



```
  <!DOCTYPE html>
 <html>
 <head>
 <meta charset="utf-8">
 <meta http-equiv="X-UA-Compatible" content="IE=edge">
 <title>Add new User</title>
 <meta name="description" content="">
 <meta name="viewport" content="width=device-width, initial-scale=1">
 <meta name="robots" content="all,follow">
 <!-- Bootstrap CSS-->
 <link rel="stylesheet" href="vendor/bootstrap/css/bootstrap.min.css">
 <!-- Font Awesome CSS-->
 <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.3.1/css/all.css"
 integrity="sha384-mzrmE5qonljUremFsqc01SB46JvROS7bZs3IO2EmfFsd15uHvIt+Y8vEf7N7fWAU"
 crossorigin="anonymous">
 <!-- Google fonts - Popppins for copy-->
 <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Poppins:300,400,800">
 <!-- orion icons-->
 <link rel="stylesheet" href="css/orionicons.css">
 <!-- theme stylesheet-->
 <link rel="stylesheet" href="css/style.default.css" id="theme-stylesheet">
 <!-- Custom stylesheet - for your changes-->
 <link rel="stylesheet" href="css/custom.css">
 <!-- Favicon-->
 <link rel="shortcut icon" href="img/logo.svg">
 <script type="text/javascript">
function showDiv(divId, element)
{
    document.getElementById(divId).style.display = element.value == "local" ? 'block' : 'none';
}
</script>
<div class="d-flex align-items-stretch">
  </div>
```

Figure 5.8: Add new User Code Snippet

Add New Server

Given below screen in “Figure 5.9” is loaded on user screen after clicked add new server button. It takes server name from user to add a new server to monitor its services through click on given server unique image.

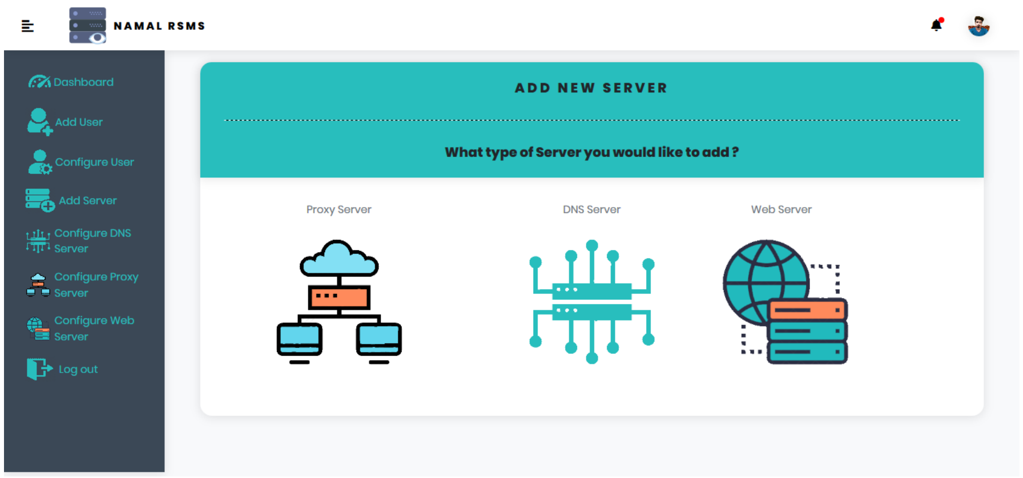
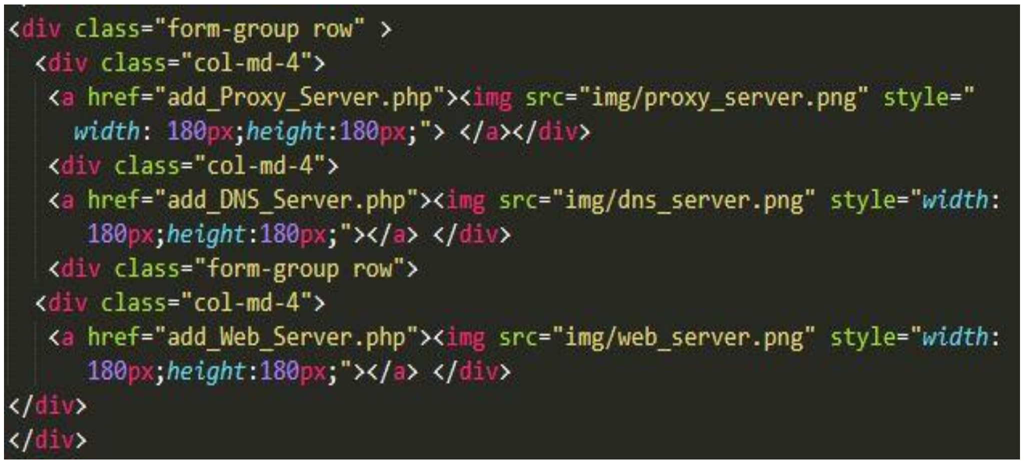


Figure 5.9: Add new Server Forum

Given below “Figure 5.10” is the code to redirect the user to “add server page” against click on the image.

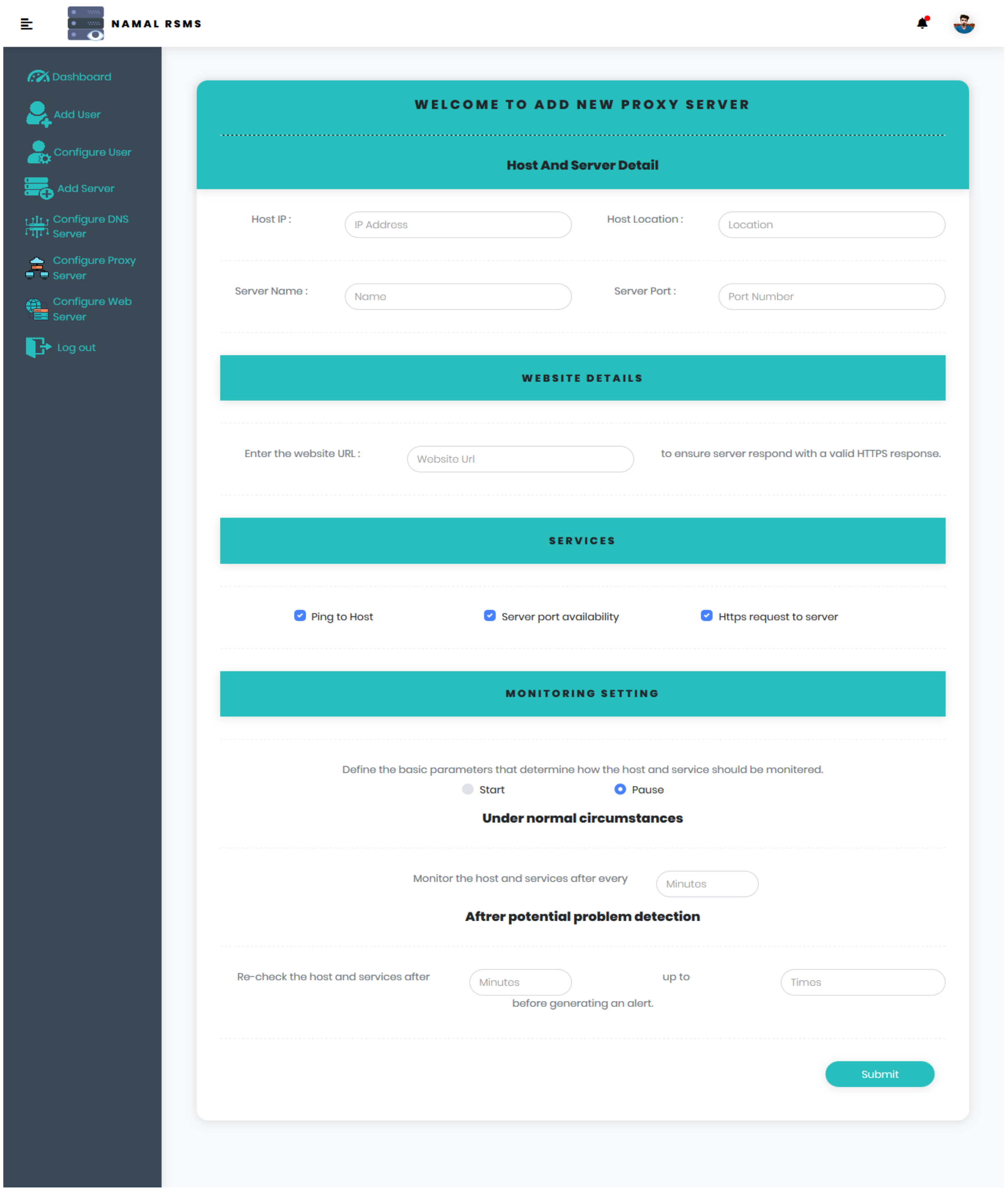


```
<div class="form-group row" >
  <div class="col-md-4">
   <a href="add_Proxy_Server.php"><img src="img/proxy_server.png" style="
     width: 180px;height:180px;"> </a></div>
   <div class="col-md-4">
   <a href="add_DNS_Server.php"><img src="img/dns_server.png" style="width:
      180px;height:180px;"></a> </div>
   <div class="form-group row">
  <div class="col-md-4">
   <a href="add_Web_Server.php"><img src="img/web_server.png" style="width:
      180px;height:180px;"></a> </div>
</div>
</div>
```

Figure 5.10: Add new Server Code Snippet

Add New Proxy Server

The screen in “Figure 5.11” is shown after clicked add new proxy server image. It takes proxy server basic info and configuration from the user which requires to monitor its services.



WELCOME TO ADD NEW PROXY SERVER

Host And Server Detail

Host IP : IP Address
Host Location : Location
Server Name : Name
Server Port : Port Number

WEBSITE DETAILS

Enter the website URL : Website Url to ensure server respond with a valid HTTPS response.

SERVICES

[✓] Ping to Host [✓] Server port availability [✓] Https request to server

MONITORING SETTING

Define the basic parameters that determine how the host and service should be monitored.

Start [● Pause]

Under normal circumstances

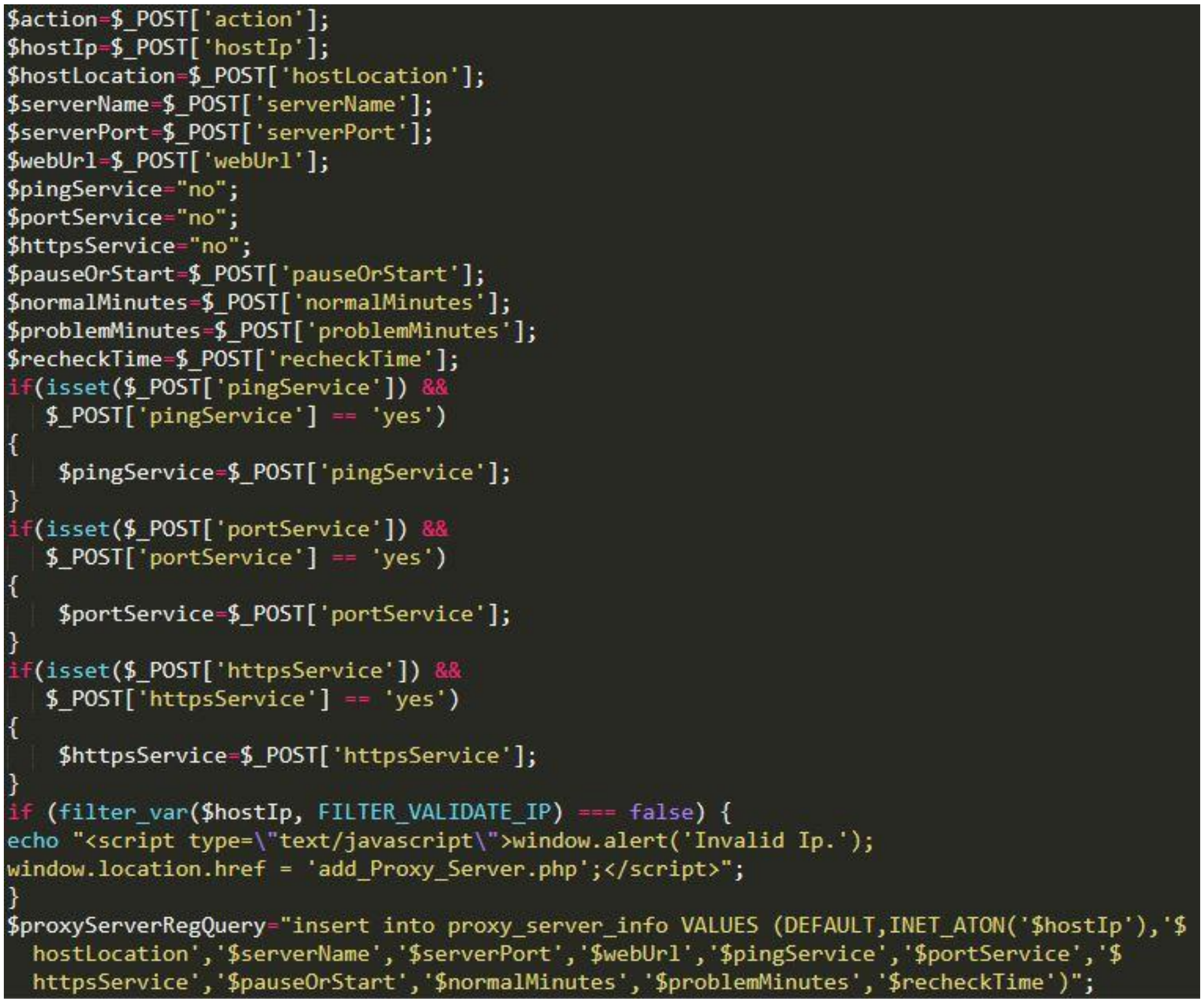
Monitor the host and services after every Minutes

Aftrer potential problem detection

Re-check the host and services after Minutes up to Times before generating an alert.

Figure 5.11: Add new Proxy Server Forum

Given below “Figure 5.12” is the code which store this new proxy server info into a database and test its services through communicating java application.

