CHAPTER 1

INTRODUCTION

1.1 Background Of Study

In power engineering, the Optimal Power Flow (OPF) is an important tool relating numerical analysis performed to a power system. OPF is essential for engineers to plan, perform economic scheduling and control an existing system. Also the study is important for future expansion planning of a power system. The objective of performing OPF is to determine the magnitudes and phases angle of voltages at each bus and active and reactive power flow in each line.

A power system is assumed to be operating under balanced conditions and a single-phase model is used in order to solve a power flow problem. Each bus is associated with four quantities which are voltage magnitude $|V|$, phase angle δ, real power P and reactive power Q [4].

RTU recorded the quantities of each bus and sent them to SCADA system through wireless communication channel. Along the data transmission process, certain data can be corrupted by noise. At the receiver, SCADA accepted all the data either corrupted or uncorrupted by noise. EMS used data collected at SCADA for power utility performance analysis. The result of the analysis is inaccurate and unreliable because the data used contained inaccurate data.

PSSE uses mathematical approach called State Estimation (SE). By using SE, inaccurate data in the SCADA can be detected and filtered out. As a result, EMS will used accurate data only in order to perform any analysis. As the redundancy of data used by PSSE increases, the accuracy of the analysis will also increases. However, redundancy decreases when SE filtered out inaccurate data. In order to maintain redundancy, new data need to be transmitted back by RTU from the inaccurate data's bus to SCADA.

1.2 Problem Statement

PSSE cannot maintain input redundancy because there is no medium for PSSE to be updated with new data because it only works one way and unable to produce notification back to the physical power system. Figure 1 shows data flows from physical power system to EMS environment. Personnel from the physical power system should be given privilege to monitor and control PSSE since they are the one who has to transmit new data to SCADA.

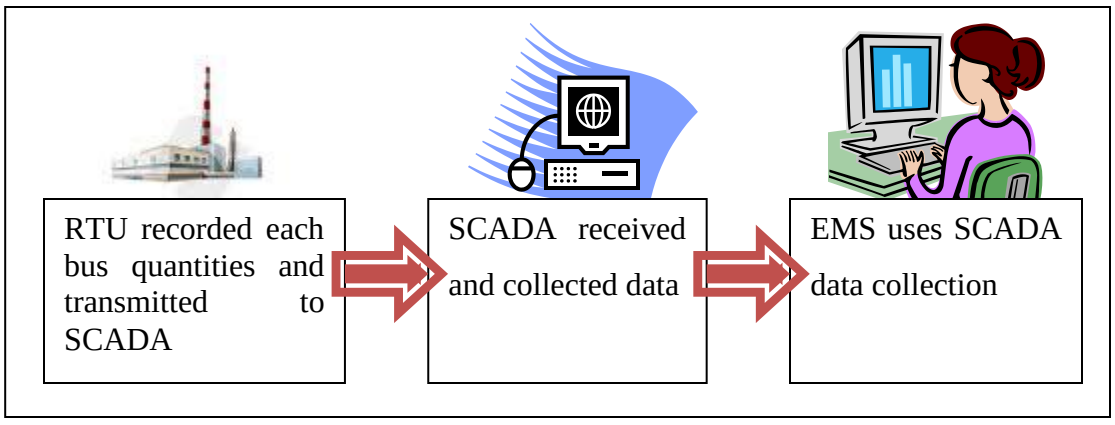


Figure 1 Data flow from physical power station to analysis phase

1.2 Objective Of Project

The objective of this project is to develop a web-based SCADA for PSSE that allows authorize user to continuously monitor and feedback new measurement value for the PSSE database from anywhere via Internet.

1.3 Scope Of Study

The web-based SCADA for PSSE is developed for the purpose of data monitoring, editing and allowing it to be viewed by authorized personnel anywhere via the Internet. It is also made to notify personnel for new data transmission.

The project aims to create a website that is able to:

1. Allow personnel to edit bus quantities in the input database
2. Notify personnel to take a new measurement reading

CHAPTER 2

LITERATURE REVIEW

2.1 PSSE

PSSE is an under development project that aims to be integrated between RTU and SCADA to aid EMS analysis accuracy. PSSE uses SE mathematical approach. In order to prove the mathematical concept, PSSE is developing using MATLAB tool.

2.1.1 Matrix Laboratory (MATLAB) Application Of PSSE

MATLAB is a powerful tool that helps in calculating numerical computing environment programming language [6]. Therefore, the PSSE coding is written in MATLAB files (M-files) and the SCADA data that is stored in text files.

MATLAB is not user friendly where is has no Graphical User Interface (GUI) that ease users in working with the tool. Since MATLAB is designed to do heavy calculations, it cannot have any fancy GUI. This will cause difficulties to users with no programming background. It is hard for them to edit input database or even run the PSSE since they need to key in certain MATLAB commands.

2.2 LabVIEW

LabVIEW is graphical programming software used in developing program for data acquisition, control, simulation and communication applications. In the LabVIEW environment, icons are interconnected to create program generally referred to as VI, and all LabVIEW program have an extension .vi.

All VIs must have two elements: Front Panel and Block Diagram. The front panel is an interactive user interface of a VI because it simulates the front panel of a physical instrument. The front panel can control push buttons, graphs, knobs and many other controls and indicators in Figure 2. The block diagram is the VI's source code, constructed in LabVIEW's graphical programming language, G in Figure 3. The block diagram is the actual executable program. The components of a block diagram are built-in functions, constants, lower-level VIs and program execution control structures [7].

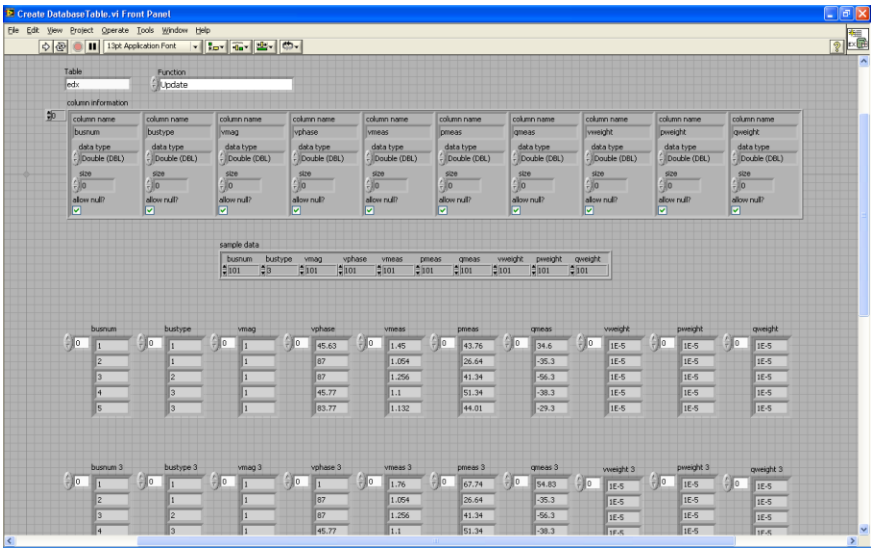


Figure 2 LabVIEW Front Panel

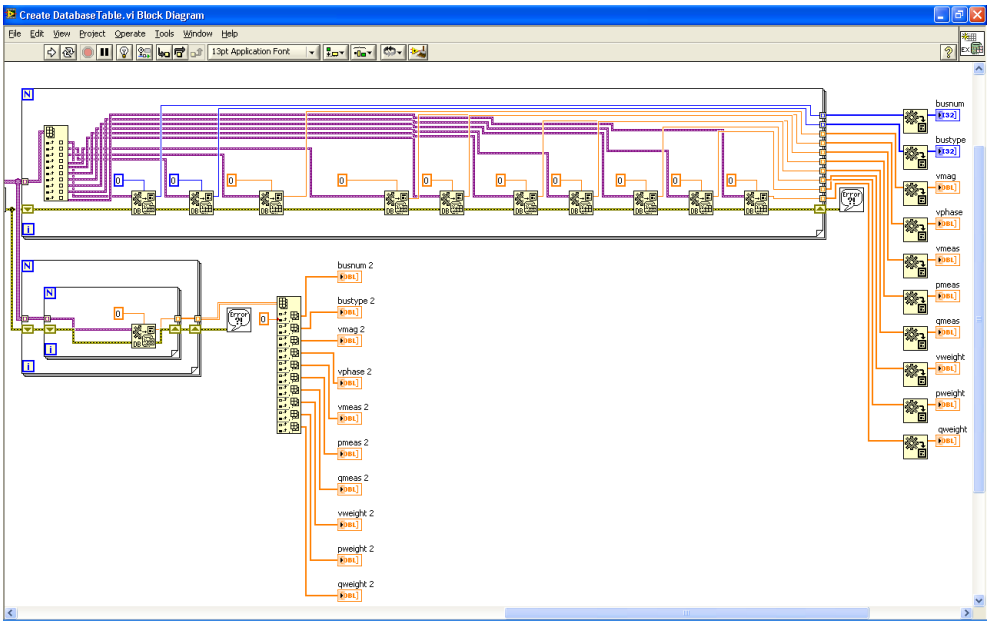


Figure 3 LabVIEW Block Diagram

2.3 Database Terminologies

A database is a collection of data in tables that have relationship between each other. A database should be able to perform basic functions such as creating table, inserting data into table, displaying queries, updating data in a table and deleting table [8].

2.4 Website Development Tools

There are several basic tools needed in the website developments.

2.4.1 MySQL

MySQL is a relational database management system shown in Figure 4 that runs as a server providing multi-user access to a number of databases [9]. MySQL is based on Structured Query Language (SQL). There are several basic operations needed to be mastered which are CREATE (creating a database), INSERT (inserting data to the database), SELECT (selecting which database to work with), UPDATE (updating data to the database) and DELETE (deleting data from the database). The challenge of handling with MySQL is the ability to create relationship between databases. Correct relationship is needed in order to make the website works properly.

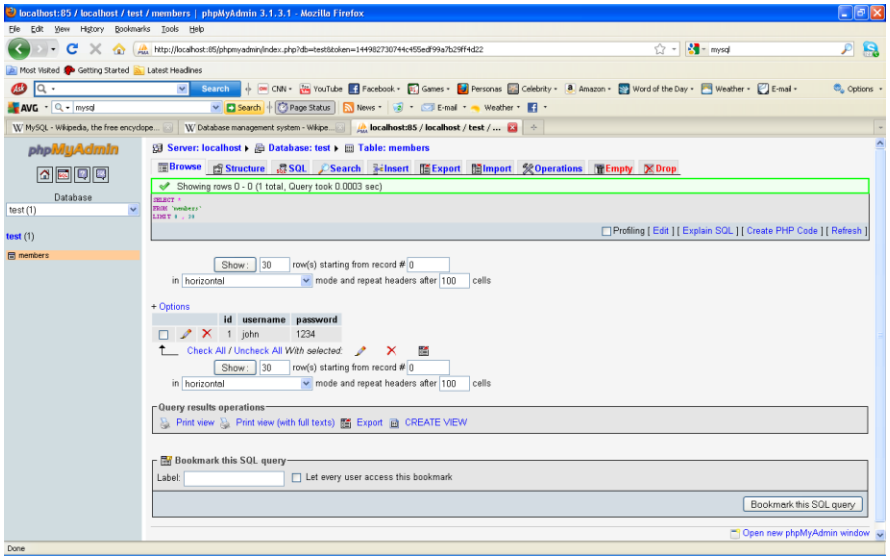


Figure 4 Screenshot of MySQL called phpMyAdmin

2.4.2 Server-Side Scripting Language

Both languages as shown in Table 1 is strong and have own capabilities. Besides, many improvements in their reliability and capability performance have made Hypertext Preprocessor (PHP) and VB become software developer choices.

