Software Design Specification

for

Web Application for NIC

Version 1.0 approved

Ronald Tony (U101116FCS103)

Sourav Upadhya (U101116FCS134)

Saloni Jain (U101116FCS107)

Sourish Das (U101116FCS135)

Shubhangi (U101116FCS127)

Sabyasachi Mishra (U101116FCS104)

NIIT University

13th November 2018

Table of Contents

1. Introduction

It is a document which will provide an overview of the software designs being implemented in the project including use case models, sequence diagrams, and other supporting requirement information.

1.1 Purpose of this document

This document will provide a detailed description of various UML design components like use-case diagram, state diagram and sequence diagram instilled in the project. The various interactions between the components are outlined at the end of this document.

1.2 Scope of the development project

Project aims to provide web application to manage the safe and secure wireless connection provided by the organization.

BENEFITS:

- Secure login.
- Easy management of devices accessing the wireless connection.

1.3 Definitions, acronyms, and abbreviations

- IEEE: Institute of Electrical and Electronics Engineers
- SDS: Software Design Specification
- UML: Unified Modelling Language

1.4 References

- R. S. Pressman, Software Engineering: A Practioner's Approach, 5th Ed, McGraw-Hill, 2001
- Software Engineering: Ian Sommerville, 9th Ed
- IEEE SDS template

1.5 Overview of document

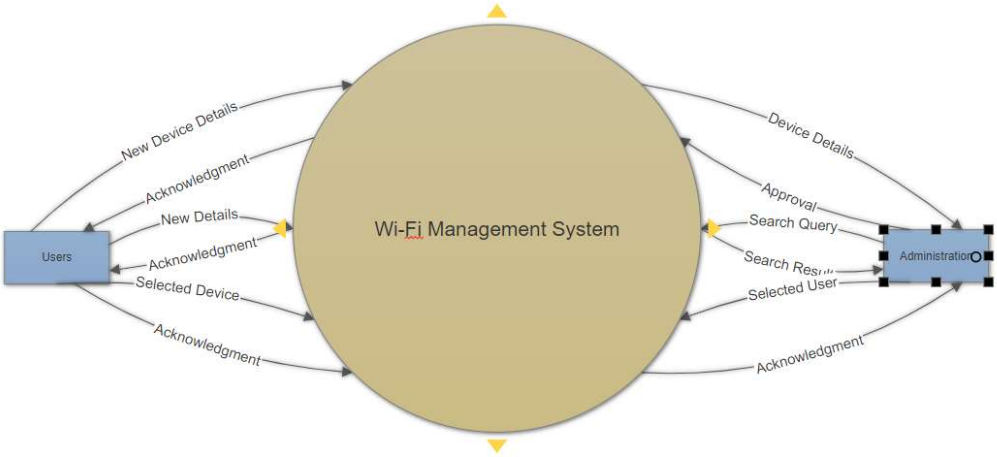
This SDS document is consist of seven section with various sub-sections. The sections of Software Design Document are:

- **Introduction**- This section recount about the document, purpose of this document, scope of the development project, definitions, acronyms and abbreviations, references related to design issues are used in the document.
- **Logical Architecture**- Depicts about logical architecture(class diagram sequence diagram, etc) and it's components.
- **Execution Architecture**- Define the runtime environment, processes, deployment view also tells about the reusability and relationships to other products.
- **Design Decisions and Trade-offs**- This section will help the reader understand the design that we are using. Also the reasons why few decisions were made over other alternatives.

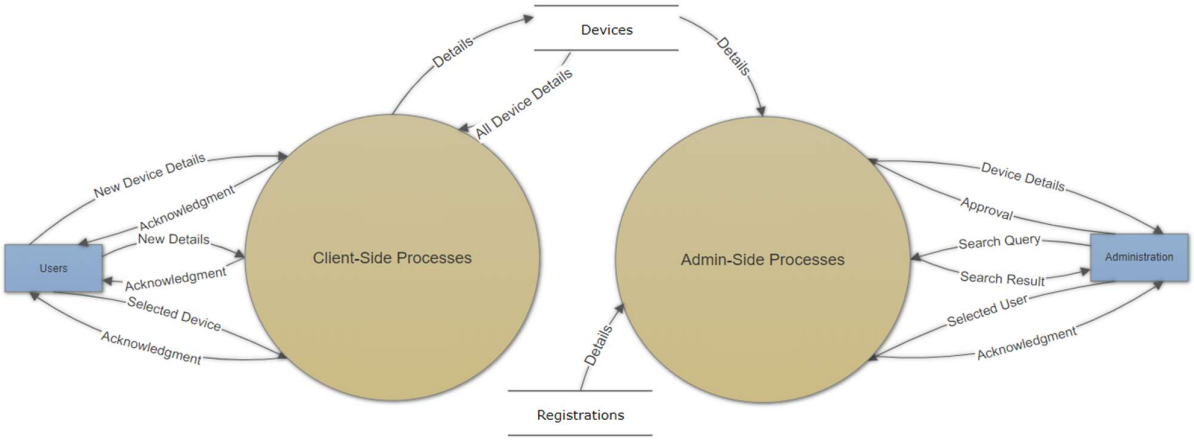
2. Logical Architecture (Data Flow Diagram, Sequence Diagram, State Diagram)