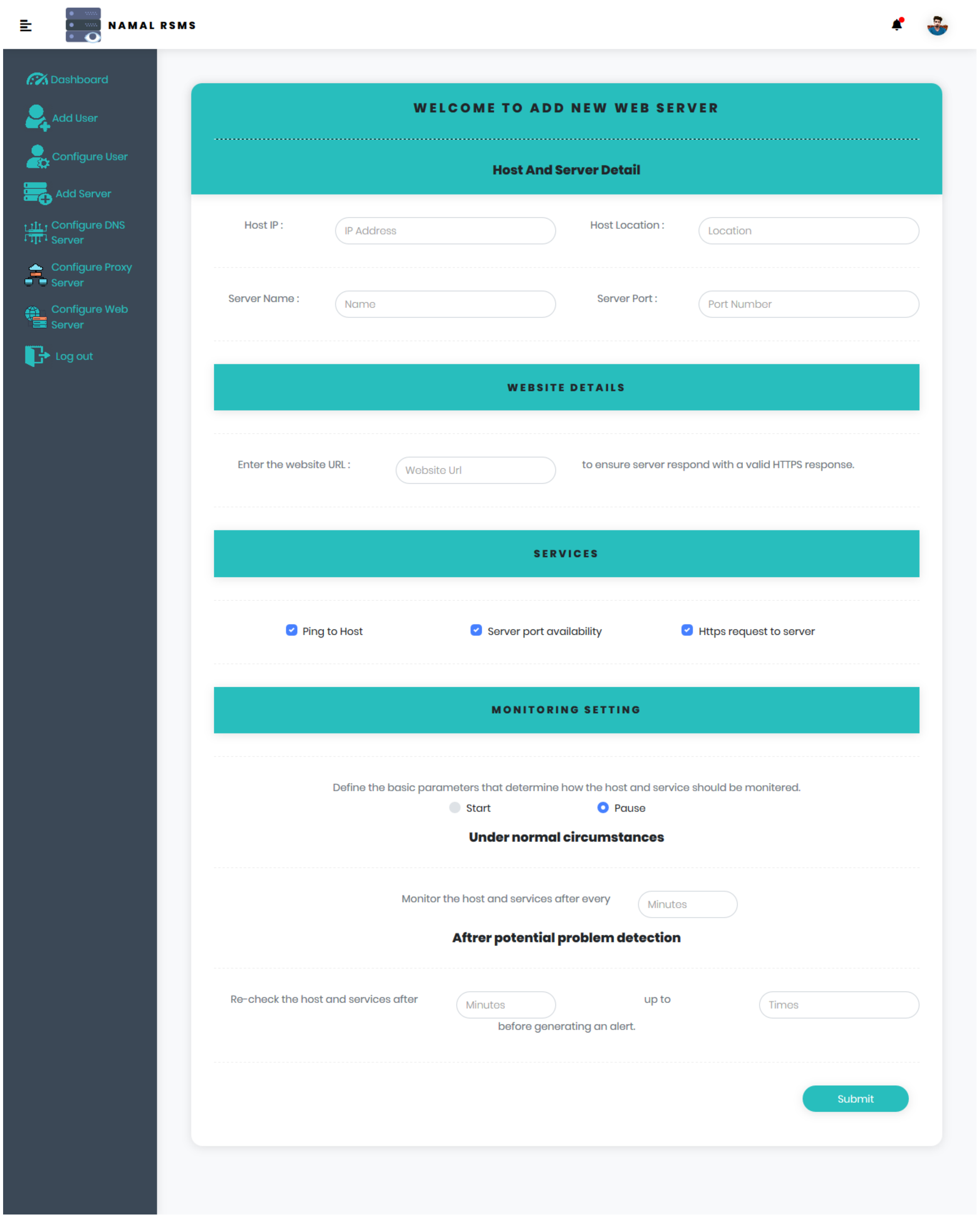


```php
$action=$_POST['action'];
$hostIp=$_POST['hostIp'];
$hostLocation=$_POST['hostLocation'];
$serverName=$_POST['serverName'];
$serverPort=$_POST['serverPort'];
$webUrl=$_POST['webUrl'];
$pingService="no";
$portService="no";
$httpsService="no";
$pauseOrStart=$_POST['pauseOrStart'];
$normalMinutes=$_POST['normalMinutes'];
$problemMinutes=$_POST['problemMinutes'];
$recheckTime=$_POST['recheckTime'];
if(isset($_POST['pingService']) &&
   $_POST['pingService'] == 'yes')
{
    $pingService=$_POST['pingService'];
}
if(isset($_POST['portService']) &&
   $_POST['portService'] == 'yes')
{
    $portService=$_POST['portService'];
}
if(isset($_POST['httpsService']) &&
   $_POST['httpsService'] == 'yes')
{
    $httpsService=$_POST['httpsService'];
}
if (filter_var($hostIp, FILTER_VALIDATE_IP) === false) {
echo "<script type=\"text/javascript\">window.alert('Invalid Ip.');
window.location.href = 'add_Proxy_Server.php';</script>";
}
$proxyServerRegQuery="insert into proxy_server_info VALUES (DEFAULT,INET_ATON('$hostIp'),'$
  hostLocation','$serverName','$serverPort','$webUrl','$pingService','$portService','$
  httpsService','$pauseOrStart','$normalMinutes','$problemMinutes','$recheckTime')";
```

Figure 5.12: Add new proxy server code snippet

Add New Web Server

Given below screen in “Figure 5.13” is shows after clicked add new web server image. It takes new web server basic info and configuration from the user which is mandatory to monitor its services.


NAMAL RSMS

WELCOME TO ADD NEW WEB SERVER

Host And Server Detail

Host IP : IP Address
Host Location : Location
Server Name : Name
Server Port : Port Number

WEBSITE DETAILS

Enter the website URL : Website Url to ensure server respond with a valid HTTPS response.

SERVICES

[✓] Ping to Host [✓] Server port availability [✓] Https request to server

MONITORING SETTING

Define the basic parameters that determine how the host and service should be monitored.

Start [✓ Pause]

Under normal circumstances

Monitor the host and services after every Minutes

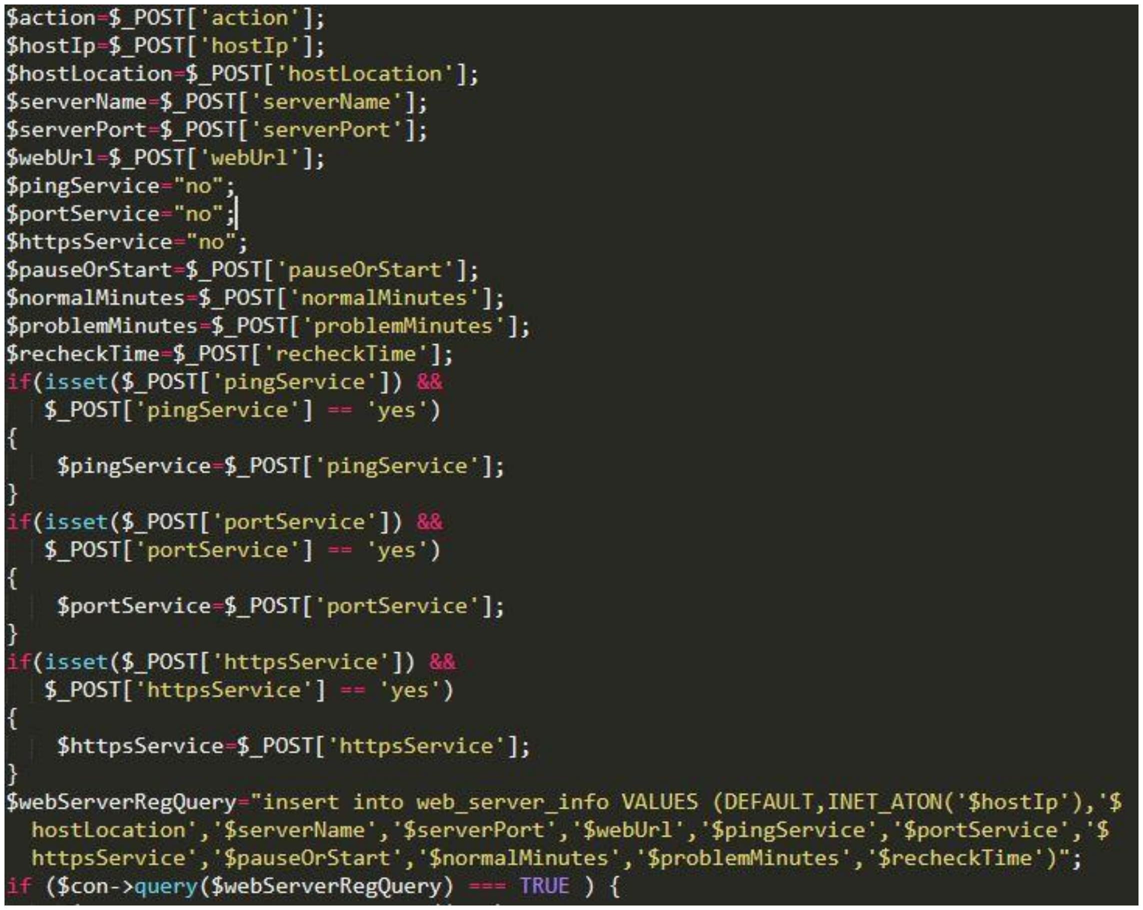
Aftrer potential problem detection

Re-check the host and services after Minutes up to Times before generating an alert.

Submit

Figure 5.13: Add new web server Forum

Given below “Figure 5.14” is the code which store this new web server info into database and test its services through communicating java application.

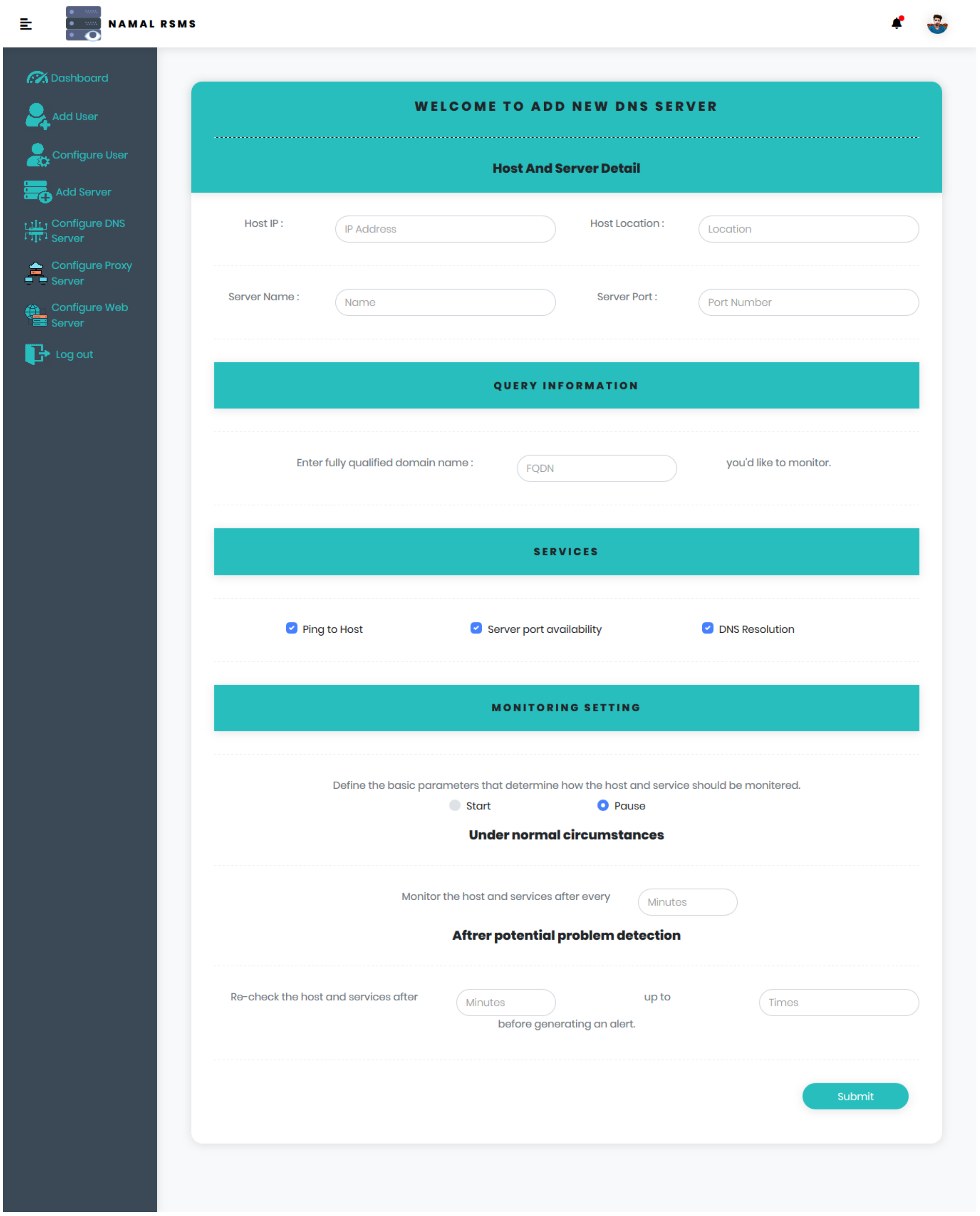


```
$action=$_POST['action'];
$hostIp=$_POST['hostIp'];
$hostLocation=$_POST['hostLocation'];
$serverName=$_POST['serverName'];
$serverPort=$_POST['serverPort'];
$webUrl=$_POST['webUrl'];
$pingService="no";
$portService="no";
$httpsService="no";
$pauseOrStart=$_POST['pauseOrStart'];
$normalMinutes=$_POST['normalMinutes'];
$problemMinutes=$_POST['problemMinutes'];
$recheckTime=$_POST['recheckTime'];
if(isset($_POST['pingService']) &&
   $_POST['pingService'] == 'yes')
{
    $pingService=$_POST['pingService'];
}
if(isset($_POST['portService']) &&
   $_POST['portService'] == 'yes')
{
    $portService=$_POST['portService'];
}
if(isset($_POST['httpsService']) &&
   $_POST['httpsService'] == 'yes')
{
    $httpsService=$_POST['httpsService'];
}
$webServerRegQuery="insert into web_server_info VALUES (DEFAULT,INET_ATON('$hostIp'),'$
  hostLocation','$serverName','$serverPort','$webUrl','$pingService','$portService','$
  httpsService','$pauseOrStart','$normalMinutes','$problemMinutes','$recheckTime')";
if ($con->query($webServerRegQuery) === TRUE ) {
```

Figure 5.14: Add new web server code snippet

Add New Domain name Server

The screen in “Figure 5.15” is shown after clicked add new DNS server image from. It takes new domain name server basic info and configuration from the user which is mandatory to monitor its services.



