Routing Scripting Version 2

MINT 709 CAPSTONE PROJECT REPORT

School of Electrical and Computer Engineering

University of Alberta

Submitted by

Gurpreet Singh

Gu9@ualberta.ca

Supervisor: Arsh Saini

Submitted to: Prof. Mike McGregor

March 17, 2016

Abstract

This is an attempt to create a GUI based application for Network Engineers .Most of the service provider router runs numerous protocols, as the network grows, network topologies get complex and its takes long time to analyzing the whole network by Network Admin. This tool will help users to know various routes information from service provider routers in different locations (PE's), search an IP address, interfaces status, routers performance parameters such as CPU, Memory and other details in the GUI tool fast and easy. User don't have to enter each and every command on routers CLI while troubleshooting.

This is the Python Scripting version-2, which is the extended project of Routing Scripting from 2015. This project has added functionality of more user friendly interface, shows the various protocol output in the GUI format.

Table of Contents

List of Figures

List of Abbreviations

AS	Autonomous System
API	Application Programming Interface
BGP	Border Gateway Protocol
CLI	Command Line Interface
CSS	Cascading Style Sheets
CPU	Central Processing Unit
CE	Router Customer Edge Router
CEF	Cisco Express Forwarding
EIGRP	Enhanced Interior Gateway Routing Protocol
FEC	Forwarding Equivalence Class
GNS3	Graphical Network Simulator-3
GRE	Generic Routing Encapsulation
HTML	Hyper Text Markup Language
HTTP	Hypertext Transfer Protocol
IGP	Interior Gateway Protocol
iBGP	Internal Border Gateway Protocol
eBGP	External Border Gateway protocol
IS-IS	Intermediate System Protocol
IOS	Internet Operating System
IP	Internet Protocol
IPv4	Internet Protocol version 4
ISP	Internet Service Provider
LDP	Label Distribution Protocol
LSP	Label Switched Path
LSR	Label Switch Router
LIB	Label Information Base
LFIB	Label Forwarding Information Base
MPLS	Multiprotocol Label Switching
MP-BGP	Multiprotocol Border Gateway Protocol
OSPF	Open Shortest Path First
PE Router	Provider Edge Router
P Router	Provider Router
RIP	Routing Informational Protocol
RD	Route Distinguisher
RT	Route Target
SSH	Secure Shell
SQL	Structured Query Language
TTL	Time To Live
URL	Uniform Resource Locator
VRF	Virtual Routing Forwarding
VPN	Virtual Private Network
VLSM	Virtual length Subnet masking

Chapter 1
Introduction

1.1 Overview

Network Admin Tool is the GUI application, which provide user friendly interface. In this, User can access the network parameters in order to troubleshoot or analyze the whole network fast and easy, without login into the CLI of each routers and execute commands manually. The routing tables created in the provider edge router for each VRF can be seen in the GUI .It provide functionality to search an IP address in a specific VRF and other network extraction commands from cisco vendor routers and display it on the web page application .This Document will explore the Front and Back end design of this web application, architecture design of the MPLS domain and BGP implementation and highlight some of the open source libraries of python used in Django Framework to meet the user end requirements.

1.2 Objectives and Goals

To design a network architecture for service provider and show the network parameter with Django Web application so user can troubleshoot network problems on provider edge routers with GUI interface. To Implement the MPLS protocol in the backbone of the service provider and BGP between the PE and CE routers. And finally extract network parameter and analyzing it according to the user requirement. All the extracted routers parameters has to be stored in the database.

1.3 Our Approach

To first design a network topology, implementing a MPLS which can be divided into three areas such as MPLS core runs a IS-IS protocol to form reachability to other routers in Service Provider backbone. Secondly, Provider edge routers runs a MP-BGP with other PE router in MPLS backbone .VPN has to be implemented with VRF's on PE routers .On the CE, *eBGP* has to be configured with PE routers.

Python Scripts runs in the Django framework and with SSH to the routers, network parameters will be extracted from the routers and store it into the database and finally, mapping has to be done with database and front end modules to provide a rich GUI.

1.4 Report Overview

In this first half of the document, I will explain the technologies used in this project. In the second half I will explain the GNS3 network topology, IP addressing and layer three protocols used in the topology design. Last but not least, python and Django Framework Front and back design implementation and showing the output of the working system in the end.

Chapter 2

Literature Overview

2.1 IP Addressing

In order to identify a router interface in the network, each interface has to be assigned a unique IP address. Router uses Network address and Subnet mask to forward the packet at network layer to particular destination. And if we represent an IP address and Subnet mask in binary format and performed a AND operation, it gives us Network ID of that particular IP address. In this section, I will explain about IPv4 Addressing system and Subnetting.

2.1.1 IPv4 Classfull Addressing

An IP address is 32 bit long, which gives information about host addresses and network addresses. It is important to understand both terminologies.
Consider an example of Class B IP address 172.16.1.1/24

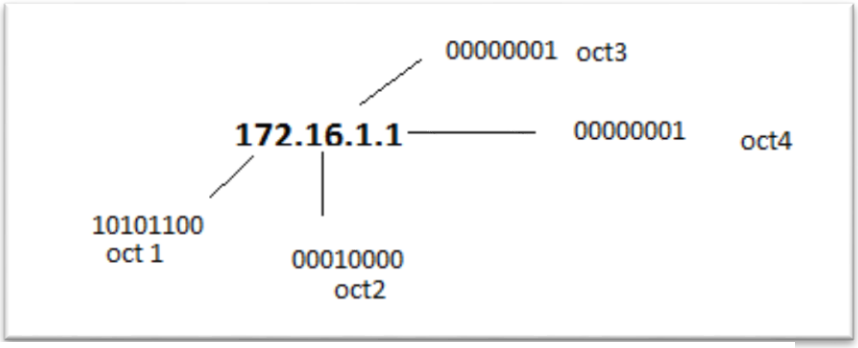


Figure 1:IPv4 Address Subnet

As each octet is 8 bit long, the first three octets represent the Network ID and oct4 represent the available IP address for the host. This can be determined from the five Classes of IP addresses.

Class	Range(First Octet)	Default Masks
A	1-127	255.0.0.0
B	128-191	255.255.0.0
C	192-223	255.255.255.0
D	224-239	None(Reserved for Multicast)
E	240-255	None(Experimental)

But now days, Only ISP provide a service of assigning a IP address to a particular Client .It is the responsibility of the ISP to allocate number of IP address according to the client requirement ,otherwise large chunk of IP address space will be useless .

2.1.2 VLSM

Class Inter Domain Routing, It provide a functionality of borrowing a bits from host and adding to the network ID, this is called subnetting. But still there is a waste of IP addresses because it offers a fixed block of subnet range in which some of the IP are not assigned to the host .For example if I need to assign a IP address to 25 hosts in the Account department and 14 hosts in the IT department, than Network Engineers uses the Class C, with borrowing bits from host and make it Subnet mask 255.255.255.224, So the fixed block range is 32 IP addresses (including Broadcast and Network address).As we can see, for Account department we need to assign 25 hosts with subnet mask of 27 and similar way for IT department 14 host are assigned with another block of 32 IP address, which cause plenty of unused IP Addresses.

With VLSM (Virtual length Subnet masking),Block size of IP addresses depend on the number of the hosts .For an Example ,IT department has 14 hosts, instead of using CIDR ,we can assign a next available block of Class C with Subnet mask of 255.255.255.240 (/28).

2.2 IP Routing Technologies

In order to provide a packet forwarding at Network layer, we need a routing protocols such as Static routing or dynamic routing .In static routing Network admin configured and advertise a static route to the neighbors routers manually .But in Dynamic routing , All the routes are learnt automatically by running a similar routing protocols (such as OSPF ,EIGRP,RIP,IS-IS ,BGP) .In this Section, I will explain, Internal protocol IS-IS and External protocol BGP basics and working Operations.

2.2.1 IS-IS

Intermediate System-to-intermediate System Protocol or IS-IS is widely used protocol in Service provider Domain. There are several benefits of IS-IS protocol. Most of them are:

- IS-IS is a Link State protocol, it has fast convergence rate as compare to Distance vector protocols like RIP etc.
- Provide functionality of Traffic engineering at Service Provider domain with MPLS.
- Easy to Scale network.

IS-IS has backbone area similar to OSPF area 0 .Backbone provide, connectivity to different areas, In case of IS-IS all the router in the backbone are L2 level routers. Within domain ,there are two level of hierarchy exits such as Level 1 ,Level 2 and Level 1-2 in IS-IS system.

The routers which are connected between two areas act same as Area Border routers in OSPF, it contain both Level tables information(L1 and L2).The routers ,which are configured to be L1 routers ,hold the information about the routing table of L1 level only.

Routing Between different areas

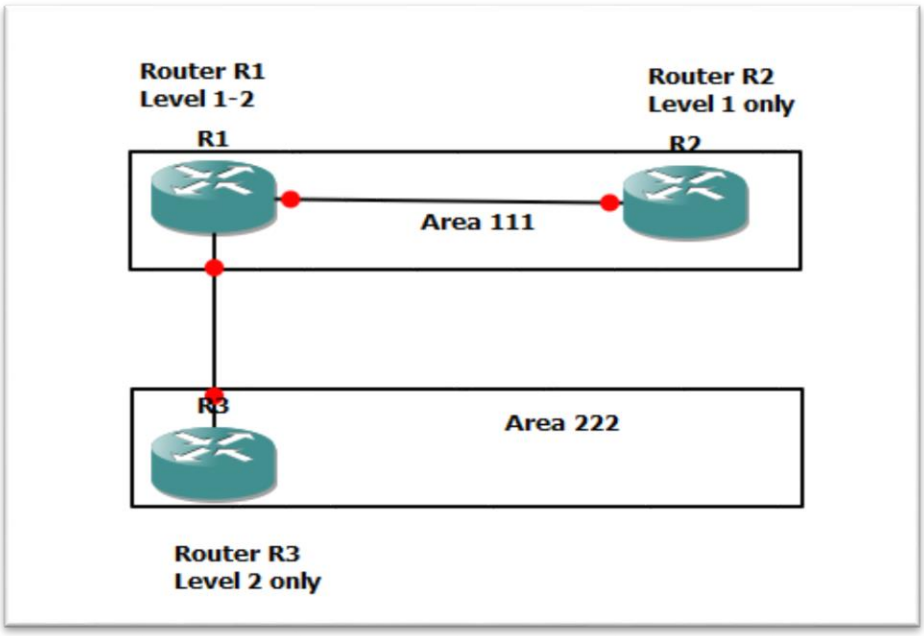


Figure 2: IS-IS Routing Different Areas

Level 1: Routing Table contain information about routes within same area.
Level 2: Routing Table contain information about routes in different areas only.
Level 1-2: Routing table has information about Both Level 1 and Level 2 routes.
IS-IS network address called Network Entity Title.NET Id has three attributes area identifier, System identifier and net selector.

For example:

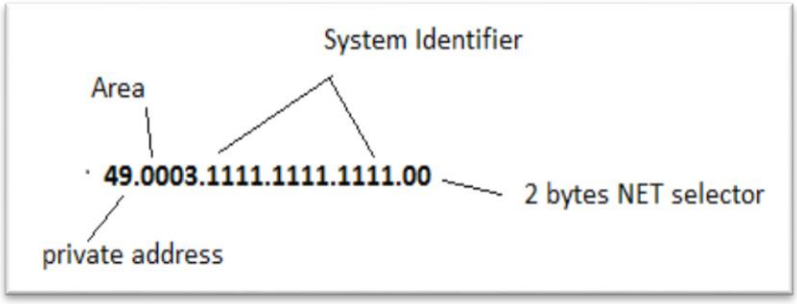


Figure 3: Network Entity Title

2.2.2 BGP

Most like IS-IS or other IGP protocols are not efficient when the routing table grows more than thousands entries. It is impossible to imagine IGP protocols such as OSPF running a whole internet in which, convergence process shuts down the whole internet while updating table entries.

Benefits of Border gateway protocol

- Traffic Engineering
- Transit network
- Support large table entries

BGP has three tables

- Neighbor Table
- BGP Table-All prefix learned.
- IP routing Table-only best routes from the BGP Table.

2.2.2.1 BGP Neighbor Relationship

BGP discovers other BGP neighbour with TCP port 179.Unlike IGP protocols, where hello packet send back and forth between the two routers in order to exchange the information, in BGP Network admin manually define the neighbours.

2.2.2.2 Internal BGP Peers

BGP connectivity between two neighbors can be *iBGP* if two peers are in the same autonomous number.*iBGP* needs a full mesh topology between all the peers reside in same autonomous number to prevent the loop.

2.2.2.3 External BGP Peers

eBGP peers are usually directly connected in the physical topology scenario .Basically this connection is between two different autonomous systems ,where TTL value is equal to 1 or more .*eBGP* has its own loop prevention mechanism different from *iBGP*.

2.2.2.4 BGP Neighbor States

There are five states of TCP connection which are Idle, Open, Open confirm, Active, Established. If the state of the peer is active, this means TCP session failed or neighbors is not formed between two BGP routers. It can be debug with cisco command *debug ip bgp*.

2.2.2.5 BGP Message Types

BGP uses four basic messages which are Open, Keep alive, Update, and Notification.

2.2.3 MPLS

MPLS or Multiprotocol Label Switching Protocol forward a packet based on label as compare to IP network where router look up for IP address in CEF but in case of MPLS, routers see the label entry in the LFIB.The reason behind the popularity of MPLS is that, it separates the concepts of control plane and forwarding plane. Control plane is Connectionless and it can be implemented with IP routing protocols (example OSPF.IS-IS). On other hand, forwarding plane is connection oriented and provide packet switching at Layer 2.MPLS does not fit in the OSI model ,it provide encapsulation of the packet at LSR routers(which is basically layer 2 functionality) and forwarding of packet based on Labels. And the LSR at egress map the Label with the specific IP address while leaving the Service provider domain .So it does not use layer 3 at all in MPLS domain. In the end, we can say hypothetically, it is a 2.5 layer protocol. In this section, I will explain MPLS basic concept and its application such as MPLS VPN architecture.

2.2.3.1 Label

MPLS label is 32 Bits long, It has four fields such as Label, EXP, BoS and TTL.

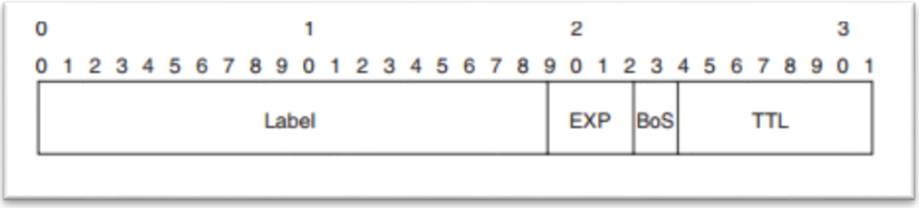


Figure 4: MPLS Label

- Label -The first 20 bits represent the Label value, which ranges from 0 to 20^{20}-1.
- EXP-This is an Experimental field, it ranges between bits 20 to 22.Mostly used for Quality of service.
- BoS-It stands for Bottom of Stack, its value is 0 or 1.
- TTL-Its value ranges from 0-255, its functionality is similar to IP network, where its value keep on decreasing when packet reaches from one hop to another. It is 8 bits long.

2.2.3.2 LSR

In the service provider domain, routers in which MPLS is configured are called label switch routers or LSR. These routers perform several operations on the label such as (push, pop, swap) and forward the packet based on the label header .There are three types of Label Switch routers.

- **Ingress LSR**- These are the routers at the edge of the MPLS domain, where they received an IP packet from Customer site .Router perform the push operation ,where a MPLS label is attached and IP packet is encapsulated .
- **Intermediate LSR**-These routers performs a Swap operation and forward packet to correct destination LSR.
- **Egress LSR**-These routers performs pop operation on the label at the edge of the MPLS domain. The label has to remove when it sends to the specific customer site.

2.2.3.3 LSP

Label switched path is the path in the MPLS domain, that packet follows until it reached the edge router where egress router remove the label. Ingress router can be any router which push label to the IP packet and intermediate LSR forward these encapsulated packet to the next hop.