Data Flow Diagrams

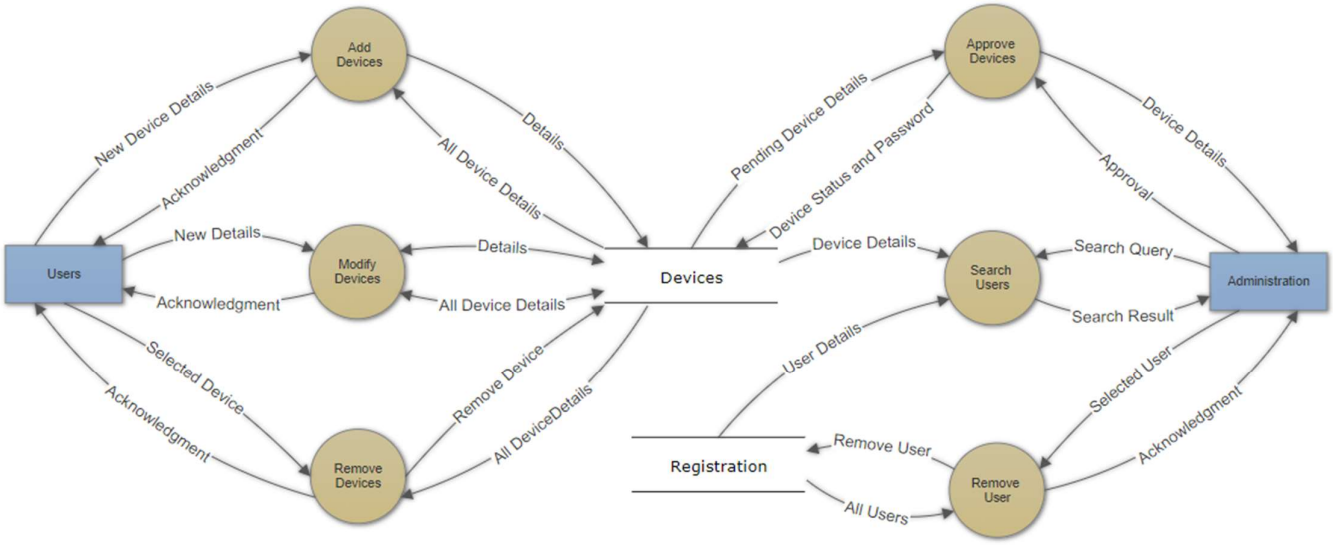
Level – 0 Diagram



Level – 1 Diagram

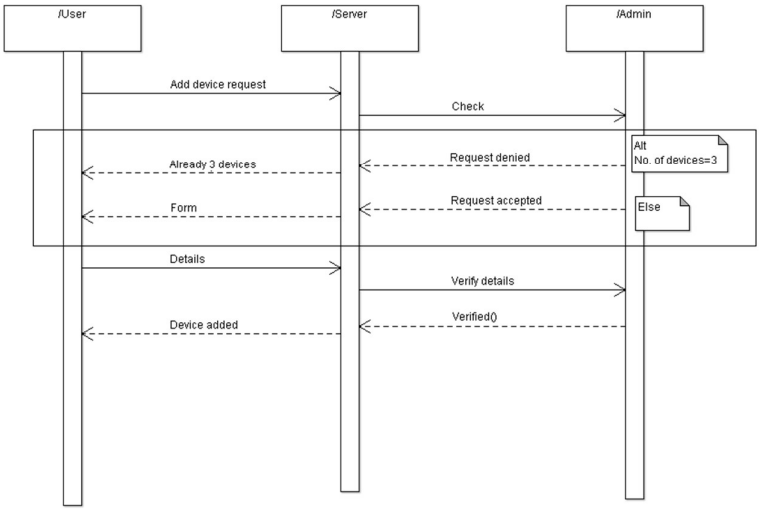


Level – 2 Diagram

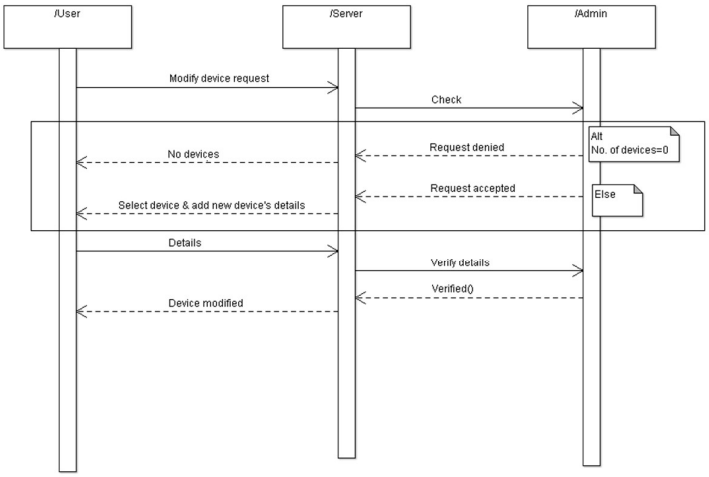