WELCOME TO ADD NEW DNS SERVER

Host And Server Detail

Host IP :	IP Address	Host Location :	Location
Server Name :	Name	Server Port :	Port Number

QUERY INFORMATION

Enter fully qualified domain name : FQDN you'd like to monitor.

SERVICES

[✓] Ping to Host [✓] Server port availability [✓] DNS Resolution

MONITORING SETTING

Define the basic parameters that determine how the host and service should be monitered.

Start [✓ Pause]

Under normal circumstances

Monitor the host and services after every Minutes

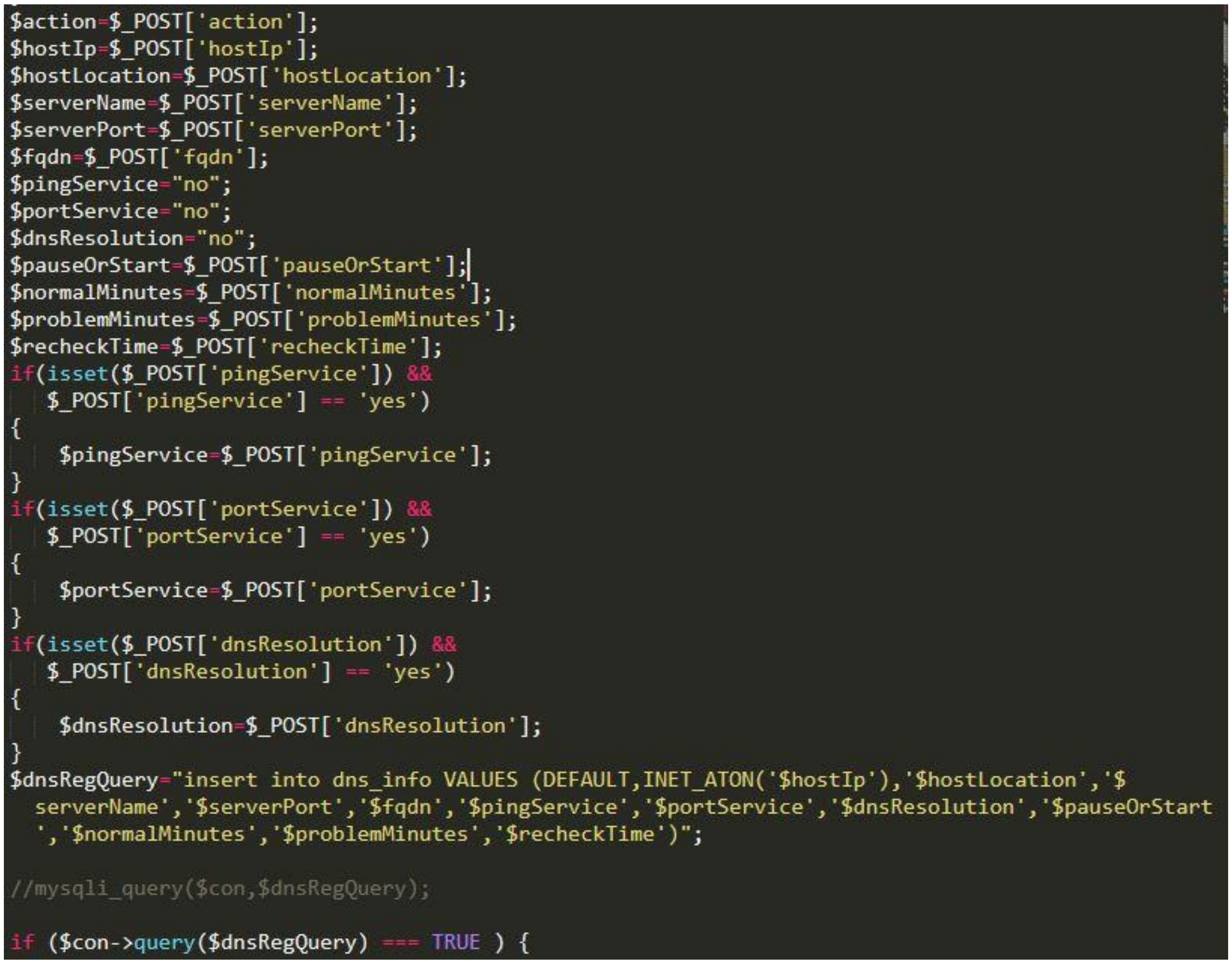
Aftrer potential problem detection

Re-check the host and services after Minutes up to Times before generating an alert.

Submit

Figure 5.15: Add new DNS Forum

Given below “Figure 5.16” is the code which store this DNS info into a database and test its services through communicating java application.

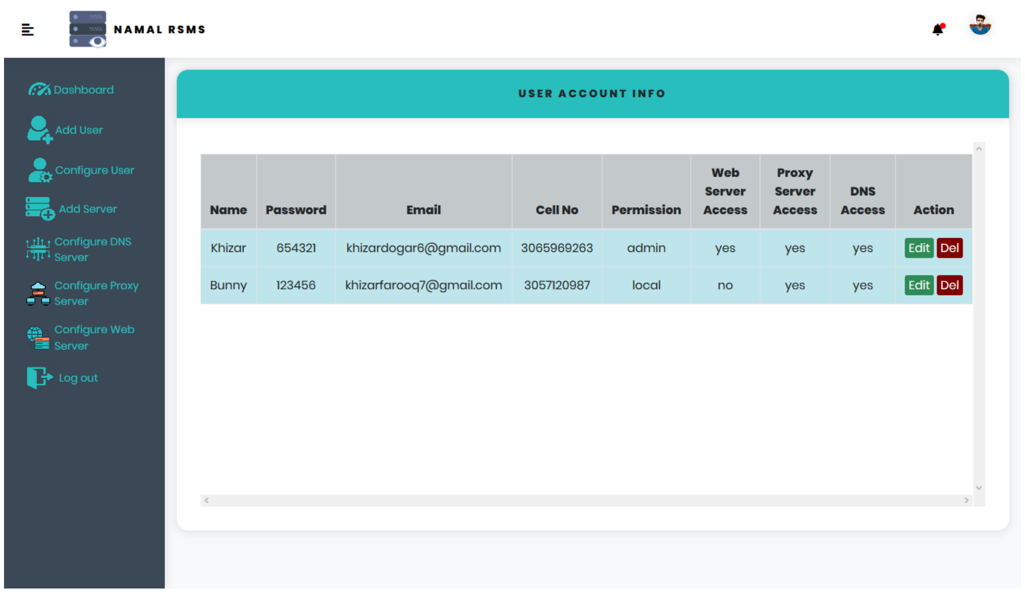


```php
$action=$_POST['action'];
$hostIp=$_POST['hostIp'];
$hostLocation=$_POST['hostLocation'];
$serverName=$_POST['serverName'];
$serverPort=$_POST['serverPort'];
$fqdn=$_POST['fqdn'];
$pingService="no";
$portService="no";
$dnsResolution="no";
$pauseOrStart=$_POST['pauseOrStart'];
$normalMinutes=$_POST['normalMinutes'];
$problemMinutes=$_POST['problemMinutes'];
$recheckTime=$_POST['recheckTime'];
if(isset($_POST['pingService']) &&
   $_POST['pingService'] == 'yes')
{
    $pingService=$_POST['pingService'];
}
if(isset($_POST['portService']) &&
   $_POST['portService'] == 'yes')
{
    $portService=$_POST['portService'];
}
if(isset($_POST['dnsResolution']) &&
   $_POST['dnsResolution'] == 'yes')
{
    $dnsResolution=$_POST['dnsResolution'];
}
$dnsRegQuery="insert into dns_info VALUES (DEFAULT,INET_ATON('$hostIp'),'$hostLocation','$
  serverName','$serverPort','$fqdn','$pingService','$portService','$dnsResolution','$pauseOrStart
  ','$normalMinutes','$problemMinutes','$recheckTime')";

//mysqli_query($con,$dnsRegQuery);

if ($con->query($dnsRegQuery) === TRUE ) {
```

Figure 5.16: Add new DNS server code snippet

Configure User

Given below screen in “Figure 5.17” is displays after pressed configure user button. This wizard shows the current list of users that is store in the database. Admin user can also edit info of any user by just click the edit button.

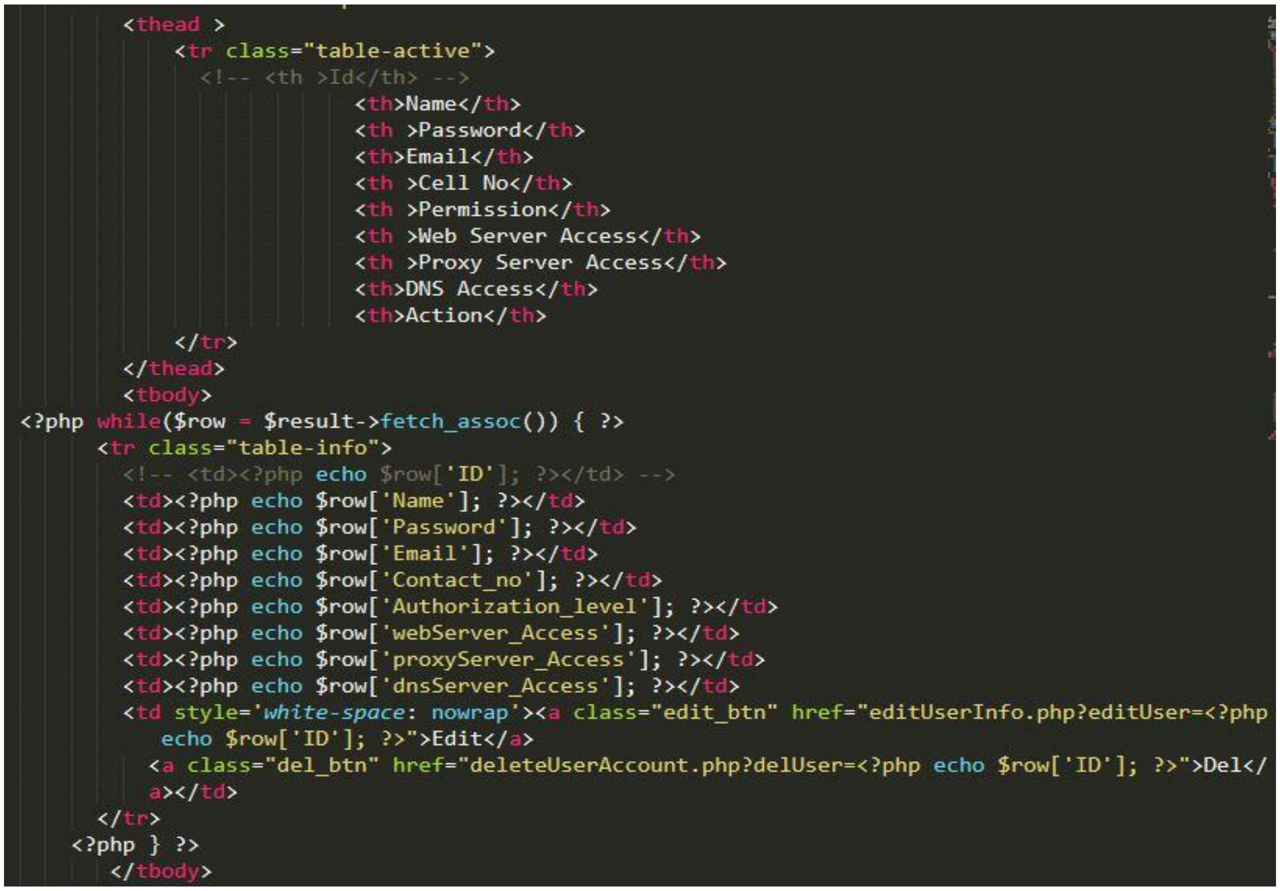


USER ACCOUNT INFO

Name	Password	Email	Cell No	Permission	Web Server Access	Proxy Server Access	DNS Access	Action
Khizar	654321	khizardogar6@gmail.com	3065969263	admin	yes	yes	yes	Edit Del
Bunny	123456	khizarfarooq7@gmail.com	3057120987	local	no	yes	yes	Edit Del

Figure 5.17: Configure user Forum

Given below “Figure 5.18” is the code which displays all users and their info in front of admin.