Table 1 Comparison between PHP and VB

PHP	VB.net
• Low maintenance and development course • High performance and reliability • Able to embed itself into HTML code • Compatible with server like Apache • Open source software • Support MySQL	• VB has been built from the ground up as an object oriented language • VB has a number of security and reliability features built right into its core design • Steadily improved in runtime performance, better than C and C#

PHP is a server-side scripting language. It allows web developer to make the website more dynamic, meaning that not only the website is capable of displaying text and images but able to receive input from user and process the data received [10]. PHP is being used widely to develop commercial websites by many developers.

2.4.3 HTML

HTML is used for website text formatting. It allows web developer to organize and arrange the texting such as using heading, list, tables, creating forms and inserting images as shown in Figure 5. HTML doesn't require any software to be installed in order to use it. HTML coding can be written in any text editor application such as Notepad. The result can be viewed through any web browser such as Mozilla or Internet Explorer [11].

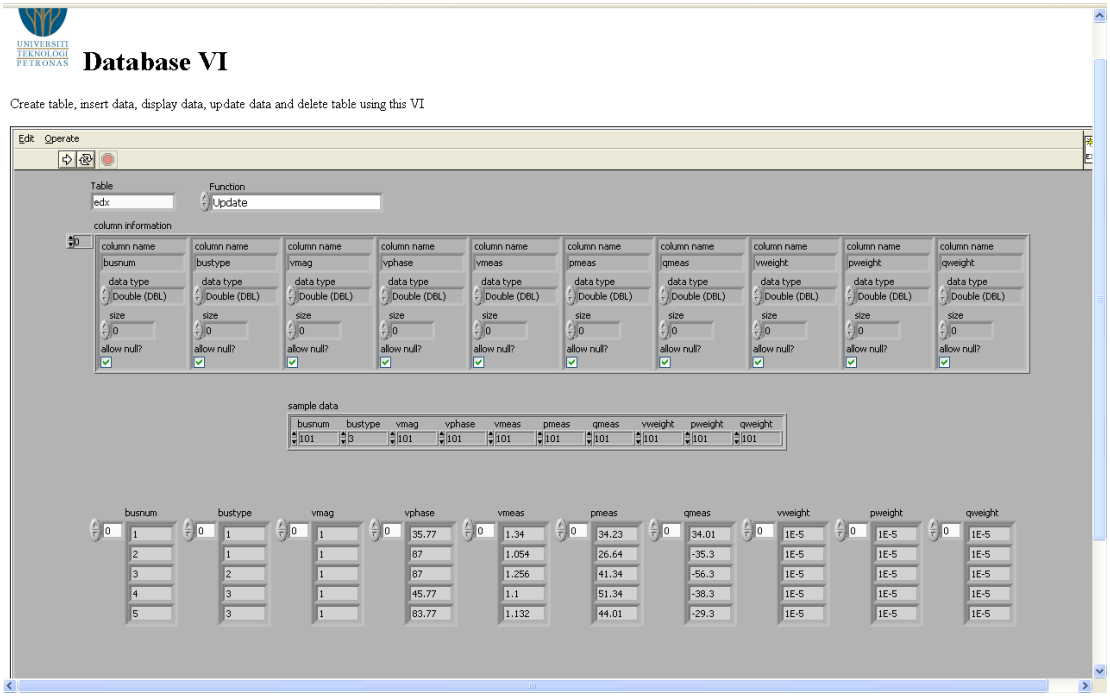


Figure 5 Screenshot of HTML file viewed in Mozilla web browser

2.4.4 CSS

To add in style to the web pages, web developers use CSS. It allows web developers to customize or decorate the website to make it more presentable. It can customize the background colour, font type, font size and other settings to make the website more interesting as shown in Figure 6 [12].

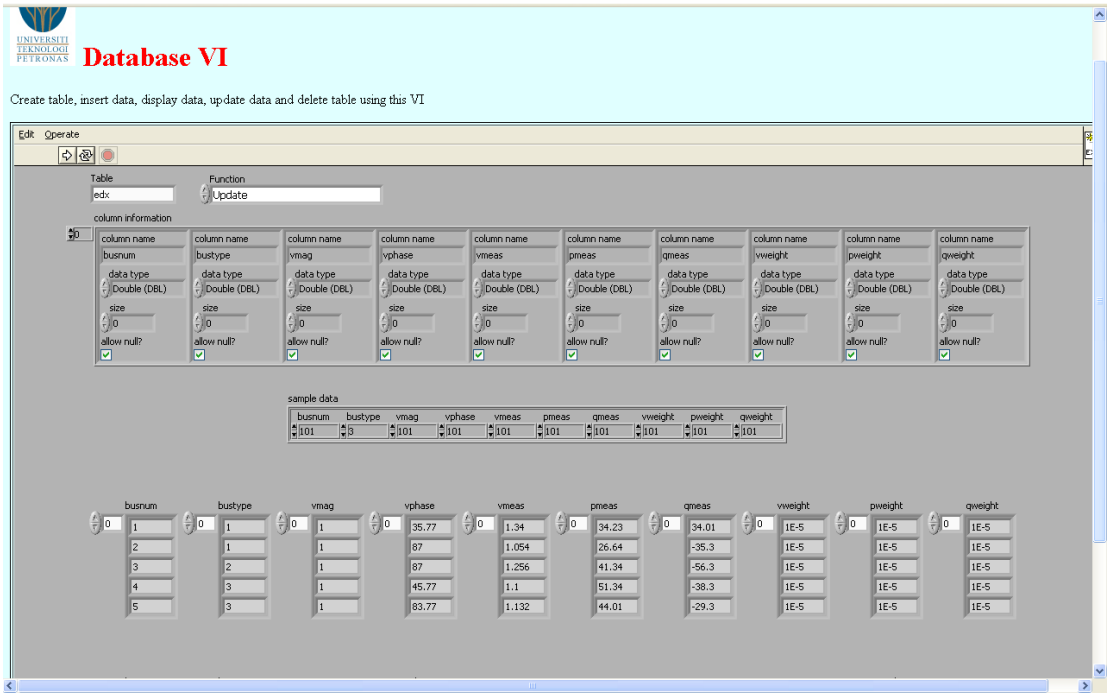


Figure 6 Screenshot of HTML webpage that uses CSS

CHAPTER 3

METHODOLOGY

3.1 Procedure Identification

A thorough study was done through the internet and also from the Information Resource Center (IRC) on the basic knowledge needed and tools required in achieving the project aim.

This project aims to develop a web-based SCADA system for PSSE. The big task is broken into small subtask to ease the development process. The work flow for the success of this project is shown in Figure 7. The work flow represents work divisions for FYP 1 & 2. Gantt charts for both FYP 1 & 2 activities are attached in Appendix K and L.

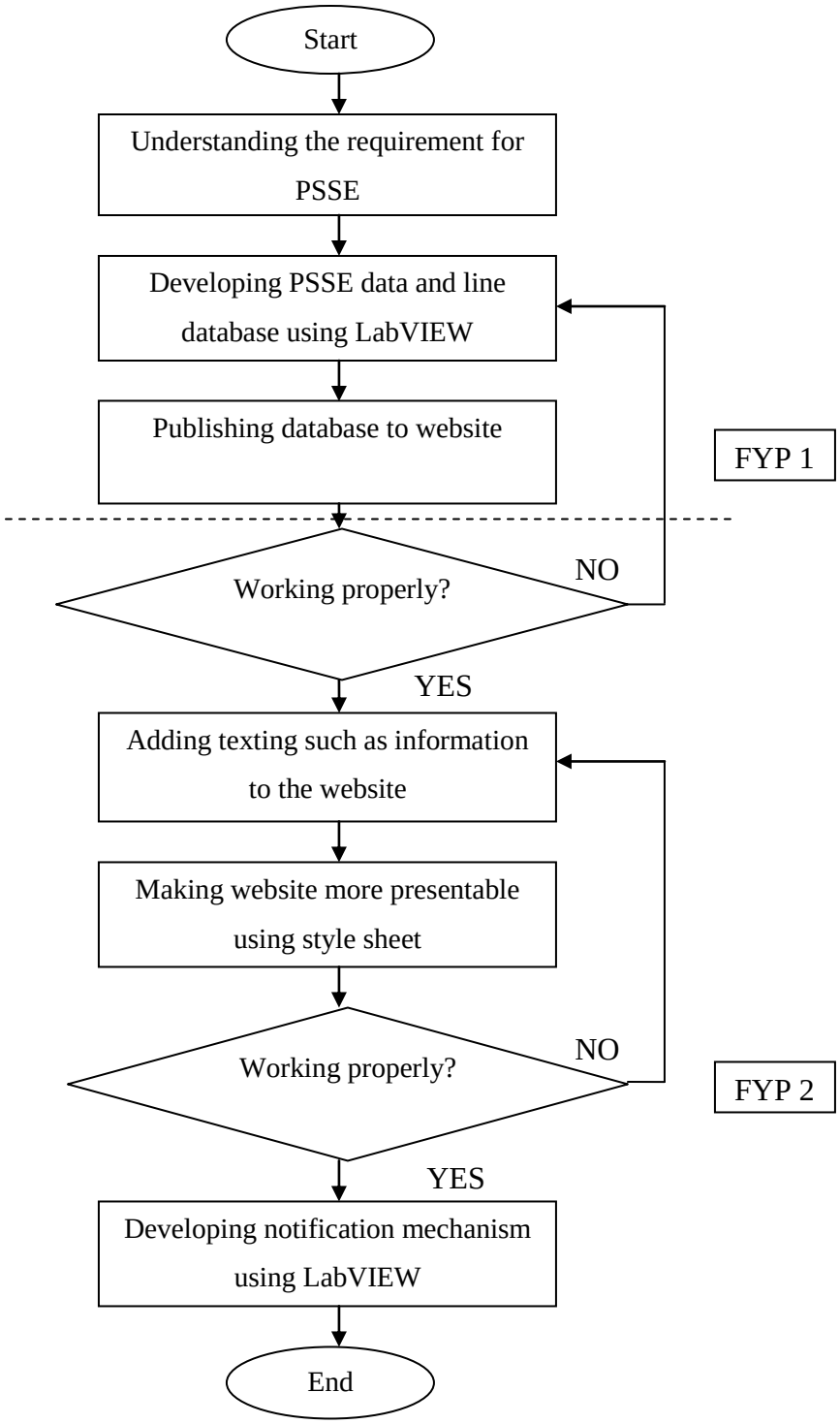


Figure 7 Work flowchart for FYP 1 & 2

3.2 Tools and Equipment Used

This project requires more programming skills and efforts. Also, software and hardware as listed in Table 2 are essential for the success of this project.