Sequence Diagrams

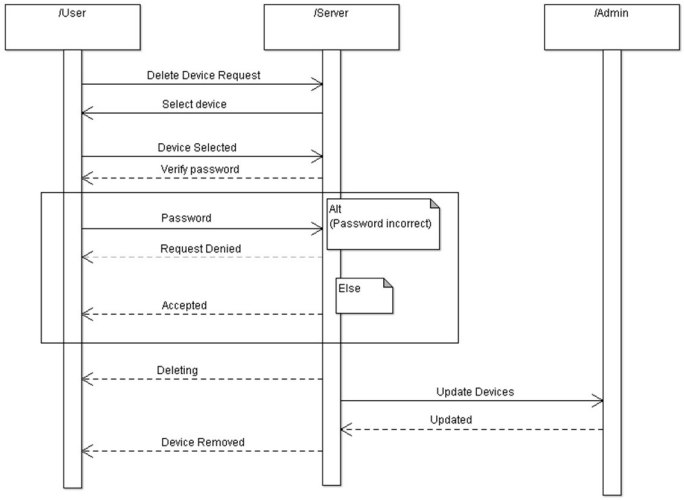
Sequence Diagram: Client Side- Add Device



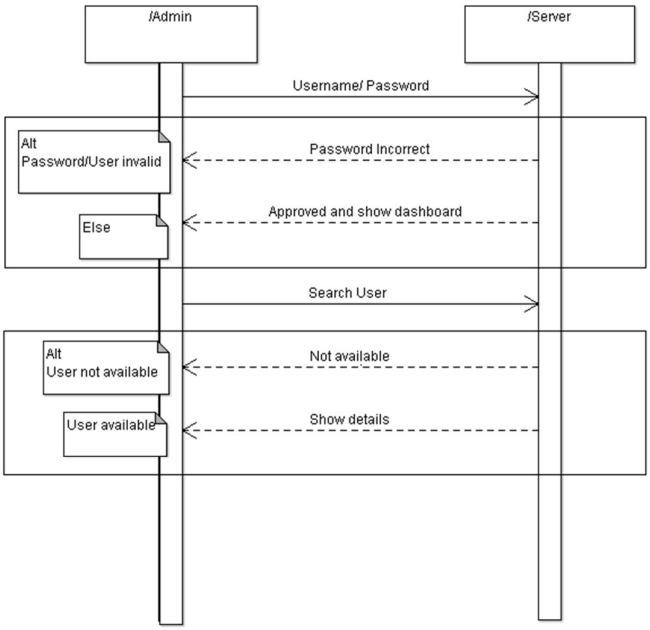
Sequence Diagram: Client Side- Modify Device