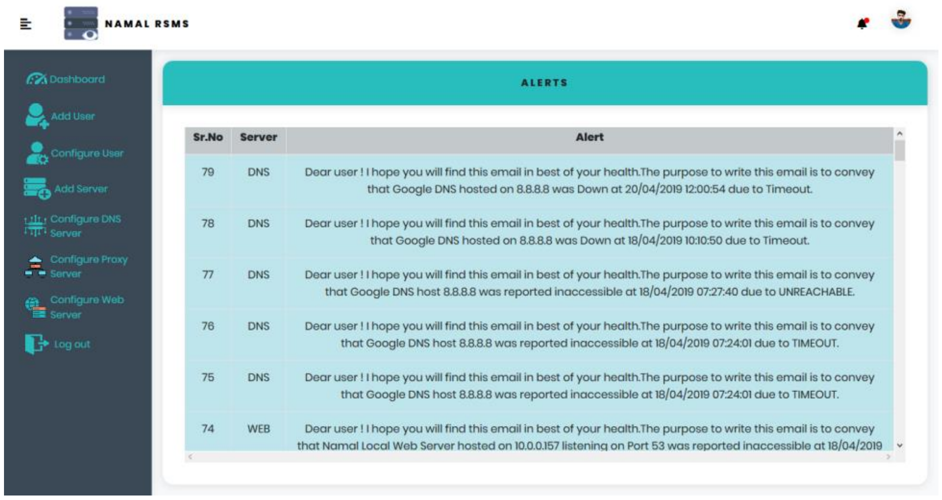


```
        <thead >
            <tr class="table-active">
              <!-- <th >Id</th> -->
                           <th>Name</th>
                           <th >Password</th>
                           <th>Email</th>
                           <th >Cell No</th>
                           <th >Permission</th>
                           <th >Web Server Access</th>
                           <th >Proxy Server Access</th>
                           <th>DNS Access</th>
                           <th>Action</th>
            </tr>
        </thead>
        <tbody>
<?php while($row = $result->fetch_assoc()) { ?>
      <tr class="table-info">
        <!-- <td><?php echo $row['ID']; ?></td> -->
        <td><?php echo $row['Name']; ?></td>
        <td><?php echo $row['Password']; ?></td>
        <td><?php echo $row['Email']; ?></td>
        <td><?php echo $row['Contact_no']; ?></td>
        <td><?php echo $row['Authorization_level']; ?></td>
        <td><?php echo $row['webServer_Access']; ?></td>
        <td><?php echo $row['proxyServer_Access']; ?></td>
        <td><?php echo $row['dnsServer_Access']; ?></td>
        <td style='white-space: nowrap'><a class="edit_btn" href="editUserInfo.php?editUser=<?php
           echo $row['ID']; ?>">Edit</a>
          <a class="del_btn" href="deleteUserAccount.php?delUser=<?php echo $row['ID']; ?>">Del</
          a></td>
      </tr>
    <?php } ?>
       </tbody>
```

Figure 5.18: Configure User Form Code snippet

View Alerts

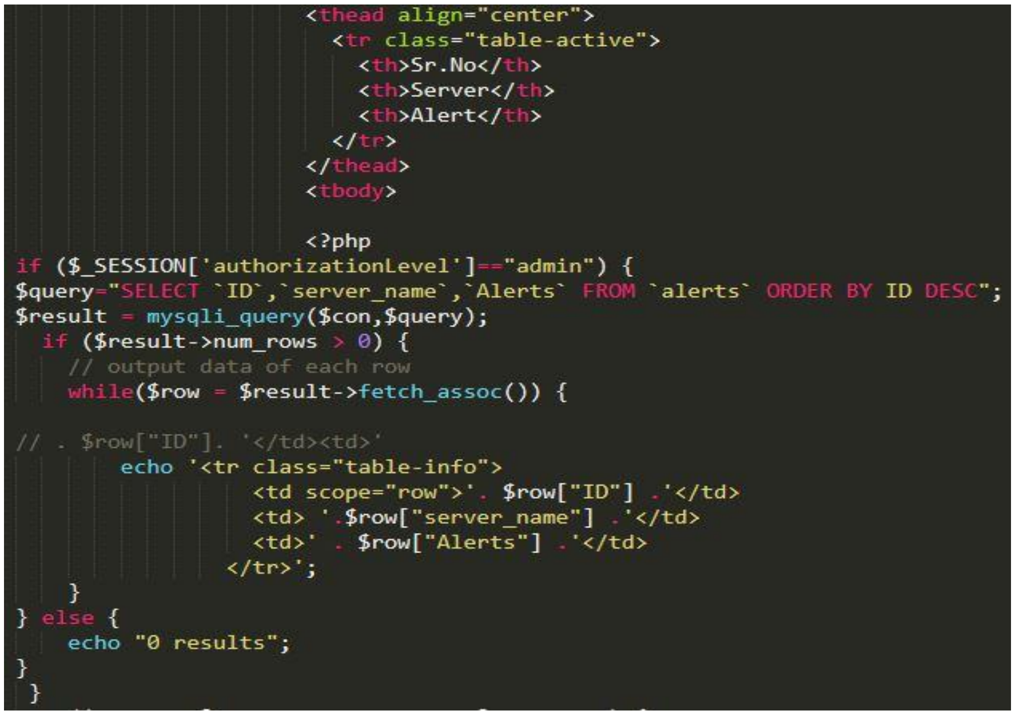
Given below screen in “Figure 5.19” shows email alerts of every server services failure through fetch from database. Whenever any kind of issue happen in monitoring server service, the admin user will receive an email alert and that user who have access. And, these alerts have also stored in the database.



Sr.No	Server	Alert
79	DNS	Dear user ! I hope you will find this email in best of your health.The purpose to write this email is to convey that Google DNS hosted on 8.8.8.8 was Down at 20/04/2019 12:00:54 due to Timeout.
78	DNS	Dear user ! I hope you will find this email in best of your health.The purpose to write this email is to convey that Google DNS hosted on 8.8.8.8 was Down at 18/04/2019 10:10:50 due to Timeout.
77	DNS	Dear user ! I hope you will find this email in best of your health.The purpose to write this email is to convey that Google DNS host 8.8.8.8 was reported inaccessible at 18/04/2019 07:27:40 due to UNREACHABLE.
76	DNS	Dear user ! I hope you will find this email in best of your health.The purpose to write this email is to convey that Google DNS host 8.8.8.8 was reported inaccessible at 18/04/2019 07:24:01 due to TIMEOUT.
75	DNS	Dear user ! I hope you will find this email in best of your health.The purpose to write this email is to convey that Google DNS host 8.8.8.8 was reported inaccessible at 18/04/2019 07:24:01 due to TIMEOUT.
74	WEB	Dear user ! I hope you will find this email in best of your health.The purpose to write this email is to convey that Namal Local Web Server hosted on 10.0.0.157 listening on Port 53 was reported inaccessible at 18/04/2019

Figure 5.19: View Alerts Page

Given below “Figure 5.20” is the code which fetches all alerts from database and display in the table.

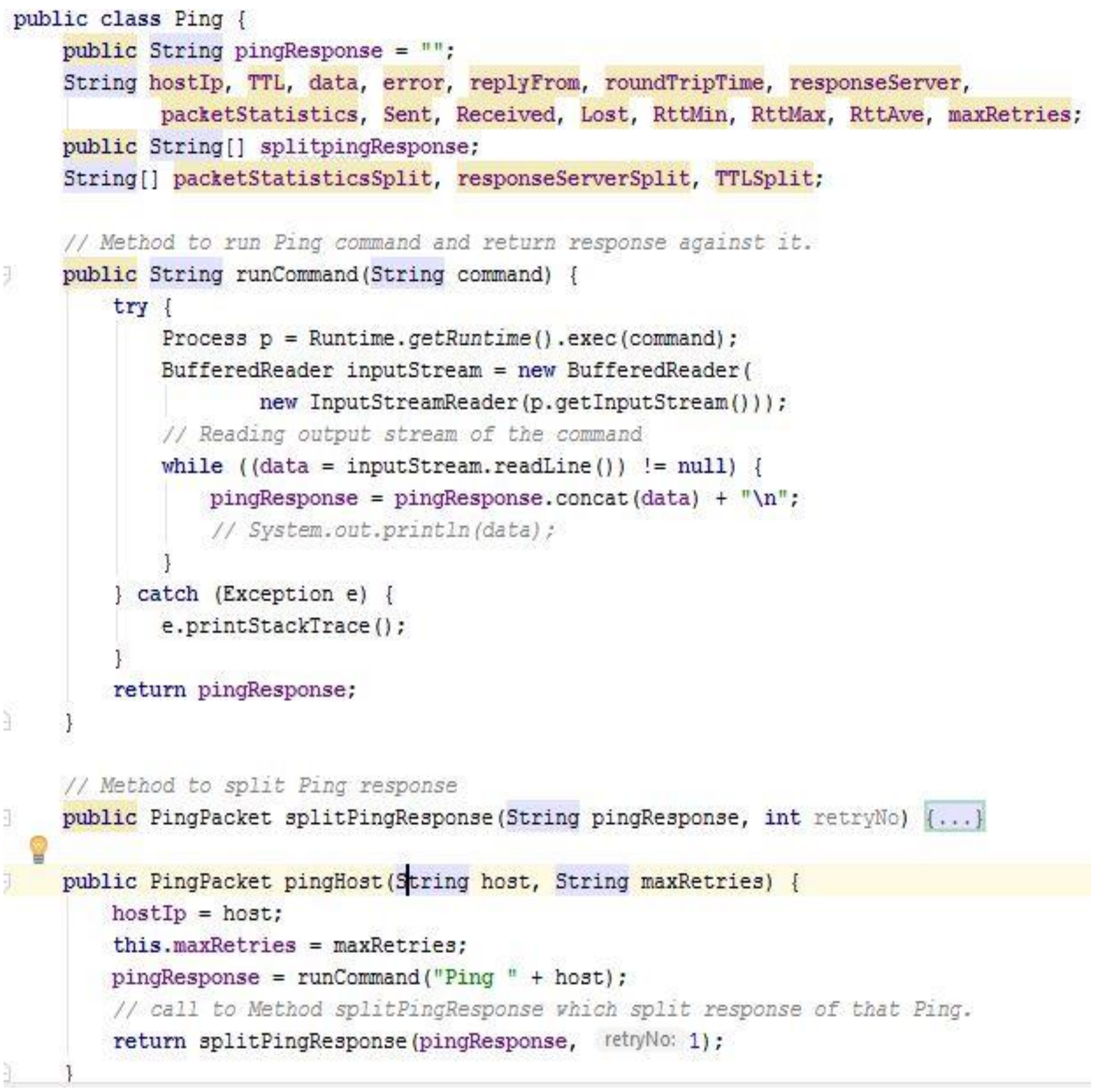


```
                            <thead align="center">
                              <tr class="table-active">
                                <th>Sr.No</th>
                                <th>Server</th>
                                <th>Alert</th>
                              </tr>
                            </thead>
                            <tbody>

                            <?php
if ($_SESSION['authorizationLevel']=="admin") {
$query="SELECT `ID`,`server_name`,`Alerts` FROM `alerts` ORDER BY ID DESC";
$result = mysqli_query($con,$query);
  if ($result->num_rows > 0) {
    // output data of each row
    while($row = $result->fetch_assoc()) {

// . $row["ID"]. '</td><td>'
        echo '<tr class="table-info">
                  <td scope="row">'. $row["ID"] .'</td>
                  <td> '.$row["server_name"] .'</td>
                  <td>' . $row["Alerts"] .'</td>
              </tr>';
    }
} else {
    echo "0 results";
}
 }
```

Figure 5.20: View Alerts page Code Snippet

Ping Class

This java class takes host IP as input to monitor host availability status through create ICMP request and send it to the host. Then, it analyses the response against each request. It sends again request to host in case of unavailable due to request time out or server unreachable.

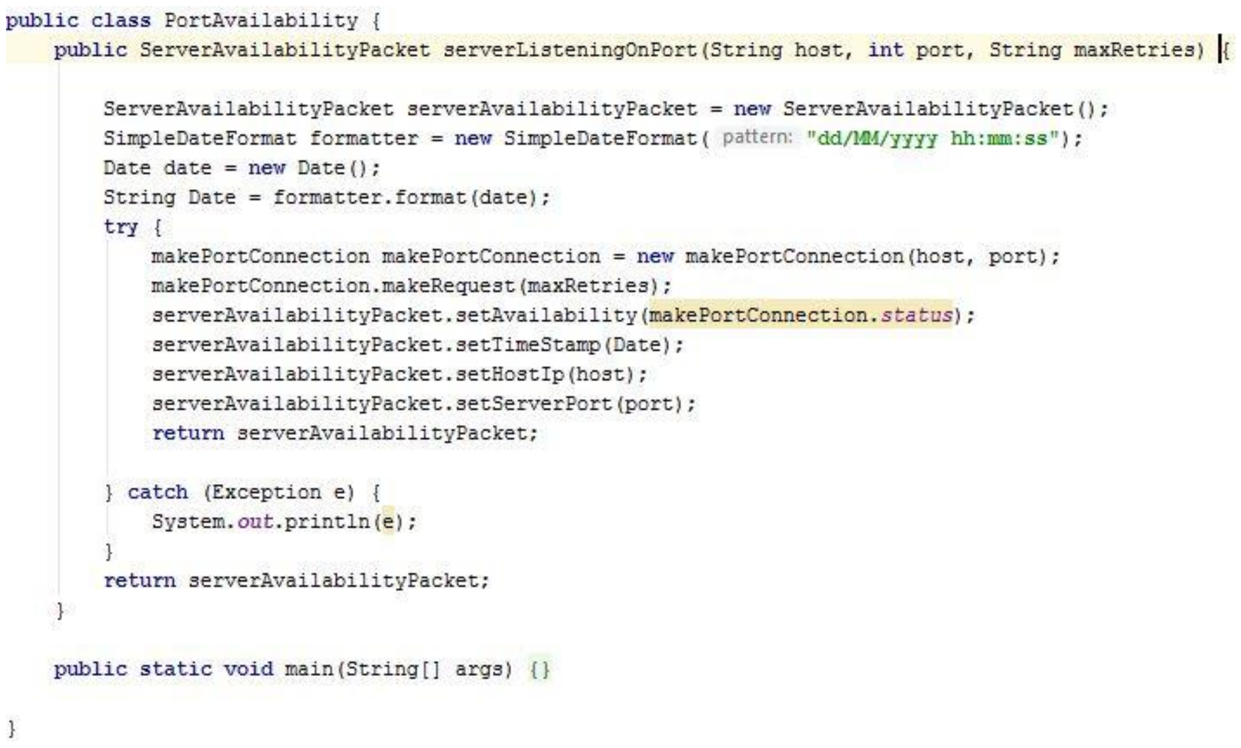


```java
public class Ping {
    public String pingResponse = "";
    String hostIp, TTL, data, error, replyFrom, roundTripTime, responseServer,
            packetStatistics, Sent, Received, Lost, RttMin, RttMax, RttAve, maxRetries;
    public String[] splitpingResponse;
    String[] packetStatisticsSplit, responseServerSplit, TTLSplit;

    // Method to run Ping command and return response against it.
    public String runCommand(String command) {
        try {
            Process p = Runtime.getRuntime().exec(command);
            BufferedReader inputStream = new BufferedReader(
                    new InputStreamReader(p.getInputStream()));
            // Reading output stream of the command
            while ((data = inputStream.readLine()) != null) {
                pingResponse = pingResponse.concat(data) + "\n";
                // System.out.println(data);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        return pingResponse;
    }

    // Method to split Ping response
    public PingPacket splitPingResponse(String pingResponse, int retryNo) {...}

    public PingPacket pingHost(String host, String maxRetries) {
        hostIp = host;
        this.maxRetries = maxRetries;
        pingResponse = runCommand("Ping " + host);
        // call to Method splitPingResponse which split response of that Ping.
        return splitPingResponse(pingResponse,  retryNo: 1);
    }
```

Figure 5.21: Ping Class Code Snippet

Port Availability Class

Given below java class code in "Figure 5.22" take host IP, server port and max retries as input to monitor the status of server availability through creating a socket connection with the server port. If the connection is unable to establish due to time out or unknown host, it creates a connection with the server port again and again until max retries value achieved.