National Instrument (NI) LabVIEW 2009 trial edition with NI LabVIEW Database Connectivity Toolkit is installed to a personal computer (PC).

In order to run PHP scripts, an Apache Server is required. XAMPP is an easy to install Apache Server that contains PHP and MySQL[13]. It is an open source installer. XAMPP is installed to the same PC.

Table 2 List of Hardware and Software required

Hardware	Software
• PC	• NI LabVIEW 2009 trial edition • NI LabVIEW Database Connectivity Toolkit • XAMPP

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Findings

4.1.1 Internet Connectivity in LabVIEW

LabVIEW allows VIs to be accessed by other people through a web browser. When allowing people to access VIs over the web, it also support authority whether people can

- Just monitor the VIs (read-only)
- Control VIs (send data and events back)

from the web browser.

To enable the connectivity features, a built-in LabVIEW Web Server can be used. The benefit of this feature is it dynamically creates web pages with images of VI's front panel.

4.1.2 LabVIEW 's Built-In Web Server Configuration

Below are the steps taken to configure the built-in Web Server feature:-

Step 1: To avoid any conflicts, make sure no other web server (e.g., Apache) is running on the same port before enable the LabVIEW web server

Step 2: If firewall is installed, ensure port 80 is not blocked. (Port 80 is the default port the web server runs on)

Step 3: In LabVIEW, go to Tools>>Options>>Web Server:Configuration as shown in Figure 8

Step 4: Check the box "Enable Web Server"

Step 5: Click OK

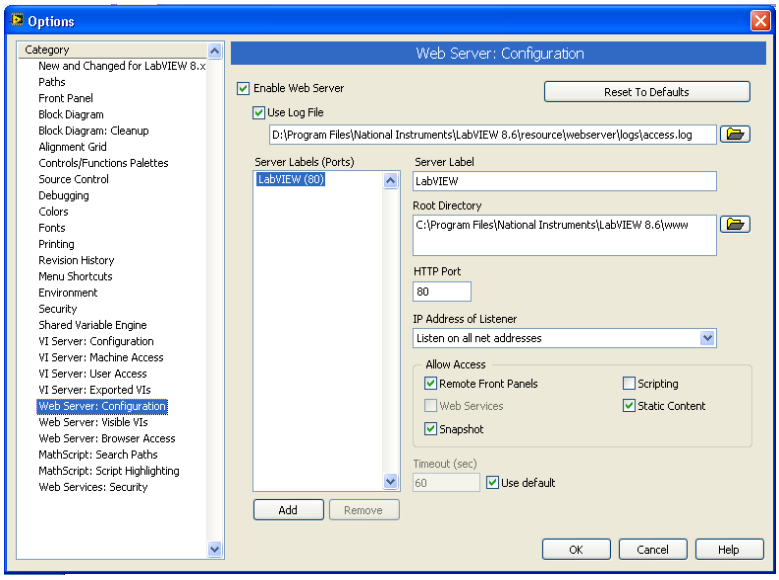


Figure 8 Web Server Configuration options

4.1.3 Publishing to HTML With LabVIEW Web Server

By using LabVIEW Web Server, task to share a VI on the Internet is been simplified. To publish a VI in a web browser, make sure that the VI exist and loaded into the LabVIEW memory to be served by the LabVIEW web server. The LabVIEW web server can publish the VI to the Web through various methods such as

- "Snapshot" : A static ("snapshot") image of a VI's front panel
- "Monitor" : An image of a VI's front panel that auto-refresh every *N* seconds
- "Embedded" : The VI in the front panel can be controlled. However, this option uses a browser plug-in to display the VI in real time and allows users to control the VI

4.1.4 Database Connectivity Toolkit Add On

LabVIEW Database Connectivity allows quick connection to local and remote databases and performs a lot of common database operations without having to know SQL. This toolkit is able to communicate with Microsoft Access.

4.1.4.1 Database Programming Model

There are 3 step processes for LabVIEW to interact with a database:

Step 1: Establish a connection with a database. This connection is through a System Data Source Name (DSN), File DSN or Universal Data Link (UDL)

Step 2: Perform operations on the database such as inserting records, updating records and querying records

Step 3: Close the connection to the database and check for errors

4.1.4.2 LabVIEW to Microsoft Access database

The first step in running database operations with the Database Connectivity Toolkit is to connect to the particular database. UDL is a type of connection that can be created for LabVIEW to communicate with a database.

Below are the steps taken to establish a connection between LabVIEW and a Microsoft Access Database through UDL:

Part 1 Create a Database

Step 1: Open Microsoft Access (Start → Programs → Microsoft Access)

Step 2: Create a new Blank Database from a New File menu, name it LabVIEW.mdb (use this Access file throughout the project)

Step 3: Save the database and close Microsoft Access. Tables and data will be added using the Database Connectivity Toolkit

Part 2 Create a UDL through LabVIEW

Step 1: Launch LabVIEW

Step 2: From a blank VI, select Tools → Create Data Link to launch the Data Link Properties window

Step 3: On the Provider tab, select the provider for the Database Management System (DBMS) that will be communicated with. By default this is the Microsoft Object Linking and Embedding Database (OLE DB) Provider for Open Database Connectivity (ODBC) driver. In this case, select the Microsoft Jet 4.0 OLE DB Provider as shown in Figure 9 to communicate with Microsoft Access database. After selecting the provider, click Next button to move to the next step

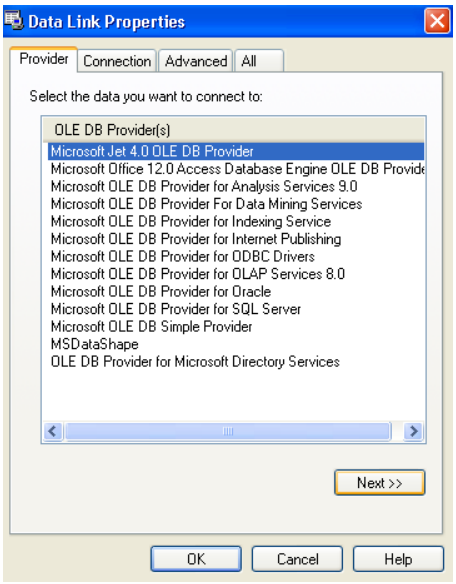


Figure 9 Data Link Properties

Step 4: On the Connection tab, browse to and select the LabVIEW.mdb database created in Part 1. Click the Test Connection button to test the UDL connection as shown in Figure 10

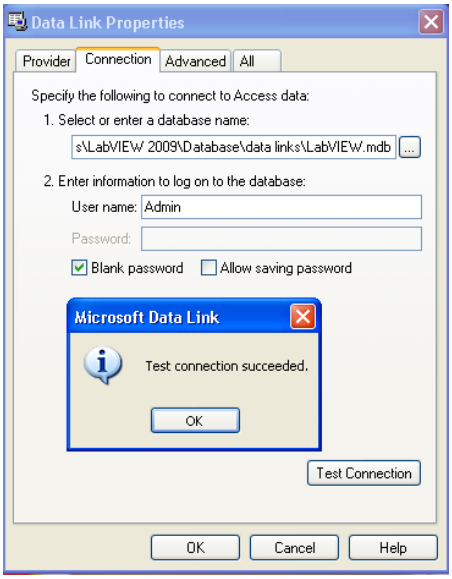


Figure 10 Connection tab in Data Link Properties

Step 5: Click the OK button to complete the setup of the UDL

Step 6: Save the UDL as LabVIEW.udl in the C:\Program Files\National Instruments\LabVIEW 2009\Database\data links folder

4.1.5 Login Page

Login page allow the administrator to control who can view the website. Since this SCADA for PSSE should be viewed by certain authorized personnel, therefore it needs a login page before one can access the main page of the website.

3 files are needed for the login mechanism. They are the main_login.php, checklogin.php and login_success.php. There are steps taken to accomplish this task. Below are the steps:

Step 1: Create table "members" in database "test" using phpMyAdmin

- CREATE TABLE 'members' ('id' int(4) NOT NULL auto_increment, 'username' varchar(65) NOT NULL default", 'password' varchar(65) NOT NULL default", PRIMARY KEY ('id')) TYPE=MyISAM AUTO_INCREMENT=2;
- INSERT INTO 'members' VALUES (1,'john',1234)

Step 2: Create file *main_login.php* in Notepad refer Appendix A

Step 3: Create file *checklogin.php* refer Appendix B

Step 4: Create file *login_success.php* refer Appendix C

Step 5: Create file *logout.php* refer Appendix D

Step 6: Save file main_login.php, checklogin.php, login_success.php and logout.php in C:\xampp\htdocs

4.2 Results

4.2.1 LabVIEW Database Connectivity Toolkit

The Database Connectivity Toolkit VIs allowed a quick and easy table creation in a Microsoft Access database, all from the LabVIEW environment. The basic database operations such as creating table, inserting data, selecting data or updating data can be easily developed using LabVIEW.

4.2.1.1 Creating Table in a Database

By developing create_database.vi, it will create a database in Microsoft Access with required table name and will key in the column information with its selected data type. The VI's front panel is shown in Figure 11 and the VI's block diagram is shown in Figure 12. The flowchart is attached in Appendix E.

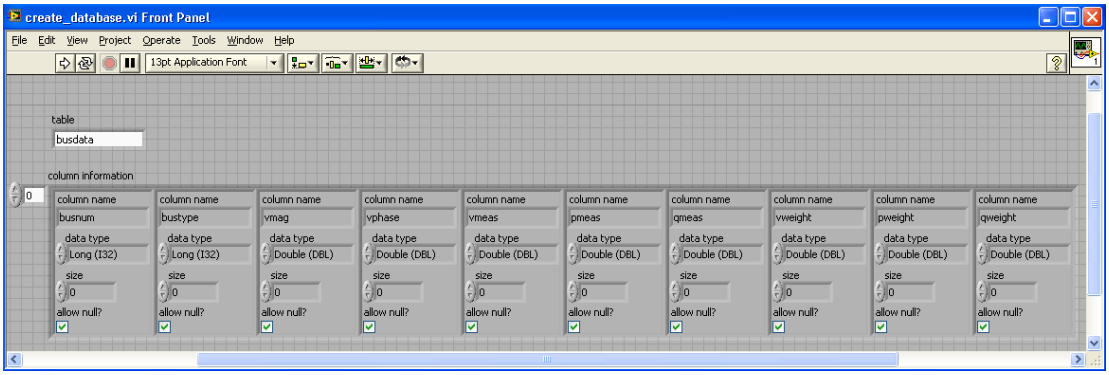


Figure 11 create_database.vi Front Panel

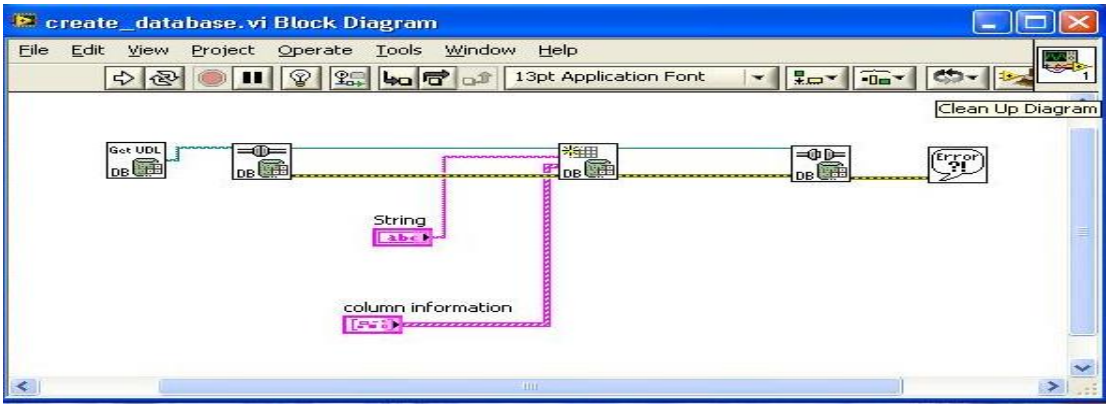


Figure 12 create_database.vi Block Diagram

This VI creates a table called busdata in LabVIEW.mdb with the specific column information. It will create 10 columns. The result in LabVIEW.mdb is shown in Figure 13.

