

# **SYMBIOSIS UNIVERSITY OF APPLIED SCIENCES**

## **INDORE**



### **PROJECT REPORT**

### **ON**

### **“MONNIT: The System Monitoring Application”**

Submitted to “Symbiosis University of Applied Sciences, Indore”  
As a final project report for the partial fulfillment of the award of degree of

### **BACHELOR OF TECHNOLOGY**

### **IN**

### **SCHOOL OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY**

Submitted To:

Dr. Neha Gupta

Director I/C (SCSIT)

Submitted By:

Atharva Pednekar (2020BTCS007)

Jaydeep Patel (2020BTCS016)

Tejas Bhati (2020BTCS042)

# **SYMBIOSIS UNIVERSITY OF APPLIED SCIENCES**

## **INDORE**

### **CERTIFICATE**

This is to certify that the Innovative Project report entitled “MONNIT: The System Monitoring Application”, submitted by Atharva Pednekar, Jaydeep Patel & Tejas Bhati students of third year towards partial fulfillment of the degree of Bachelor of Technology in School of Computer Science and Information Technology in year 2022-2023.

Symbiosis University of Applied Sciences, Indore (M.P.)

Place: Indore

Date: 05/12/2022

INTERNAL EXAMINER

EXTERNAL EXAMINER

# **SYMBIOSIS UNIVERSITY OF APPLIED SCIENCES**

## **INDORE**

### **RECOMMENDATION**

The work entitled “MONNIT: The System Monitoring Application”, submitted by Atharva Pednekar, Jaydeep Patel & Tejas Bhati students of third year Computer Science and Information Technology, towards the partial fulfillment for the award of degree of Bachelor of Technology in Computer Science and Information Technology of Symbiosis University of Applied Sciences Indore (M.P.) is a satisfactory account of their Innovative Project and is recommended for the award of the degree.

Endorsed By:

Dr. Neha Gupta

Director I/C, SCSIT

## **STUDENT UNDERTAKING**

I hereby undertake that the project work entitled “MONNIT: The System Monitoring Application” has been carried out by me and my team from the period August to December and the report so prepared is a record of work done by me during my Innovative Project. I further declare that we have completed the innovative project in accordance with the policy of the University. This Project report is submitted towards fulfillment of my academic requirement and not for any other purpose.

I hereby undertake that the material of this Project is my original work and we have not copied anything from anywhere else. The material obtained from other sources has been duly acknowledged. I understand that if at any stage, it is found that I have indulged in any malpractice or the project and the project report has been copied or not completed by us, the university shall cancel my degree/withhold my result and appropriate disciplinary action shall be initiated against our team.

Student Name and Signature

Mentor Name & Signature

Enrollment Number:

Date:

Head of School Name & Signature

# **SYMBIOSIS UNIVERSITY OF APPLIED SCIENCES**

## **INDORE**

### **ACKNOWLEDGEMENT**

The successful completion of any work is generally not an individual effort. It is an outcome of dedicated and cumulative efforts of a number of people, each having its own importance to the objective. This section is a value of thanks and gratitude towards all those persons who have implicitly or explicitly contributed in their own unique way towards the completion of the project. For their invaluable comments and suggestions, I wish to thank them all.

Positive inspiration and right guidance are must in every aspect of life. Especially, when we arrive at academic stage for instance. For the success of our project a number of obligations have been taken. We have performed solemn duty of expressing a heartfelt thanks to all who have endowed us with their precious perpetual guidance, suggestions and information. Any kind of help directly or indirectly has proved importance to us.

# Table of Contents

<b>Chapter-1: INTRODUCTION .....</b>	<b>7</b>
1. Introduction .....	8
2. Literature Review .....	9
3. Problem Statement.....	10
<b>Chapter-2: THE PROJECT .....</b>	<b>11</b>
1. Project Definition .....	12
1.1 Objective .....	12
1.2 Project Scope .....	12
<b>Chapter-3: REQUIREMENT ANALYSIS .....</b>	<b>13</b>
1. Functional Requirements .....	14
2. Non-Functional Requirements.....	15
3. Use-Case Specification.....	15
<b>Chapter-4: DESIGN .....</b>	<b>16</b>
1. Database Design .....	17
2. Use Case Diagram .....	18
3. Class Diagram.....	19
4. Sequence Diagram .....	20
5. Collaboration Diagram .....	20
6. Deployment Diagram .....	21
7. Activity Diagram .....	22
8. State Diagram .....	21
<b>Chapter-5: EXPERIMENT AND TESTING.....</b>	<b>23</b>
1. Test cases Developed.....	24
2. Test cases Used.....	24
<b>Chapter-6: CONCLUSION .....</b>	<b>28</b>
1. Problems and Issues in Current System .....	29
2. Future Extensions .....	29
<b>Chapter-7: APPENDIX.....</b>	<b>30</b>
1. Product's Code Snippets.....	31
2. Product's User Interface .....	35
<b>Chapter-8: REFERENCES .....</b>	<b>38</b>

# **Chapter-1: INTRODUCTION**

## 1. Introduction

The following is a final report for the Innovative Project titled “MONNIT: The Server Monitoring Tool” developed by Atharva Pednekar, Jaydeep Patel & Tejas Bhati the students of B. Tech CSIT V semester (2020-24 batch).

Some brief on the background of a Watchdog tool; A Watchdog tool is a product name given to a category of tools and software that serve the core purpose of monitoring single or in some cases multiple systems/paradigms at a time to ensure the quality of performance from these machines.

The application developed in this project “MONNIT” derives it’s inspiration from “Data Dog”, an existing Watchdog Tool.

The current application “Data Dog” is a Web based Service that offers to monitor the Performance of website(s) by monitoring and displaying information like Latency and Error Rate.

Our project “MONNIT” is intended to deliver similar functionality for servers rather than just Web applications, it is also an upgrade over the conventional Task Manager in the sense that the new application will be capable of monitoring the state of resource consumption of remote systems as well and also raise alerts upon reaching levels of Overconsumption and store the occurrence of these alerts in a relational database, a feature that is not available in current conventional Task Manager or DataDog.



## 2. Literature Review

As mentioned above, MONNIT derives its inspiration from “DataDog”, a popular Watchdog tool for server monitoring; some brief on “DataDog” tool.

Watchdog is an algorithmic feature for APM performance and infrastructure metrics that automatically detects potential application and infrastructure issues. It leverages the same seasonal algorithms that power anomalies and dashboards. Watchdog observes trends and patterns in:

APM metrics:

- Hits (request rate)
- Error rate
- Latency

Infrastructure metrics from integrations:

- System, for the Host-level memory usage (memory leaks) and TCP retransmit rate.
- Redis
- PostgreSQL
- NGINX
- Amazon Web Services, for the S3, ELB/ALB/NLB, CloudFront, and DynamoDB Amazon services.
- Alerting

Watchdog looks for irregularities in metrics, like a sudden spike in the hit rate. For each irregularity, the Watchdog page displays a Watchdog alert. Each alert includes a graph of the detected metric irregularity and gives more information about the relevant time frame and endpoint or endpoints. Watchdog automatically monitors data sent by the Datadog Agent or by integrations.

MONNIT is an attempt to develop an application for the purpose of administering a server’s resource consumption metrics while providing the fast and reliable service model that the current application offers as a Free to use service that is much efficient and easy to use and configure than the original application.

To develop the application following methodology was put to use:

- Studying the problem statement to understand the requirement better.
- Studying the documentation/services offered by “DataDog” to understand the basics of resource monitoring.
- Developing the Agile model to be put to use to develop the application in the series of sprints.
- Developing the initial documentation for the application.
- Working in the above mentioned sprint cycles to develop the product.
- Testing in between the sprint cycles and at the final stage of product development to ensure quality.

### **3. Problem Statement**

To develop an application for monitoring the server’s resource consumption and networking resources that will also raise alerts for Over Consumption and store the occurrence of alerts in a relational database.

The application will be a two component product, a Desktop Agent software that will operate upon the server systems to gather the data for Resource Consumption metrics and a Web Application for the user to visualize the obtained data using graphs and Alerts for overconsumption.

## **Chapter-2: THE PROJECT**

# **1. Project Definition**

## **1.1 Objective**

To develop an application for server monitoring that will enable the user to view the current status of system's resource consumption and raise alerts at overconsumption, also it would store the occurrence of alerts in a relational database.

## **1.2 Project Scope**

The scope of this project lies in the industries that rely heavily on the use of servers or environments that require centralized monitoring of multiple systems remotely.

In such use cases it is vital for the System administrator to be able to visualize the current state of system's resource consumption to avoid over usage.

Also, the application could be used by Network admins to remotely monitor the systems

## **Chapter-3: REQUIREMENT ANALYSIS**

## 1. Functional Requirements

#	Title of System Feature	Description	Priority	Functional Requirements
1	Register at the Agent.	Register User and systems via the agent	High	Register the user with Password & Email ID along with System ID.
2	Login at the Agent	Login user at the Agent	High	Login the user via authenticating the Login request against the database.
3	Run at the Agent	Runs the agent to run the CMD commands and produce the data for Resource consumption status and alerts.	High	Execute the CMD commands for the resource consumption data and alerts.
4	Stop at the Agent	Stops the agent to run the CMD commands and produce the data for Resource consumption status and alerts.	High	Stops the Agent.
5	Login at the Website	Login the user using Email ID, password & System ID (Optional)	High	Authenticate the user against the users present in the database.
6	Server Network Metrics	Display the network metrics of the selected system.	High	Display the server network metric's graph based on the data from Database that is generated by the Agent.
7	Resources Metrics	Display the resources metrics of the selected system	High	Display the Resource metric's graph based on the data from Database that is generated by the Agent.
8	Error logs	Display the logs of the selected system.	High	Display the Error Logs stored in the Database that is generated by the Agent.
9	Logout	Logout the user from the website/Session time out	High	Terminates the session.