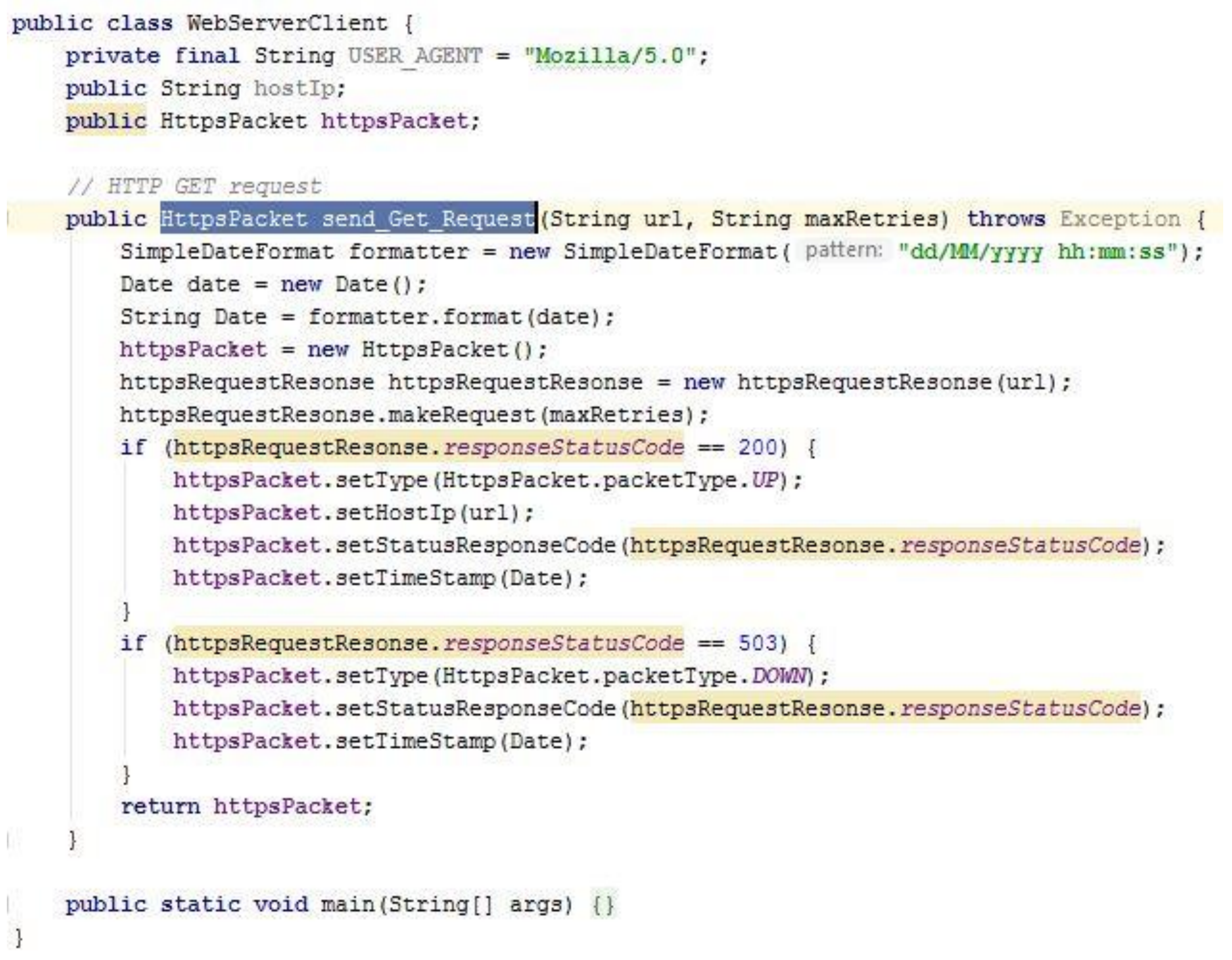


```java
public class PortAvailability {
    public ServerAvailabilityPacket serverListeningOnPort(String host, int port, String maxRetries) {

        ServerAvailabilityPacket serverAvailabilityPacket = new ServerAvailabilityPacket();
        SimpleDateFormat formatter = new SimpleDateFormat( pattern: "dd/MM/yyyy hh:mm:ss");
        Date date = new Date();
        String Date = formatter.format(date);
        try {
            makePortConnection makePortConnection = new makePortConnection(host, port);
            makePortConnection.makeRequest(maxRetries);
            serverAvailabilityPacket.setAvailability(makePortConnection.status);
            serverAvailabilityPacket.setTimeStamp(Date);
            serverAvailabilityPacket.setHostIp(host);
            serverAvailabilityPacket.setServerPort(port);
            return serverAvailabilityPacket;

        } catch (Exception e) {
            System.out.println(e);
        }
        return serverAvailabilityPacket;
    }

    public static void main(String[] args) {}

}
```

Figure 5.22: Port Availability Class Code Snippet

Web Server Client Class

This java class which is used to monitor web server service status. This class method names "HttpsPacket send_Get_Request" take web URL and max retries as input and send https requests to that server and analyze response against each request. It sends, again and again, https request until max retries value achieved in case of service unavailable due to load on the server. Code snapshot of this class is given below in "Figure 5.23".

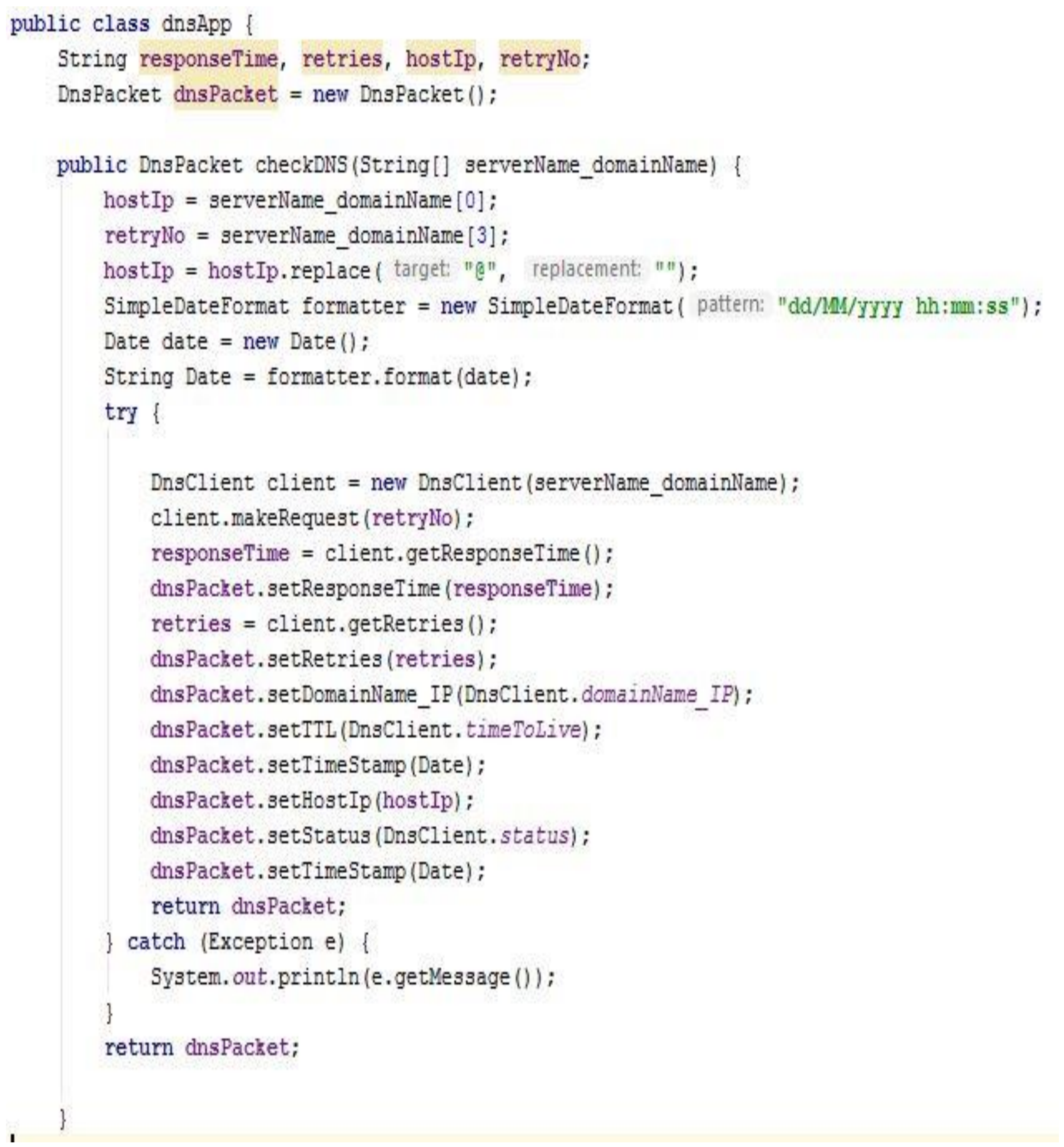


```java
public class WebServerClient {
    private final String USER_AGENT = "Mozilla/5.0";
    public String hostIp;
    public HttpsPacket httpsPacket;

    // HTTP GET request
    public HttpsPacket send_Get_Request(String url, String maxRetries) throws Exception {
        SimpleDateFormat formatter = new SimpleDateFormat( pattern: "dd/MM/yyyy hh:mm:ss");
        Date date = new Date();
        String Date = formatter.format(date);
        httpsPacket = new HttpsPacket();
        httpsRequestResonse httpsRequestResonse = new httpsRequestResonse(url);
        httpsRequestResonse.makeRequest(maxRetries);
        if (httpsRequestResonse.responseStatusCode == 200) {
            httpsPacket.setType(HttpsPacket.packetType.UP);
            httpsPacket.setHostIp(url);
            httpsPacket.setStatusResponseCode(httpsRequestResonse.responseStatusCode);
            httpsPacket.setTimeStamp(Date);
        }
        if (httpsRequestResonse.responseStatusCode == 503) {
            httpsPacket.setType(HttpsPacket.packetType.DOWN);
            httpsPacket.setStatusResponseCode(httpsRequestResonse.responseStatusCode);
            httpsPacket.setTimeStamp(Date);
        }
        return httpsPacket;
    }

    public static void main(String[] args) {}
}
```

Figure 5.23: Web Server Class Code Snippet

dnsApp Class

This java class is used to monitor domain name server service status. This class method names “DnsPacket checkDNS” takes a string array which has testing web URL, server host Ip, server port and max retries as input and send DNS query to that server and analyze response against each request. It sends DNS query until max retries value achieved in case of service unviable due to load on the server. Given below “Figure 5.24” is the code of this java class.