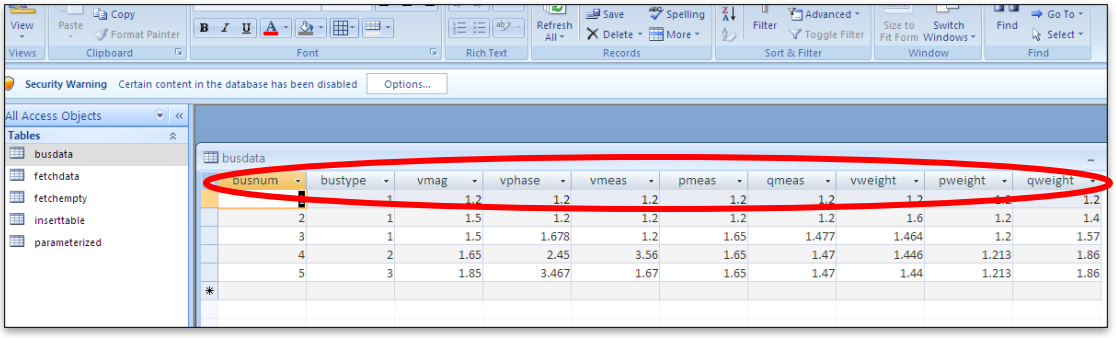


Figure 13 create_database.vi effects on LabVIEW.mdb

4.2.1.2 Inserting Data to a Database

By developing insert_data.vi, it will allow user to insert a record of data one at a time in the sample data column. It will save the data inside LabVIEW.mdb. It is as if data been inserted manually from Microsoft Access; however it can also be done in LabVIEW environment. The VI's front panel is shown in Figure 14 and the VI's block diagram is shown in Figure 15. The flowchart is attached in Appendix F.

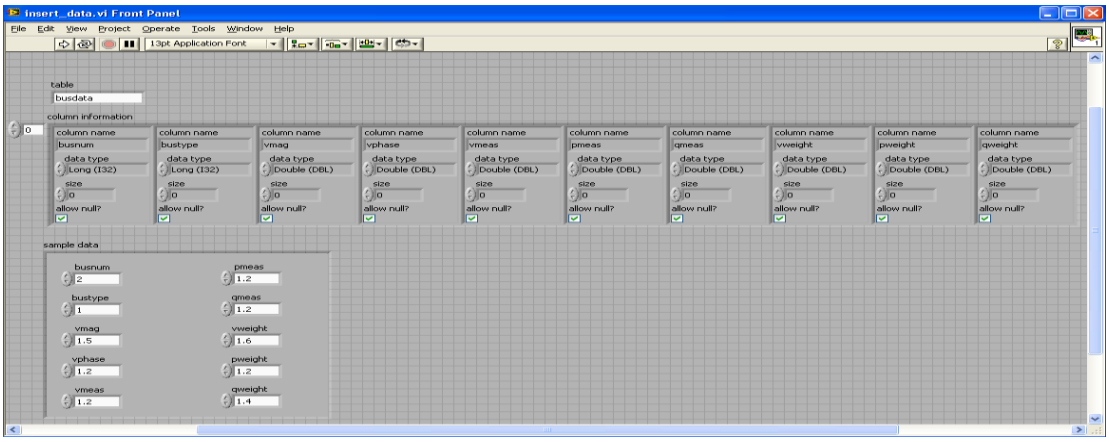


Figure 14 insert_data.vi Front Panel

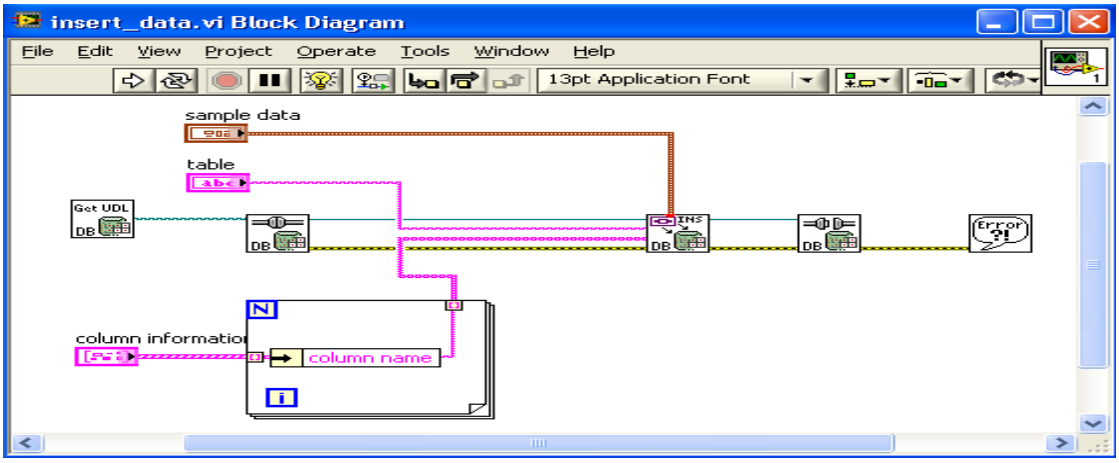


Figure 15 insert_data.vi Block Diagram

This VI will insert data from sample data area to the specific column and save it in the required table name. The changes occurred at LabVIEW.mdb as insert_data.vi is run is shown in Figure 16.

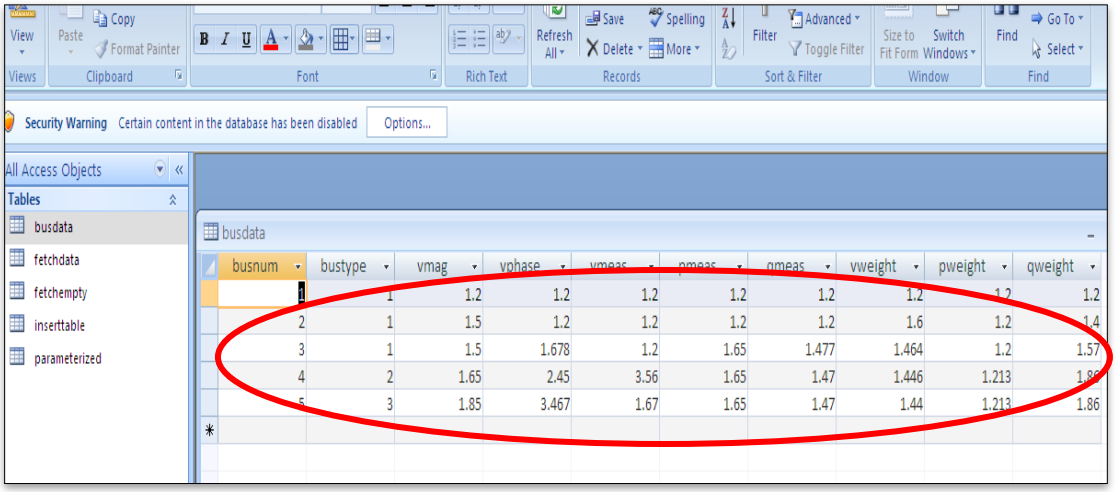


Figure 16 insert_data.vi effects on LabVIEW.mdb

4.2.1.3 Retrieving Data from a Database

By developing display_data.vi, it will extract all of the data from the database and display it in the appropriate indicators. The VI's front panel is shown in Figure 17 and the VI's block diagram is shown in Figure 18. The flowchart is attached in Appendix G.