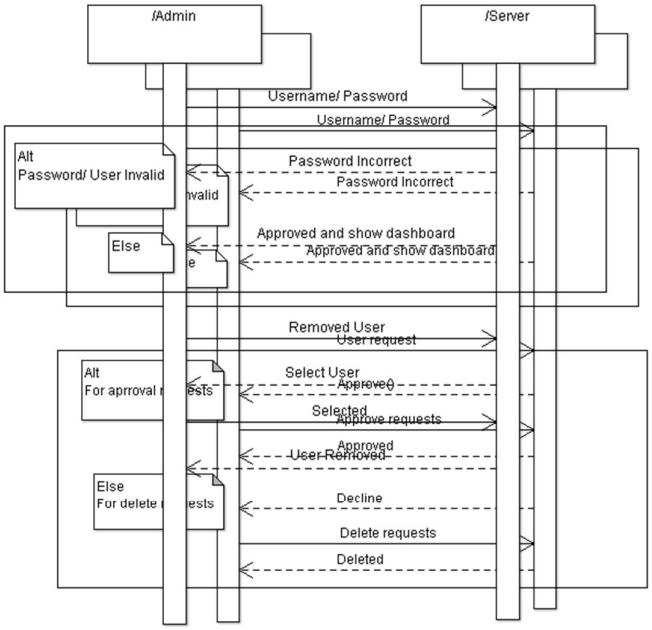
Sequence Diagram: Client Side- Remove Device



Sequence Diagram: Admin Side- Search User



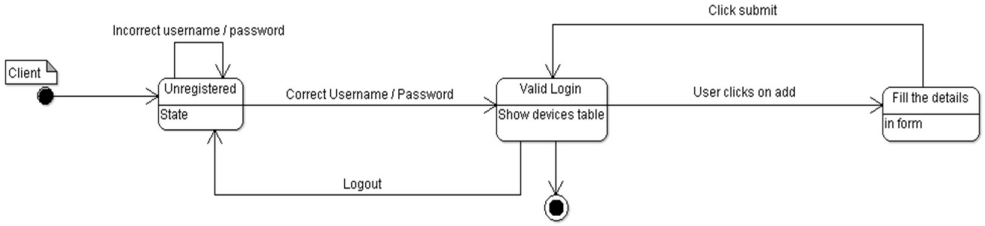
Sequence Diagram: Admin Side- Remove User



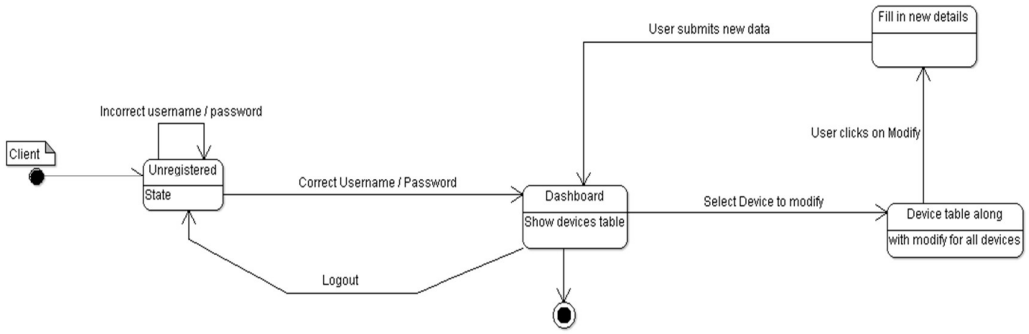
Sequence Diagram: Admin Side- Approve Requests