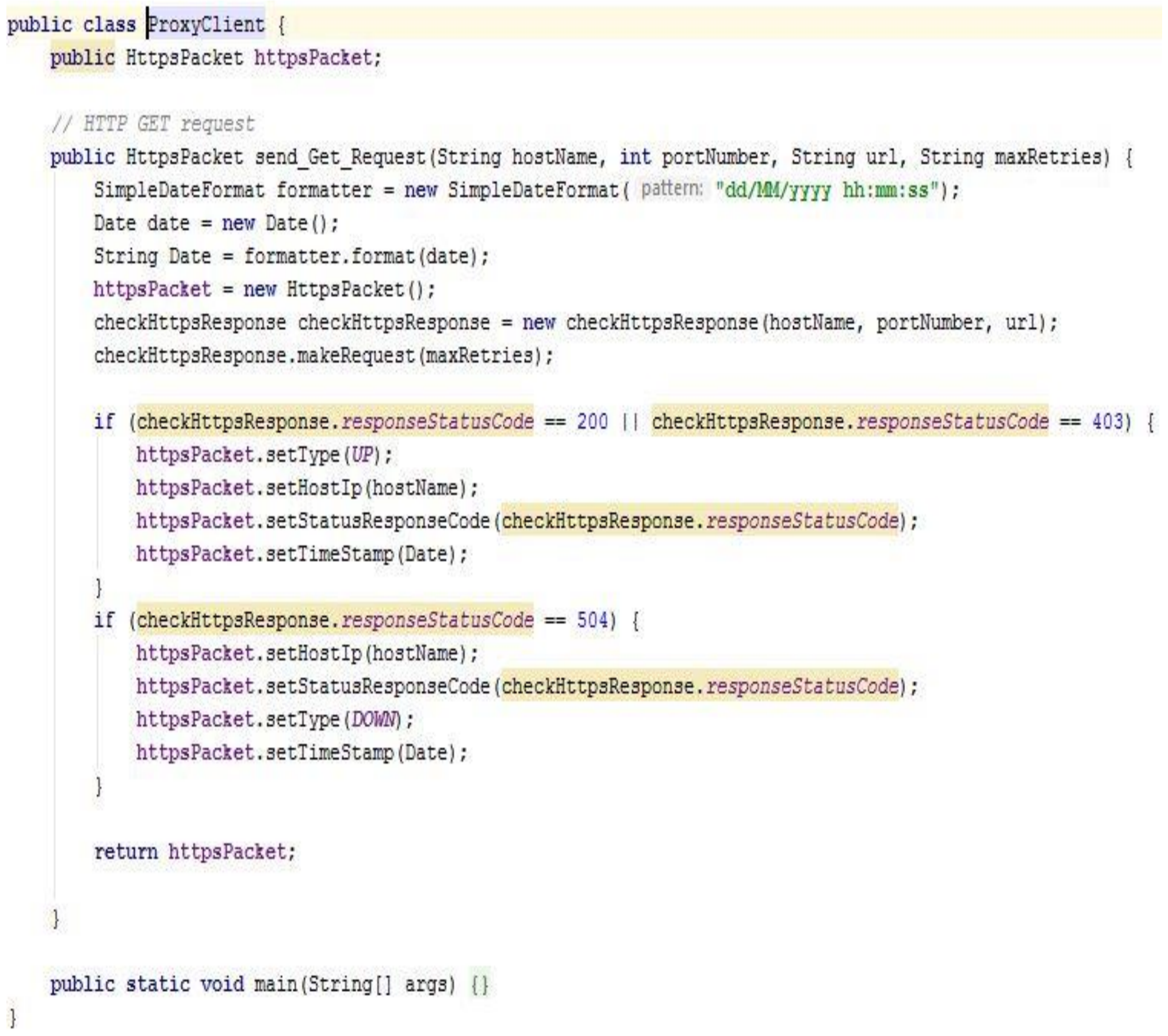


```
public class dnsApp {
    String responseTime, retries, hostIp, retryNo;
    DnsPacket dnsPacket = new DnsPacket();

    public DnsPacket checkDNS(String[] serverName_domainName) {
        hostIp = serverName_domainName[0];
        retryNo = serverName_domainName[3];
        hostIp = hostIp.replace( target: "@",  replacement: "");
        SimpleDateFormat formatter = new SimpleDateFormat( pattern: "dd/MM/yyyy hh:mm:ss");
        Date date = new Date();
        String Date = formatter.format(date);
        try {

            DnsClient client = new DnsClient(serverName_domainName);
            client.makeRequest(retryNo);
            responseTime = client.getResponseTime();
            dnsPacket.setResponseTime(responseTime);
            retries = client.getRetries();
            dnsPacket.setRetries(retries);
            dnsPacket.setDomainName_IP(DnsClient.domainName_IP);
            dnsPacket.setTTL(DnsClient.timeToLive);
            dnsPacket.setTimeStamp(Date);
            dnsPacket.setHostIp(hostIp);
            dnsPacket.setStatus(DnsClient.status);
            dnsPacket.setTimeStamp(Date);
            return dnsPacket;
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
        return dnsPacket;

    }
```

Figure 5.24: dnsApp class Code Snippet

Proxy Client

This java class is used to monitor proxy server service status. This class method names “HttpsPacket send_Get_Request” take web URL and max retries as input and send https requests to that server and analyze response against each request. It sends, again and again, https request until max retries value achieved in case of service unavailable due to load on the server. Given below “Figure 5.25” is the code snippet of this class.

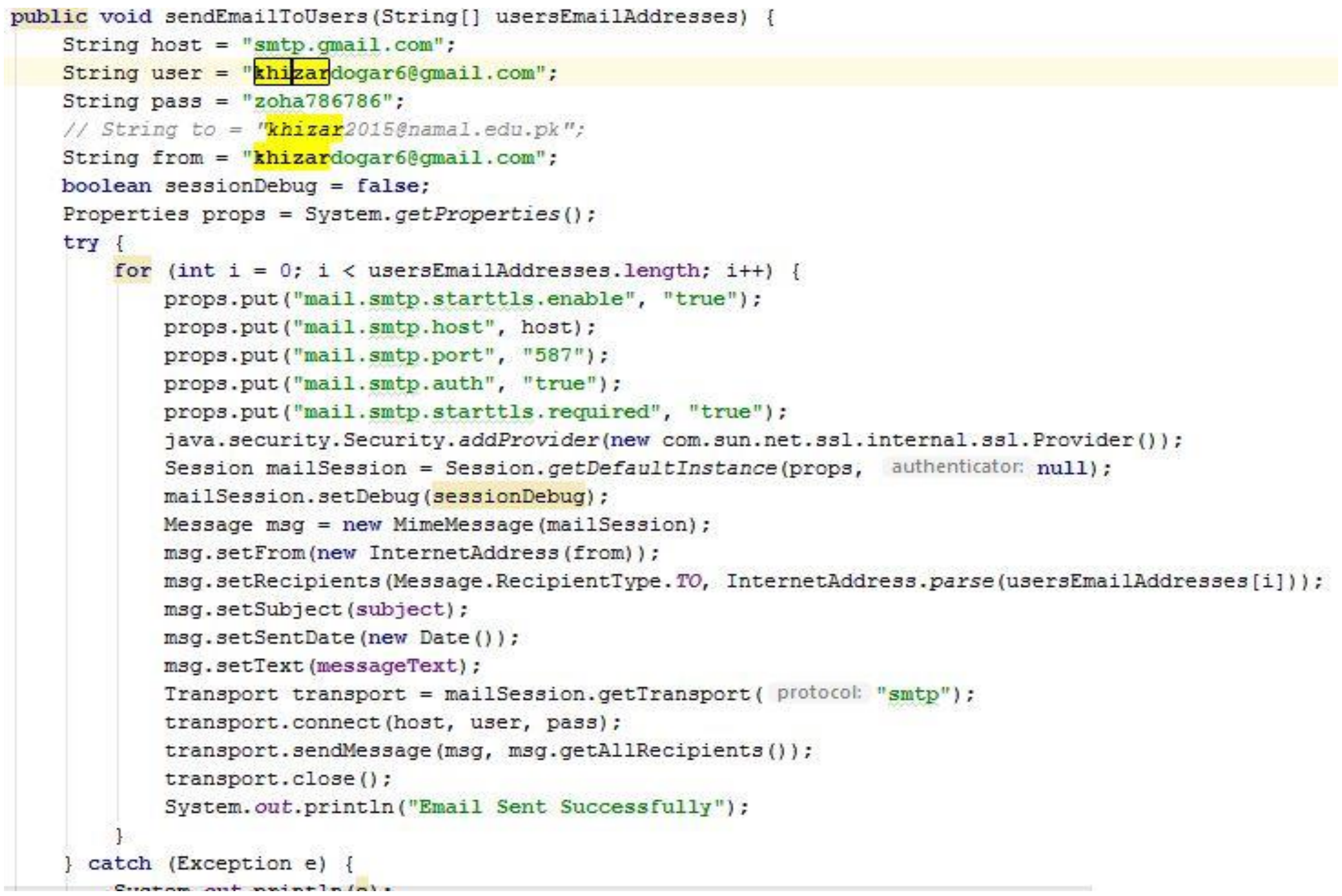


```java
public class ProxyClient {
    public HttpsPacket httpsPacket;

    // HTTP GET request
    public HttpsPacket send_Get_Request(String hostName, int portNumber, String url, String maxRetries) {
        SimpleDateFormat formatter = new SimpleDateFormat( pattern: "dd/MM/yyyy hh:mm:ss");
        Date date = new Date();
        String Date = formatter.format(date);
        httpsPacket = new HttpsPacket();
        checkHttpsResponse checkHttpsResponse = new checkHttpsResponse(hostName, portNumber, url);
        checkHttpsResponse.makeRequest(maxRetries);

        if (checkHttpsResponse.responseStatusCode == 200 || checkHttpsResponse.responseStatusCode == 403) {
            httpsPacket.setType(UP);
            httpsPacket.setHostIp(hostName);
            httpsPacket.setStatusResponseCode(checkHttpsResponse.responseStatusCode);
            httpsPacket.setTimeStamp(Date);
        }
        if (checkHttpsResponse.responseStatusCode == 504) {
            httpsPacket.setHostIp(hostName);
            httpsPacket.setStatusResponseCode(checkHttpsResponse.responseStatusCode);
            httpsPacket.setType(DOWN);
            httpsPacket.setTimeStamp(Date);
        }

        return httpsPacket;

    }

    public static void main(String[] args) {}
}
```

Figure 5.25: Proxy Client Code Snippet

Email Alert

The role of this java class in this system is to send an email alert to the admin user and local user in case of issue happened in server services or hardware failure. This class method names “sendEmailToUsers” which takes system users email address and play a duty to send alerts to all users. A snippet of this class code is given below in “Figure 5.26”.

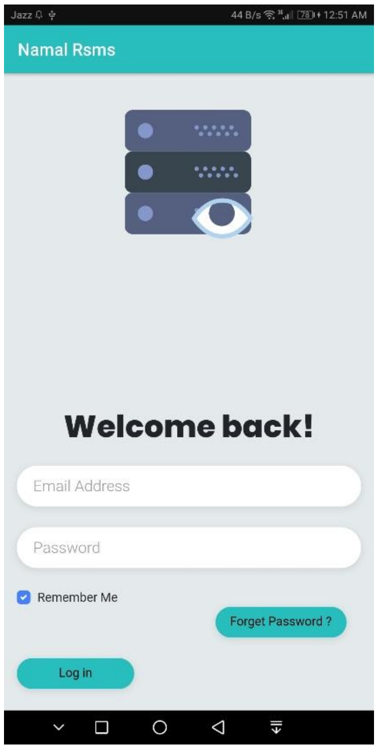


```java
public void sendEmailToUsers(String[] usersEmailAddresses) {
    String host = "smtp.gmail.com";
    String user = "khizardogar6@gmail.com";
    String pass = "zoha786786";
    // String to = "khizar2015@namal.edu.pk";
    String from = "khizardogar6@gmail.com";
    boolean sessionDebug = false;
    Properties props = System.getProperties();
    try {
        for (int i = 0; i < usersEmailAddresses.length; i++) {
            props.put("mail.smtp.starttls.enable", "true");
            props.put("mail.smtp.host", host);
            props.put("mail.smtp.port", "587");
            props.put("mail.smtp.auth", "true");
            props.put("mail.smtp.starttls.required", "true");
            java.security.Security.addProvider(new com.sun.net.ssl.internal.ssl.Provider());
            Session mailSession = Session.getDefaultInstance(props,  authenticator: null);
            mailSession.setDebug(sessionDebug);
            Message msg = new MimeMessage(mailSession);
            msg.setFrom(new InternetAddress(from));
            msg.setRecipients(Message.RecipientType.TO, InternetAddress.parse(usersEmailAddresses[i]));
            msg.setSubject(subject);
            msg.setSentDate(new Date());
            msg.setText(messageText);
            Transport transport = mailSession.getTransport( protocol: "smtp");
            transport.connect(host, user, pass);
            transport.sendMessage(msg, msg.getAllRecipients());
            transport.close();
            System.out.println("Email Sent Successfully");
        }
    } catch (Exception e) {
```

Figure 5.26: Email Alert Code Snippet

5.3 Android Application

As it had been explained earlier, it is a hybrid application. There are three types of the mobile app. One is a native that is developed to target a particular mobile operating system. The second one is a web app and, the other one is hybrid. The reason to choose a hybrid application is to target every smart mobile operating system. This application runs in a native container that uses the mobile browser engine to render this system web application from the server where it hosts and show web view to the user in the mobile app. Given below is the code snippet of the mobile application user interface of this system.



Jazz 44 B/s 78 12:51 AM

Namal Rsms

Welcome back!

Email Address

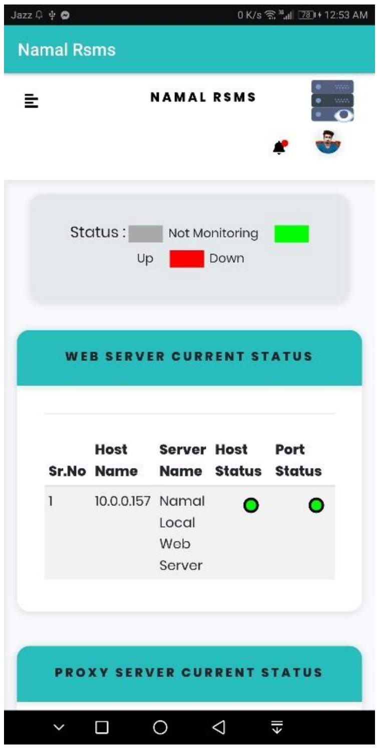
Password

☑ Remember Me

Forget Password ?

Log in

Figure: 5.27 Login Screen



Jazz 0 K/s 78 12:53 AM

Namal Rsms

NAMAL RSMS

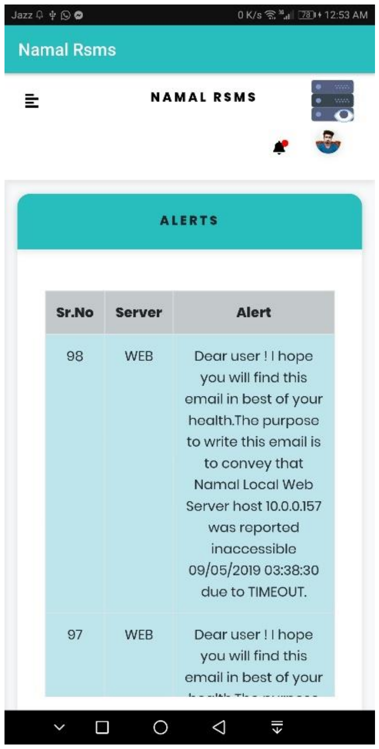
Status : ▨ Not Monitoring 🟩 Up 🟥 Down

WEB SERVER CURRENT STATUS

Sr.No	Host Name	Server Name	Host Status	Port Status
1	10.0.0.157	Namal Local Web Server	🟢	🟢

PROXY SERVER CURRENT STATUS

Figure: 5.28 Dashboard



Jazz 0 K/s 78 12:53 AM

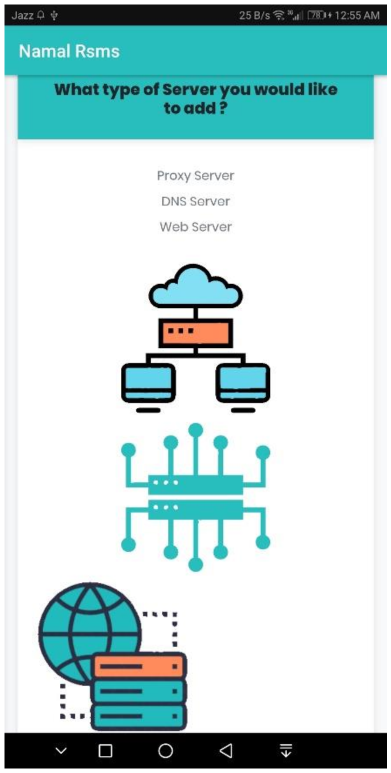
Namal Rsms

NAMAL RSMS

ALERTS

Sr.No	Server	Alert
98	WEB	Dear user ! I hope you will find this email in best of your health.The purpose to write this email is to convey that Namal Local Web Server host 10.0.0.157 was reported inaccessible 09/05/2019 03:38:30 due to TIMEOUT.
97	WEB	Dear user ! I hope you will find this email in best of your

Figure: 5.29 Alerts Screen



Jazz 25 B/s 78 12:55 AM

Namal Rsms

What type of Server you would like to add ?