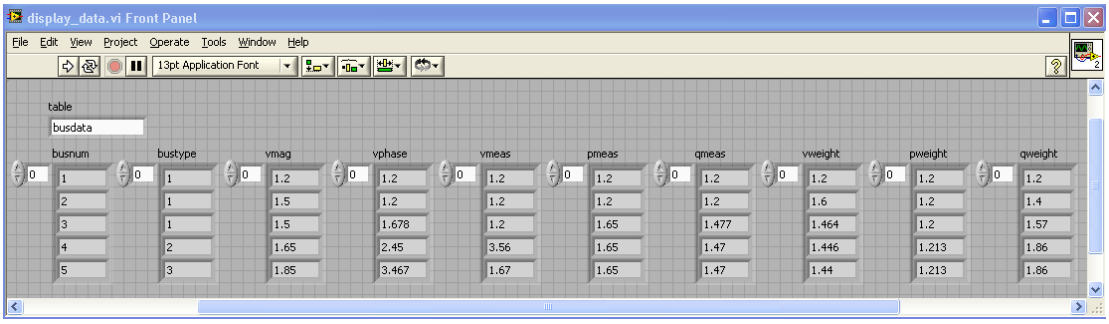


Figure 17 display_data.vi Front Panel

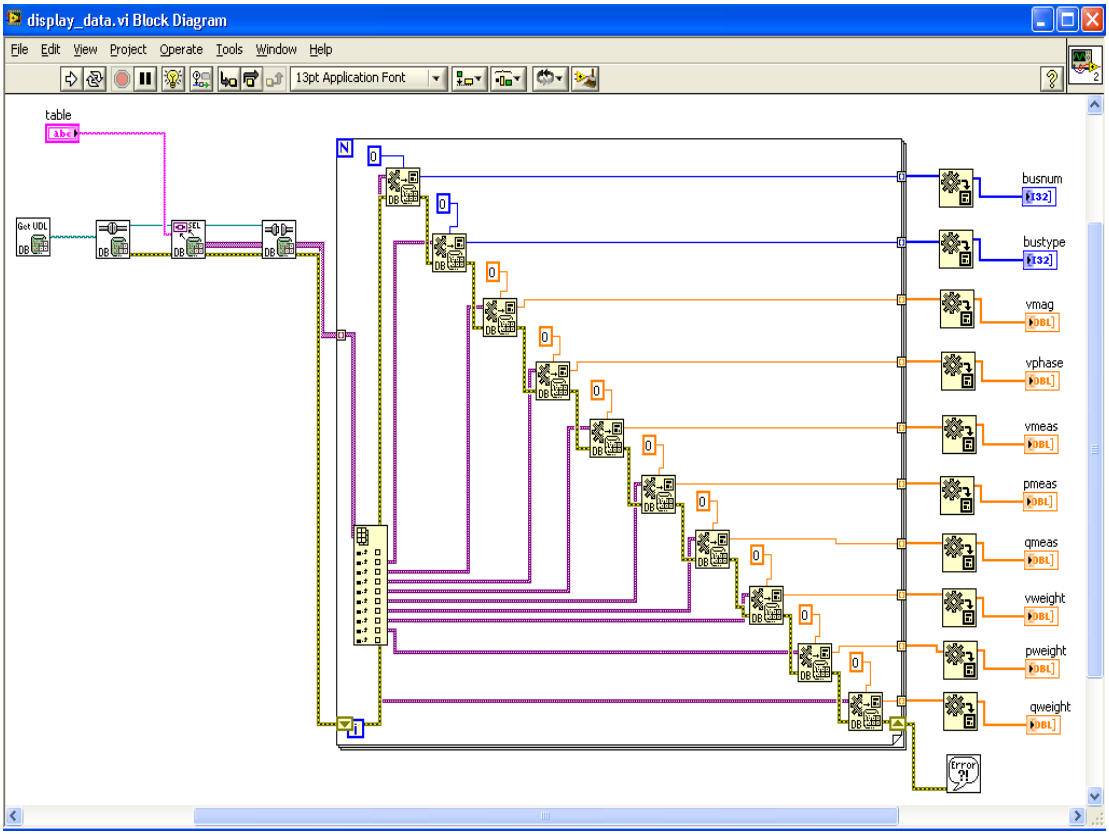


Figure 18 display_data.vi Block Diagram

4.2.1.4 Combined subVI

The entire sub VIs has been combining as one VI to ease user to do all the functions in one VI only. Also, delete and update functions has been developed in the main VI called PSSE.vi. The delete mechanism flowchart is attached in Appendix H. The update mechanism flowchart is attached in Appendix I. The front panel of PSSE.vi is shown in Figure 19. The PSSE.vi flowchart is attached in Appendix J.

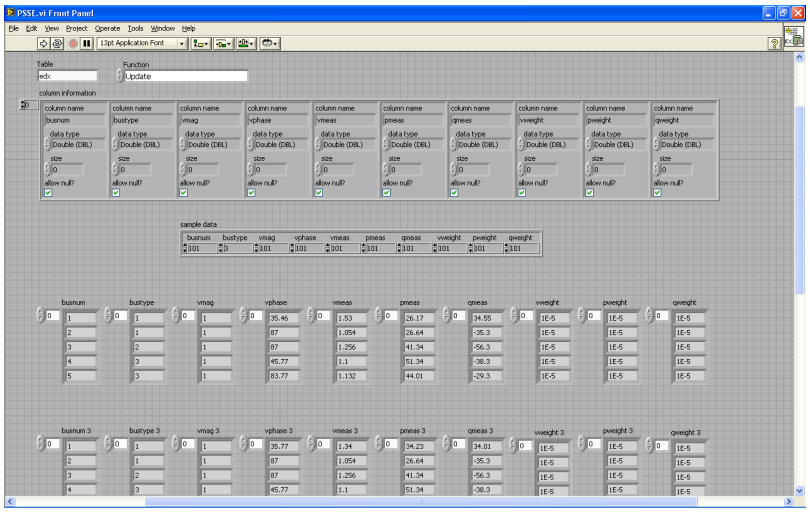


Figure 19 PSSE.vi Front Panel

4.2.2 LabVIEW Web Publishing Tool

PSSE.vi as shown in Figure 20 is published to the Internet as a HTML webpage using LabVIEW Web Publishing Tool as shown in Figure 21.

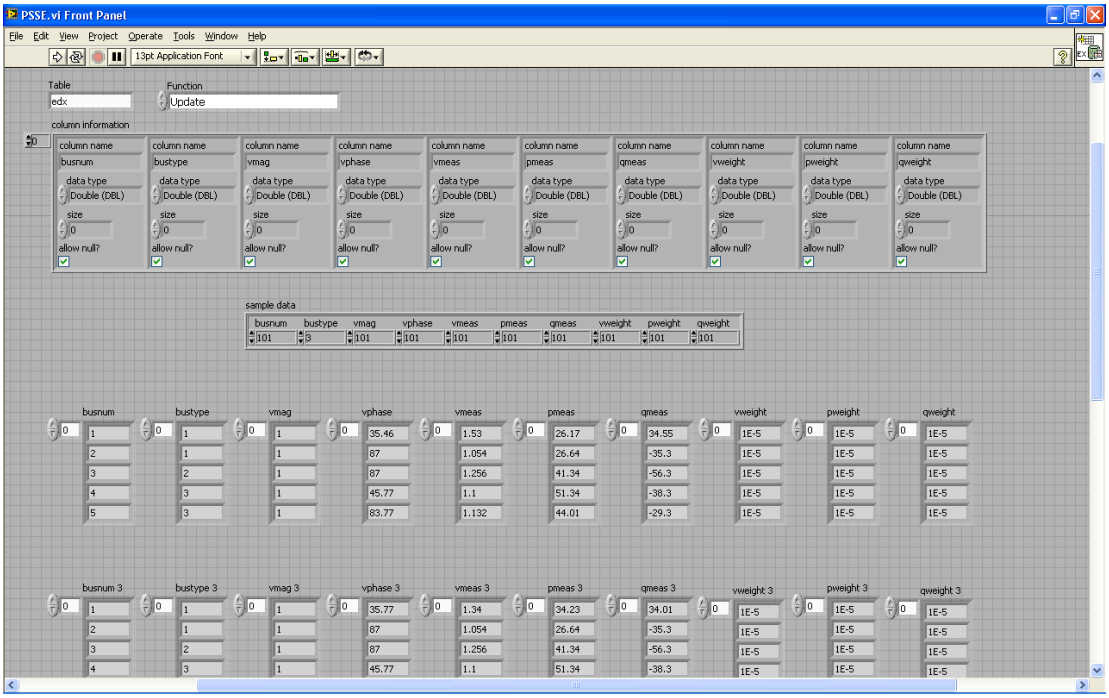


Figure 20 PSSE.vi in server control

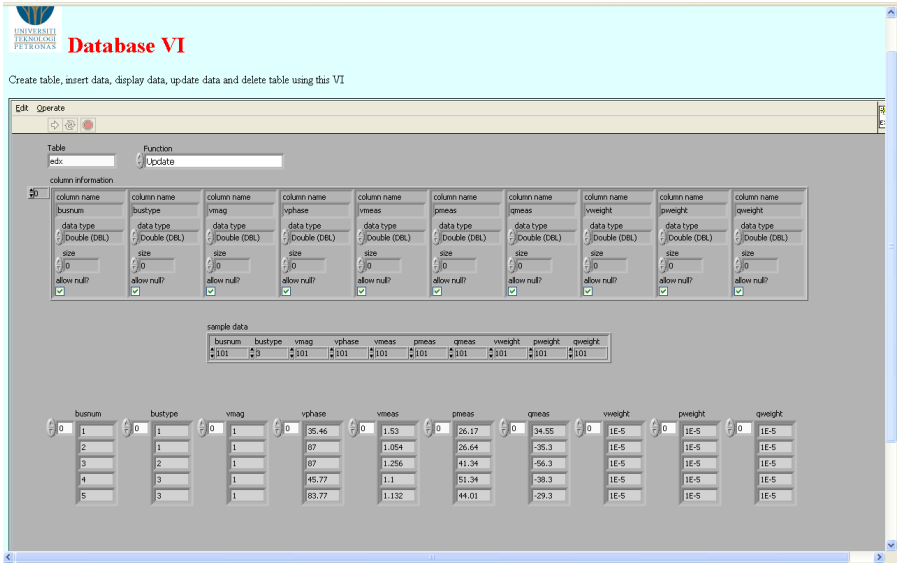


Figure 21 Published HTML PSSE.vi

The LabVIEW Web Server is enabled by conducting all the steps mention in section 4.1.2. Since LabVIEW Web Server also allows users to publish the VI to HTML, then it is easier to develop a web based SCADA by following the steps below:

Step 1: Make sure the built-in web server is turned on, as described earlier

Step 2: Load the Add.vi into the LabVIEW memory by opening the VI

Step 3: Open the Web Publishing Tool (from the Tools menu) as shown in Figure 22

Step 4: Set the VI name drop-down list to "PSSE.vi" and set the Viewing Mode option to any of the three ways that is mentioned in section 4.1.3. Then press the Preview in Browser button to open a preview HTML page in the default web browser or press Next to permanently save the HTML file

Step 5: Open the HTML file in a web browser. The entire VI front panel will be displayed

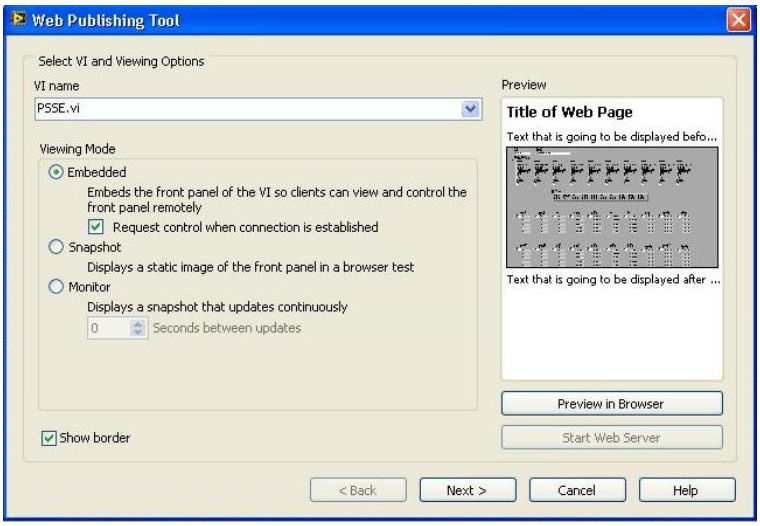


Figure 22 Web Publishing Tool Snapshot

4.2.3 Viewing Mode

All the three modes which are the snapshot, monitor and embedded are managed to be performed. The snapshot mode actually displays a static image of the VI front panel. Therefore, the VI in the web cannot be controlled. This condition is suitable for people to just monitor the VI (read-only). The monitor mode also can be read-only. The only different is it will refresh the VI snapshot every N seconds. Therefore, if the is any changes in the real VI, it will be displayed in the web after N seconds. This is suitable for VIs that produced moving waveforms. The embedded mode is chosen for this project because this mode embeds the front panel of the VI so clients can view and control the front panel remotely.

4.2.4 "Embedded" Viewing Mode

This mode needs extra configuration. Below are the steps taken to allow client to control the VI.