## 2. Non-Functional Requirements

#	Title of System Feature	Description	Priority	Functional Requirements
1	Consistent User Interface	The Application should have a consistent User Interface(UI).	High	Media Queries for responsive design of the Web Application.
2	Appropriate Graphs	The plotting of graphs should be appropriate for the of resource (Ex: Pie Chart for showing the consumption of Memory).	High	Appropriate command of Graph plotting in Chart.Js for each type of graph.
3	Being able to see the Occurrence of alerts.	The user should be able to view the database of Raised Alerts to observe if there is a pattern of occurrence in the past alerts.	High	Running the command to enter the data for overconsumption into the Relational database.

## 3. Use-Case Specification

The application MONNIT requires the user persona of a Network/Resource administrator.

Such users are regularly required to monitor the current status of consumption of resources to avoid overconsumption which can prove damaging to systems' health.

## **Chapter-4: DESIGN**



# 1. Database Design

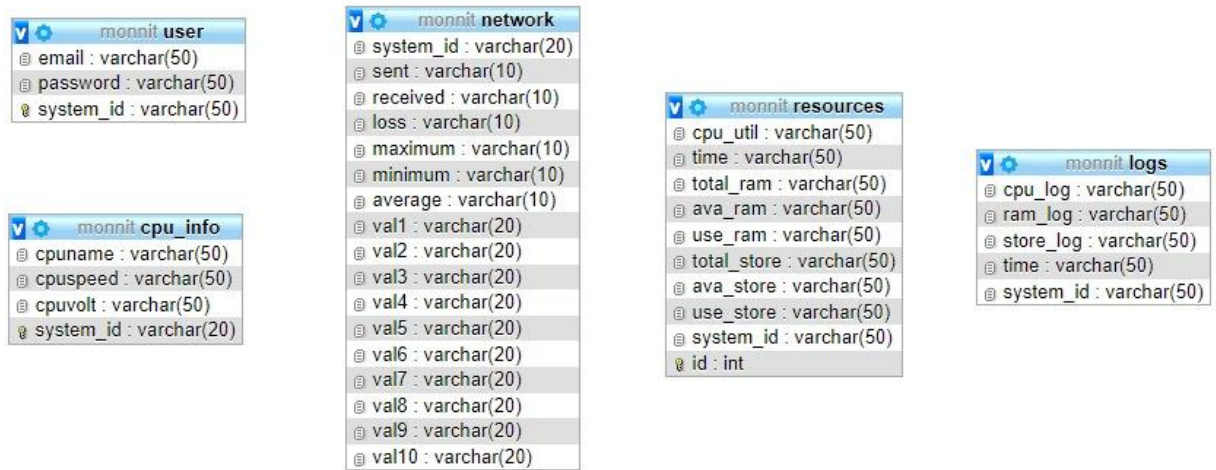


Diagram 4.1: Database Schema for the MONNIT Application

The database for MONNIT exists as a collection of the Data Tables:

- User: Email-Id, Password & System ID which are of type Varchar(50)
- CPU Info: CPU name, CPU speed, CPU Voltage & System Id which are of type Varchar(50)
- Network: System Id, Sent, Received & Lost data Packets along with Min. Max. & Avg. latency which are of type Varchar(50)
- Resources: CPU\_util, Time, Total, Available & Used RAM along with Total, Available & Used Storage which are of type Varchar(50)
- Logs: Which include Error Logs for CPU, RAM & Storage consumption along with time & System ID which are of the type Varchar(50)