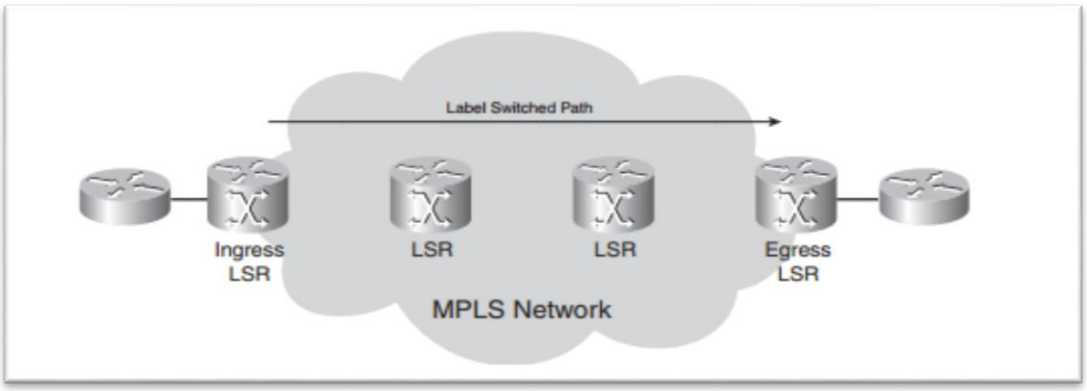


Figure 5: ISP Label Switch Path

So, the path followed by the packet in the service provider domain, between two different LSR is called LSP. The flow of LSP is unidirectional between ingress to egress routers, where each node is LSR.

2.2.3.4 FEC

The packets with similar destination address, destination port and source port are treated to the same FEC or Forwarding Equivalence Class in the MPLS. The FEC will be same for the packet received at ingress LSR which assigned the label that belong to that FEC. In this way different destination has different LSP based on the destination address and other parameters.

2.2.3.5 LDP

In order to distribute Label to the neighbour LSRs, there is a need of protocol similar to IGP protocols which does local and remote bindings of prefix. IETF created a protocol called LDP, which is supported on all the vendors routers .It holds two separate tables for local and remote bindings and assigned a unique label number on it. Binding is created for each interface of the LSR. Binding number is unique so local binding on peer router for the same label can be different from the receiving LSR local binding.

2.2.3.6 MPLS VPN Overview

In traditional VPN based networking possible with Point to Point Tunneling Protocol, Layer 2 Tunneling Protocol and GRE (Generic Routing Encapsulation) .Basically VPN or Virtual Private network is a private network of one company which has two or more branches separated globally and these branches are connected through VPN through public internet infrastructure. In frame relay model all the branches needed to be full mesh with all the branches and with the ISP for the internet connection**.** In particle this model takes more time to configure (because of security and IP addressing)and not so scalable .It need all the service provider router to learn about customers routes ,which is IP address lookup at layer 3 ,which is done by BGP. All the routes are stored in One table of different VPN sites. So it does not provide efficient solution and required more configuration and need to consider security parameters .It also cause conflict of same IP addressing at ISP, if two VPN sites has same addressing scheme.

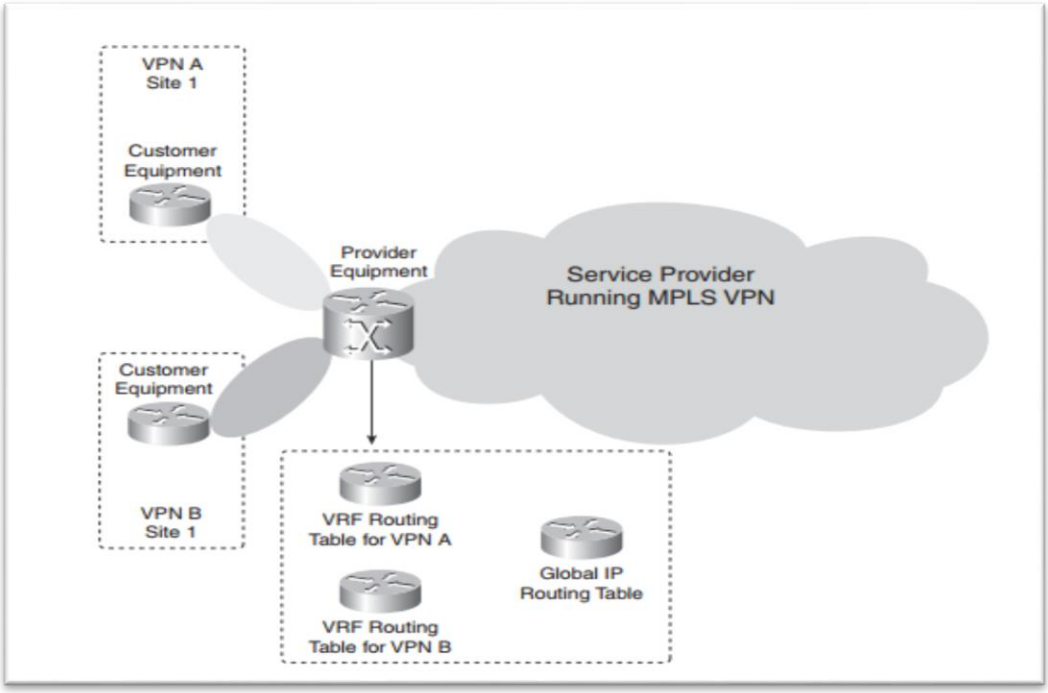


Figure 6: PE Routers VRF's

But in case of MPLS VPN based model, there is a VRF (Virtual routing forwarding) for each VPN site. Only Provider edge routers need to peer with CE routers virtual interface .Intermediate LSR routers in the MPLS domain does not required the information of customer IP addresses, because of the label forwarding. So, there is no need of BGP at the P routers, only edge routers ingress and egress LSR need the BGP peering in order to do IP lookup and mapping to label of the customer routes.

In the above Figure PE router is being connected to VPN Site A and VPN Site B with Layer 3 IGP protocols .PE router has two separate table entries for VPN A and VPN B which is possible with VRF.
Virtual routing forwarding provide functionality of separating different VPN sites IP addresses. The PE router exchange this information with other LSR through LDP and peering with other PE routers in the MPLS domain with BGP extended community. P routers provide packet forwarding only.

2.3 Python

Python is a high level object oriented scripting language .Python has in built data structure such as List, Dictionaries and tuples and standard libraries. Code written in python is more readable as compare to other Programming languages like C or C++.There is a various open source python libraries available, which provide user to do programming in Web Framework, Linux scripting and desktop GUI applications.

2.4 Django

Django is a model-view-controller (MVC) framework, It is written in python. It is lightweight Web programming Framework for rapid development of Web App. It provide backend support with MySQL, SQLite or various other databases and front end with HTML and CSS or bootstrap.

There are several components of Django Framework.

View- This is the brain of the Django framework where, URL's are mapped to the methods in the view.py file. It takes a web request and returns a web response. This response could be HTML, Redirect, Error 404 or image.

URL- URL's are a table of contents for Django-powered website. HttpRequest comes from once the pattern has matched with the user request. URL's basically mapped with views method and view return the response back with business logic.

Model- This is the business logic of the Django, where user creates database and connect it with view.py by importing model.py file to views.

As we can see these components are independent to each other before implementation ,This is the perfect model for light weight application ,where all the Web designer can work on the HTML and CSS where backend developers works on model independently .And in the end the project can put together to complete the Application.

2.5 Web Technologies

In this project I have used HTML, CSS and JavaScript to provide GUI and client side validation.

2.5.1 HTML

HTML is Hyper Text markup language provide the basic structure of the web page with various tags such as <p>,<form> or many other features such as adding images ,videos etc.

2.5.2 CSS

Cascading styling sheets provide functionality of styling webpage such as background color of the body elements, styling of links, menu and other page layouts. Now days bootstrap provide in built basic CSS styling of the web page that user can modify according to the needs, this speed up the web development process .

2.5.3 JavaScript

In Django , there are python classes which provide form validation at the server side .In order to provide client side validation , a client side script is included in HTML page ,so when user submit the form with incorrect field or blank field ,it shows the error or redirect to the form page. So, there will not be any server validation required from Django Framework.

I have used *parsley.js* to validate forms. Parsley is JavaScript form validation library. Parsley automatically detect the changes in the form and provide the validation. There are numerous validators available with Parsley to validate forms and it is an open source MIT licensed.

2.5.4 Database

Models in Django support various backend software's such as SQLite (default), PostgreSQL and MySQL.

Django libraries make it easy to write SQL queries to retrieve information from database. Django retrieve values in the form of python object, of all the entries of the database and which can be manipulated or used according the programmer needs. The implementation details of MySQL is describe in the section chapter 4 with Django Framework and code review.

For Example:

- *Model_name.objects.all ()* -This query will return all the database entries in the form of objects.
- *Model_name.objects.get (variable_name=”key_to_search”)* - This query will return the specific matched key value in the database columned name “variable_name” and returns a multiple or single matched objects. If above query return nothing, it raised an ObjectNotFound error.

2.5.5 Eclipse IDE

Eclipse is an integrated development environment for major programming languages such as Java, C, C++, python and various more. Eclipse has various plugin available according to the programmer needs. In this projects, I have to install the Pydev plugin from eclipse official website and add a python PATH variable in windows environment. Basically, Eclipse provide a common integrated path for GUI, Web and backend application. Each of the modules are connected to each other with different plugins.

All the basic python libraries and backend software need to be installed before this step.

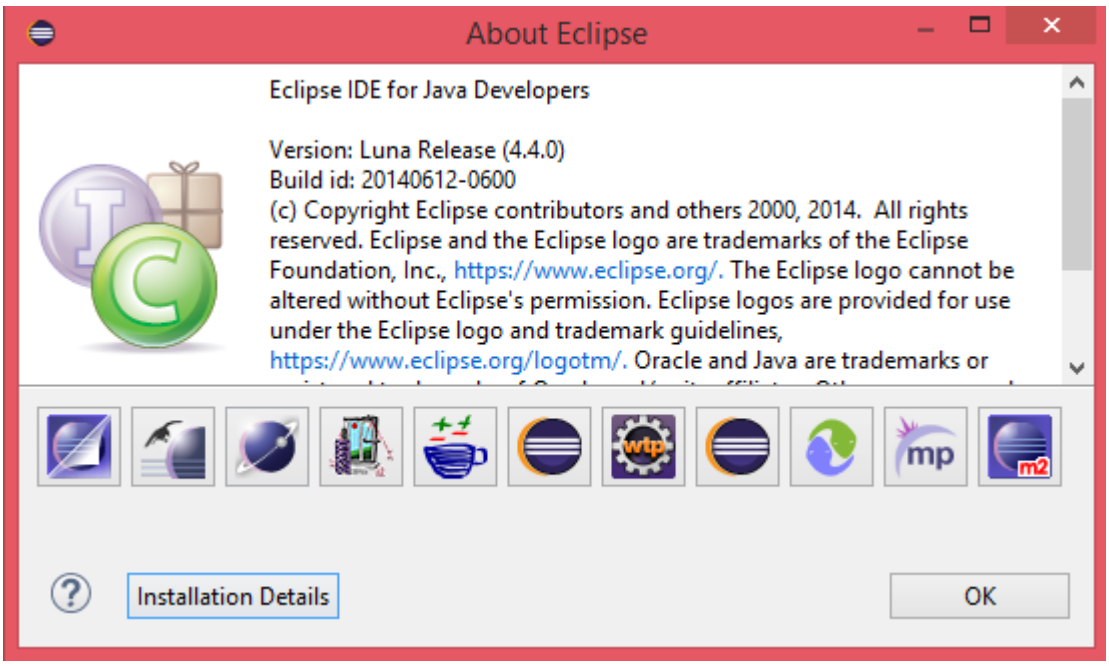


Figure 7: About Eclipse

Eclipse IDE use to develop or debug various programs written in different languages .In order to run python source code we need to install Pydev Plugin for eclipse extension to execute python code in Eclipse IDE. In this section I will explain basic setup required for any Django project in eclipse IDE.

1. Execute eclipse.exe and set the workspace location to save the project.

2. While browsing the eclipse, click on the help tab and choose the new software install option in drop down menu .it will pop up the window like shown below.

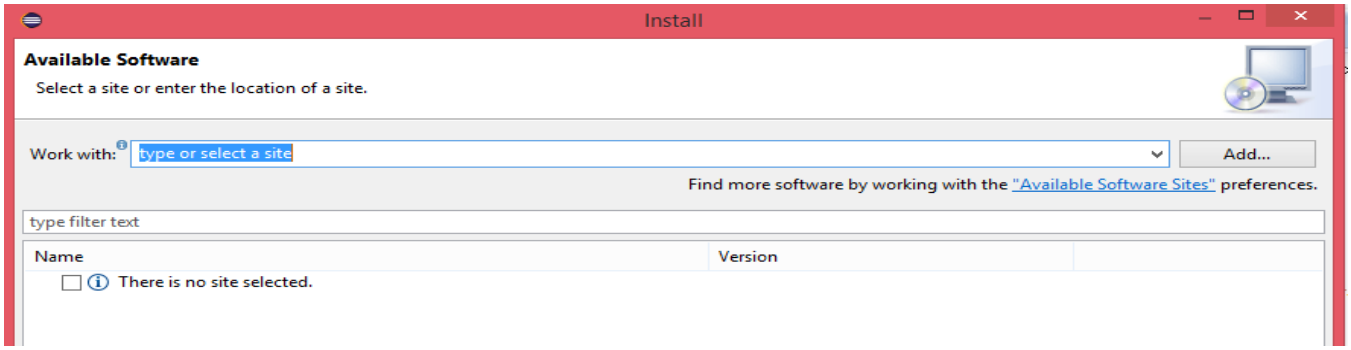


Figure 8: Install Software Package

Click on the add button

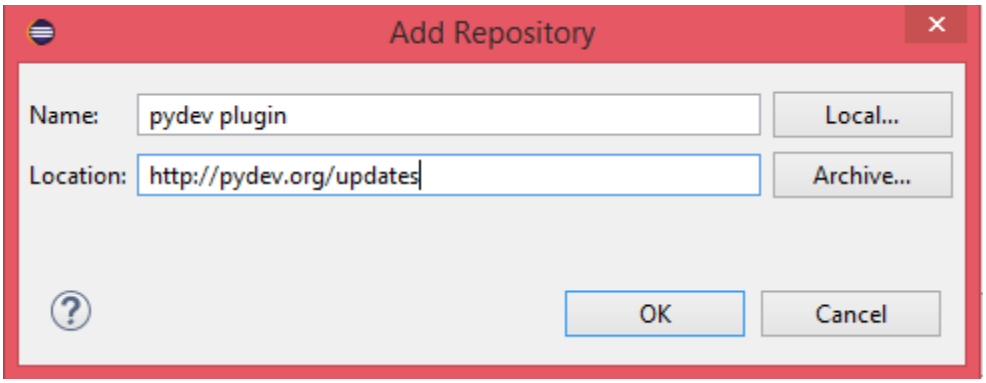


Figure 9: Add Repository

Input the site location http://pydev.org/updates and name field.