Step 1: In the page opened in the web browser, clients will see the entire VI front panel (including the toolbar and menubar, if the VI is configured to show these while running). Right click on the panel and select Request Control of VI, as shown in Figure 23

Step 2: If control is granted (based on the Web Server:Browser Access setting), then clients are able to control the VI in the web. Now, when clients try changing the value through the web, it will work just fine as if they are using the real VI

Step 3: Main VI user can release control, from the panel's pop-up menu via the Remote Panel Client>>Release Control of VI option. Note that the real user and client cannot control the VI at the same time.

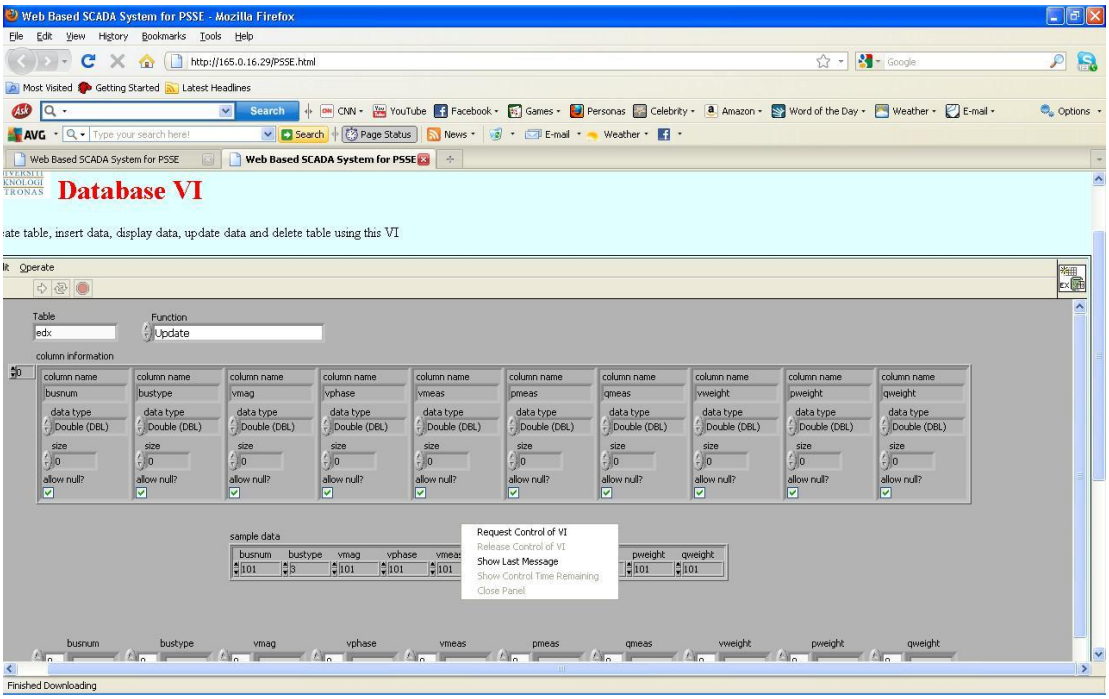


Figure 23 Requesting control of the VI in the web browser

4.2.5 Notification Mechanism

Alert mechanism in a form of sound triggering is developed using LabVIEW sound VI. Alert mechanism is placed at the voltage vectors to allow it to alert if the voltage vectors values are acceptable in accordance to its tolerance. If one of the data doesn't meet the requirement, the VI will play an alert sound to alert the user of the bad data. The front panel is shown in Figure 24 and the block diagram is shown in Figure 25.

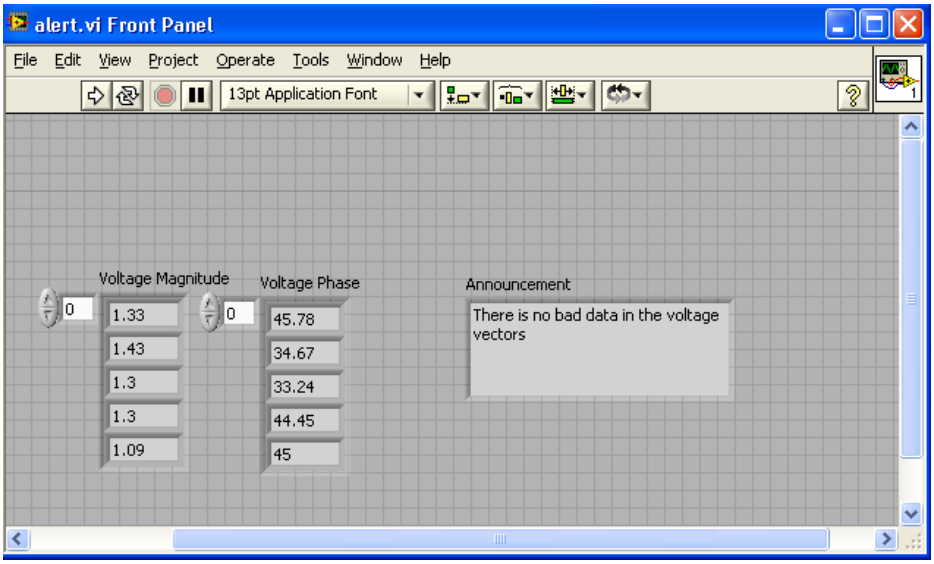


Figure 24 alert.vi Front Panel

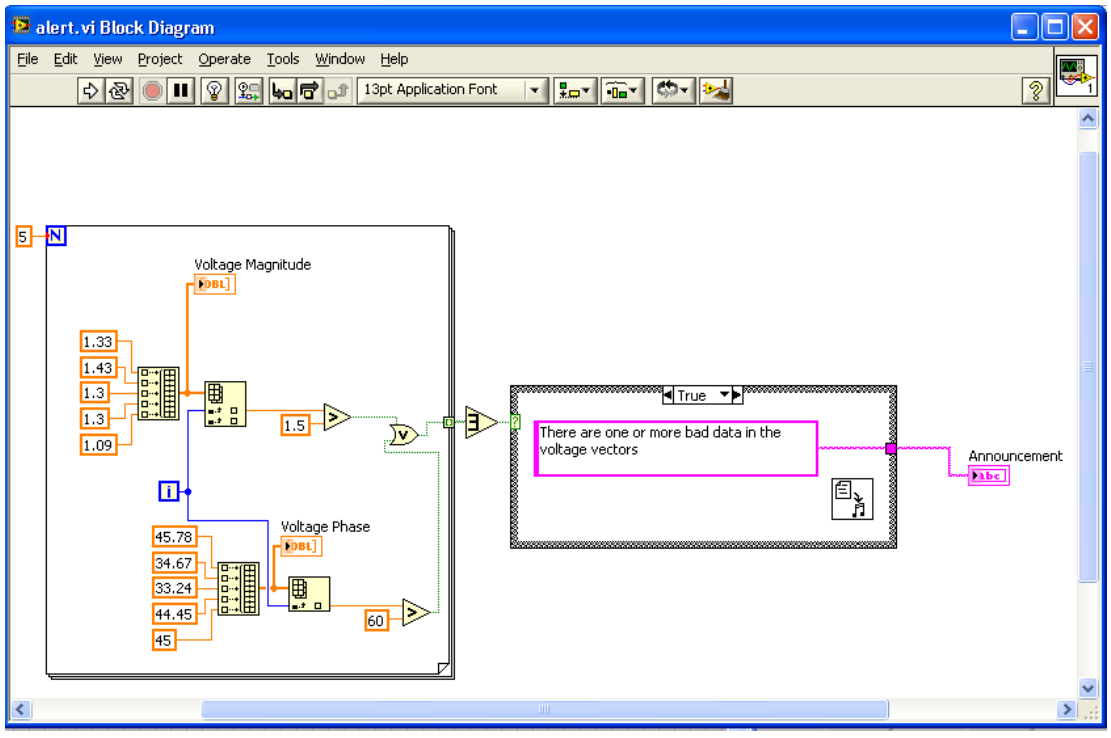


Figure 25 alert.vi Block Diagram

4.3 Discussions

4.3.1 PSSE.vi

The Database Connectivity Toolkits allowed database development using LabVIEW where database operations such as creating table, inserting, retrieving and updating data can be performed in LabVIEW environment. However, there are still several issues regarding the LabVIEW VIs that is being developed.

The issues are:

1. At this point of time, the VIs works on Microsoft Access only. It has not been tested in other databases.
2. The VIs are not stable. Every time the VIs is opened, the information contained in it will gone and need to be type again.
3. The VIs are developed independently. It is more efficient to combined all the subVIs and integrate them together as one VI.
4. Some other subVIs need to be developed such as updating data and deleting a table.

4.3.1.1 Combining subVIs to main VI

It is easier to develop some subVIs and then combined it as one VI. SubVI for creating table, inserting data, displaying data, updating data and deleting data has been developed earlier. When all the subVI works, then they are combined as one main VI. This is to ease the controlling and monitoring activity. Plus, it is easier to publish just one main VI in one web page.

4.3.2 Website Development

Since LabVIEW allows VI to be launch in HTML as image, therefore, only some wording or style will be added in the web page. Also, login page has been developed to enable the feature of accessibility for authorized user only.

4.3.3 Notification Mechanism

The notification mechanism feature is built using LabVIEW. LabVIEW has the ability to allow user to play sound through it. The notification mechanism works by comparing the data in the voltage vectors table that consists of voltage magnitudes and phases. They are compared with certain fix tolerance. If the data doesn't meet the tolerance, then the alert.vi will play the alert sound and prompt an announcement for the user to take note.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

A good monitoring system would help PSSE to be more efficient and reliable. Web-based SCADA system would further enhance the normal PSSE application in LabVIEW by adding other features to it like online access monitoring and notification mechanism.