## 2. Use Case Diagram

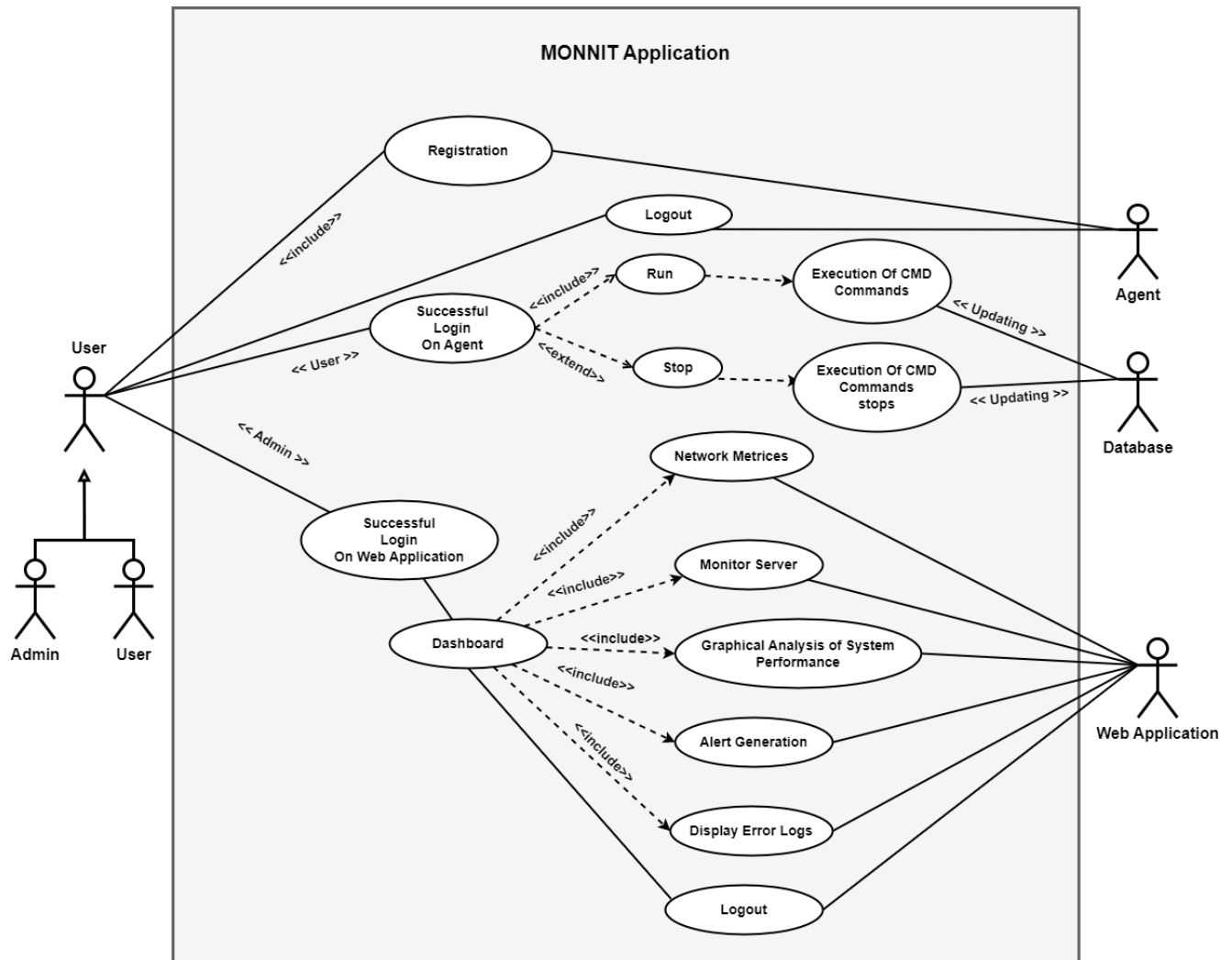


Diagram 4.2: Use Case Diagram for the MONNIT application

### 3. Class Diagram

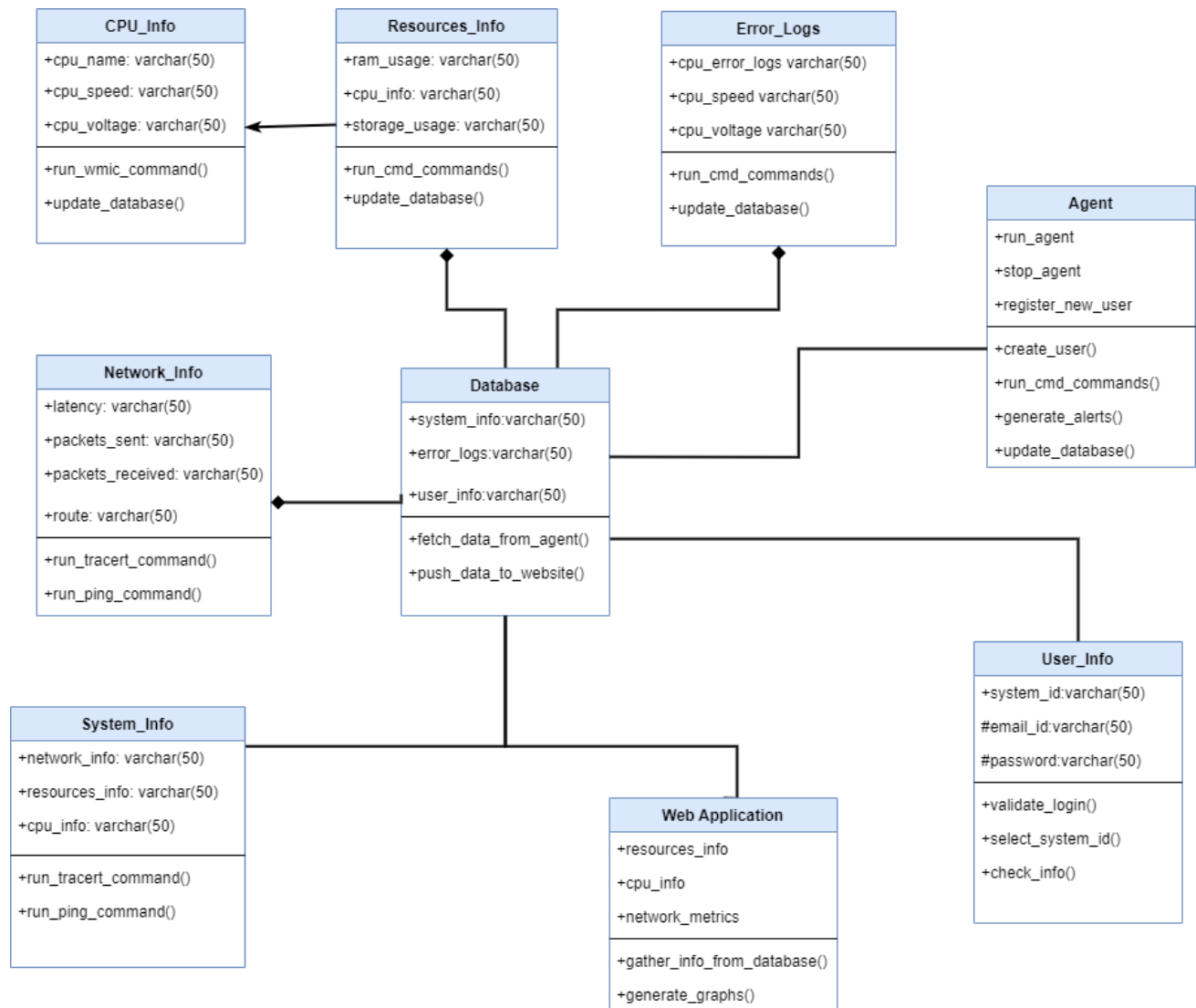


Diagram 4.3: Class Diagram for the application

## 4. Sequence Diagram

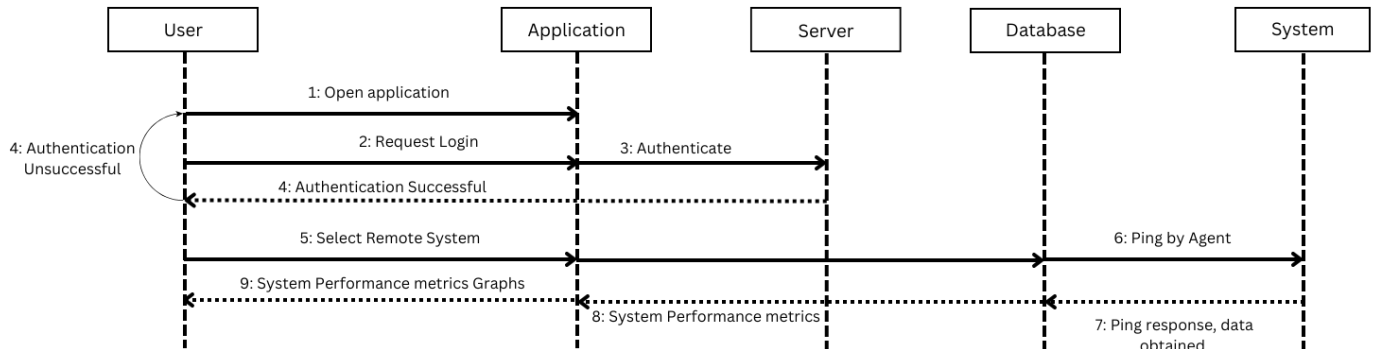


Diagram 4.4: Sequence Diagram for the application

## 5. Collaboration Diagram

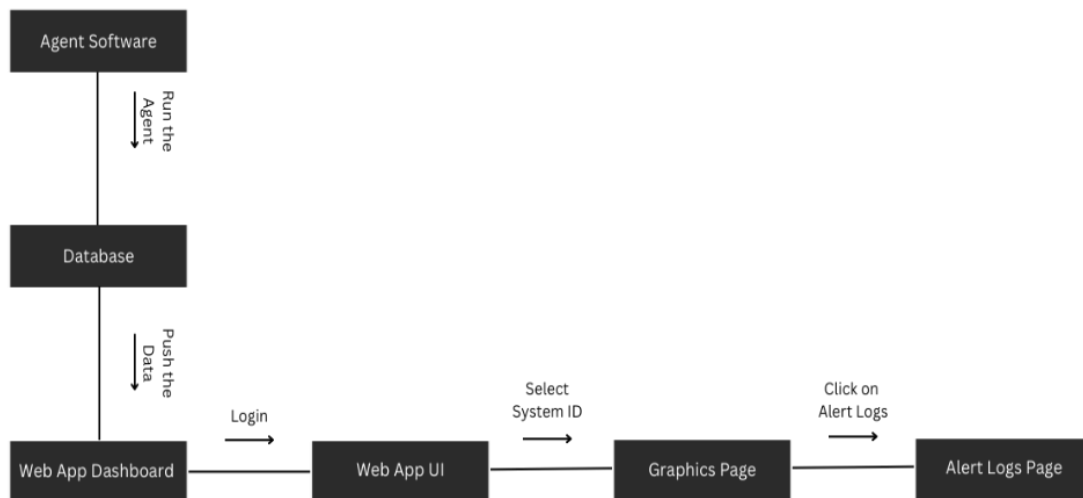


Diagram 4.5: Collaboration Diagram for the application

## 6. Deployment Diagram

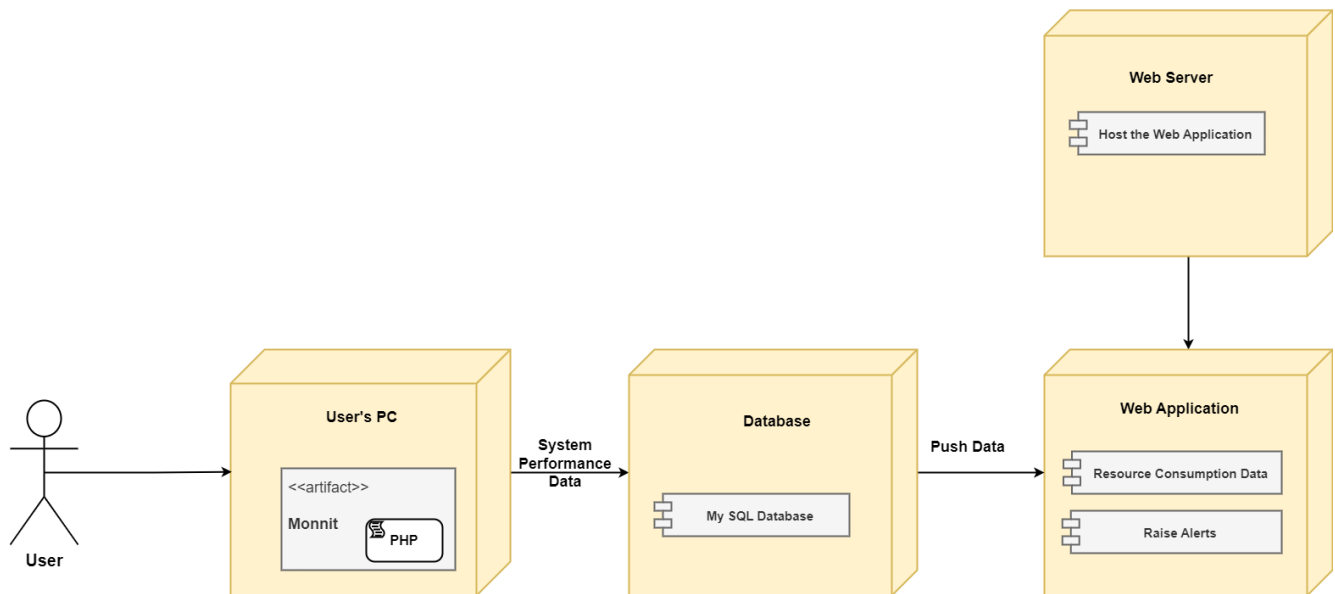


Diagram 4.6: Deployment Diagram for the application

## 7. State Diagram

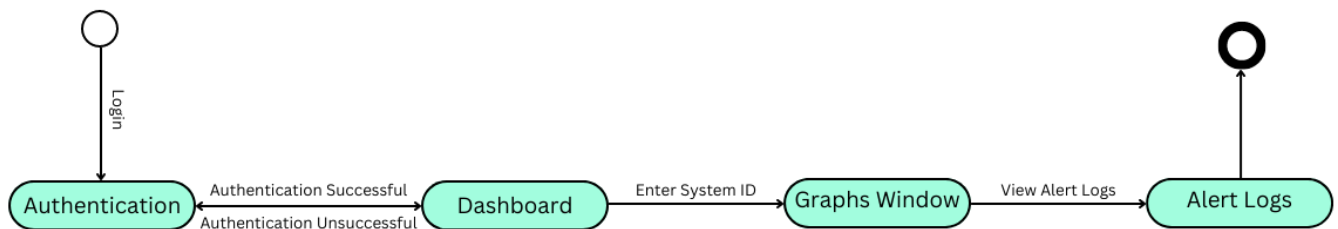


Diagram 4.8: State Diagram for the application

## 8. Activity Diagram

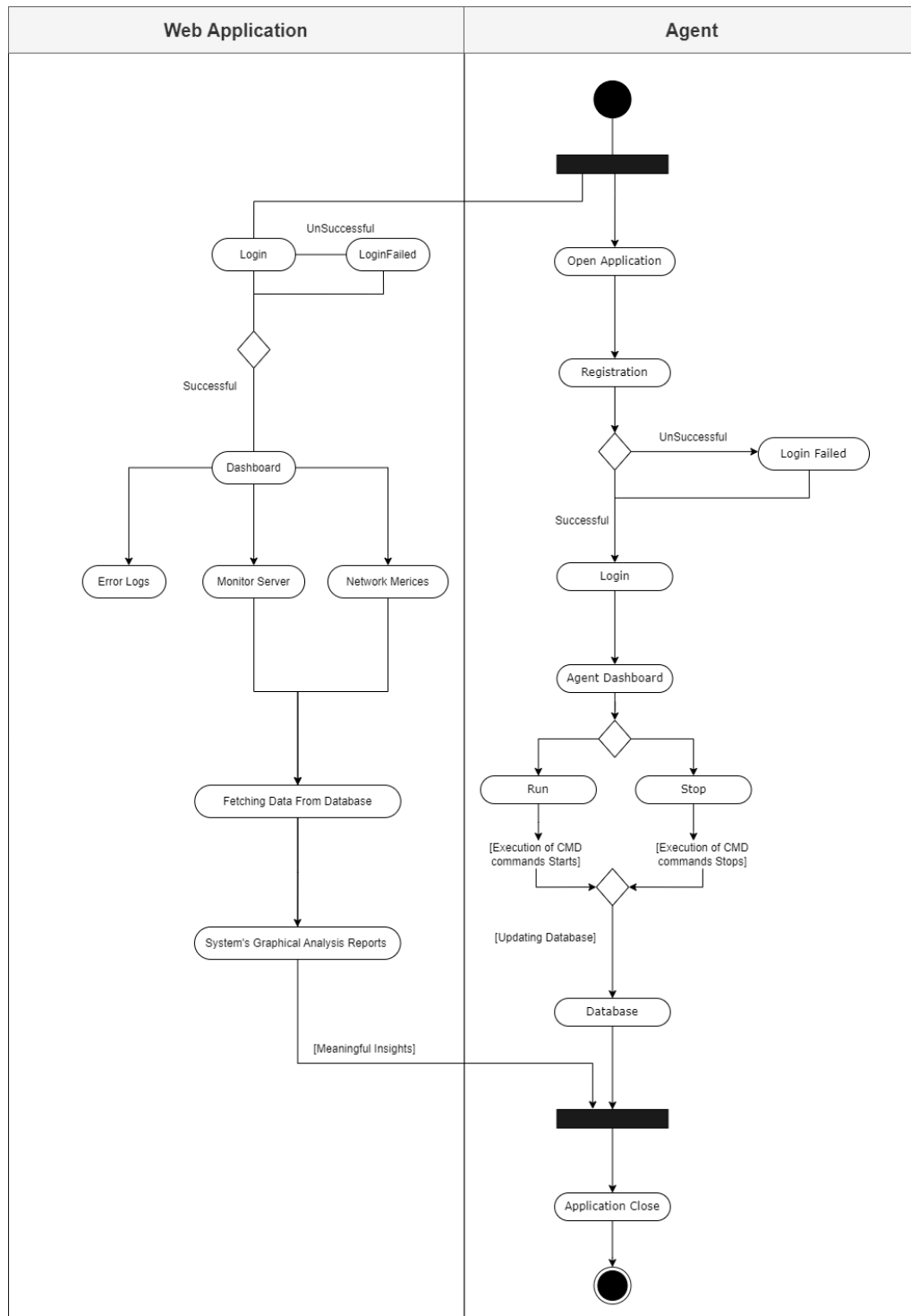