State Diagram

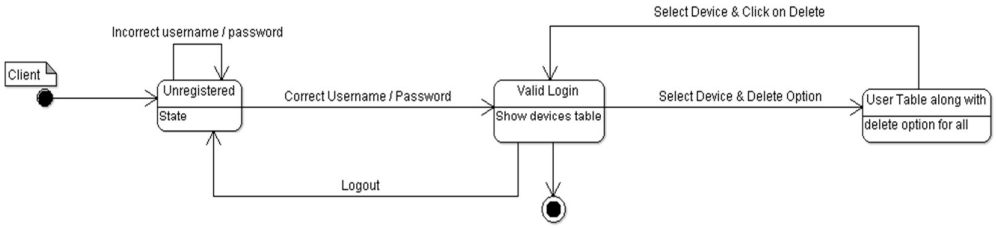
State Diagram: Client Side- Add Device



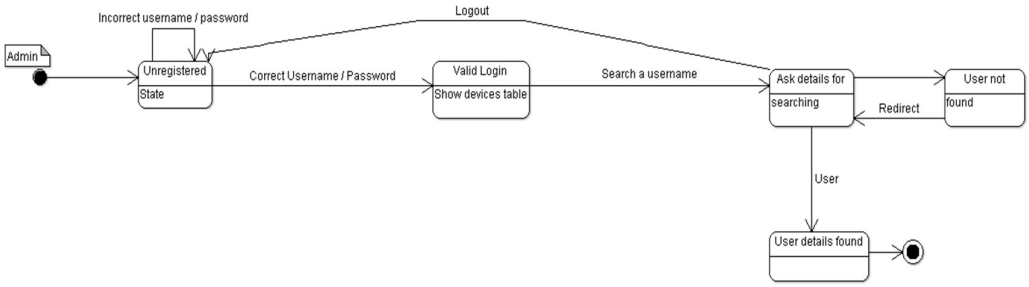
State Diagram: Client Side- Modify Device



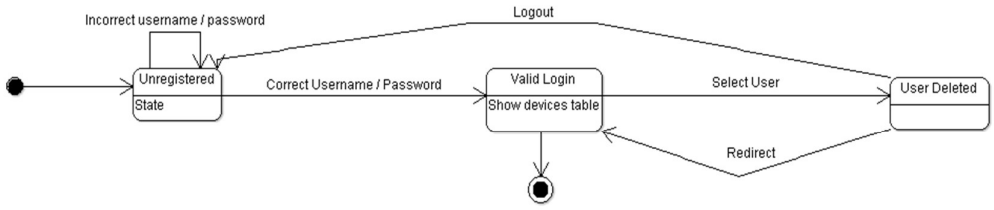
State Diagram: Client Side- Delete Device



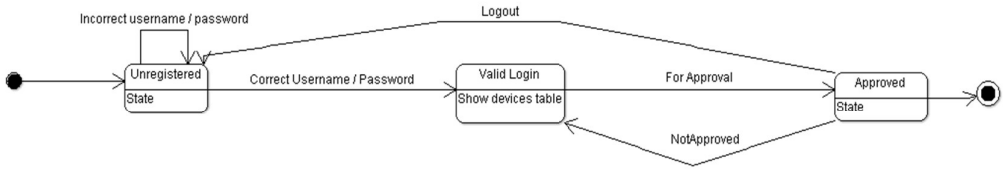
State Diagram: Admin Side- Search User



State Diagram: Admin Side- Remove User



State Diagram: Admin Side- Approve Requests



2.1 Logical Architecture Description

The Architecture is based on Client – Server communication where the data is stored in MySQL Database and the server retrieves the data as and when the client requires it to.

2.2 Database Diagrams

2.2.1 Level – 0 Diagram

This Diagram shows 2 external components and 1 major process – The user and admin, and the Wi – Fi management system. The user sends the following data,

- New device details for adding devices
- New details for modifying current devices
- Selection of device to be removed

The system returns acknowledgments for the same. The admin sends the following data,

- Approval and password of verified devices
- Search query for database
- Selected user entry for removal from database

The system returns acknowledgments for the same.

2.2.2 Level – 1 Diagram

This diagram depicts the breakdown of the Wi – Fi management system into 2 separate processes; client – side and Admin – side. This diagram also shows the data storage systems; Devices and Registrations. The client – side process sends the details received from the user to the Devices storage and retrieves all the details of the stored devices. The admin – side process receives all the device details from both the device storages for its functionality.

2.2.3 Level – 2 Diagram

This diagram depicts the breakdown of client – side and admin – side processes to their respective sub – processes. The client – side process is divided into,

- Add Device
- Modify Device
- Remove Device

The admin – side process is divided into,

- Approve Devices
- Search Users
- Remove User

The inputs are then directed to respective processes and their outputs are given back to users and admins.

2.3 Sequence Diagrams

2.3.1 Client - Side Add Devices

The user first sends a request to add devices to the server. The server checks with the admin if the device limit is reached or not. If not so, a form is produced to the user. The details filled is then sent to the admin via server, to be verified and added to the Devices database.

2.3.2 Client - Side Modify Devices

The user first sends a request to modify devices to the server. The server checks with the database and asks the client to select the device he/she wants to replace then a form is produced to the user. The details of the new device filled by the user is then sent to the admin via server, to be verified and modified to the Devices database.