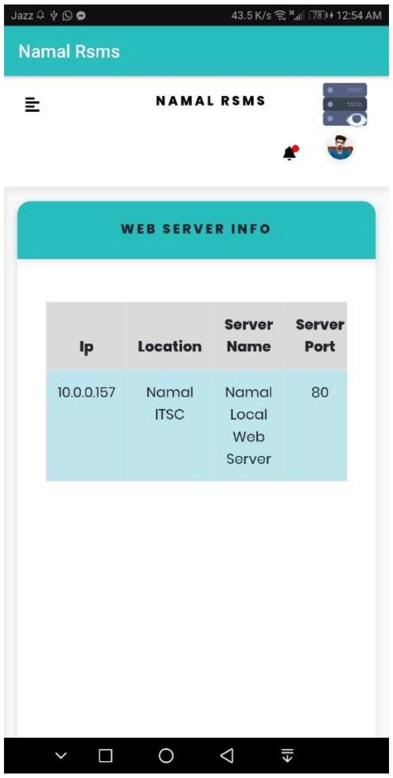
Proxy Server

DNS Server

Web Server

Figure:5.30 Add New Server

Screen



Jazz 43.5 K/s 78 12:54 AM

Namal Rsms

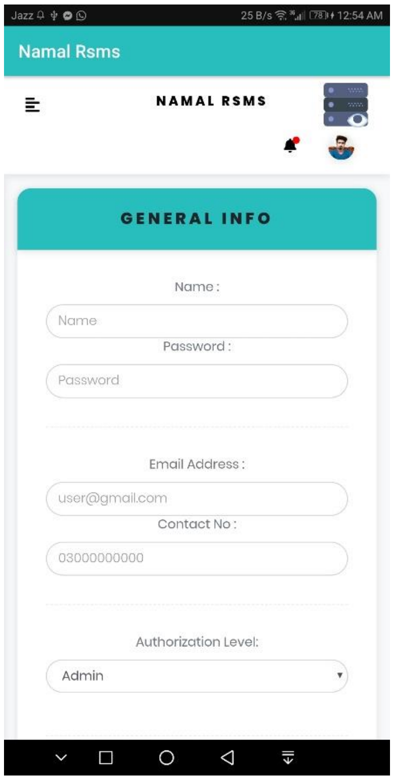
NAMAL RSMS

WEB SERVER INFO

Ip	Location	Server Name	Server Port
10.0.0.157	Namal ITSC	Namal Local Web Server	80

Figure: 5.31 Web Server

Configuration Wizard



Jazz 25 B/s 78 12:54 AM

Namal Rsms

NAMAL RSMS

GENERAL INFO

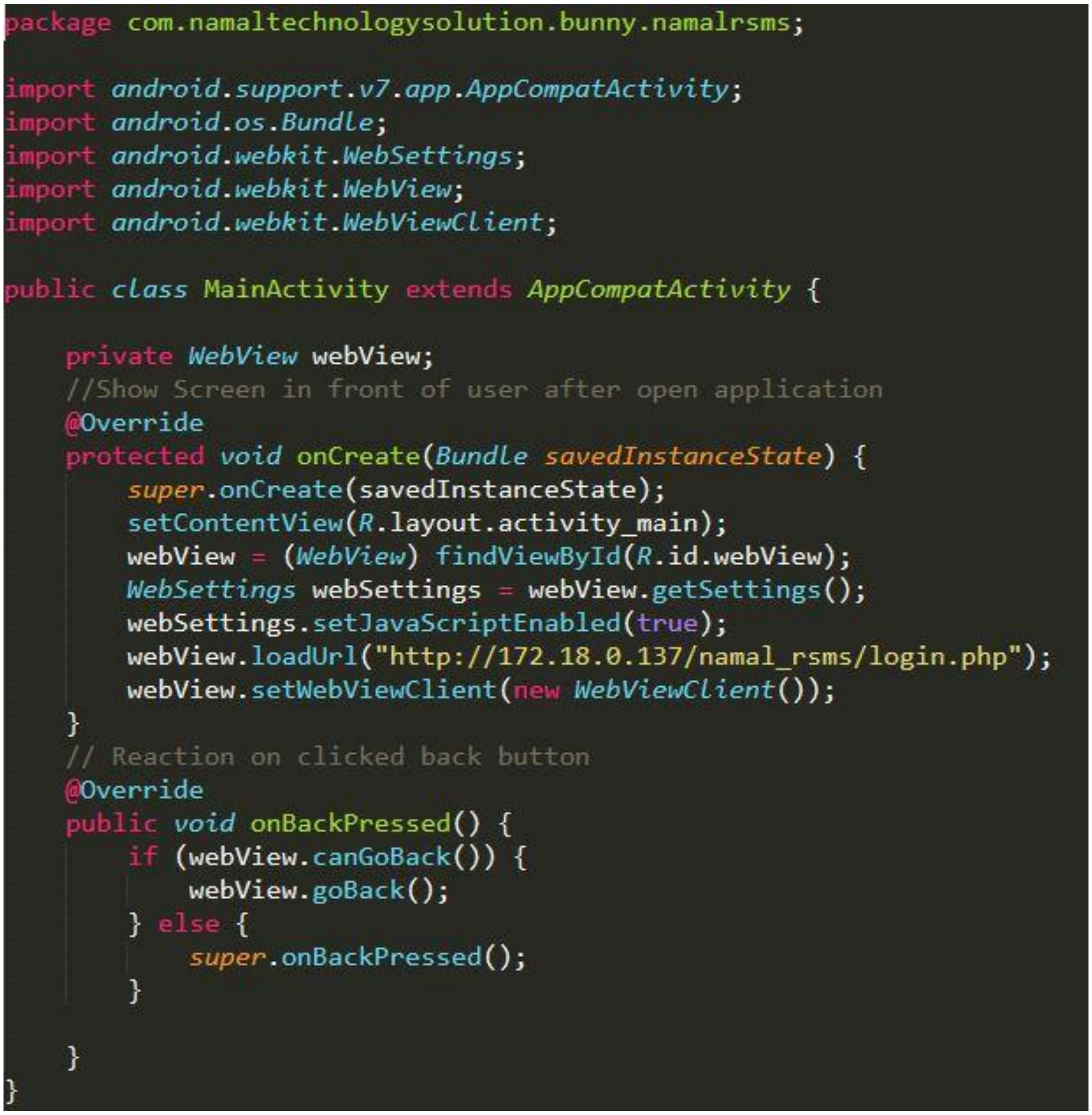
Name :	Name
Password :	Password
Email Address :	user@gmail.com
Contact No :	03000000000
Authorization Level:	Admin

Figure:5.32 Add new user

Screen

Main Activity

Since we had decided to target every smart mobile operating system in order to access of this system so we have used browser bundle known as “web view” to build a hybrid app using web technologies (HTML, CSS, JavaScript). Given below code snippet in “Figure 5.33” clearly shows how the system web application is streaming from localhost server address “172.16.0.137” in the mobile application. To use android application, please set above mentioned address as localhost Ip address.



```java
package com.namaltechnologysolution.bunny.namalrsms;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.webkit.WebSettings;
import android.webkit.WebView;
import android.webkit.WebViewClient;

public class MainActivity extends AppCompatActivity {

    private WebView webView;
    //Show Screen in front of user after open application
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        webView = (WebView) findViewById(R.id.webView);
        WebSettings webSettings = webView.getSettings();
        webSettings.setJavaScriptEnabled(true);
        webView.loadUrl("http://172.18.0.137/namal_rsms/login.php");
        webView.setWebViewClient(new WebViewClient());
    }
    // Reaction on clicked back button
    @Override
    public void onBackPressed() {
        if (webView.canGoBack()) {
            webView.goBack();
        } else {
            super.onBackPressed();
        }

    }
}
```

Figure 5.33: Main Activity Code Snippet

Chapter 6

6.Testing and Results

6.1 Testing

Testing is very important in the development phase of the software. The reason for testing is to find out mistakes what we have done. These can be due to a bad assumption or it can be a blind spot. So, we can test our product by ourselves or by someone else because it is possible that we may not notice the flaw before testing and notice after it. So, there are three different kinds of testing, Usability testing, Functionality testing, and unit testing. I have done all three of them and these are elaborate below in detail.

6.1.1 Usability Testing

Usability testing has played an important role in my development process. I have done this testing by asking employees of Namal information technology support center to use and test it as a user and give me feedback on its features and the user interfaces. These guys helped me to find bugs in my system and I have removed them from the system. Given below are the features the users were asked to check.

- Add User
- Edit and delete user
- Add Server
- Edit server configuration
- View Alerts

Most of the users were felt comfortable during the use of this system and they did not feel confusion anywhere. Everything was correctly in place they said. They also said that the system dashboard was continuously updating them about the current status of the monitoring servers.

Except for a few little bit issues, no one had reported any bug or anything else that could have created problems for them while they were using it. Some of them had suggested some improvements in the UI design that were noted instantly and I had worked to improve those features as well. Given below is the rating they all gave to NAMAL REMOTE SERVER MONITORING SYSTEM.

ID	User-Friendly	Functionality	Will use it in the future
3	8	9	8
2	10	8	10
4	9	10	9

Table 6.1: Usability testing rating

6.1.2 Functionality Testing

Functionality testing was done to test the functionalities specified in the requirement section. Given below “Table 6.2” and “Table 6.3” shows the small description of all functional requirements and their test results.

Requirement ID	Description
1	Admin or Local User can log in through his Gmail account
2	Admin user can add new user
3	Admin user can configure users
5	Admin User can add a new server
6	Admin user can configure all servers
7	Local user can configure server according to admin permission
8	Admin user can get alerts of all monitoring servers
9	Local users can get alerts of that server who they have access

Table 6.2: Functional requirement description

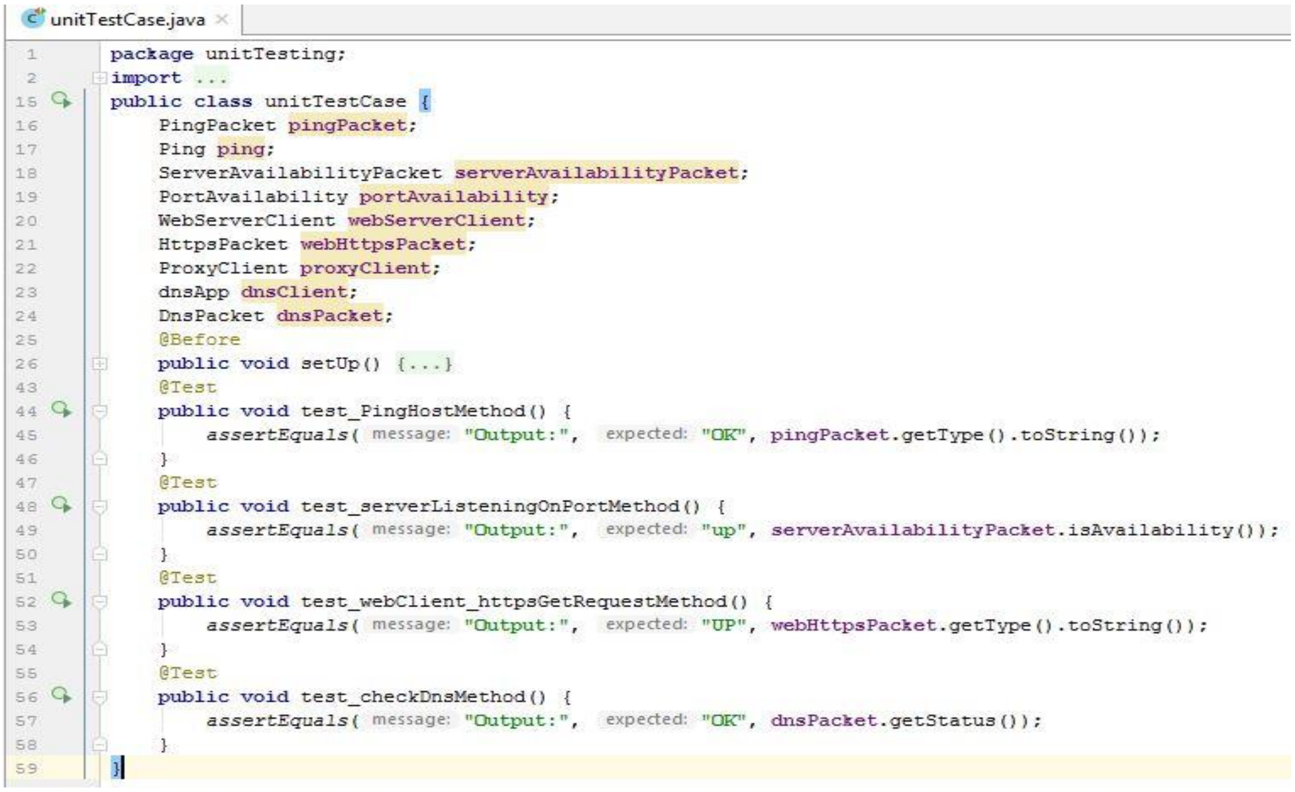
Requirement ID	Action	Output
1	Pressing the login button on the login screen and providing correct email and password	User successfully logins
2	Pressing the login button on the login screen and providing wrong information	Login process fails
3	Pressing the login button on the login screen and input fields are empty	Login process fails
4	Admin can add a new user by filling all required info like name, email address, password, contact number, Authorization level	User successful registered
5	Admin can add a new server by filling all required info like host IP, location, server name, port number, testing URL, monitoring configuration.	The server successfully added and monitoring start
6	Admin can configure the server by edit all required info	The server successfully configured.
7	Admin can configure added user permission	User successfully configured.
8	Pressing the notification button in the action bar	All notifications are shown
9	On clicked profile icon of the action bar to change the profile image	Profile image successfully changed

Table 6.3: Functional requirement testing output

6.1.3 Unit Testing

I have done unit testing for the java applications. I have written test cases in “IntelliJ IDEA” and run these to test methods of these java classes “Ping”, “PortAvailability”, “WebServerClinet”, and “dnsApp”. Moreover, they have passed. Given below “Figure 6.1” shows the test cases for these classes. And, a snippet of these test cases result is also given below in “Figure 6.2” at the end of “Figure 6.1”.