Diagram 4.7: Activity Diagram for the application

## **Chapter-5: EXPERIMENT AND TESTING**

## **1. Test cases Developed**

White box Testing Used.

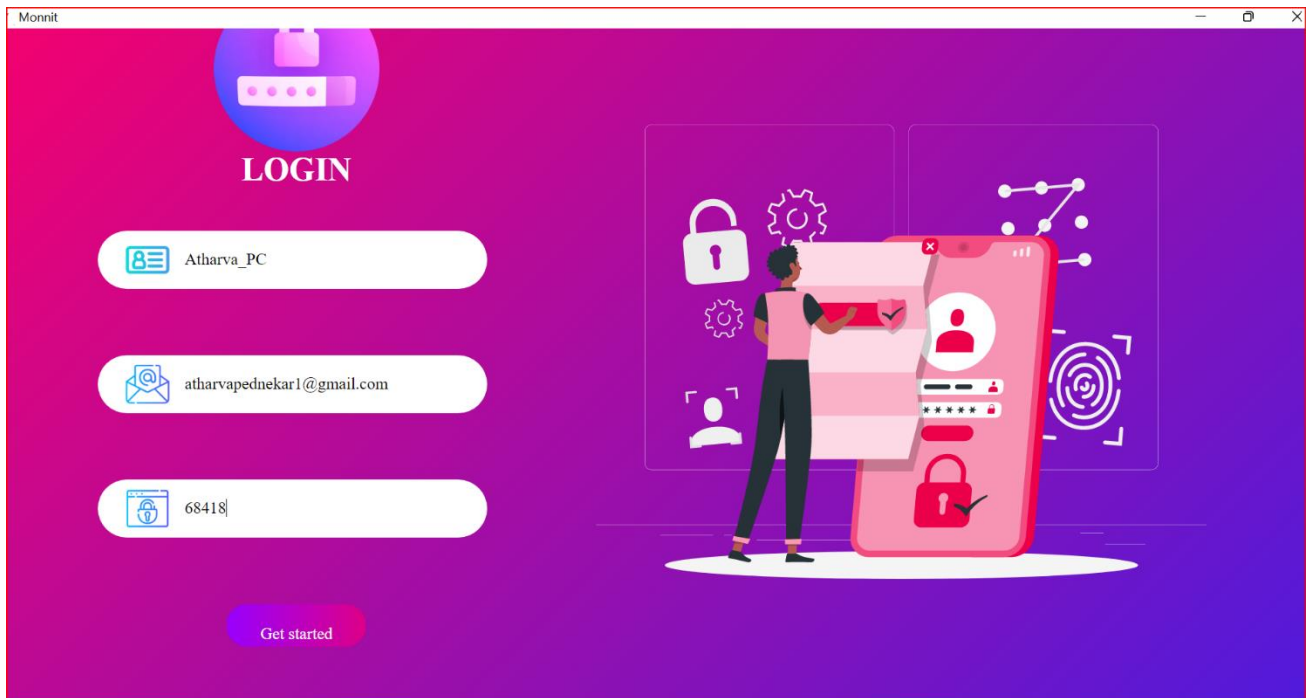
White box testing techniques analyze the internal structures the used data structures, Internal design, code structure, and the working of the software rather than just the Functionality as in black box testing. It is also called Glass Box testing or Clear Box testing or structure testing.

- Overloading resource for testing the alert raising system.
- Checking the responsivity of the web app.
- Checking the Database for the alerts that are being stored.
- Checking the Register function at the Agent.
- Checking the Run/Stop Agent at the Agent.
- Checking the Alert Logs to ensure all stored alerts are being displayed accurately