2.3.3 Client - Side Remove Devices

The user first sends a request to remove devices to the server. The server checks with the admin if there is any device connected to remove or not followed by which server asks the client which device is to be removed. After selecting the device server verifies the password from the user, if the password is correct then the server removes the device and update for the same to admin.

2.3.4 Admin - Side Search User

The admin first logs into the portal which directs the admin to the dashboard if the password and username is correct. Then the admin search for the user for which the server checks into the database if the details of that user is available or not. If yes then the details are displayed.

2.3.5 Admin - Side Remove User

The admin first logs into the portal which directs the admin to the dashboard if the password and username is correct. Admin requests for the removal of a user to which the server shows the table of existing users to the admin. From the displayed table admin selects the user which has to be removed, followed by which server removes the selected user and updates the database.

2.3.6 Admin - Side Approve Requests

The admin first logs into the portal which directs the admin to the dashboard if the password and username is correct. Admin selects an action out of approve devices or delete devices. After selecting, admin performs the task of accepting or declining the pending requests accordingly.

2.4 State Diagrams

The initial states are depicted with a black dot. Final state is being shown by a black dot with a surrounding circle.

2.4.1 Client side Add Device

The client logs in with the credentials. Upon login, the updated devices table is shown. User then chooses the “Add a Device” option and then fills in the details of new device. The new device details are updated in the Devices table and the updated table is displayed to the user.

2.4.2 Client side Modify Device

The user can modify any existing device details by logging in to the system and going to the update page. The user first gets a list of existing devices with the current details and then makes changes by filling in the new details for the respective details.

2.4.3 Client side Delete Device

A logged in authenticated user can remove/delete a device from the devices table by selecting the devices existing on the network and then deleting them. The table is updated and then displayed to the user.

2.4.4 Admin side Search Device

A user is first authenticated to be an admin. Once authenticated, the devices table is shown to the admin user. The user can search for a particular device by entering in the details. Upon successful search, the device details are shown. Else, an error page is shown and then the user is re-directed to the search query page of the web app.

2.4.5 Admin side Delete Device

An authenticated admin can delete a device from the devices table by selecting a device from the table and then clicking on the delete option. The same is updated and shown to the admin user using a re-direct.

2.4.6 Admin side Approve Users

An authenticated admin has the right to approve. Upon logging in, the admin is shown the devices table. The admin individually approves/declines request for each of the users on the network. Once the action of approval/disapproval is taken, it is updated in the Devices table and the same is shown to the admin user.

3. Execution Architecture

3.1 Execution Architecture Description

The system is a Client-Server system, where the client is any web-browser which supports the following softwares: -

- HTML 5
- JavaScript
- PHP 7.2

The server is an apache server (version 2.4.34) using MariaDB as the data-server. The Protocols being used are TCP/IP (version 10).

3.1 Reuse and relationships to other products

This module is a part of NIC's management system for managing the devices connected to their organization's secure wireless service. This module maybe used as a separate management system by the organization or maybe a part of the department(s) under the organization.

4. Design decisions and tradeoffs

1. Initially, if any of the user's device(s) violated the terms and conditions, the user's device was removed from the database. Later, it was decided that the user would be removed from the database in such circumstances.

2. Data flow diagrams were adopted instead of using class diagrams to depict the Wi – Fi Management system.

5.Pseudocode for PHP components

5.1 Client Side

5.1.1 Login(Username, Password)

```
{
      Start session;
      If (Request method is Post)
      {
            Get database_connection object;
            Store the data username in a variable User;
            Filter the variable User;
            Store the data password in a variable Pass;
            Filter and encrypt the variable Pass;

            SQL = "SELECT * FROM registration WHERE
username = '".$user."' and password = '".$pass."'";
            Run query and store result in variable Count;
            If(Count == 1)
            {
                  Store the variable User in session;
                  Close database_connection;
                  Redirect to dashboard.php;
            }
            Else
            {
                  Close database_connection;
                  Print error message;
            }
      }
      Else
            Redirect to login.html;
}
```