3. Select the PyDev and optional package for installation and press the next button.

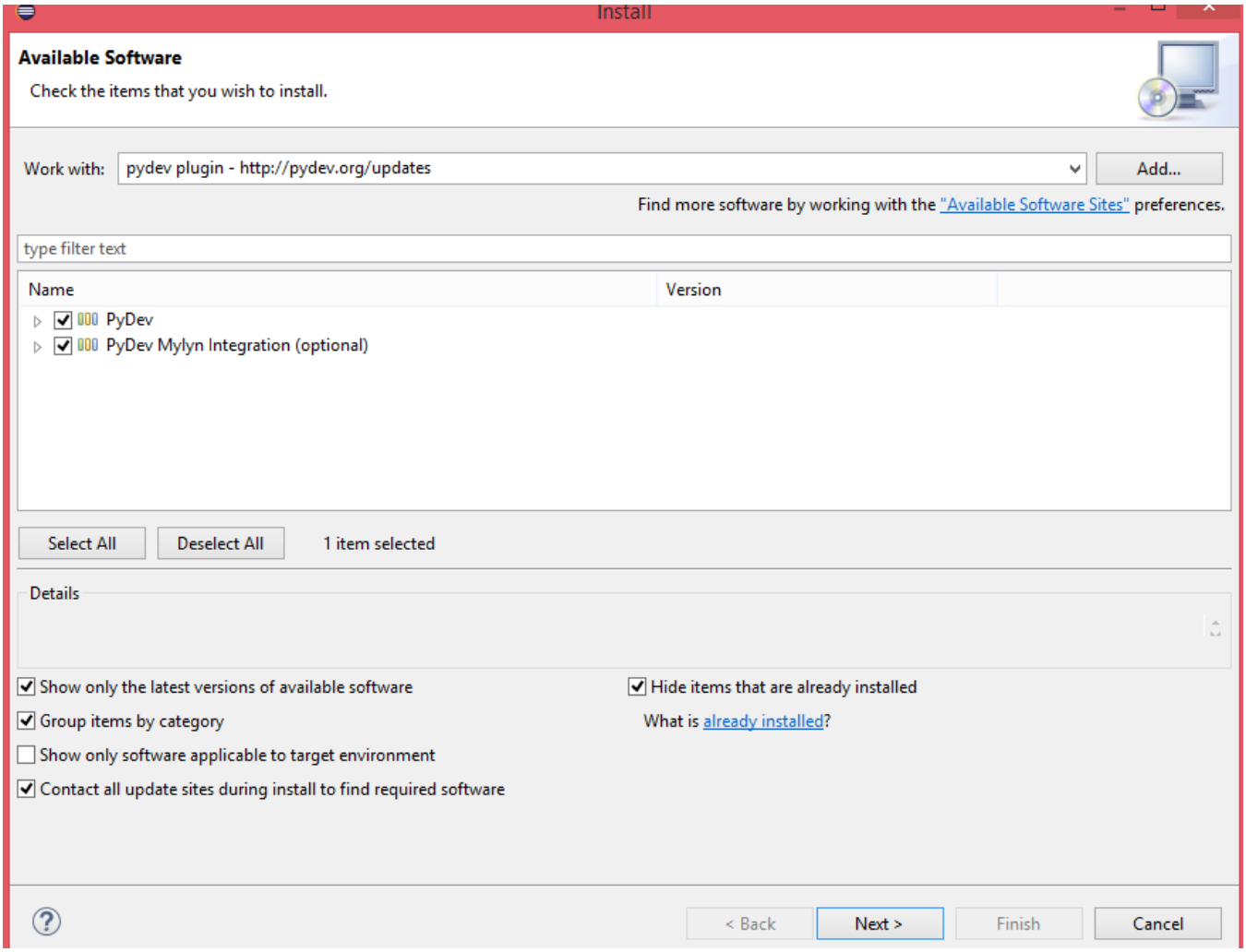


Figure 10: Available Package to Install

Now when you complete the above process, you can make your first PyDev project with eclipse by navigating to new->other.

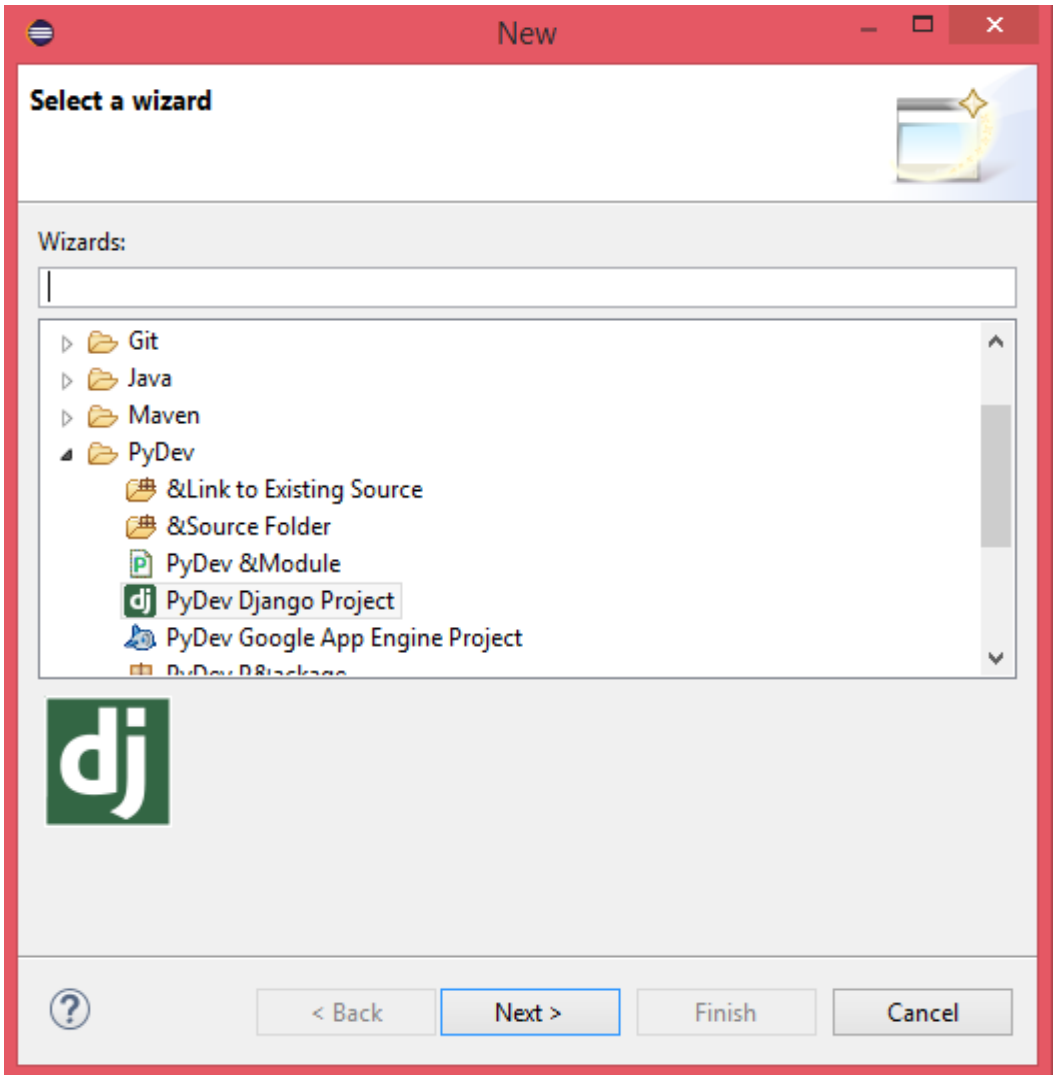


Figure 11: New Project

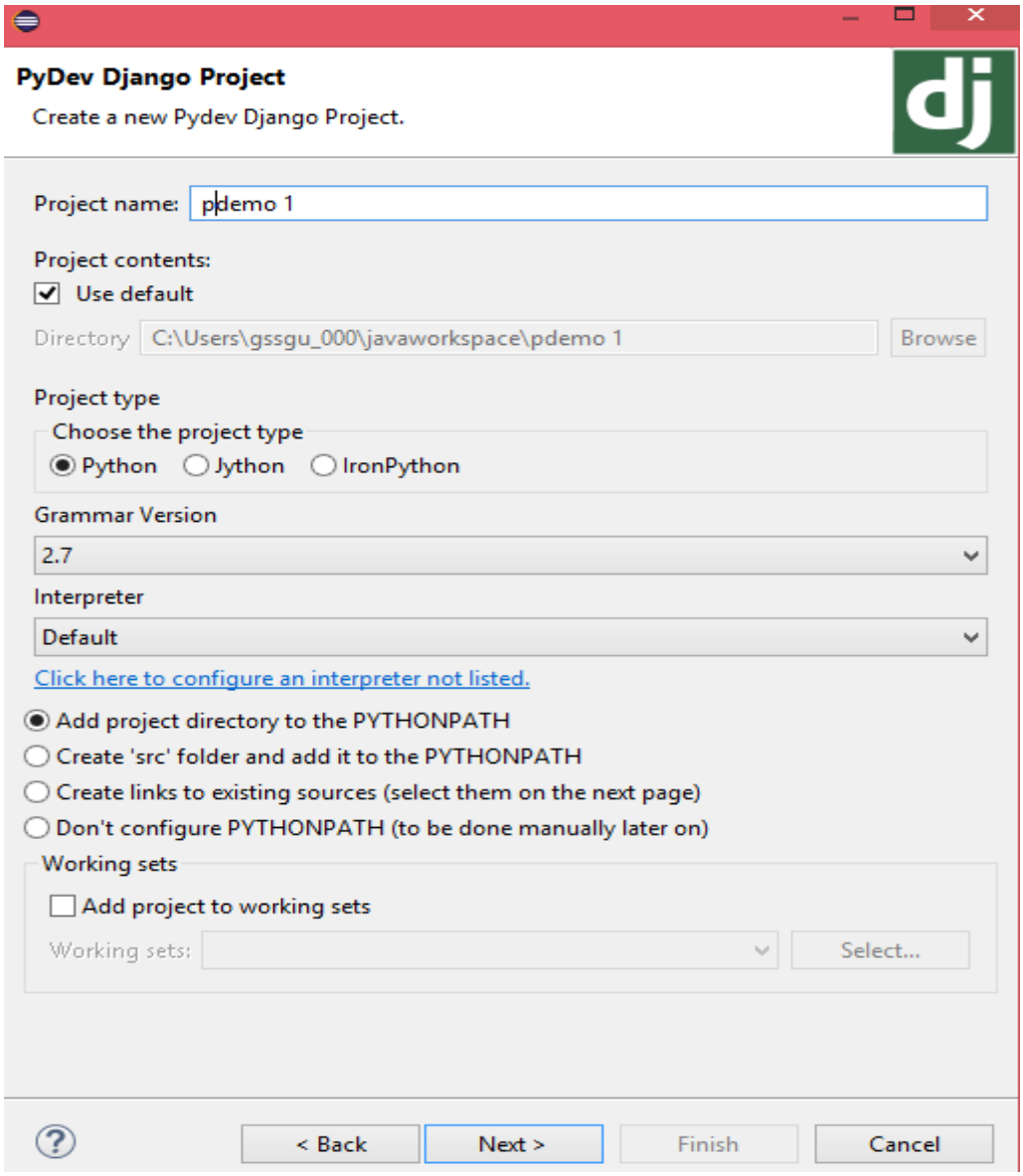


Figure 12: New Django Project

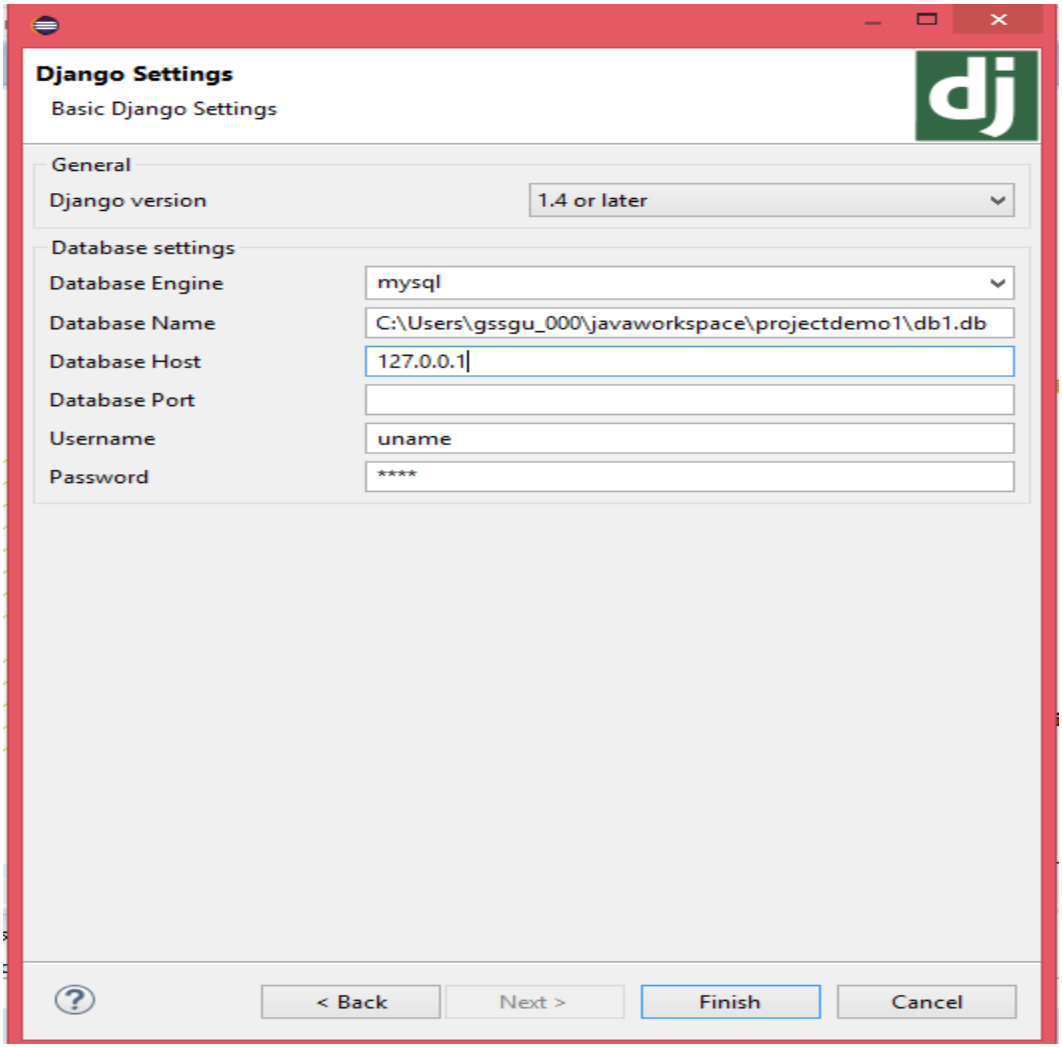


Figure 13: Django Settings

Database engine must be MySQL and input the database name you have created in MySQL.

4. Create first Django app

This can be done by right click to the Django project (pdemo1) and navigate to the create application option and it will pop up input this window.

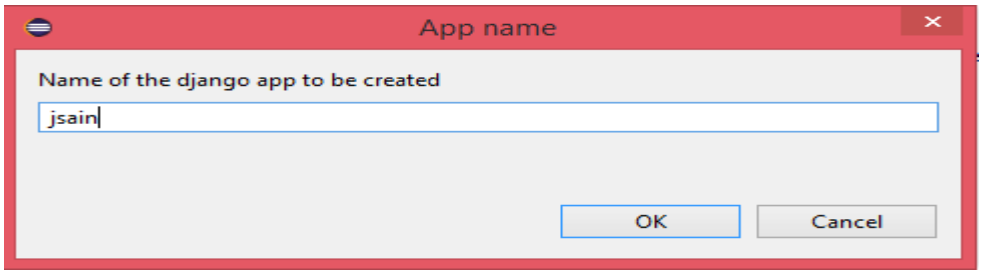
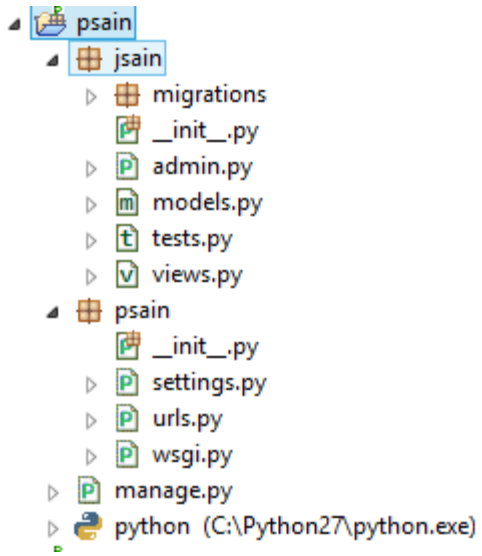


Figure 14: App Name

This will create a sub branch of jsain including the view.py, model.py and other python classes required for development of this app.



5. Settings.py file has other middleware classes information which may or may not be required for the projects and it can be changed by deleting the code from setting.py file. We can add the HTML File, CSS, and JavaScript and assigned the location of the local directory in the settings.py file .For example changing the MySQL settings if you want to change the database name, it can be done in setting.py file.