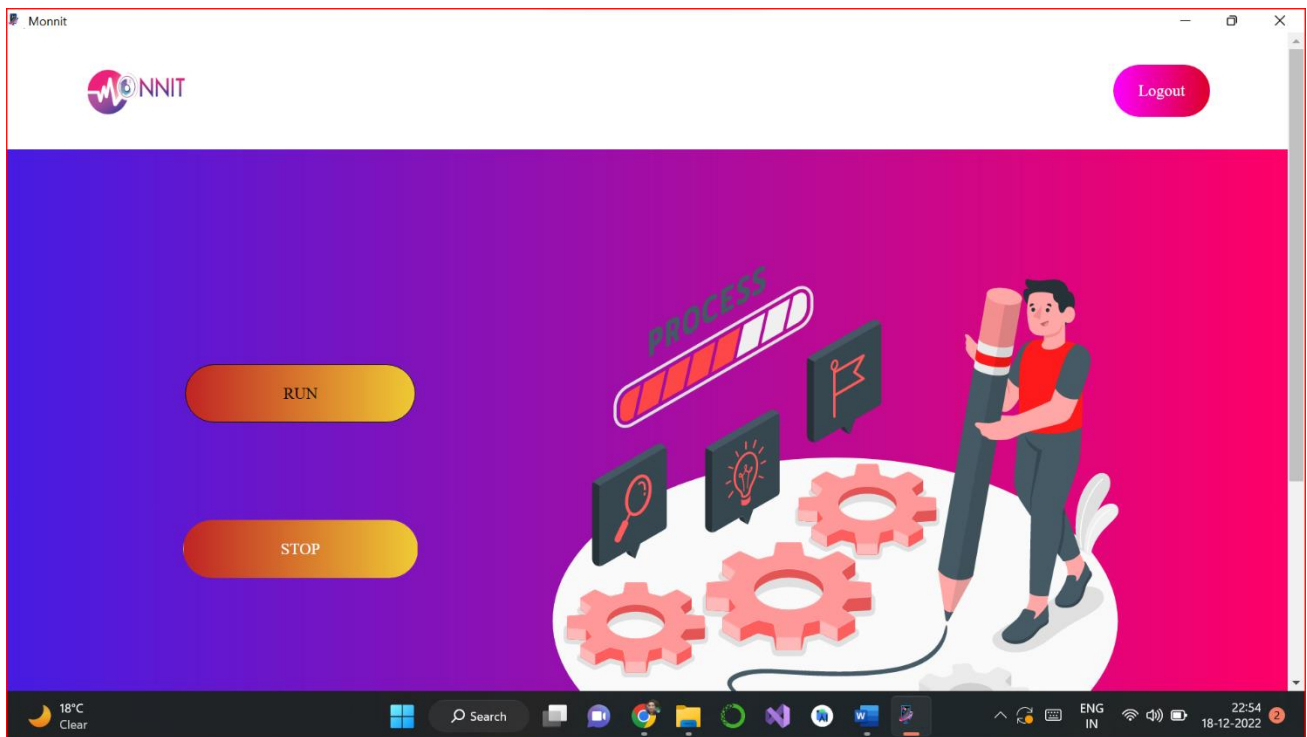
## **2. Test cases used**

- Overloading resource for testing the alert raising system.
- Checking the Database for the alerts that are being stored.
- Checking the Register function at the Agent.
- Checking the Run/Stop Agent at the Agent.
- Checking the Alert Logs to ensure all stored alerts are being displayed accurately





*Diagram 5.1: Test Case phase-1*



*Diagram 5.2: Test Case phase-2*

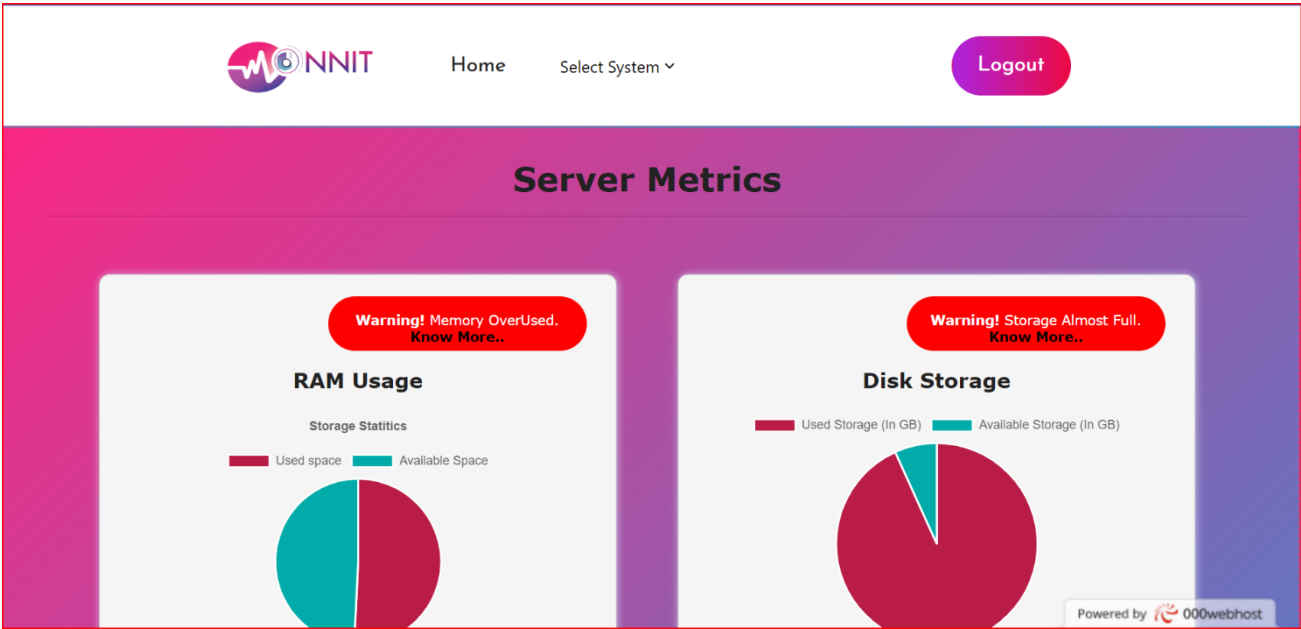


Diagram 5.3: Test Case phase-3

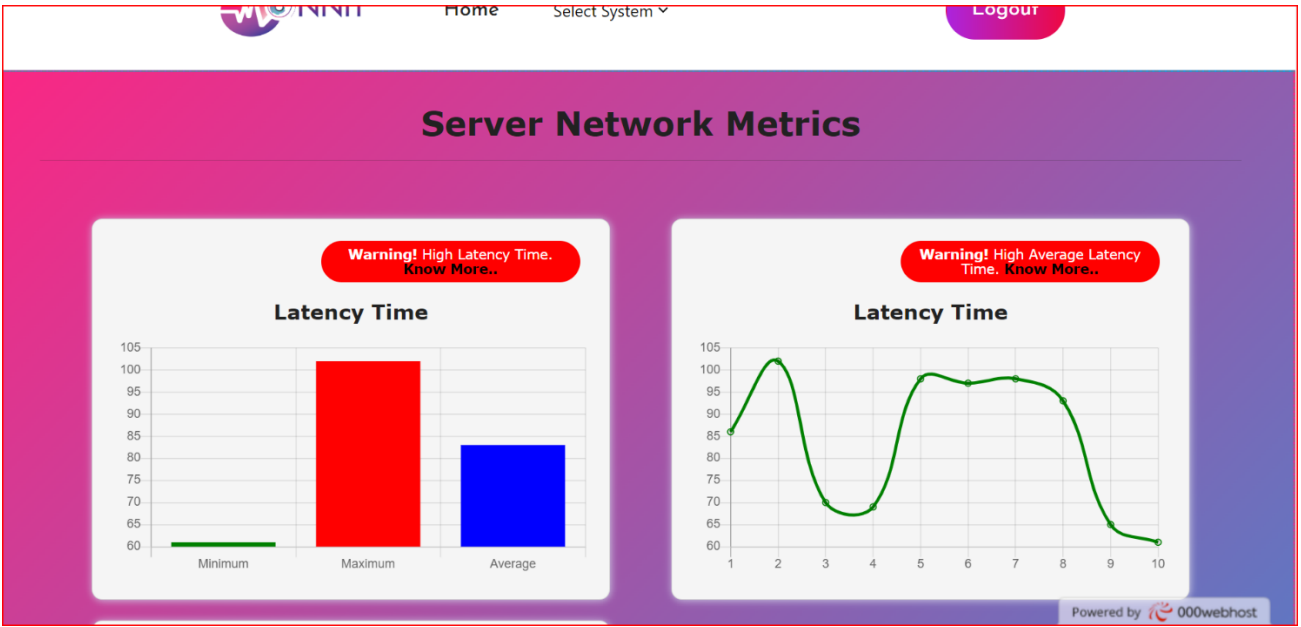


Diagram 5.4: Test Case phase-4

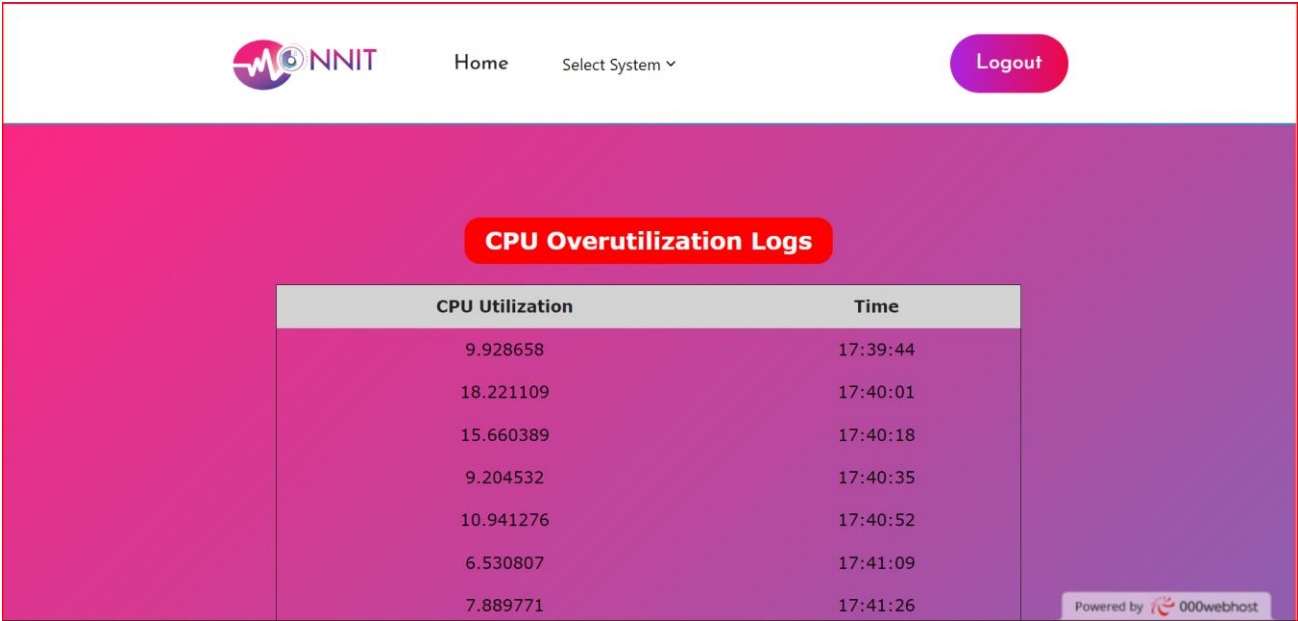


Diagram 5.5: Test Case phase-5

## **Chapter-6: CONCLUSION**

## **1. Problems and Issues in Current System**

It is truly said “There is no perfection, only beautiful versions of brokenness.”, similar to us the developers, the project has some imperfections we came across and would like you to know them, some of the imperfections we came across are as follows:

- Currently the application lack security fixes.
- Currently the application does not allow the user to download the Report sheet of the data that is being monitored, a feature we would surely like to append in the future versions of the project.