5.1.2 Register (Fullname, Designation, EmployeeCode, MobileNum, Email, Division, Location, Ministry, Username, Password, CnfPassword, UserCategory)

```
{
      If(Input_parameters not valid)
      {
            Print Error message;
      }
      Else
      {
            Include the file 'connection.php';
            If(Server request method is post)
            {
                  Get the database_connection object;

                  Store data FullName in variable fname;
                  Filter the variable fname;

                  Store data Designation in variable
            designation;
                  Filter the variable designation;

                  Store data EmployeeCode in variable empcode;
                  Filter the variable empcode;

                  Store data MobileNum in variable mobileno;
                  Filter the variable mobileno;

                  Store data Email in variable email;
                  Filter the variable email;

                  Store data Division in variable nic_div;
                  Filter the variable nic_div;

                  Store data Location in variable nic_loc;
                  Filter the variable nic_loc;

                  Store data Ministry in variable noninc_dept;
                  Filter the variable noninc_dept;

                  Store the data Username in a variable user;
                  Filter the variable user;

                  Store the data Password in a variable pass;
                  Filter and encrypt the variable pass;

                  Store the data UserCategory in a variable cat;
                  Filter the variable cat;

                  SQL = "INSERT INTO registration
VALUES('".$fname."','".$designation."','".$empcode."',".$mobil
eno.",'".$email."','".$nic_div."','".$nic_loc."','".$nonic_dep
t."','".$user."','".$pass."','".$cat."')";

                  If(SQL_database_connection fails)
                  {
```

```
                        Close the database_connection;
                        Print an error message;
                        Redirect the user to login.html;
                  }
                  Else
                  {
                        Close the database_connection;
                        Print a success message notifying that
            the particular user has been registered;
                        Redirect to login.html;
                  }
      }
      Else
      {
            Stay in login.html;
      }
}
```

5.1.3 Add (DeviceName, OS, MACAddress)

```
{
      Start the session;
      If(user_is_set)
      {
            If(User has 3 devices registered)
            {
                  Disable the add button;
            }
            Else
            {
                  If(Input_parameters are not valid)
                  {
                        Print Error message;
                  }
                  Else
                  {
                        Include the file 'connection.php';
                        If(Server request method is post)
                        {
                              Get the database_connection object;
                              Store data DeviceName in variable
      device_name;
                              Filter the variable device_name;
                              Store data MACAddress in variable
                        mac_address;
                              Filter the variable mac_address;
                              Store data OS in variable os;
                              Filter the variable os;
                              SQL = "insert into
      devices(username,device_name,mac_address,os,status,from_d
      uration,to_duration)
      values('$user','$device_name','$mac_address','$os','$stat
      us',now(),'')";

                              Get the status;
                              If(Status)
                              {
                                    Close the database_connection;
                                    Print the success message;
                              }
                              Else
                              {
                                    Close the database_connection;
                                    Print a failed message;
                              }
                        }
                  }
            }
      }
}
```

5.1.4 Modify (ModifiedDeviceID, NewDeviceName NewOS, NewMACAddress)

```
{
      Start the session;
      If(user_is_set)
      {
            If(Input_parameters are not valid)
            {
                  Print Error message;
            }
            Else
            {
                  Include the file 'connection.php';
                  If(Server request method is post)
                  {
                        Get the database_connection object;
                        Store the device ID to be modified in a
      variable rid;
                        Store data NewDeviceName in variable
      device_name;
                        Filter the variable device_name;
                        Store data NewMACAddress in variable
                  mac_address;
                        Filter the variable mac_address;
                        Store data NewOS in variable os;
                        Filter the variable os;
                        SQL = "update devices set
      device_name='$device_name',mac_address='$mac_address',os=
      '$os', from_duration=now(), to_duration='', password='',
      status='pending' where sno='$rid'"

                              Get the status;
                              If(Status)
                              {
                                    Close the database_connection;
                                    Print the success message;
                              }
                              Else
                              {
                                    Close the database_connection;
                                    Print a failed message;
                              }
                        }
                  }
            }
      }
}
```

5.1.5 Remove(removedDeviceID)

```
{
      Start the session;
      If(user_is_set)
      {
            Get database_connection object;
            Store the data removeDeviceID in variable id;
            SQL = "delete from devices where sno='$id'";
            Get the status;
            If(Status)
            {
                  Close the database_connection;
                  Print the success message;
            }
            Else
            {
                  Close the database_connection;
                  Print a failed message;
            }
      }
}
```

5.1.6 Dashboard(Username)

```
{
      Start the session;
      If(user_is_set)
      {
            Get the database_connection object;
            Store the data username in a variable user;
            SQL = "select * from devices where
username='$user'";
            Store the data obtained from connection_status in a
      variable row;
            While(There is a row available in table)
            {
                  Print the values in tabular form;
            }
      }
}
```