5.2 Recommendations

There are few recommendations that could help in further enhancing the project. Those recommendations are:

1. Varies the bus system database from only 5-Bus System to more than that
2. Develops line database
3. Launches PSSE offline web-based SCADA online
4. Adds other notification method besides sound alert like SMS and email alert

REFERENCES

[1] http://en.wikipedia.org/wiki/Energy_management_system

[2] http://en.wikipedia.org/wiki/Remote_Terminal_Unit

[3] http://www.ni.com/labview/whatis/

[4] Power System Analysis, 2nd Edition, Mc Graw Hill International, Hadi Saadat

[5] http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=00575782

[6] http://en.wikipedia.org/wiki/MATLAB

[7] LabVIEW for Everyone: Graphical Programming Made Easy and Fun, 3rd Edition, Prentice Hall, Jeffrey Travis and Jim Kring,

[8] http://en.wikipedia.org/wiki/Database

[9] http://en.wikipedia.org/wiki/MySQL

[10] http://www.php.net/

[11] http://en.wikipedia.org/wiki/HTML

[12] http://en.wikipedia.org/wiki/Cascading_Style_Sheets

[13] http://www.apachefriends.org/en/xampp.html

APPENDICES

APPENDIX A

main_login.php coding script

```
<table width="300" border="0" align="center" cellpadding="0" cellspacing="1" bgcolor="#CCCCCC">
<tr>
<form name="form1" method="post" action="checklogin.php">
<td>
<table width="100%" border="0" cellpadding="3" cellspacing="1" bgcolor="#FFFFFF">
<tr>
<td colspan="3"><strong>Member Login </strong></td>
</tr>
<tr>
<td width="78">Username</td>
<td width="6">:</td>
<td width="294"><input name="myusername" type="text" id="myusername"></td>
</tr>
<tr>
<td>Password</td>
<td>:</td>
<td><input name="mypassword" type="password" id="mypassword"></td>
</tr>
<tr>
<td> </td>
<td> </td>
<td><input type="submit" name="Submit" value="Login"></td>
</tr>
</table>
</td>
</form>
</tr>
</table>
```

APPENDIX B

checklogin.php coding script

```
<?php
$host="localhost"; // Host name
$username="fyp2"; // Mysql username
$password="maryam"; // Mysql password
$db_name="test"; // Database name
$tbl_name="members"; // Table name

// Connect to server and select databse.
mysql_connect("$host", "$username", "$password")or die("cannot connect");
mysql_select_db("$db_name")or die("cannot select DB");

// username and password sent from form
$myusername=$_POST['myusername'];
$mypassword=$_POST['mypassword'];

// To protect MySQL injection (more detail about MySQL injection)
$myusername = stripslashes($myusername);
$mypassword = stripslashes($mypassword);
$myusername = mysql_real_escape_string($myusername);
$mypassword = mysql_real_escape_string($mypassword);

$sql="SELECT * FROM $tbl_name WHERE username='$myusername' and
password='$mypassword'";
$result=mysql_query($sql);

// Mysql_num_row is counting table row
$count=mysql_num_rows($result);
// If result matched $myusername and $mypassword, table row must be 1 row

if($count==1){
// Register $myusername, $mypassword and redirect to file "login_success.php"
session_register("myusername");
session_register("mypassword");
header("location:login_success.php");
}
else {
echo "Wrong Username or Password";
}
?>
```

APPENDIX C

login_success.php coding script

```
<?php
// Check if session is not registered, redirect back to main page.
// Put this code in first line of web page.

session_start();
if(!session_is_registered(myusername)){
header("location:main_login.php");
}
?>

<html>
<body>
Login Successful
</body>
</html>
```

APPENDIX D

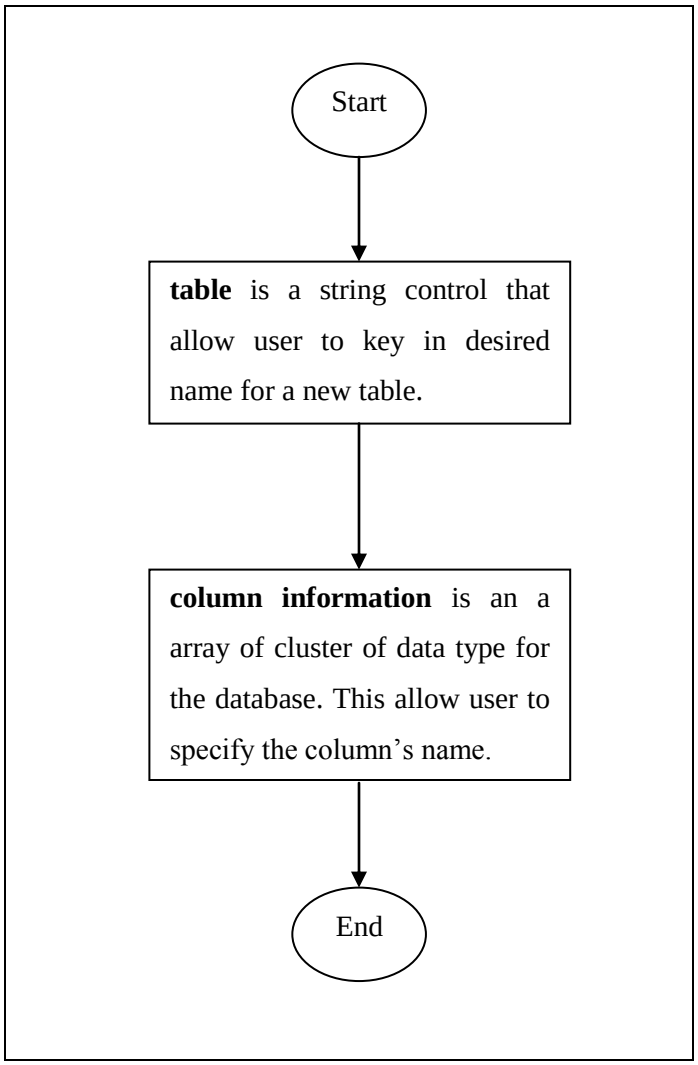
logout.php coding script

```
<?php
// Put this code in first line of web page.

session_start();
session_destroy();
?>
```