## **2. Future Extensions**

Just like us the human beings, our work evolves over time and graces the beauty of perfection. Sticking on this note here are some additions we would love to make to this project over the course of time:

- Add the feature to monitor multiple systems at a time.
- Develop a software version of the application to be used as a software of its kind.
- Add a feature to download the Report of system analytics done so far.
- Enhance the documentation to be even more user friendly.
- To Make the Application Platform Independent .
- To Scale the Availability of Web Application.

## **Chapter-7: APPENDIX**

## 1. Product's Code Snippets

```
26
27 }
28
29 | //print_r($CpuUtilization);
30 $query="(SELECT * FROM resources ORDER BY id DESC LIMIT 1) ORDER BY id ASC";
31 $result = mysqli_query($conn, $query);
32
33 while ($row = mysqli_fetch_assoc($result)) {
34
35     $use_ram=$row['use_ram'];
36     $total_ram=$row['total_ram'];
37     $ava_ram=$row['ava_ram'];
38     $use_store=$row['use_store'];
39     $total_store=$row['total_store'];
40     $ava_store=$row['ava_store'];
41 }
42 $query="SELECT * FROM network";
43 $result = mysqli_query($conn, $query);
44
45 while ($row = mysqli_fetch_assoc($result)) {
46
47     $sent=$row['sent'];
48     $received=$row['received'];
49     $loss=$row['loss'];
50     $maximum=$row['maximum'];
51     $minimum=$row['minimum'];
52     $average=$row['average'];
53 }
54
55
```

*Diagram 7.1: Application Code for the desktop agent*

```

<?php require "runstop.php"; ?>

<?php
session_start();
require "conn.php";
// Code to fetch RAM Usage
// Command to get available Physical Memory

$cmd1 = shell_exec("systeminfo | find \"Available Physical Memory\"");

// Command to get Total Physical Memory
$cmd2 = shell_exec("systeminfo | findstr \"Total Physical Memory\"");
$pattern = '!\\d+!';

if (preg_match_all($pattern, $cmd1, $matches)) {
    $elems = count($matches[0]);
    if($elems == 2){
        $v1 = $matches[0][0];
        $v2 = $matches[0][1];
        $freeSpace = $v1 . $v2;
    }
    else{
        $freeSpace = $matches[0][0];
    }
}

if (preg_match_all($pattern, $cmd2, $matche)) {

    $v3 = $matche[0][0];
    $v4 = $matche[0][1];

    $totalSpace = $v3 . $v4;

    $usedSpace = $totalSpace - $freeSpace;
}

// Code to fetch Disk Storage
// Command to Get Full Disk Storage
$st = shell_exec("wmic diskdrive get size");

// Command to Get Free Disk Storage
$sk = shell_exec("wmic logicaldisk get freespace");

$pattern = '!\\d+!';

if (preg_match_all($pattern, $st, $matches)) {
    $TotalStorage = $matches[0][0]; // Free Storage -- //38 for sent reciev loss data pack multiple line
    $TotalStorage = $TotalStorage / (1000 * 1000 * 1000);
}

if (preg_match_all($pattern, $sk, $matche)) {
    $availableStorage = $matche[0][0]; //38 for sent reciev loss data pack multiple line
    $availableStorage = $availableStorage / (1024 * 1024 * 1024);
}

$usedStorage = $TotalStorage - $availableStorage;

```

*Diagram 7.2: Backend of the Agent*



```

// To fetch any critical condition of Server (for storing in alert table)
$CpuAlert = 0;
$RamAlert = 0;
$diskStorageAlert = 0;

if ((int) $CpuUtilVal1 > 90) {
    $CpuAlert = 1;
}
if ($usedSpace > (0.9 * $totalSpace)) {
    $RamAlert = 1;
}
if ($usedStorage > (0.9 * $TotalStorage)) {
    $diskStorageAlert = 1;
}

// Inserting Data in Database
$system_id=$SESSION['system_id'];
echo $system_id;
$query = " INSERT INTO resources(cpu_util, time, total_ram, ava_ram, use_ram, total_store, ava_store, use_store, system_id) values('$CpuUtilVal1',
'$t1','$totalSpace','$freeSpace','$usedSpace','$TotalStorage','$availableStorage','$usedStorage','$system_id' )";

if (mysqli_query($conn, $query)) {
} else {
    echo 'Unable to Not Inserted';
}

$query1 = "update network set
system_id=$system_id,sent=$sent',received='$received',loss='$lost',maximum='$maximum',minimum='$minimum',average='$average',val1='$val1',val2='$val1',val3='$val2',val4='$val3',val5='$val4',val6='$val5',
val7='$val6',val8='$val7',val9='$val8',val10='$val9' where system_id='$system_id'";

if (mysqli_query($conn, $query1)) {
} else {
    echo 'Unable to Not Inserted';
}

if ($CpuAlert == 1 || $RamAlert == 1 || $diskStorageAlert == 1) {
    $query3 = "INSERT INTO logs (cpu_log, ram_log, store_log, time, system_id) values('$CpuUtilVal1','$usedSpace','$usedStorage', '$t1','$system_id' )";

    if (mysqli_query($conn, $query3)) {
    } else {
        echo 'Unable to Not Inserted';
    }
}

// To execute code again and again
echo("<meta http-equiv='refresh' content='1'>");

```

*Diagram 7.3: Backend of the Agent continued*

```

1  <?php
2
3  // Connection to Database (Server Monitor)
4
5  $host = "mysql-100425-0.cloudclusters.net";
6  $username = "admin";
7  $pass = "9AAL1aRk";
8  $db = "monnit";
9  $dbServerPort = 10126;
10 $conn = mysqli_connect($host, $username, $pass, $db,$dbServerPort);
11
12 ▼ if(!$conn){
13     echo "Unable to Connect to Database";
14 }
15 else{
16 }
17 ?>

```

*Diagram 7.4: Code for Connection*

```

k?php

require "conn.php";
$drop=$_POST['drop'];
$query1 = "select * from logs where system_id='$drop'";

$CpuUtilizationLogs = mysqli_query($conn, $query1);
$cpu_util_rows = mysqli_num_rows($CpuUtilizationLogs);

$CpuLogTableBody = '';
$MemoryOverUsedTableBody = '';
$DiskStorageTableBody = '';

while ($row = mysqli_fetch_assoc($CpuUtilizationLogs)) {
    global $tableBody;
    $col1 = $row["cpu_log"];
    $col2 = $row["time"];

    $currRowData = "<tr class='table-secondary'><td class='table-secondary'>$col1</td><td class='table-secondary'>$col2</td></tr>";
    $CpuLogTableBody = $CpuLogTableBody . $currRowData;
}

$MemoryOverUsedLogs = mysqli_query($conn, $query1);
$memory_used_rows = mysqli_num_rows($MemoryOverUsedLogs);
while ($row = mysqli_fetch_assoc($MemoryOverUsedLogs)) {
    global $tableBody;

    $col1 = $row["ram_log"];
    $col2 = $row["time"];

    $currRowData = "<tr class='table-secondary'><td class='table-secondary'>$col1</td><td class='table-secondary'>$col2</td></tr>";
    $MemoryOverUsedTableBody = $MemoryOverUsedTableBody . $currRowData;
}

$DiskStorageFullLogs = mysqli_query($conn, $query1);
$disk_storage_rows = mysqli_num_rows($DiskStorageFullLogs);
while ($row = mysqli_fetch_assoc($DiskStorageFullLogs)) {
    global $tableBody;

    $col1 = $row["store_log"];
    $col2 = $row["time"];

    $currRowData = "<tr class='table-secondary'><td class='table-secondary'>$col1</td><td class='table-secondary'>$col2</td></tr>";
    $DiskStorageTableBody = $DiskStorageTableBody . $currRowData;
}
}
?>
<!DOCTYPE html>