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'db2',
        'USER':'root',
        'PASSWORD':'1234',
    }
}
```

Figure 15: Django Setting.py Code

In the section 4.0 I will explain the setting.py code according to my project requirement and other implementation of the project.

Chapter 3
Design and Simulation Setup

3.1 Network Topology

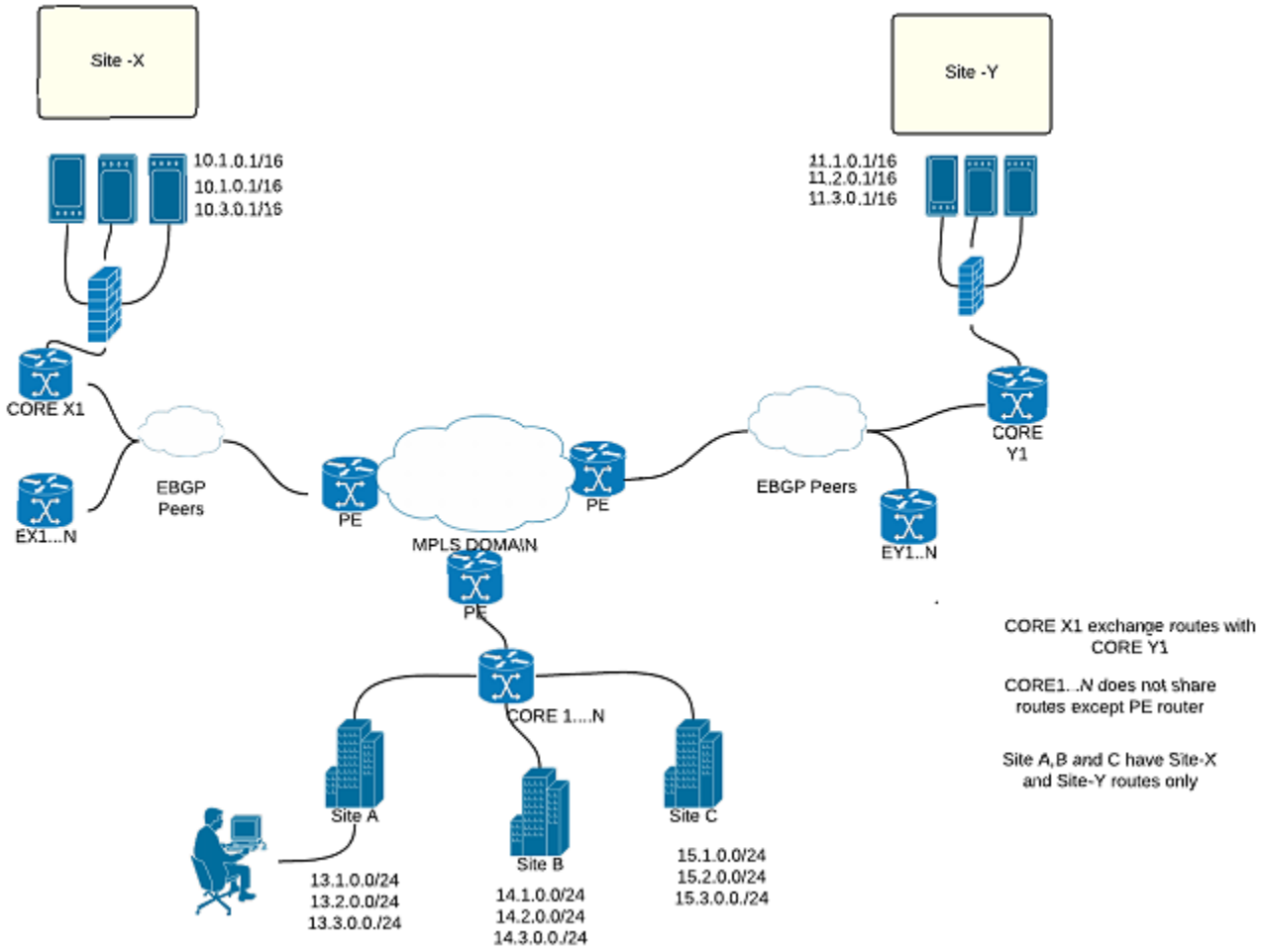


Figure 16: High Level Network Design

As in the above figure, company has two remotely connected web file servers, which can accessible from all the remote Sites of the companies A, B and C. Routers Core X1 and Core Y1 have all the routes information about Site-X and Site Y networks and all the routes of Site A ,Site B and Site C respectively. However, each Site does not share any route information to the other sites, they can be reachable to Site-X and Site-Y networks through PE router. In the service provider area, IS-IS is implemented to provide reachability to other MPLS peer routers. *MP-BGP* is configured between the edge routers respectively, and VRF_A, VRF_B, and VRF_C.

In order to connect two remote Web server Sites (X and Y), BGP is configured between PE and CE routers. In the above design CE routers run BGP between the intermediate CE routers and share information with Core X and Core Y respectively, I will explain more about the topology in GNS3 section.

3.1.1 GNS3 Network Topology

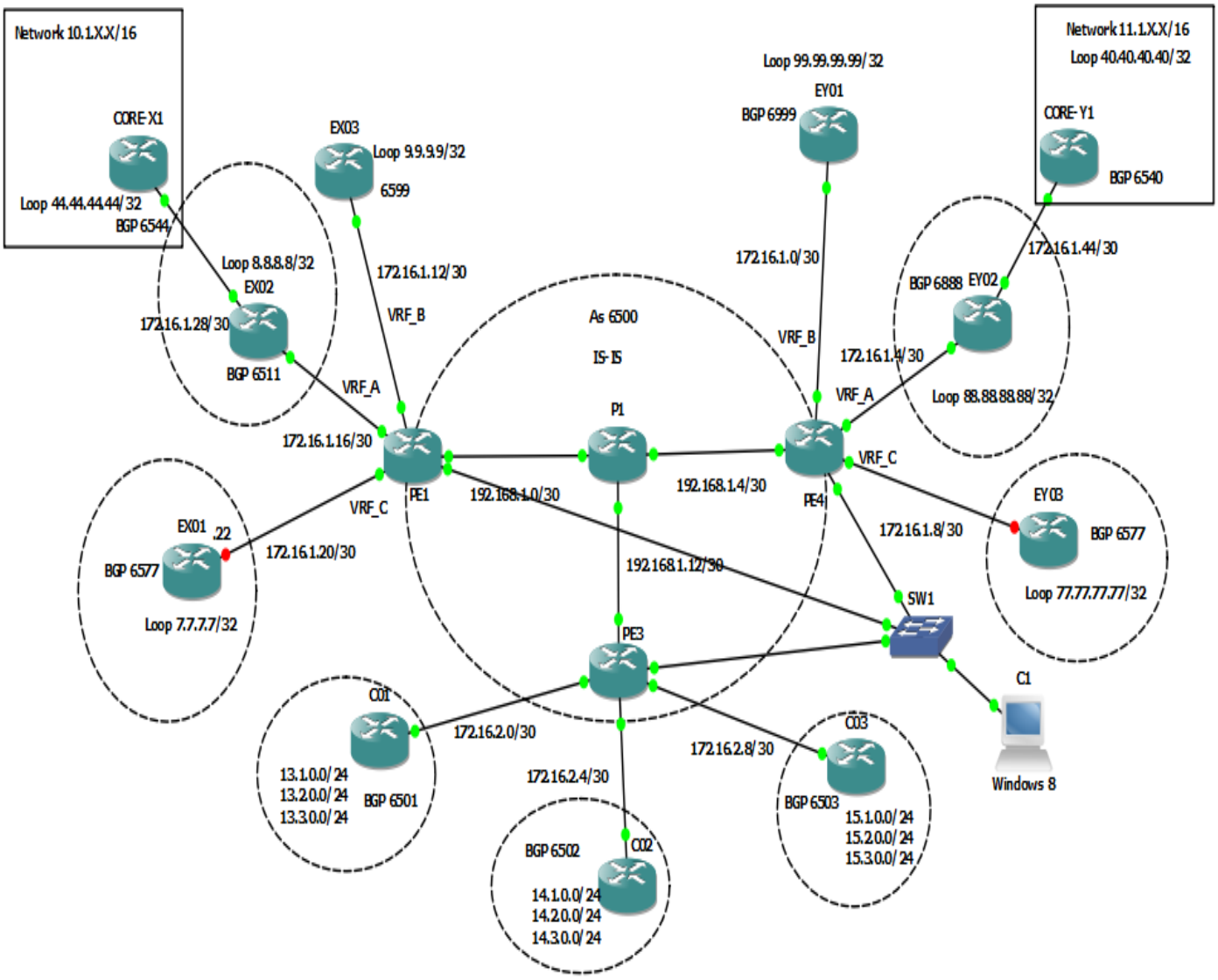


Figure 17: GNS3 Network Design

In the MPLS domain IS-IS and BGP 6500 is implemented. In MPLS, we have configured several VRF's and import/export the RT's to the other VRF's.VRF_A is connecting internal company network Site X and Site Y .In which we are importing route targets of Site A ,Site B and Site C to the PE1 and PE4 routers .In similar manner,PE3 router is importing RT's from VRF_A.

PE1 and PE4 MPLS follows the below RD's and RTs values

VRF name	RD	Import	Export
VRF_A	1:100	1:100,1:6,2:6,3:6	1:100
VRF_B	2:100	2:100	2:100
VRF_C	3:100	3:100	3:100

PE3

VRF name	RD	Import	Export
Site_A	1:6	1:100	1:6
Site_B	2:6	1:100	2:6
Site_C	3:6	1:100	3:6

The P1 router in the Middle of the service provider domain does packet forwarding based on the label tag, so there is a no need of configuring VRF's at P routers. VRF are only implemented on the edge routers or Ingress or egress router of ISP. Some of the core design requirements are given below.

CORE X1 Directly connected network -> (10.1.0.0/16,10.2.0.0/16,10.2.0.0/16).
CORE Y1 Directly connected networks-> (11.1.0.0/16,11.2.0.0/16,11.3.0.0/16).
CORE X1 router has routes of C01, C02 and CORE Y1.
COREY1 router has routes of C01, C02 and CORE X1.
C01 has only routes of CORE Y1 and CORE X1 (excluding the routes of C02 and C03).
C02 has only routes of CORE Y1 and CORE X1 (excluding the routes of C01 and C03).
C03 has only routes of CORE Y1 and CORE X1 (excluding the routes of C02 and C01).

3.1.2 IP Addressing

Nodes	**IP**
COREX1- EX02	172.16.1.28/30
EX02-PE1	172.16.1.16/30
EX01-PE1	172.16.1.20/30
PE1-EX03	172.16.1.12/30
PE1-P1	192.168.1.0/30
P1-PE4	192.168.1.4/30
PE4-EY01	172.16.1.0/30
PE4-EY02	172.16.1.4/30
PE4-EY03	172.16.1.8/30
EY02-COREY1	172.16.1.44/30
P1-PE3	192.168.1.12/30
PE3-C01	172.16.2.0/30
PE3-C02	172.16.2.4/30
PE3-C03	172.16.2.8/30

3.1.3 BGP Scenario

BGP AS Number	**Nodes**
6500	PE1,PE3,PE4
6544	COREX1

6511	EX02
6599	EX03
6577	EX01
6999	EY01
6888	EY02
6540	COREY1
6577	EY03
6501	C01
6502	C02
6503	C03

3.2 Django Setup

To build Django app in windows machine, essential software need to be installed in the system.

3.2.1 Python

Most of the UNIX system comes per-installed with python 2.7, in windows we need to download the binaries file of python 2.7 from official website. After installation, the global variable has to be set

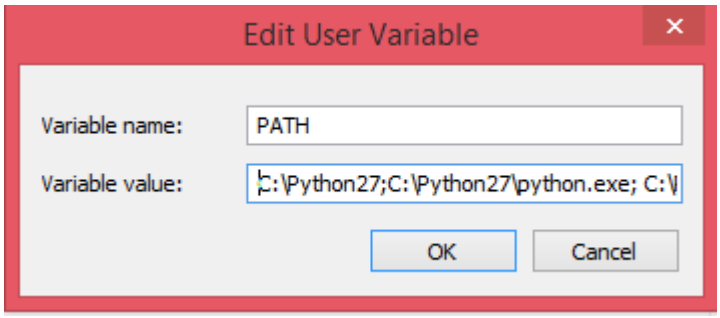


Figure 18: Environment Variable

Python has a package Manager software such as pip, easy_intall , from which we can install or upgrade various python packages from online python directory.

3.2.1.1 Paramiko

It is an implementation of SSHv2 written in python. Paramiko API has server-client architecture, in which a secure channel is established between two Nodes. When user connected to the server, server look for the host keys in the local directory. If the host key is not found in the list than Auto reject policy of the paramiko raise an exception. But In my project I have used *AutoAddPolicy()* ,so whenever user connect to the remote host, host key will be added automatically. Default behavior of the paramiko is to reject the unknown host connecting to the server and raise an exception.

Paramiko API provide handlers such as for Input/output, these are stdin,sout,stderr. For executing the command on the shell, ssh_*object.exec_command()*.

3.2.2 Virtualenv

Most of the projects now days developed with a virtual environment.it provide an isolation to the different packages or libraries running on the same system. For example one of the python project runs python 3.0 and Django version 1.9 and other python projects runs python 2.7 and Django 1.5 libraries and packages. If we don't have a virtual env variable set, we can run into the problems of different version of python or Django.So virtualenv is a container in which we can install specific libraries which are required for the python project, provide isolation to the other libraries package installed on the same system.
To install virtual environment there is a command.

pip install virtualenv

3.2.3 MySQL Setup

Python default backend support is SQLite, In order to install MySQL, we need to install the MySQL distribution such as MySQL 5.7 in my case, it can be install easily with execution of setup.exe file in windows.

Python need to connect with the MySQL, so there is a python connector package which can be installed with

pip install MySQL-python

In order to login into the MySQL database username and password is required which can be set on installation of MySQL.

3.3 GNS3 and Eclipse IDE Connectivity

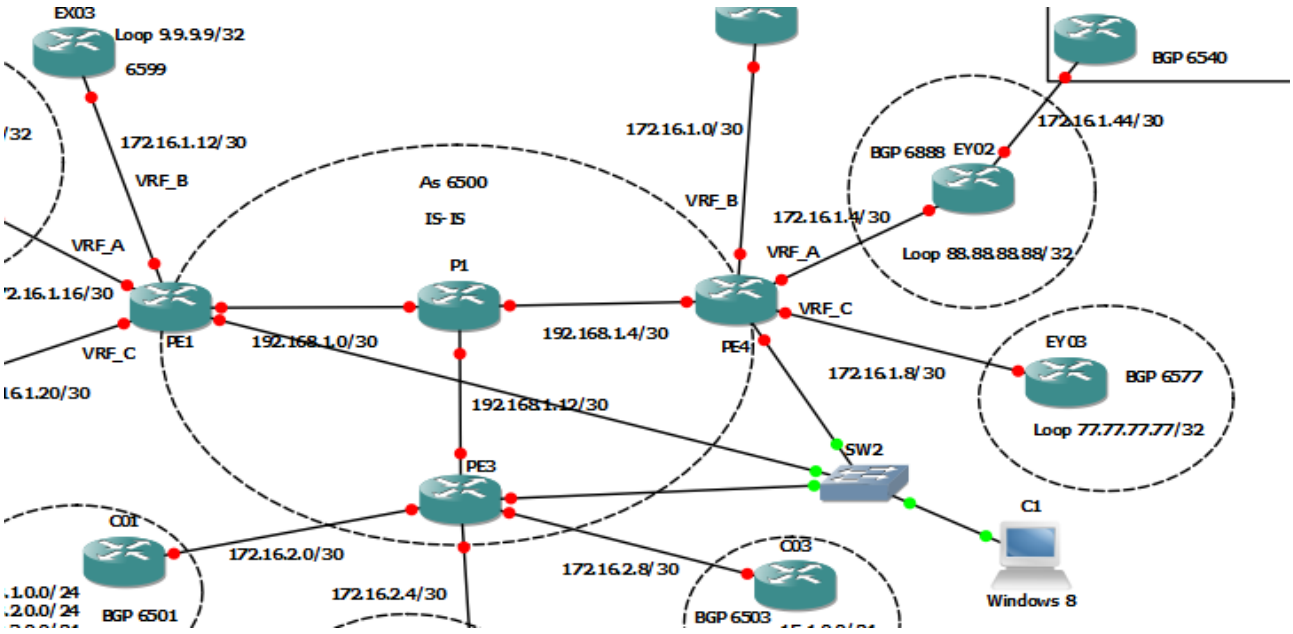


Figure 19: GNS3 and Eclipse Connectivity

Python scripts extract the network parameters from provider edge routers PE4, PE1 and PE3 .The management port has to configure at PE routers and the IP address are given according to the table shown below.

Node	IP
PE1	172.168.16.1/28
PE4	172.168.16.3/28
PE3	172.168.16.4/28

Chapter 4

Implementation and Code overview

4.1 Description of Work

In this project, Network admin tool website has several pages which provide user to login into the system, Add host, search IP address and other network parameters details such as VRF, Interface status, router performance page which shows memory, vendor and several other details.

4.1.1 Django Models

According to our requirement I need to declare few model classes in Model.py.Models provide a backend support in Django framework, where it is connected to MySQL. This code will create a tables in db2 database of MySQL.

With the object of this model classes, we can create and retrieve the database values from the MySQL.