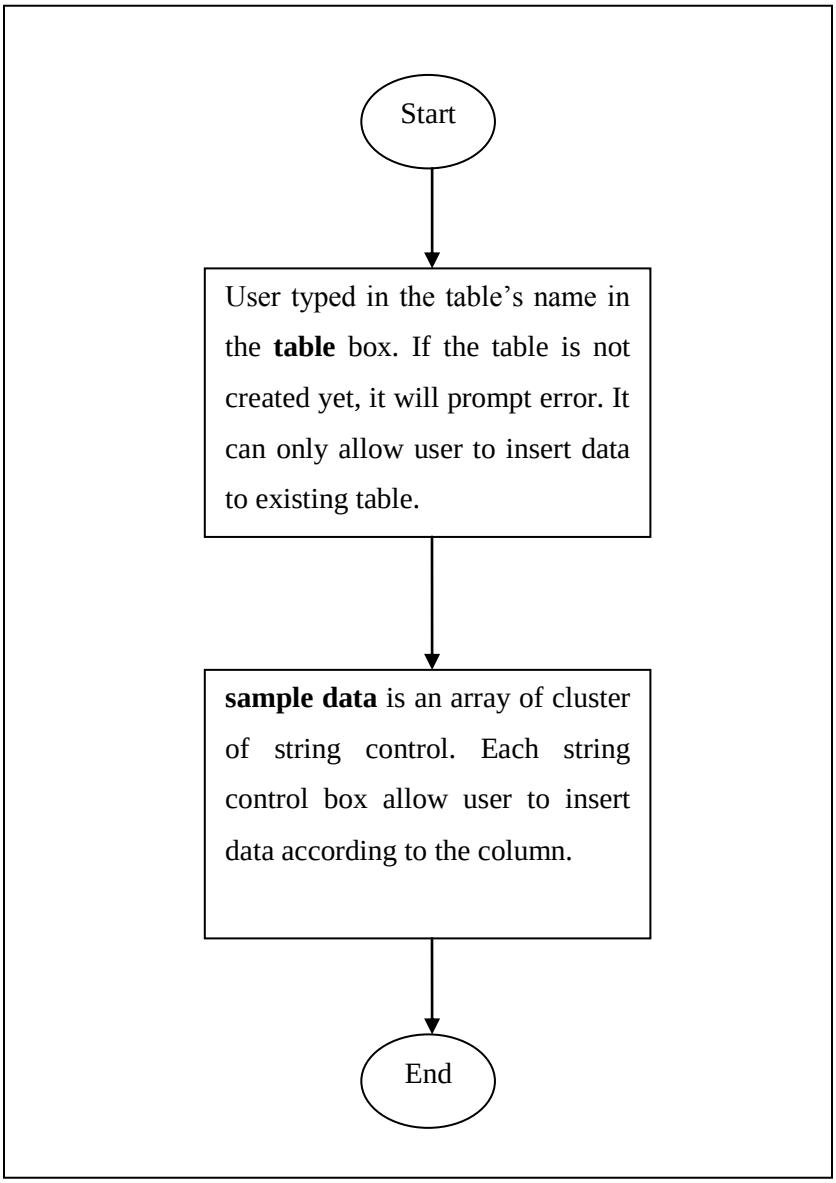
APPENDIX E

create_database.vi flowchart



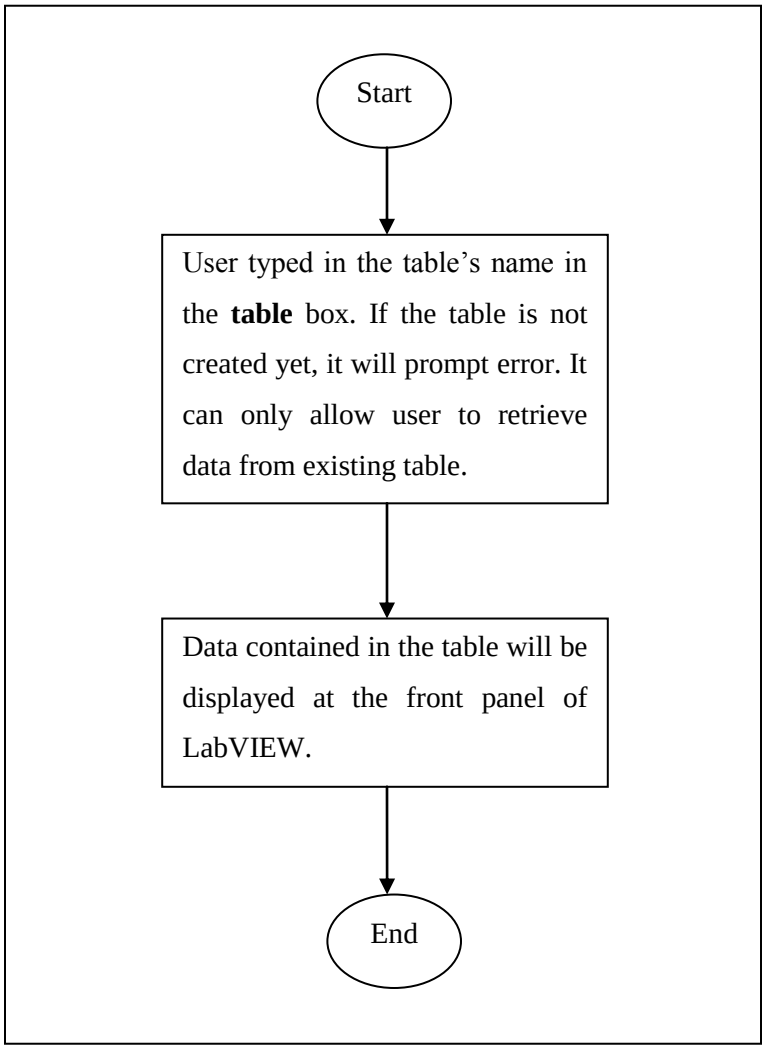
APPENDIX F

insert_data.vi flowchart



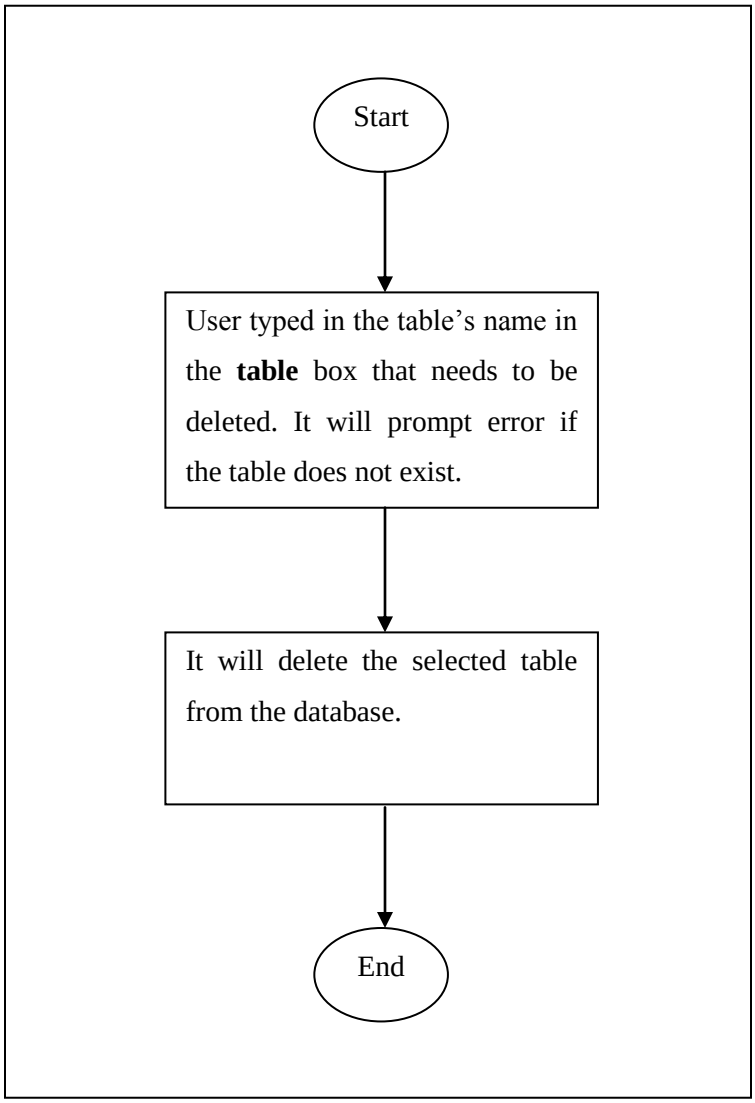
APPENDIX G

display_data.vi flowchart



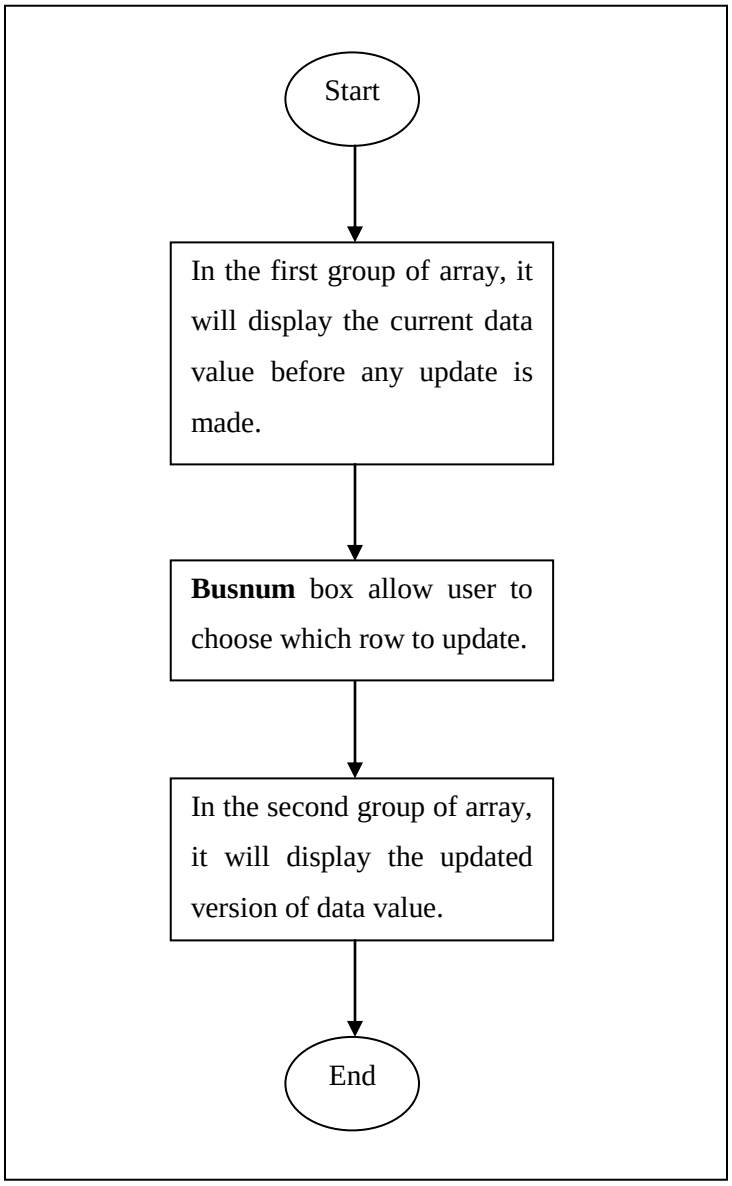
APPENDIX H

Deleting table mechanism flowchart



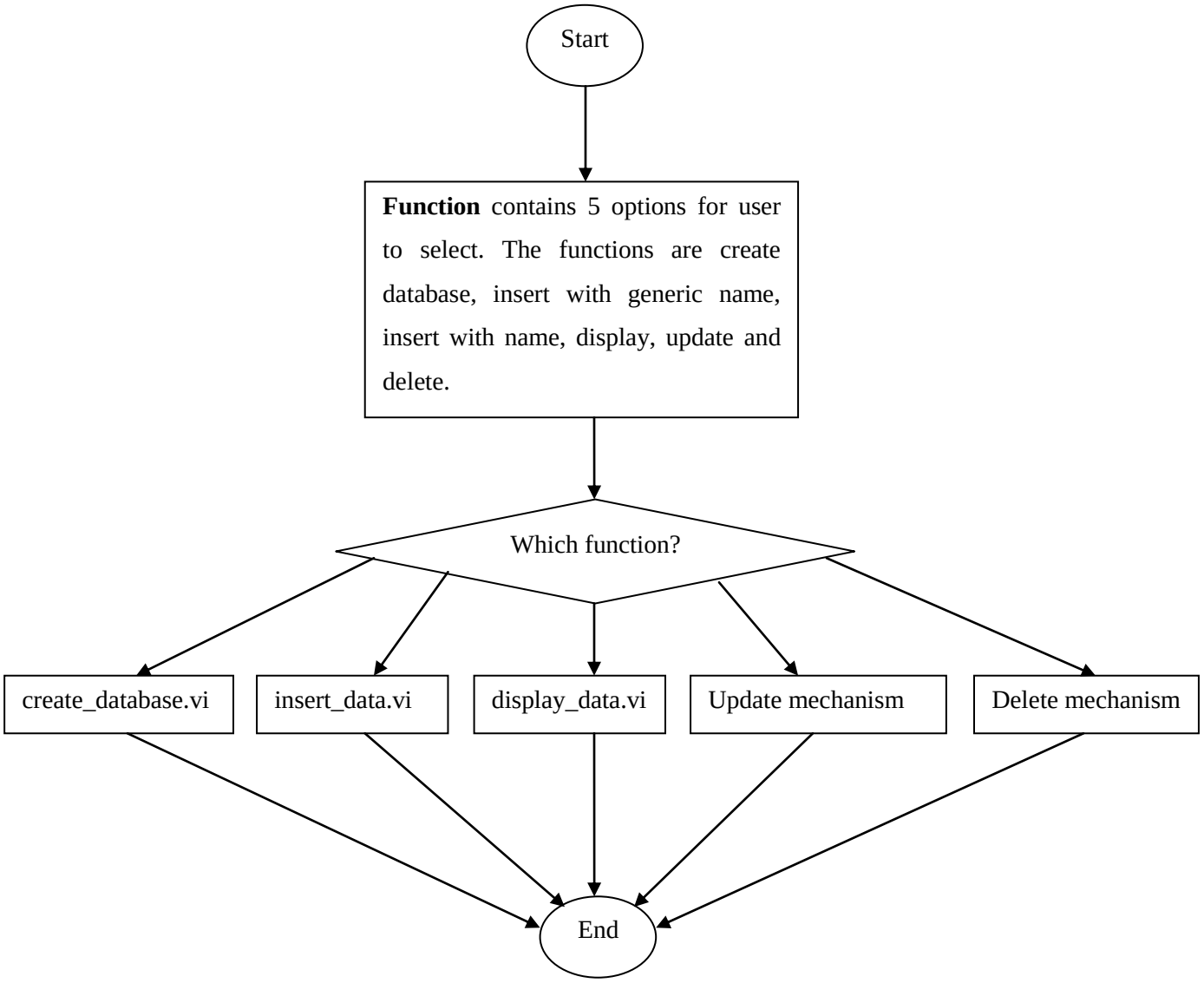
APPENDIX I

Updating data in a table mechanism



APPENDIX J

PSSE.vi flowchart



APPENDIX K

GANTT CHART FOR FYP 1

		JAN	FEB				MAC				APRIL				MEI - JUN				
No.	**Task Name**	**1**	**2**	**3**	**4**	**5**	**6**	**7**		**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16-18**	**19-20**
1	Selection & Confirmation on topic								**M**										
2	Research Reading on SE and LabVIEW								**I**										
3	Preparation for Preliminary Report								**D**									**E**	
4	Submission of Preliminary Report		●															**X**	
5	Assemble data & study on LabVIEW								**S**									**A**	
6	Learning LabVIEW programming language								**E**									**M**	
7	Submission Progress Report								**M**	●									
8	Learning LabVIEW Database Connectivity Toolkit																	**W**	
9	Develop PHP Login Script								**B**									**E**	
10	Seminar								**R**		●							**E**	
11	Debugging Login Script								**E**									**K**	
12	Preparation for Interim Final Report								**A**										
13	Submission of Interim Final Report								**K**						●				
14	Oral Presentation																		

APPENDIX L

GANTT CHART FOR FYP 2

		JUL	AUGUST				SEPTEMBER				OCTOBER				NOV-DEC				
No.	Task Name	**1**	**2**	**3**	**4**	**5**	**6**		**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16-18**	**20**
1	Resolve the problems encountered during FYP 1																		
2	Develop create_database.vi VI							**M**											
3	Preparation for Progress Report 1							**I**											
4	Submission of Progress Report 1							**D**										**E**	
5	Develop insert_data.vi VI																	**X**	
6	Develop display_data.vi VI							**S**										**A**	
7	Preparation for Progress Report 2							**E**										**M**	
8	Submission of Progress Report 2							**M**											
9	Develop PSSE.vi VI																	**W**	
10	Pre-EDX Exhibition (Wednesday Week 11)																	**E**	
11	System Testing and Fixing							**B**										**E**	
12	Preparation for the Final Report							**R**										**K**	
13	Submission Draft of Final Report							**E**											
14	Submission Final Report (Soft copy) and Technical Report							**A**											
15	Oral Presentation (week 18)							**K**											
16	Submission Final Report (Hard Bound)																		