```

Diagram 7.5: Code for Error Logs

```

k?php

$drop=$_POST['drop']; echo $drop;
// Connection to Database (Server Monitor)
$host = "mysql-99212-0.cloudclusters.net";
$username = "admin";
$password = "75QnFlt1";
$db = "monniti";
$dbServerPort = 10007;
$conn = mysqli_connect($host, $username, $password, $db,$dbServerPort);

if(!$conn){
    echo "Unable to Connect to Database";
}
else{
}
$CpuUtilization=array();
$t=array();

$query=" (SELECT * FROM resources where system_id='$drop' ORDER BY id DESC LIMIT 5) ORDER BY id ASC";
$result = mysqli_query($conn, $query);

while ($row = mysqli_fetch_assoc($result)) {
    array_push($CpuUtilization, (float) $row["cpu_util"]);
    array_push($t, (string)($row["time"]));
}

//print_r($CpuUtilization);
$query="(SELECT * FROM resources where system_id='$drop' ORDER BY id DESC LIMIT 1) ORDER BY id ASC";
$result = mysqli_query($conn, $query);

while ($row = mysqli_fetch_assoc($result)) {

    $use_ram=$row['use_ram'];
    $total_ram=$row['total_ram'];
    $ava_ram=$row['ava_ram'];
    $use_store=$row['use_store'];
    $total_store=$row['total_store'];
    $ava_store=$row['ava_store'];
}
$query="SELECT * FROM network where system_id='$drop'";
$result = mysqli_query($conn, $query);

while ($row = mysqli_fetch_assoc($result)) {

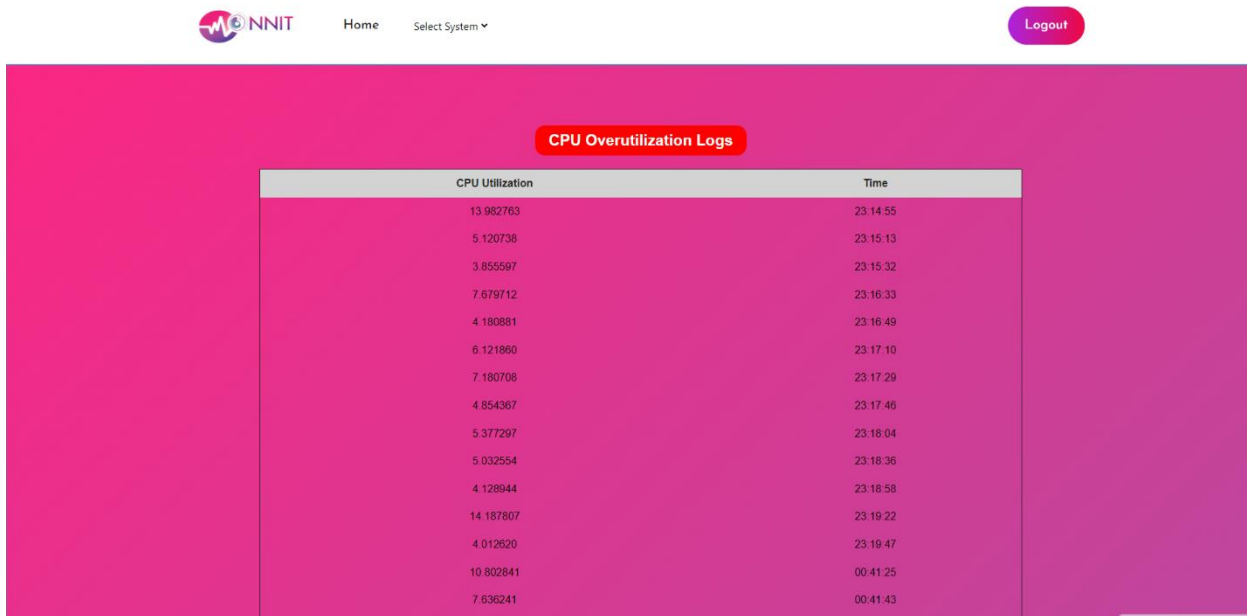
    $sent=$row['sent'];
    $received=$row['received'];
    $loss=$row['loss'];
    $maximum=$row['maximum'];
    $minimum=$row['minimum'];
    $average=$row['average'];

    $val1 = $row['val1']; //latency value
    $val2 = $row['val2'];
    $val3 = $row['val3'];
    $val4 = $row['val4'];
    $val5 = $row['val5'];
    $val6 = $row['val6'];
}

```

Diagram 7.6: Code for Graph Generator

## 2. Product's User Interface



The screenshot displays the MONNIT web application interface. At the top, there is a navigation bar with the MONNIT logo, a 'Home' link, a 'Select System' dropdown menu, and a 'Logout' button. The main content area features a red header for 'CPU Overutilization Logs'. Below this header is a table with two columns: 'CPU Utilization' and 'Time'. The table contains 15 rows of data, showing various CPU utilization values and their corresponding times.

CPU Utilization	Time
13.982763	23:14:55
5.120738	23:15:13
3.855597	23:15:32
7.679712	23:16:33
4.180881	23:16:49
6.121860	23:17:10
7.180708	23:17:29
4.854367	23:17:46
5.377297	23:18:04
5.032554	23:18:36
4.128944	23:18:58
14.187807	23:19:22
4.012620	23:19:47
10.802841	00:41:25
7.636241	00:41:43

Diagram 7.7: User Interface for Error Logs

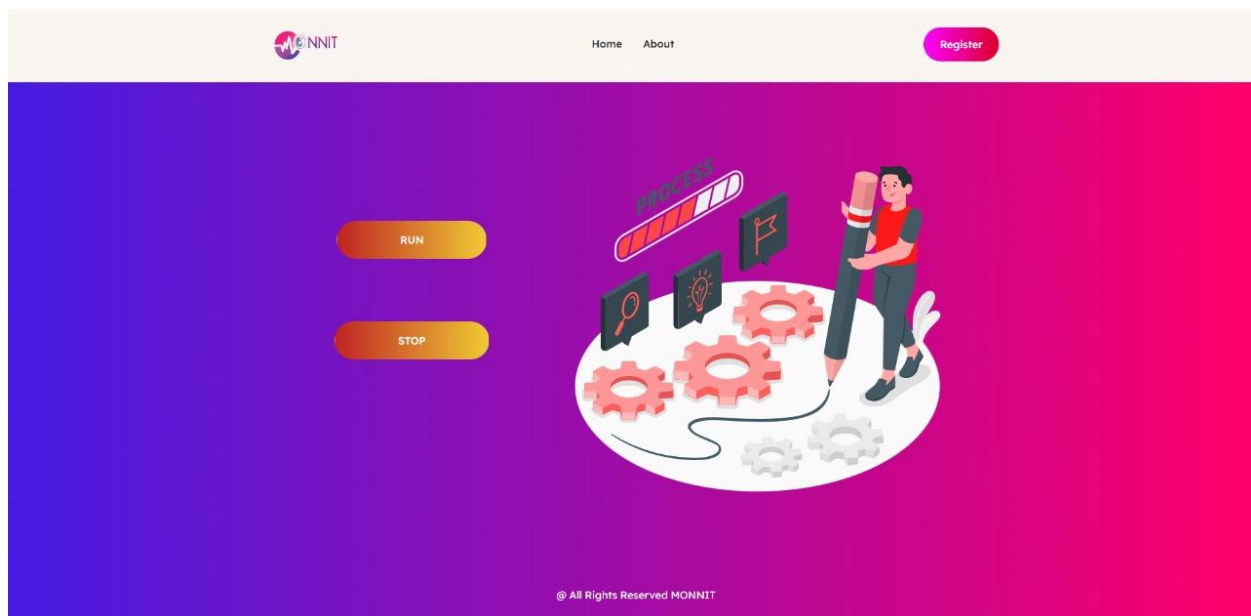
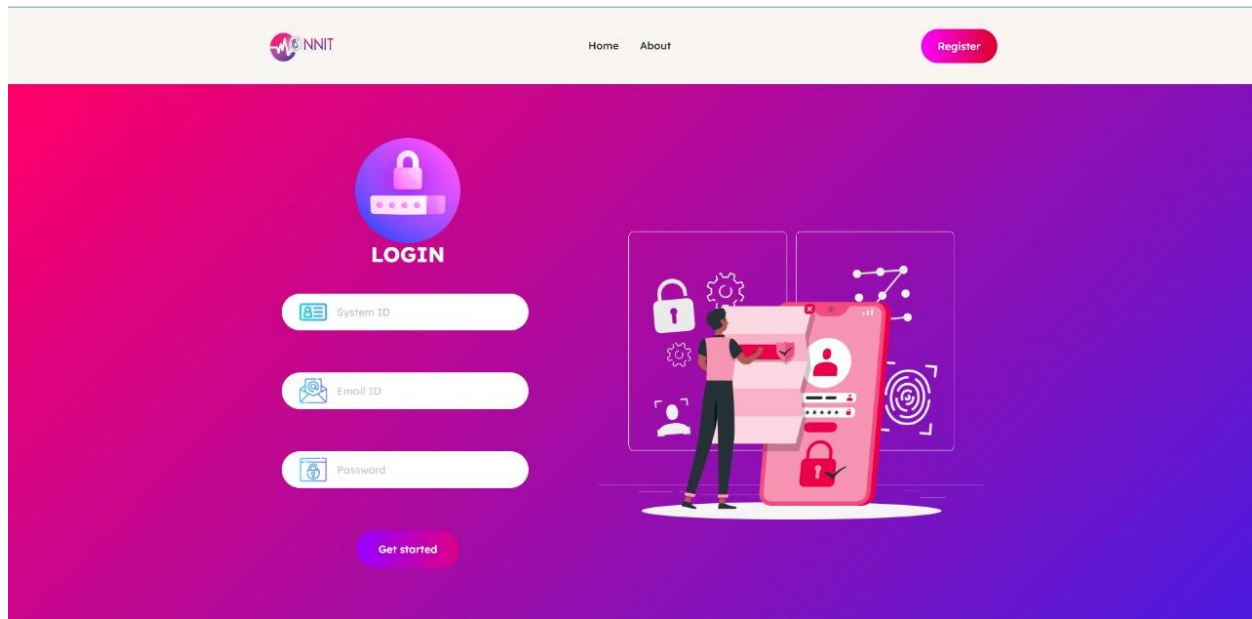
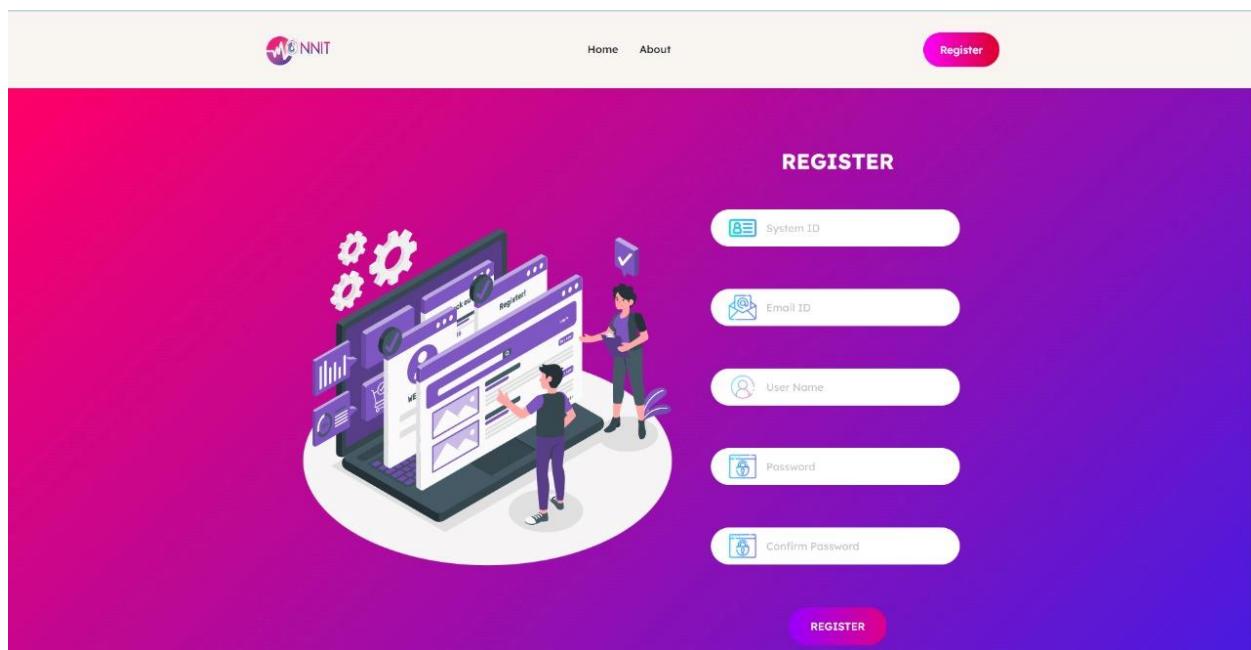