```
from django.db import models

class RouterAd(models.Model):
    host_name = models.CharField(max_length=32)
    host_ip = models.CharField(max_length=32)
    user_name=models.CharField(max_length=32)
    password=models.CharField(max_length=32)


class Routerint(models.Model):
    hostip = models.CharField(max_length=32)
    interface_name=models.CharField(max_length=32)
    ip_addr=models.CharField(max_length=32)
    admin_up=models.CharField(max_length=30)
    protocol_up=models.CharField(max_length=20)
    t_stamp = models.DateTimeField()

class Routerprotocol(models.Model):
    hostip=models.CharField(max_length=32)
    outer_protocols= models.CharField(max_length=200)

class Routerpara(models.Model):
    host_name = models.CharField(max_length=32)
    vendor_name= models.CharField(max_length=32)
    model_name=models.CharField(max_length=32)
    image_name=models.CharField(max_length=32)
    version_name = models.CharField(max_length=32)
    cpu_usage = models.CharField(max_length=32)
    cpu_speed=models.CharField(max_length=32)
    mem_tot=models.CharField(max_length=32)
    mem_used=models.CharField(max_length=32)

class Routervrfs(models.Model):
    hostip = models.CharField(max_length=32)
    vrf_name = models.CharField(max_length=32)
    vrf_RD = models.CharField(max_length=32)
    vrf_int=models.CharField(max_length=32)
```

```
class Routervrf_lookup(models.Model):
    router_id = models.CharField(max_length=50)
    customer_name= models.CharField(max_length=50)
    network=models.CharField(max_length=50)
    protocol=models.CharField(max_length=50)
    nexthop = models.CharField(max_length=50)
    timestamp = models.DateTimeField()
```

In this code I will explain how to insert, update and retrieve the values from database with Routerint class object.

Insert

```
p1=Routerint(hostip=ip_address,interface_name=key,ip_addr=value[0],admin_up=value[1],
protocol_up=value[2],t_stamp=value[4])
p1.save()
```

Passing variables in *Routerint()* will initialize the attributes in MySQL tables. *Save()* function run the commit command in the MySQL.

Update

```
Routerint.objects.filter(ip_addr=value[0]).update(hostip=ip_address,interface_name=key,ip_addr=val
ue[0], admin_up=value[1],protocol_up=value[2],t_stamp=value[4])
```

Filter() function retrieve the specific object from the database which matches to the column value required by the user.

Update() after the match found in the database ,update operation is performed by passing all the column values.

Retrieve

```
Routerint.objects.all(hostip=ip_address)
```

Objects.all() -This will return all the values of the database in the form of object, which can be pass to the view.py and display it on the html page.

In the similar manner all the other model classes uses the above function to perform database operations.

4.1.2 Django Views

Views in Django has methods which is the functional requirement of the whole project. Mostly views methods receive a request from user and returns a response object according to the logic defined in the method.

1. **RouterRegisteration**

In this method, User input the required field and submit the form. User input the router information such as IP address, hostname, username and password. If the form field is invalid such as IP address field is invalid than it will show an error page to the user. If form submitted successfully, it redirect the user to homepage where User can choose the router to connect and retrieve the required information. In the backend *RouterAdd* table is used to update and retrieve values from the table.

2. **AllhostRequest**

In this method, User choose the router from drop down menu and submit the form. When the form is submitted, in the background script session variables are set such as IP address, Username, Password, which are required for SSH to the router.

If IP Address, Username and Password information is correct than it will retrieve all the network parameters such as VRFs, Routes and performance parameter from the router and store it into the MySQL.

If the connection to the router fails than error page will be shown to the user.

3. IP_Lookup_View

In this method, User input the IP address, subnet mask and choose the VRF from drop down menu.If the form is successfully submitted than it will show the route information for the particular IP address in specific VRF.

If the IP address does not match with the VRF routes than respective no IP found page will be shown to the User.

4. Show_Interface_View

It retrieves the table entries from *Routerint* table in the MySQL database and show it on the web page.

5. Performance_View

It retrieves the table values of *Routerpara* in MySQL database and show it on the web page.

6. Vrf_Show

It retrieves the table values of *RouterVrfs* in MySQL database and show it on the web page.

4.1.3 Templates

Django provide a tool to display the information to the user on the website. Template in Django has tags which provide the python data structure to render on the HTML page.

In Django, HTML pages can be inherited from base html page which has the common html code for all the pages.

I have created a *base.html* page which has common JavaScript code and CSS file for all the pages. In dashbord_base.html page provide basic layout, when the router SSH connection is established.
HTML page output will be shown in the Section 5.0 of this document.

Make Sure the static path has to be set in the *setting.py* otherwise Django would not know where is the CSS and HTML file in the windows machine.

```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(os.path.dirname(__file__), 'templates')],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
```

```
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]


STATICFILES_DIRS=['C:/Users/gssgu_000/javaworkspace/FinalProject/router/static']
STATIC_ROOT = os.path.join(BASE_DIR, 'static')
```

4.2 Router Configuration

In the MPLS domain, IS-IS is configured between all the Provider edge routers.MP-BGP neighbor is established between the edge routers in ISP.

On the PE, BGP is configured to exchange the routes of VRF with the specific customers. In the section 3.0 I have explained the requirements of the network design in the GNS3 tool. The output of the respective module is shown below.

4.2.1 CORE X1 Routing table

```
C       10.2.0.0 is directly connected, Loopback12
C       10.3.0.0 is directly connected, Loopback13
C       10.1.0.0 is directly connected, Loopback11
     11.0.0.0/16 is subnetted, 3 subnets
B       11.3.0.0 [20/0] via 8.8.8.8, 00:22:45
B       11.2.0.0 [20/0] via 8.8.8.8, 00:22:45
B       11.1.0.0 [20/0] via 8.8.8.8, 00:22:45
     13.0.0.0/24 is subnetted, 3 subnets
B       13.1.0.0 [20/0] via 8.8.8.8, 00:22:46
B       13.3.0.0 [20/0] via 8.8.8.8, 00:22:46
B       13.2.0.0 [20/0] via 8.8.8.8, 00:22:46
     44.0.0.0/32 is subnetted, 1 subnets
C       44.44.44.44 is directly connected, Loopback0
     14.0.0.0/24 is subnetted, 3 subnets
B       14.2.0.0 [20/0] via 8.8.8.8, 00:22:46
B       14.3.0.0 [20/0] via 8.8.8.8, 00:22:46
B       14.1.0.0 [20/0] via 8.8.8.8, 00:22:46
     15.0.0.0/24 is subnetted, 3 subnets
B       15.3.0.0 [20/0] via 8.8.8.8, 00:22:46
B       15.2.0.0 [20/0] via 8.8.8.8, 00:22:46
B       15.1.0.0 [20/0] via 8.8.8.8, 00:22:46
```

CORE-X1 has a directly connected routes of 10.X.0.0/16 as shown above. It also has the reachability to the internal server site Site-Y network and Customer site A, B and C networks.

4.2.2 CORE Y1 Routing table

```
B       10.2.0.0 [20/0] via 88.88.88.88, 00:21:03
B       10.3.0.0 [20/0] via 88.88.88.88, 00:21:03
B       10.1.0.0 [20/0] via 88.88.88.88, 00:21:04
     11.0.0.0/16 is subnetted, 3 subnets
C       11.3.0.0 is directly connected, Loopback13
C       11.2.0.0 is directly connected, Loopback12
```

```
C       11.1.0.0 is directly connected, Loopback11
     88.0.0.0/32 is subnetted, 1 subnets
S       88.88.88.88 [1/0] via 172.16.1.45
     13.0.0.0/24 is subnetted, 3 subnets
B       13.1.0.0 [20/0] via 88.88.88.88, 00:22:12
B       13.3.0.0 [20/0] via 88.88.88.88, 00:22:12
B       13.2.0.0 [20/0] via 88.88.88.88, 00:22:12
     14.0.0.0/24 is subnetted, 3 subnets
B       14.2.0.0 [20/0] via 88.88.88.88, 00:22:12
B       14.3.0.0 [20/0] via 88.88.88.88, 00:22:12
B       14.1.0.0 [20/0] via 88.88.88.88, 00:22:12
     15.0.0.0/24 is subnetted, 3 subnets
B       15.3.0.0 [20/0] via 88.88.88.88, 00:22:12
B       15.2.0.0 [20/0] via 88.88.88.88, 00:22:12
B       15.1.0.0 [20/0] via 88.88.88.88, 00:22:12
```

CORE-Y1 has directly connected 11.X.0.0/16 and Site-X (internal company server routes) and customer routes of all the sites.

4.2.3 C01, C02 and C03 Routing Tables

```
C01
B       10.2.0.0 [20/0] via 172.16.2.2, 00:34:02
B       10.3.0.0 [20/0] via 172.16.2.2, 00:34:02
B       10.1.0.0 [20/0] via 172.16.2.2, 00:34:02
     11.0.0.0/16 is subnetted, 3 subnets
B       11.3.0.0 [20/0] via 172.16.2.2, 00:35:22
B       11.2.0.0 [20/0] via 172.16.2.2, 00:35:22
B       11.1.0.0 [20/0] via 172.16.2.2, 00:35:22
     13.0.0.0/24 is subnetted, 3 subnets
C       13.1.0.0 is directly connected, Loopback10
C       13.3.0.0 is directly connected, Loopback13
C       13.2.0.0 is directly connected, Loopback11
```

As we can see above at C01, VRF Site A import the routes of Site-X and Site-Y only. Similar to this, Routing table of C02 and C03 are shown below.

```
C02
B       10.2.0.0 [20/0] via 172.16.2.6, 00:42:46
B       10.3.0.0 [20/0] via 172.16.2.6, 00:42:46
B       10.1.0.0 [20/0] via 172.16.2.6, 00:42:46
     11.0.0.0/16 is subnetted, 3 subnets
B       11.3.0.0 [20/0] via 172.16.2.6, 00:44:07
B       11.2.0.0 [20/0] via 172.16.2.6, 00:44:07
B       11.1.0.0 [20/0] via 172.16.2.6, 00:44:07
     14.0.0.0/24 is subnetted, 3 subnets
C       14.2.0.0 is directly connected, Loopback1
C       14.3.0.0 is directly connected, Loopback2
C       14.1.0.0 is directly connected, Loopback10


C03
B       10.2.0.0 [20/0] via 172.16.2.9, 00:43:10
B       10.3.0.0 [20/0] via 172.16.2.9, 00:43:10
B       10.1.0.0 [20/0] via 172.16.2.9, 00:43:10
     11.0.0.0/16 is subnetted, 3 subnets
B       11.3.0.0 [20/0] via 172.16.2.9, 00:44:31
B       11.2.0.0 [20/0] via 172.16.2.9, 00:44:31
```