Test Case

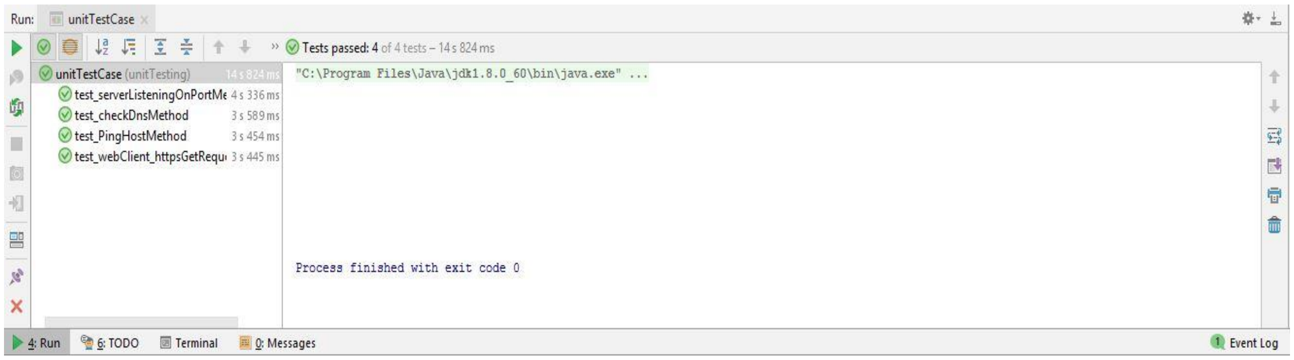


unitTestCase.java

```java
package unitTesting;
import ...
public class unitTestCase {
    PingPacket pingPacket;
    Ping ping;
    ServerAvailabilityPacket serverAvailabilityPacket;
    PortAvailability portAvailability;
    WebServerClient webServerClient;
    HttpsPacket webHttpsPacket;
    ProxyClient proxyClient;
    dnsApp dnsClient;
    DnsPacket dnsPacket;
    @Before
    public void setUp() {...}
    @Test
    public void test_PingHostMethod() {
        assertEquals( message: "Output:",  expected: "OK", pingPacket.getType().toString());
    }
    @Test
    public void test_serverListeningOnPortMethod() {
        assertEquals( message: "Output:",  expected: "up", serverAvailabilityPacket.isAvailability());
    }
    @Test
    public void test_webClient_httpsGetRequestMethod() {
        assertEquals( message: "Output:",  expected: "UP", webHttpsPacket.getType().toString());
    }
    @Test
    public void test_checkDnsMethod() {
        assertEquals( message: "Output:",  expected: "OK", dnsPacket.getStatus());
    }
}
```

Figure 6.1: Unit Test Case

Result



```
Tests passed: 4 of 4 tests – 14 s 824 ms

unitTestCase (unitTesting)                14 s 824 ms
  test_serverListeningOnPortMe            4 s 336 ms
  test_checkDnsMethod                     3 s 589 ms
  test_PingHostMethod                     3 s 454 ms
  test_webClient_httpsGetRequ             3 s 445 ms

"C:\Program Files\Java\jdk1.8.0_60\bin\java.exe" ...

Process finished with exit code 0
```

Figure 6.2: Unit Test Result

6.2 Results

This part of the chapter is about this system results after test it on a different type of local servers of NAMAL. Testing had done through manually on or off locally deployed three type of servers (Web, DNS, Proxy) and their services and visualize this system results on the behavior of servers. Following graph in "Figure 6.3" shows the nature of results on different time slot.

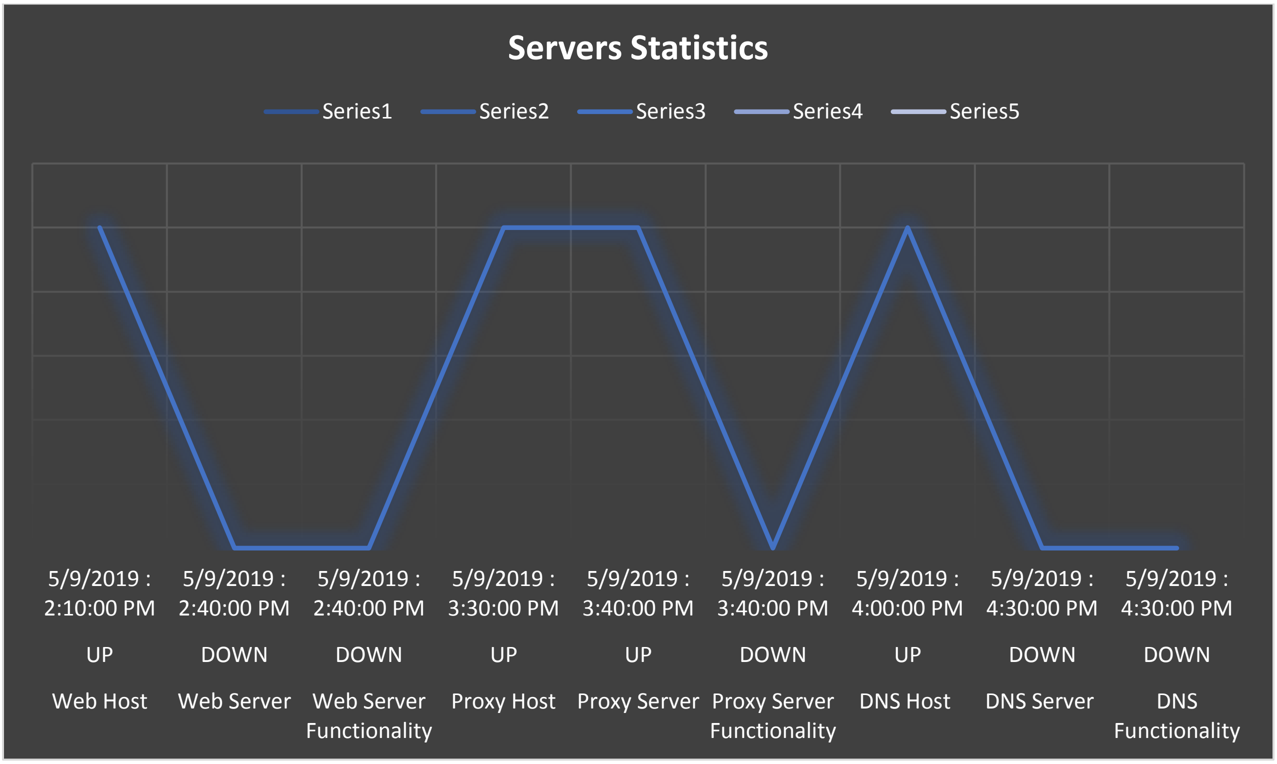


Figure 6.3: Server Statistics

Chapter 7

7.Conclusions and Future Work

7.1 Conclusions

This is the web app and hybrid app based remotely server monitoring solution for the organizations who have a huge datacenter having a lot of servers (Web, DNS, Proxy). These servers and their services are currently monitored by manually or by using customized software with very limited functionalities. So, this system has an ability to monitored this type of multiple serves at a time according to users' configurations and throw alerts by email to them in case of miss happening.

I have managed to implement a lot of committed functionalities. I have also used multi-threading in a very efficient way to monitor multiple servers and their services status. This tool is ready to use organizations. We are also planning to deploy the system for testing purpose. Although this system is independent but the addition of a few things would make it more efficient.

7.2 Future Work

This system has space for the addition of new features, a lot of features could be added to make this system more efficient in order to alert the user about server's failure issue. For example, alert to the user could be through mobile SMS or mobile call. Moreover, now this system is restricted to only three servers (Web, DNS, Proxy). So, the future plan is to implement previously mentioned alert services and to add functionality to monitor the more different type of servers along with these servers.

References

Bootstrap, 2019. *Bootstrap.* [Online]
Available at: https://getbootstrap.com/
[Accessed 28 04 2019].

Jetbrains, 2019. *IntelliJ Platform.* [Online]
Available at: https://www.jetbrains.com/opensource/idea/
[Accessed 28 04 2019].

kjkoster.com, n.d. *Architecture.* [Online]
Available at: http://www.kjkoster.org/zapcat/Architecture.html
[Accessed 12 January 2019].

kjkoster, 2019. *Architecture.* [Online]
Available at: http://www.kjkoster.org/zapcat/Architecture.html
[Accessed 12 January 2019].

Mysql, 2019. *What is MySQL?.* [Online]
Available at: https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html
[Accessed 28 04 2019].

Nagios, 2016. *Nagios Support Knowledgebase.* [Online]
Available at: https://support.nagios.com/kb/article.php?id=35
[Accessed 12 January 2019].

Nagios, 2019. *Nagios.* [Online]
Available at: https://www.nagios.org/about/overview/
[Accessed 11 January 2019].

PHP, 2019. *What is PHP?.* [Online]
Available at: https://www.php.net/manual/en/intro-whatis.php
[Accessed 28 4 2019].

researchgate, 2019. *Ganglia architecture | Download Scientific Diagram.* [Online]
Available at: https://www.researchgate.net/figure/Ganglia-architecture_fig1_299973832
[Accessed 12 January 2019].

Stanford Linear Accelerator Center, 2001. *Passive vs. Active Monitoring.* [Online]
Available at: https://www.slac.stanford.edu/comp/net/wan-mon/passive-vs-active.html
[Accessed 12 January 2019].

Toit, J. d., 2016. *Active vs. Passive network monitoring: an infographic.* [Online]
Available at: https://www.irisns.com/active-vs-passive-network-monitoring-an-infographic/
[Accessed 12 January 2019].

WampServer, 2019. *WampServer.* [Online]
Available at: http://www.wampserver.com/en/
[Accessed 28 04 2019].

‘2. What is Zabbix’ (2019), p. 2019.

Andreozzi, S., Ciuffoletti, A. and Ghiselli, A. (2019) ‘ON THE INTEGRATION OF PASSIVE AND ACTIVE NETWORK MONITORING IN GRID SYSTEMS’.

Cottrell, L. (2001) ‘Passive and Active Monitoring on a High-performance g p Network’, (February), pp. 1–35.

‘Ganglia Distributed monitoring system’ (2019).

Kind, A. N. Y. *et al.* (2010) ‘Nagios Core Version 3 . x Documentation Table of Contents’, pp. 1–354.

Massie, M. L., Chun, B. N. and Culler, D. E. (2004) ‘The ganglia distributed monitoring system: design’.

Mathapati, V. and Aswatha, A. R. (2013) ‘Performance Analysis of System Resources By Server Monitoring’, *International Journal of Innovative Research in Science, Engineering and Technology*, 2(7), pp. 3153–3157. Available at: www.ijirset.com.

Mongkolluksamee, S., Pongpaibool, P. and Issariyapat, C. (2010) ‘Strengths and limitations of Nagios as a network monitoring solution’, *Proceedings of the 7th International Joint Conference on Computer Science and Software Engineering (JCSSE 2010). Bangkok, Thailand*, (February), pp. 96–101.

Zabbix, A. *et al.* (2019) ‘8 Known issues’, pp. 12–14.

Zabbix SIA (2019) ‘4 Zabbix overview [Zabbix Documentation 3.4]’, pp. 11–12. Available at: https://www.zabbix.com/documentation/3.4/manual/introduction/overview.

‘2. What is Zabbix’ (2019), p. 2019.

Andreozzi, S., Ciuffoletti, A. and Ghiselli, A. (2019) ‘ON THE INTEGRATION OF PASSIVE AND ACTIVE NETWORK MONITORING IN GRID SYSTEMS’.

Cottrell, L. (2001) ‘Passive and Active Monitoring on a High-performance g p Network’, (February), pp. 1–35.

‘Ganglia Distributed monitoring system’ (2019).

Kind, A. N. Y. *et al.* (2010) ‘Nagios Core Version 3 . x Documentation Table of Contents’, pp. 1–354.

Massie, M. L., Chun, B. N. and Culler, D. E. (2004) ‘The ganglia distributed monitoring system: design’.

Mathapati, V. and Aswatha, A. R. (2013) ‘Performance Analysis of System Resources By Server Monitoring’, *International Journal of Innovative Research in Science, Engineering and Technology*, 2(7), pp. 3153–3157. Available at: www.ijirset.com.

Mongkolluksamee, S., Pongpaibool, P. and Issariyapat, C. (2010) ‘Strengths and limitations of Nagios as a network monitoring solution’, *Proceedings of the 7th International Joint Conference on Computer Science and Software Engineering (JCSSE 2010). Bangkok, Thailand*, (February), pp. 96–101.

Zabbix, A. *et al.* (2019) ‘8 Known issues’, pp. 12–14.

Zabbix SIA (2019) ‘4 Zabbix overview [Zabbix Documentation 3.4]’, pp. 11–12. Available at: https://www.zabbix.com/documentation/3.4/manual/introduction/overview.