Diagram 7.8: Dashboard of the Agent



*Diagram 7.9: Login screen for the Agent*



*Diagram 7.10: Register screen for the Agent*

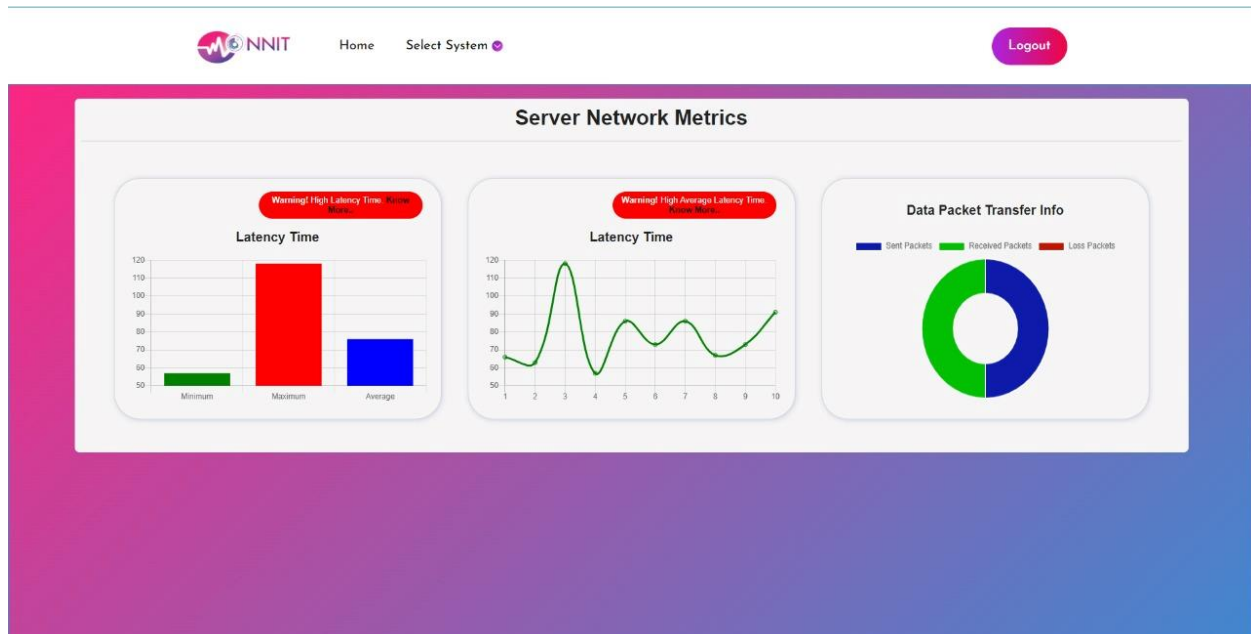


Diagram 7.11: Performance Metrics graphs

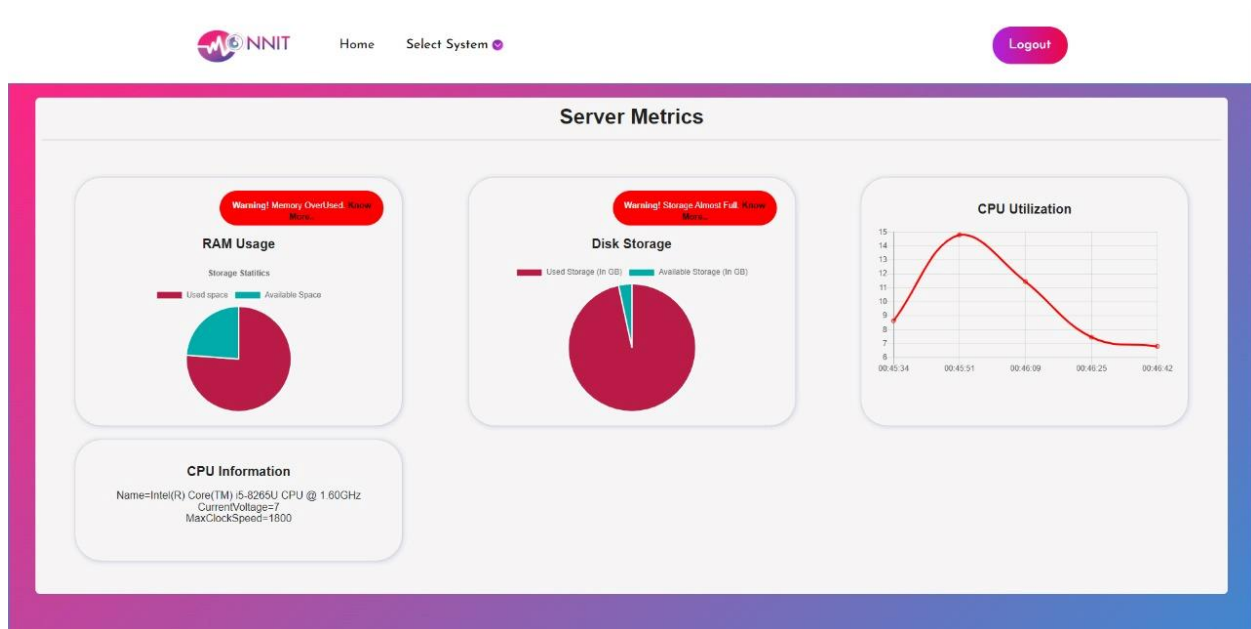


Diagram 7.12: Performance Metrics graphs continued

## **Chapter-8: REFERENCES**

- **Documentation: DataDog Server monitoring tool**

[https://www.datadoghq.com/product/?\\_gl=1\\*1uldps0\\*\\_ga\\*MTQwNjg1NjQ4MC4xNjU0NTE1MTUw\\*\\_ga\\_KN80RDFSQK\\*MTY3MTQzNjE1My42LjEuMTY3MTQzNjE1My40OS4wLjA](https://www.datadoghq.com/product/?_gl=1*1uldps0*_ga*MTQwNjg1NjQ4MC4xNjU0NTE1MTUw*_ga_KN80RDFSQK*MTY3MTQzNjE1My42LjEuMTY3MTQzNjE1My40OS4wLjA).

- **DataDog Tool**

<https://docs.datadoghq.com/watchdog/>

- **Draw.io for UML Diagrams**

<https://app.diagrams.net/#G1MWM92HZspGT8cIAzxqddw9qtl89TFUcD>

- **Chart.JS (JS Library) for creating graphs**

<https://www.chartjs.org/>