```
B       11.1.0.0 [20/0] via 172.16.2.9, 00:44:31
     15.0.0.0/24 is subnetted, 3 subnets
C       15.3.0.0 is directly connected, Loopback3
C       15.2.0.0 is directly connected, Loopback1
C       15.1.0.0 is directly connected, Loopback0
```

Chapter 5
Working of the System

5.1 Django Server

In order to run the website, Django server has to be in the running state. We can run the server by giving a command *runserver* in the eclipse.

If there I any change done in MySQL tables, Django need to run the *migrate* command in order to sync with the latest database log information**.**

After successful execution of the Django server, there are some message will be shown on the console.

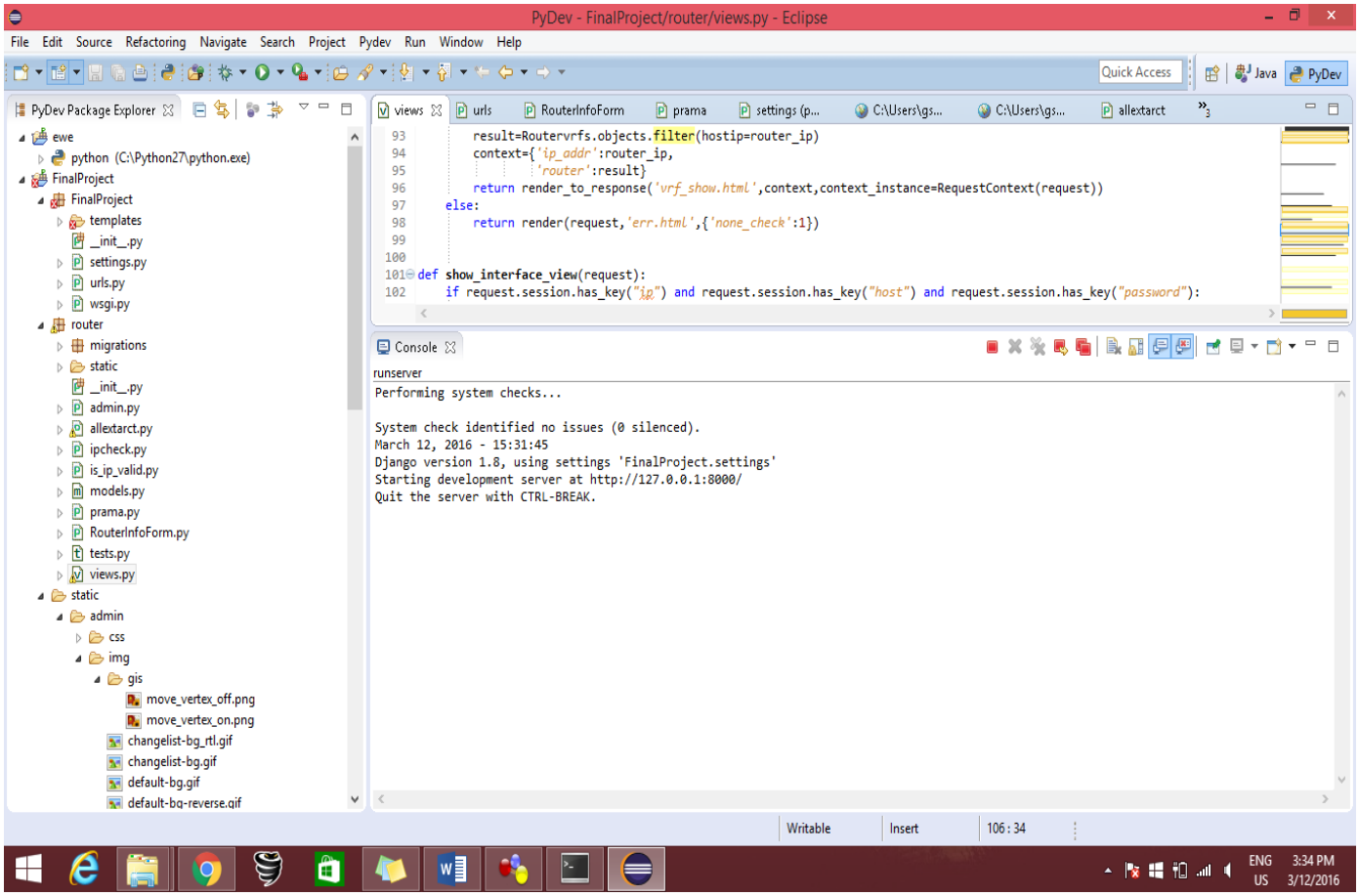


Figure 20: Run Django Server

User has to input the URL in the web browser http://127.0.0.1:8000/

5.1.1 Homepage

URL for the homepage is *http://127.0.0.1:8000/router/index/*

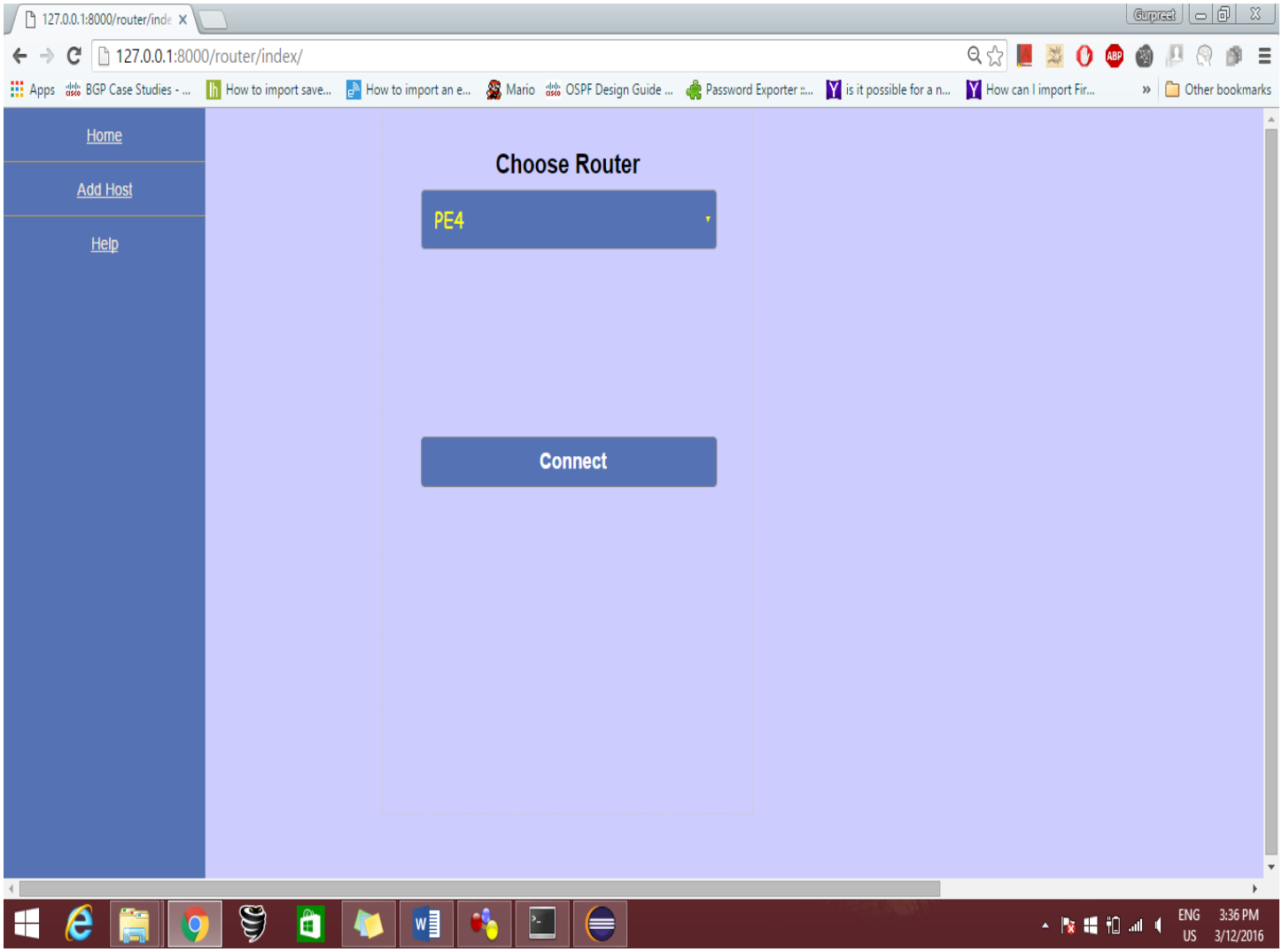


Figure 21: Homepage