5.1.7 Logout()

```
{
      Session start;
      If(session_destroy)
      {
            Session_unset();
            Redirect to login.html;
      }
      Else
            Print the error message;
}
```

5.2 Admin Side

5.2.1 Login(Username, Password)

```
{
     Start session;
     If (Request method is Post)
     {
          Store the data username in a variable User;
          Filter the variable User;
          Store the data password in a variable Pass;
          Filter and encrypt the variable Pass;

          if (usertype = "admin" && password_verify(pass,
hash))
          {
               Redirect to admin dashboard;
          }
          Else
          {
               Print the error message;
          }
     }
     Else
          Redirect to login.html;
}
```

5.2.2 Search(Input)

```
{
     Store the data input in a variable input;
     Filter it to uppercase;
     Get the table body information and store it in a
variable tbody;
     Get each row from the table body and store it in a
variable tr;
     for (i = 0; i < tr.length; i++)
     {
          For each row, get each and every data. Store it
in td;
          Compare each data with the input given;
          If(Data matched)
               Display the data;
          Else
               Display nothing;
     }
     If(Registration is Set)
     {
          Open database_connection;
          Get database_connection object;
          SQL = "Select * from Registration";
```

```
          Store the data obtained from connection_status
     in a variable row;       Store the data input in a
     variable input;
          Filter it to uppercase;
          Get the table body information and store it in
a variable tbody;
          Get each row from the table body and store it
in a variable tr;
          for (i = 0; i < tr.length; i++)
          {
          For each row, get each and every data. Store it
in td;
          Compare each data with the input given;
          If(Data matched)
               Display the data;
          Else
               Display nothing;
     }
          Close database_connection;
     }
     If(Devices is Set)
     {
          Open database_connection;
          Get database_connection object;
          SQL = "Select * from Devices";
          Store the data obtained from connection_status
     in a variable row;
     Store the data input in a variable input;
     Filter it to uppercase;
     Get the table body information and store it in a
variable tbody;
     Get each row from the table body and store it in a
variable tr;
     for (i = 0; i < tr.length; i++)
     {
          For each row, get each and every data. Store it
in td;
          Compare each data with the input given;
          If(Data matched)
               Display the data;
          Else
               Display nothing;
     }
          Close database_connection;
     }
}
```

5.2.3 Remove(removedDeviceID)

```
{
     Start the session;
     If(user_is_set)
     {
          Get database_connection object;
          Store the data removeDeviceID in variable id;
          SQL = "Delete from devices where sno='$id'";
          Get the status;
          If(Status)
          {
               Close the database_connection;
               Print the success message;
          }
          Else
          {
               Close the database_connection;
               Print a failed message;
          }
     }
}
```

5.2.4 Dashboard()

```
{
     Start the session;
     If(user_is_set)
     {
          Get the database_connection object;
          Store the data username in a variable user;
          SQL = "select * from devices where
status='pending'";
          Store the data obtained from connection_status
     in a variable row;
          While(There is a row available in table)
          {
               Print the values in tabular form;
          }
     }
}
```

5.2.5 Approve(RID, Password, DeviceName, ToDuration)

```
{
     Start the session;
     If(User is set)
     {
          Store the device ID to be modified in a
     variable rid;
          Store the data DeviceName in variable
     device_name;
          Store the data ToDuration in variable
     to_duration;
          Store the data Password in variable pass;
          SQL = "update devices set
     to_duration='$to_duration',password='$pass',
     status='approved' where sno='$rid'";
          Get the status of query execution in a variable
     status;
          If(status)
          {
               Print success message;
          }
          Else
          {
               Print Error message;
          }
}
```

5.2.6 Decline(RID, DeviceName, Reason)

```
{
     Start the session;
     If(User is set)
     {
          Store the device ID to be modified in a
     variable rid;
          Store the data DeviceName in variable
     device_name;
          Store the data Reason in variable reason;
          SQL = "update devices set
     to_duration='$to_duration',password='$reason',
     status='declined' where sno='$rid'";
          Get the status of query execution in a variable
     status;
          If(status)
          {
               Print success message;
          }
          Else
          {
               Print Error message;
          }
}
```

5.2.7 Logout()

```
{
      Session start;
      If(session_destroy)
      {
            Session_unset();
            Redirect to login.html;
      }
      Else
            Print the error message;
}
```