5.1.2 Adding New Host Page

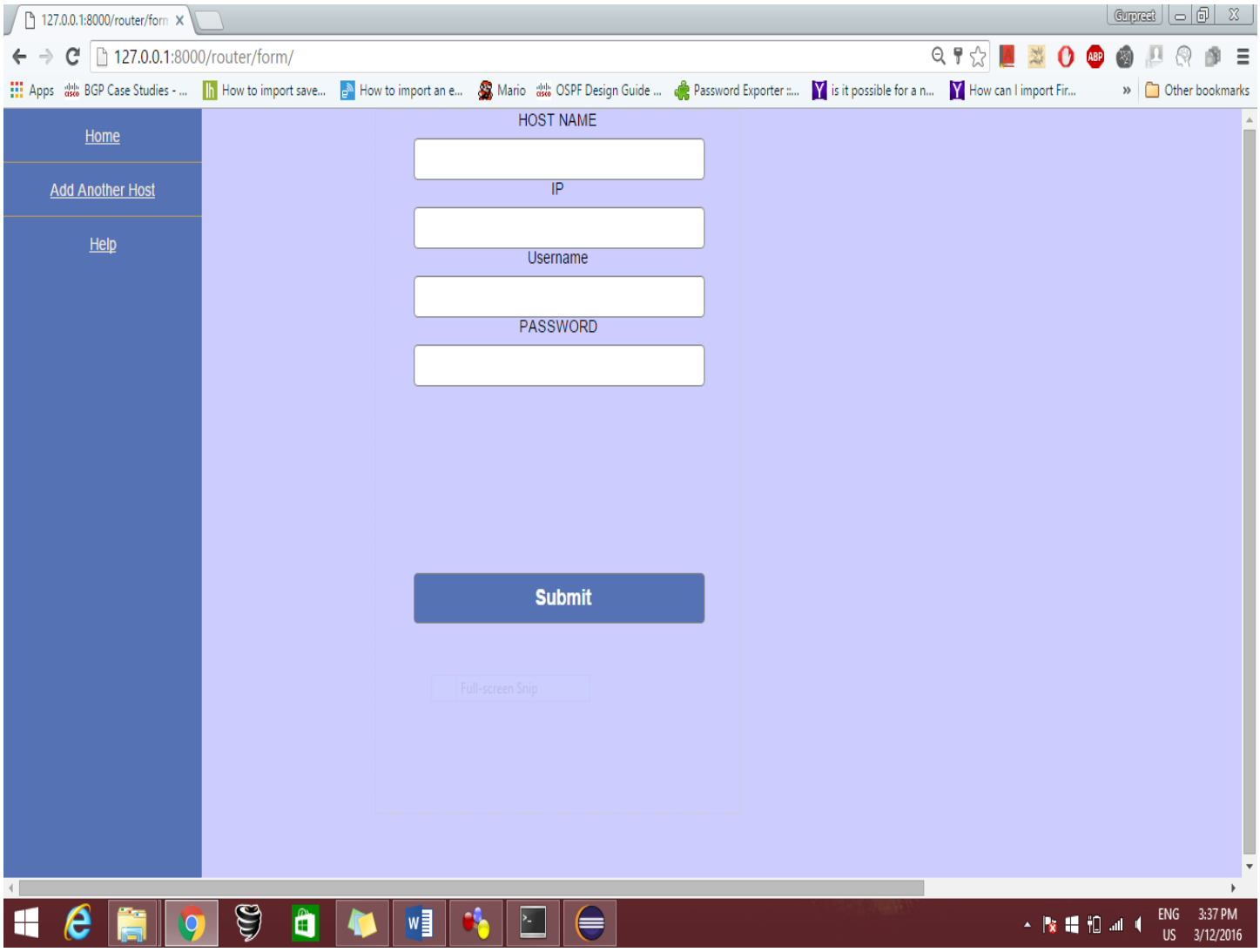


Figure 22: Add New Host

When the user choose the router from the drop down menu and click the connect button SSH connection is established.

5.1.3 Dashboard Page

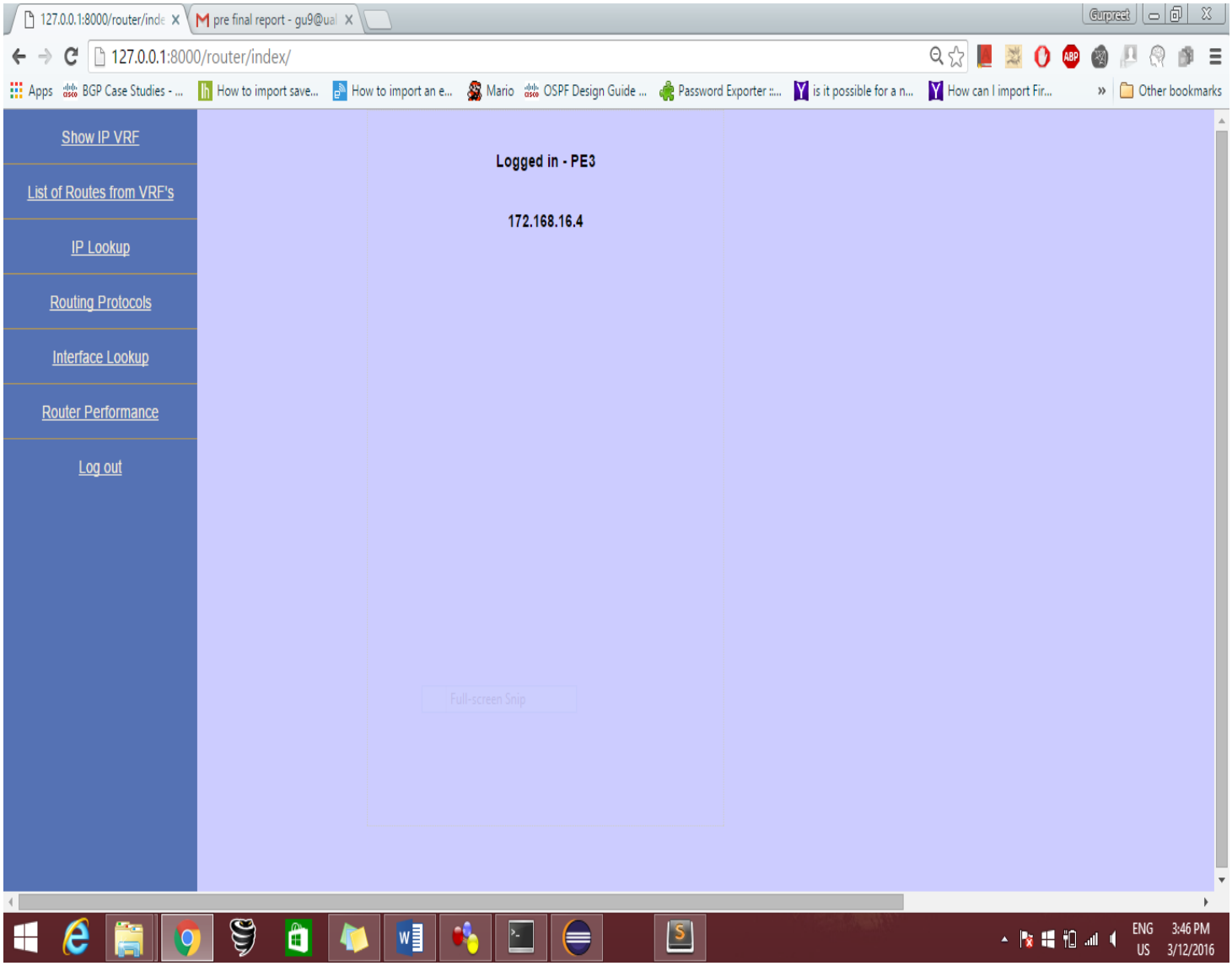


Figure 23: Logged In Dashboard

5.1.4 Show IP VRF Page

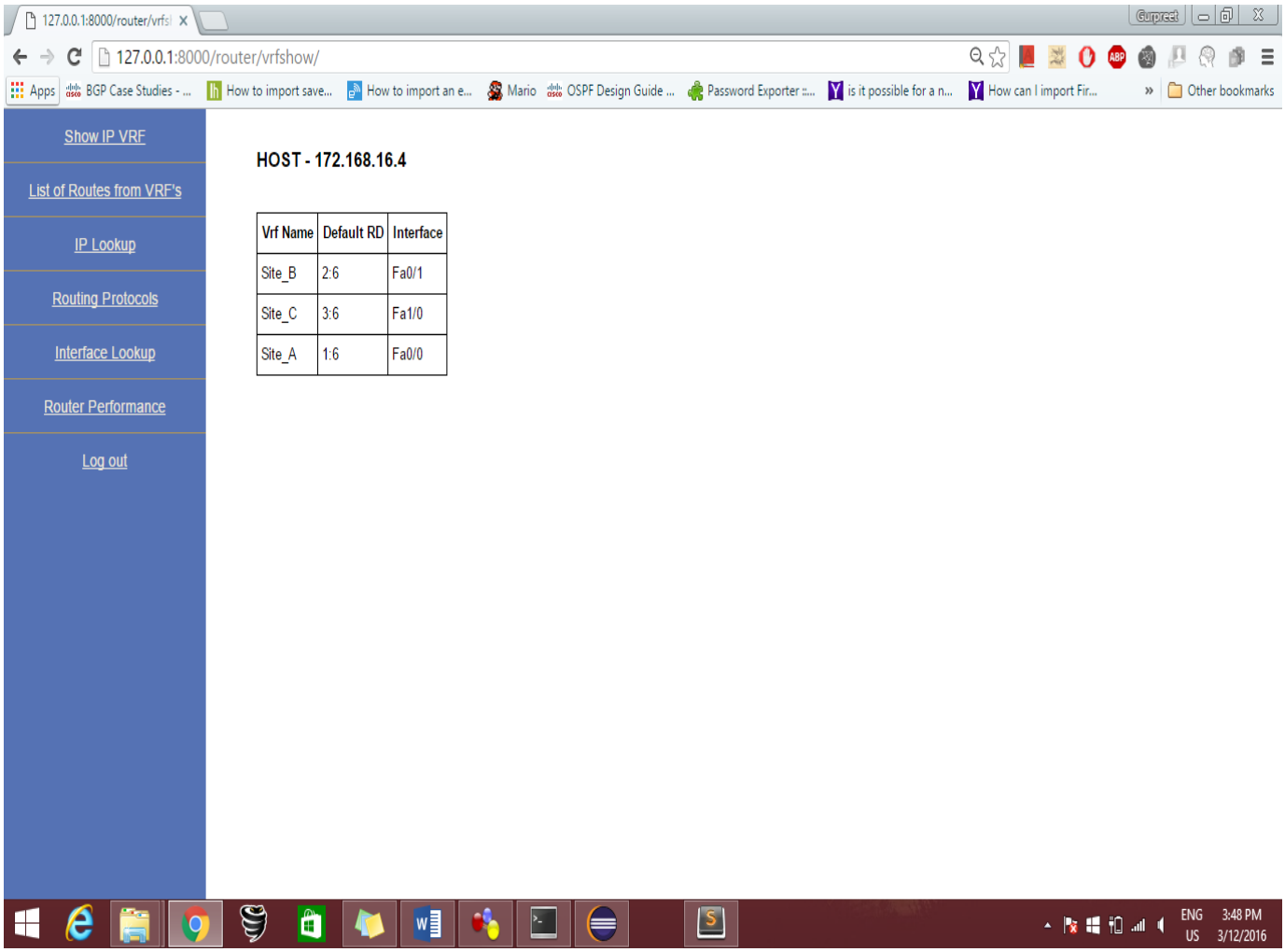


Figure 24: VRF Page

5.1.5 List of Routes from VRF Page

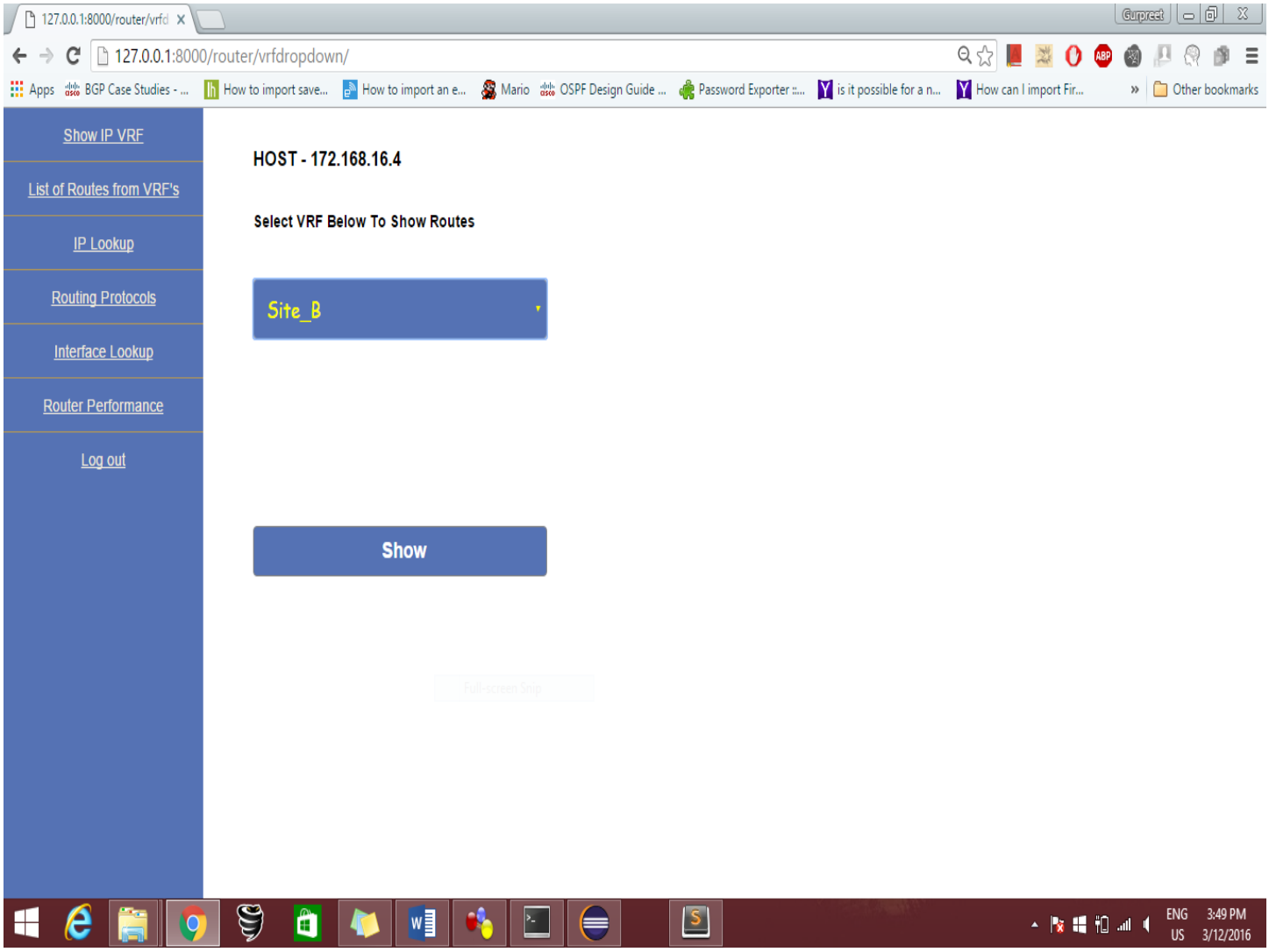


Figure 25: Choose VRF

Choose the VRF from drop down menu and click the show button.

Figure 26: VRF Routes

5.1.6 IP Lookup Page

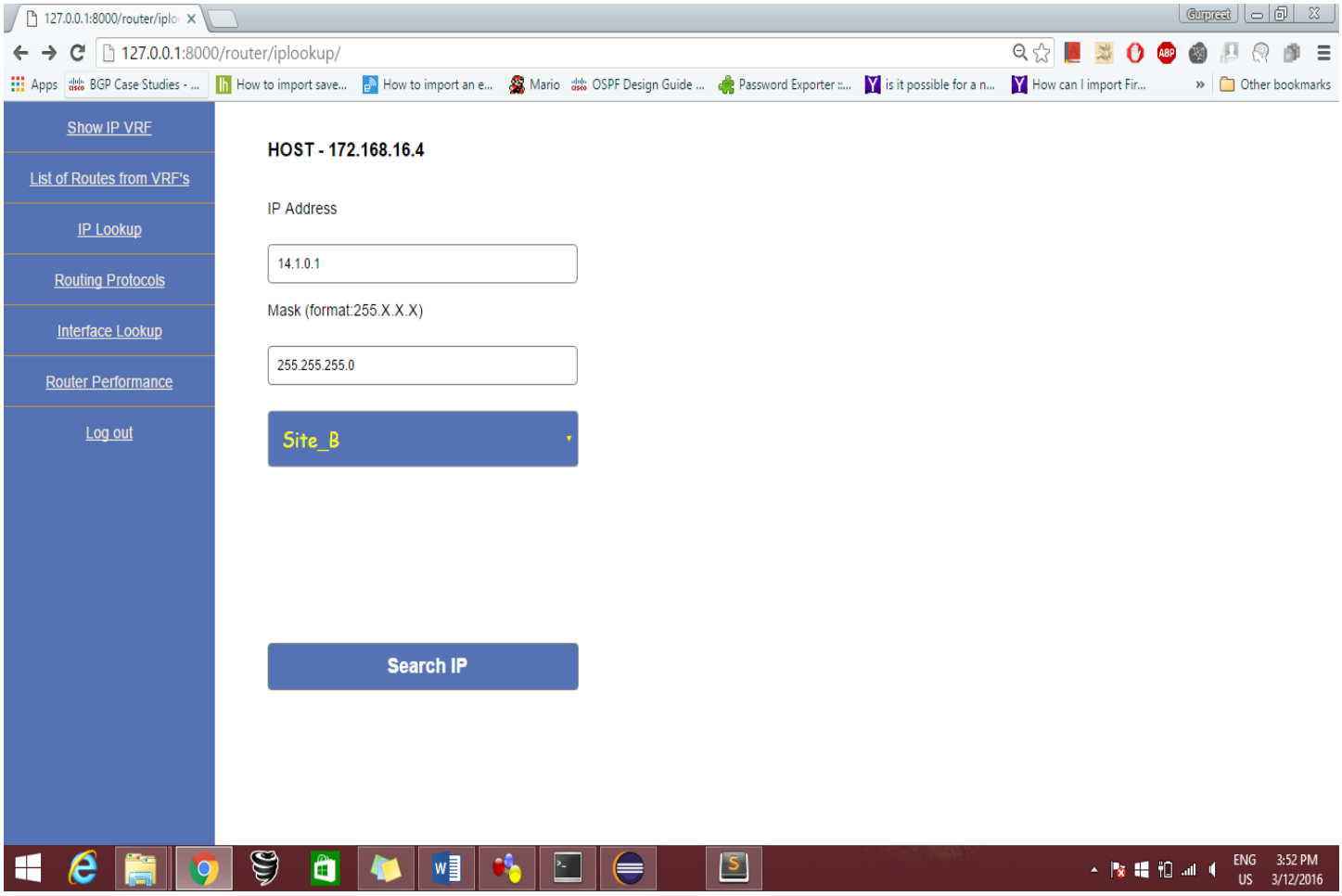


Figure 27: IP Lookup Page

To search an IP in specific VRF, input the IP, mask and click on search IP button.

After submitting the form, IP will be lookup in the specific VRF and respective output will be shown to the user. See the figure 28 for the output.

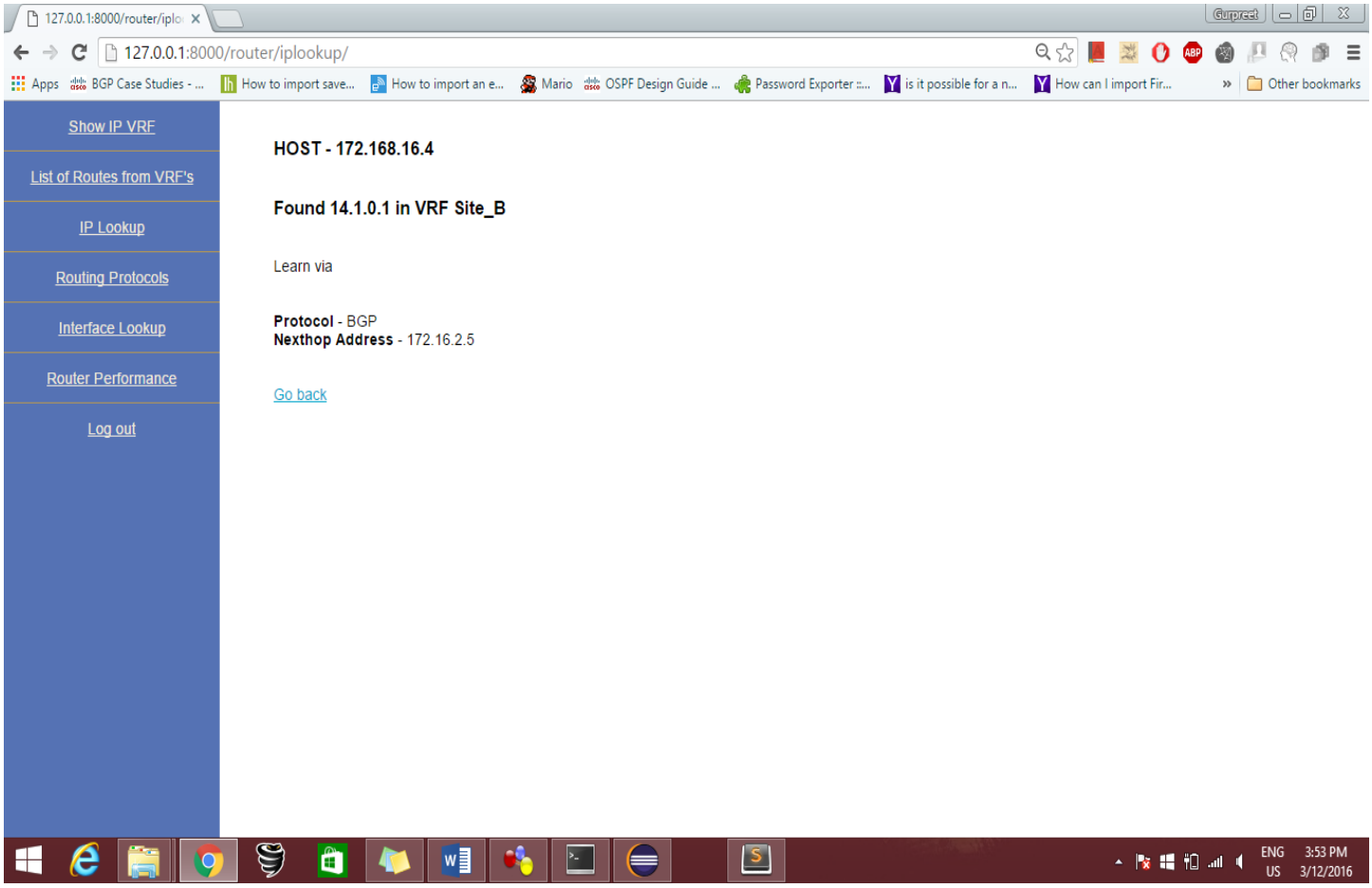


Figure 28: IP Lookup Result

5.1.7 Show IP Interface Page

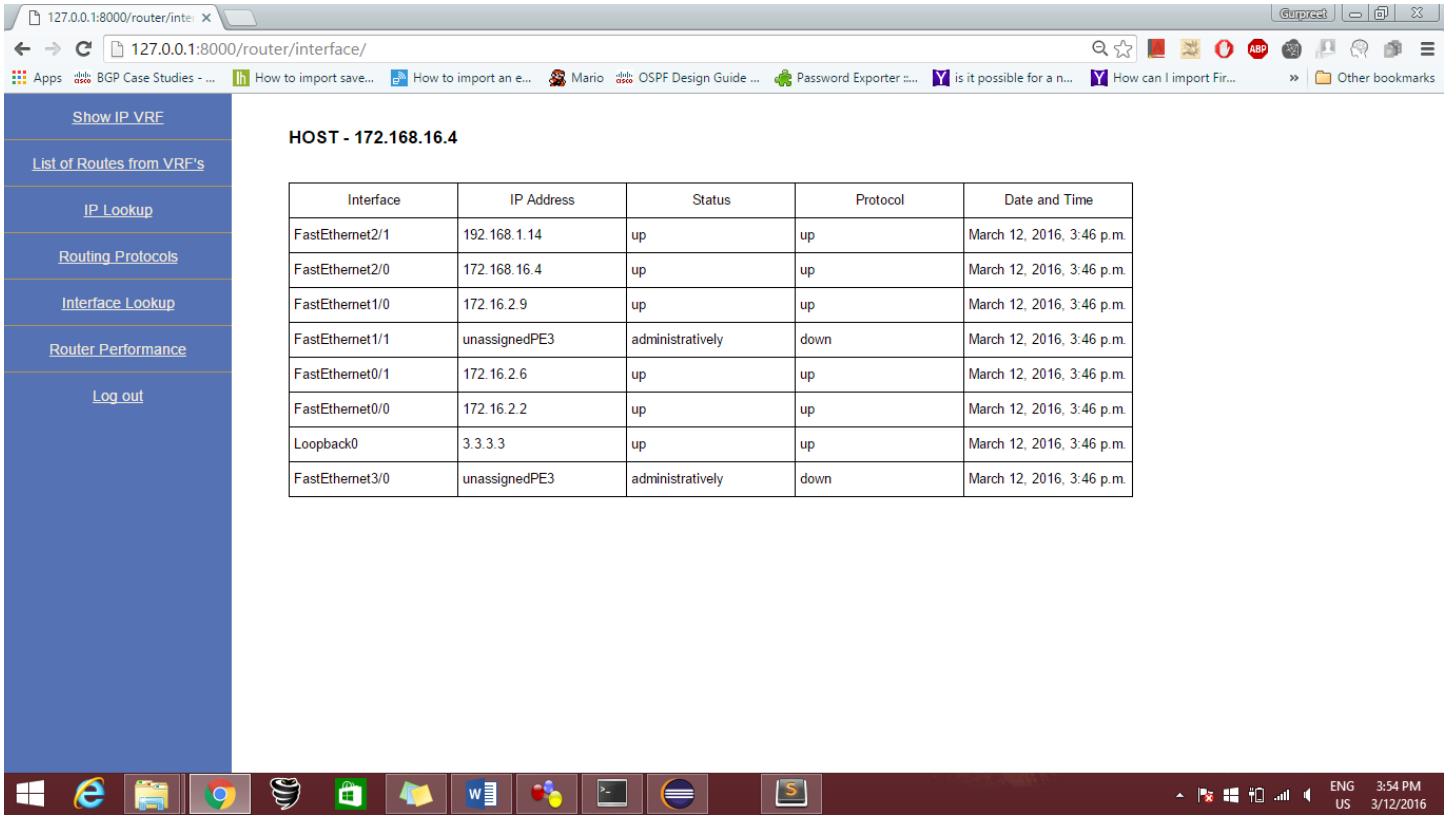


Figure 29: IP Interface

5.1.8 Router Performance Page

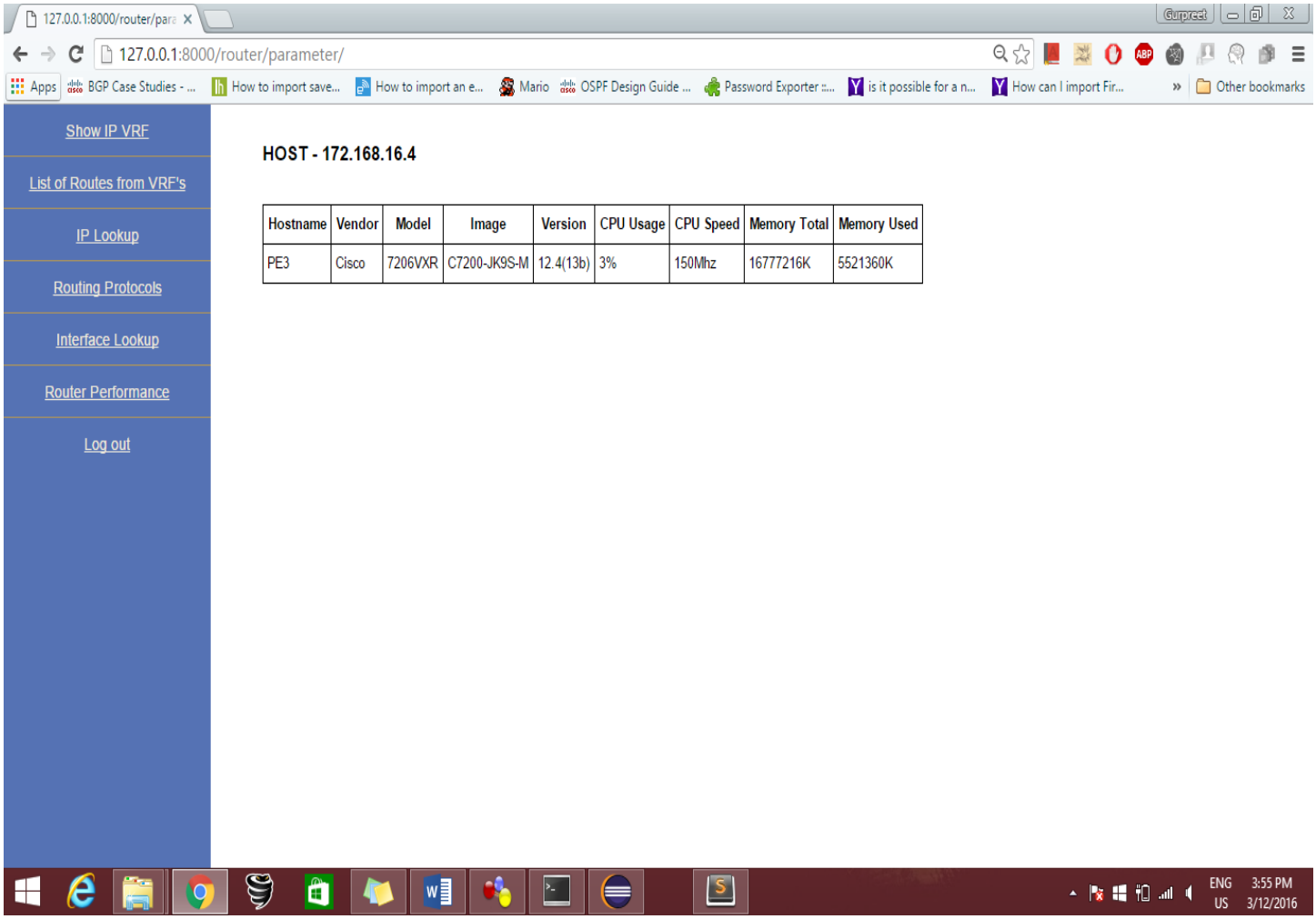


Figure 30: Router Performance

5.1.9 Error SSH page

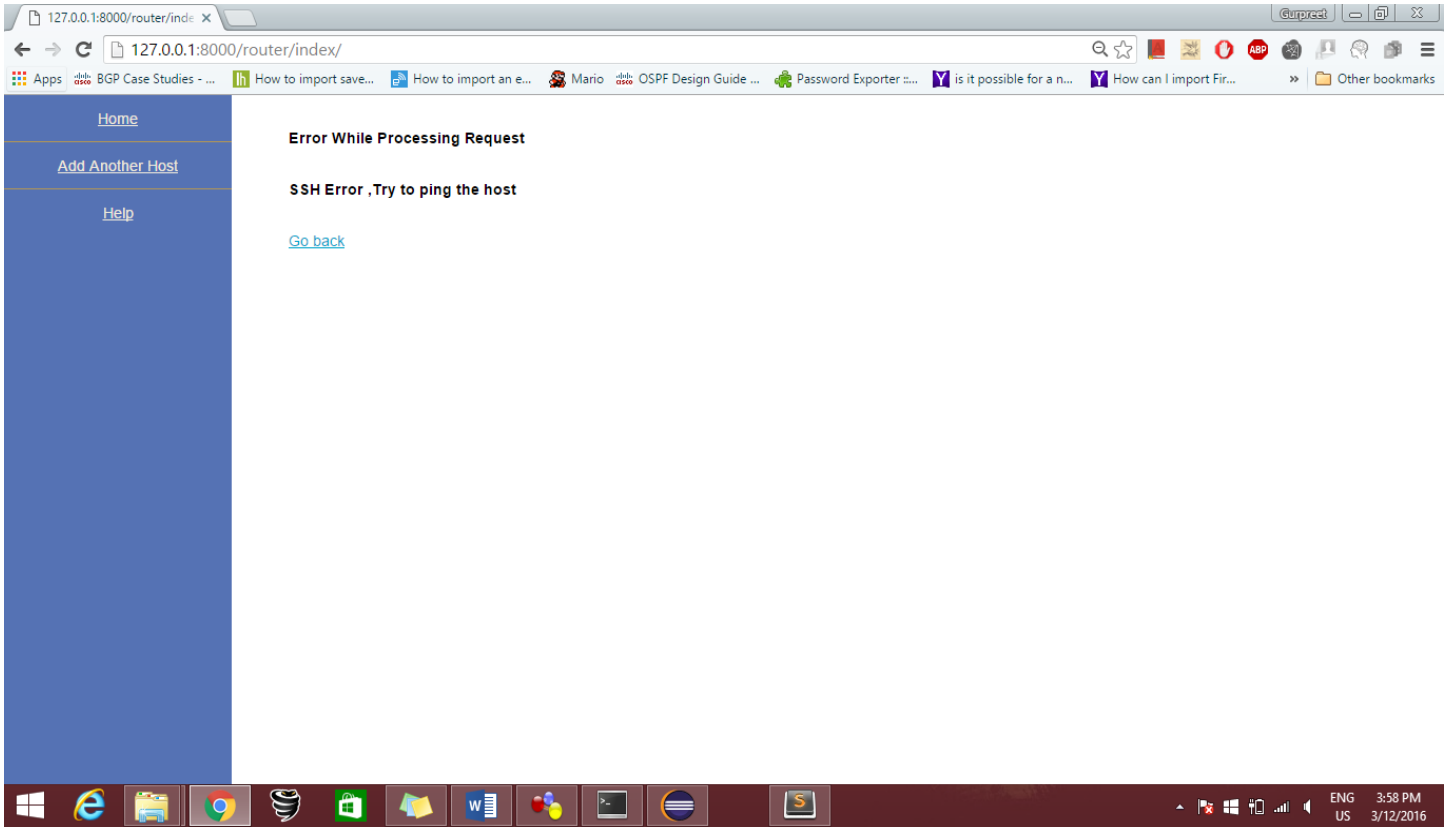


Figure 31: SSH Error

This page will show up, when SSH connection is refused by the router or router is not connected to the network.

When user try to search an IP which does not Exits

5.1.10 IP Error Page

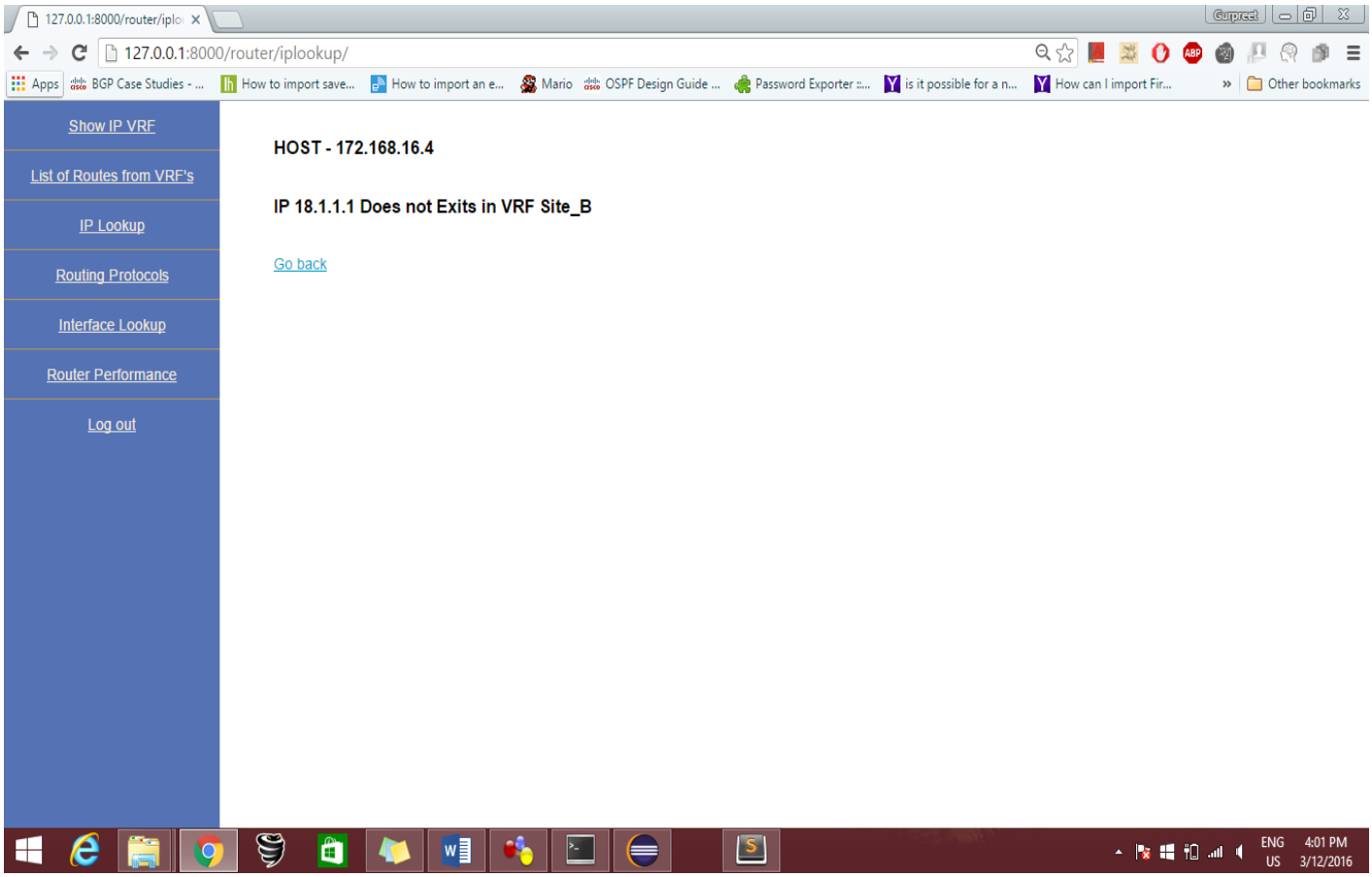


Figure 32: IP not found

5.1.11 Form Submission Validation

When the User try to submit the form without input the field information. *"This field is required*" error will be shown.

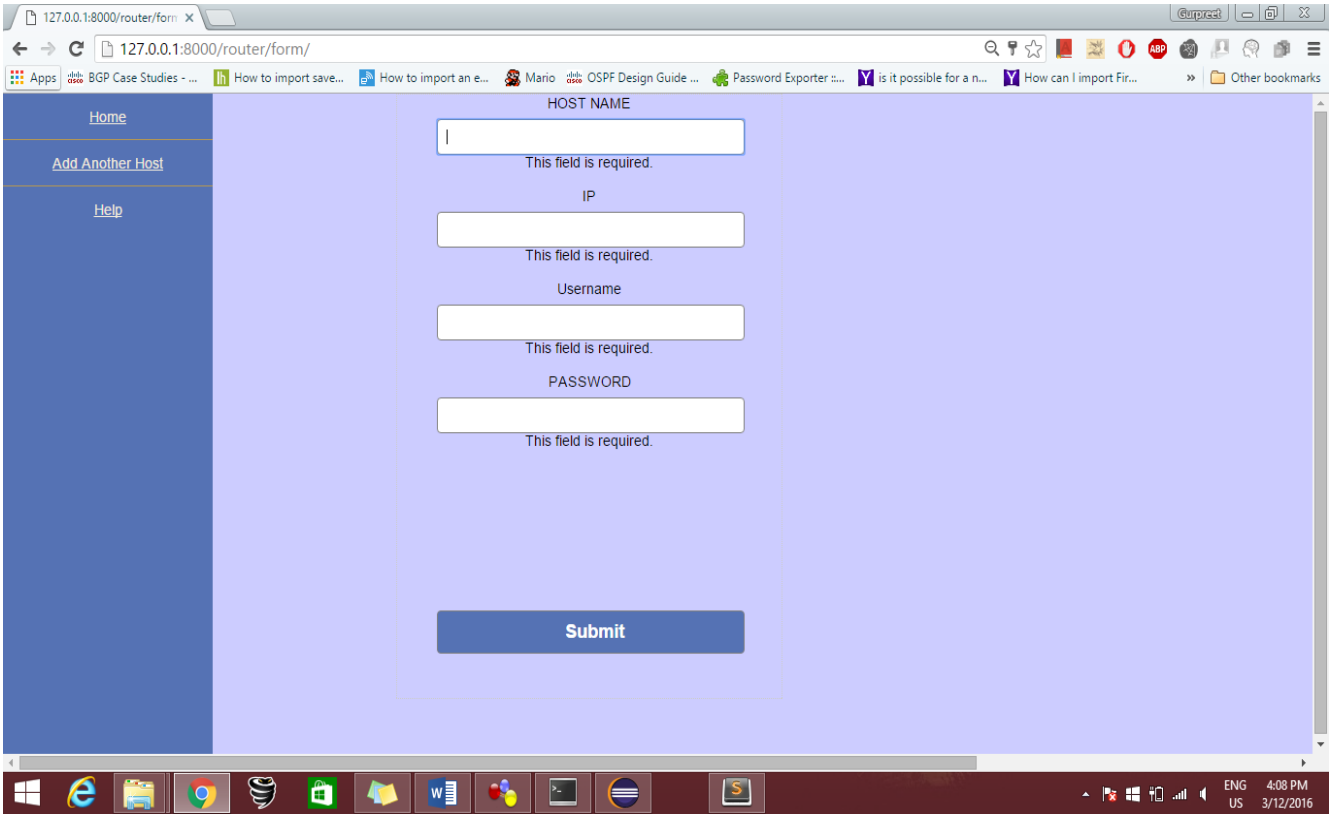


Figure 33: Form Error Page

5.2 MySQL

Logged into the MySQL CLI and we can see available database, from which db2 is used in this project**.**

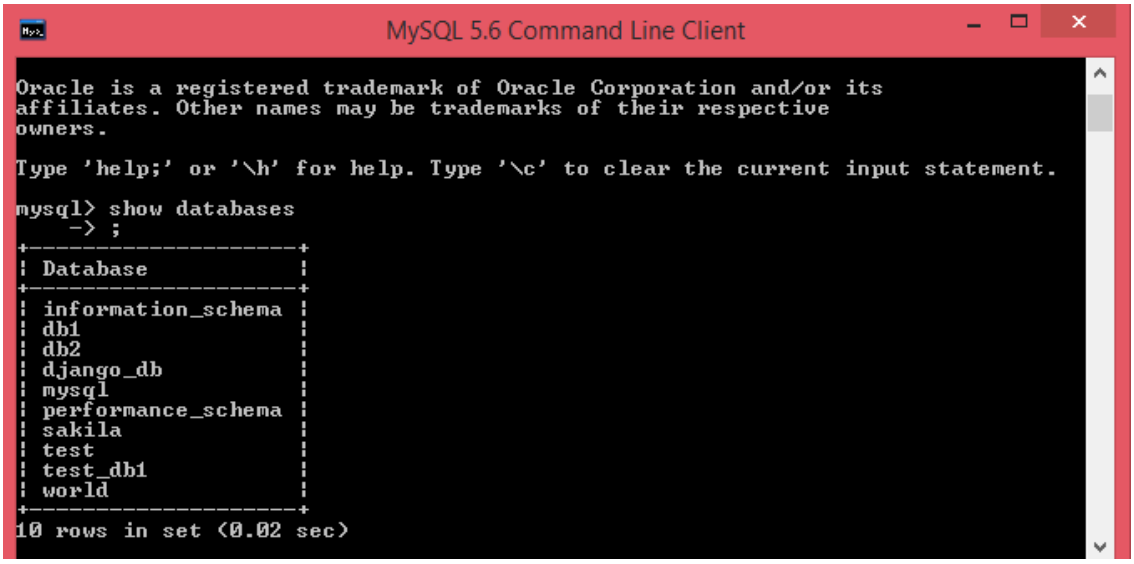


Figure 34: Database

5.2.1 Tables Available in db2 Database

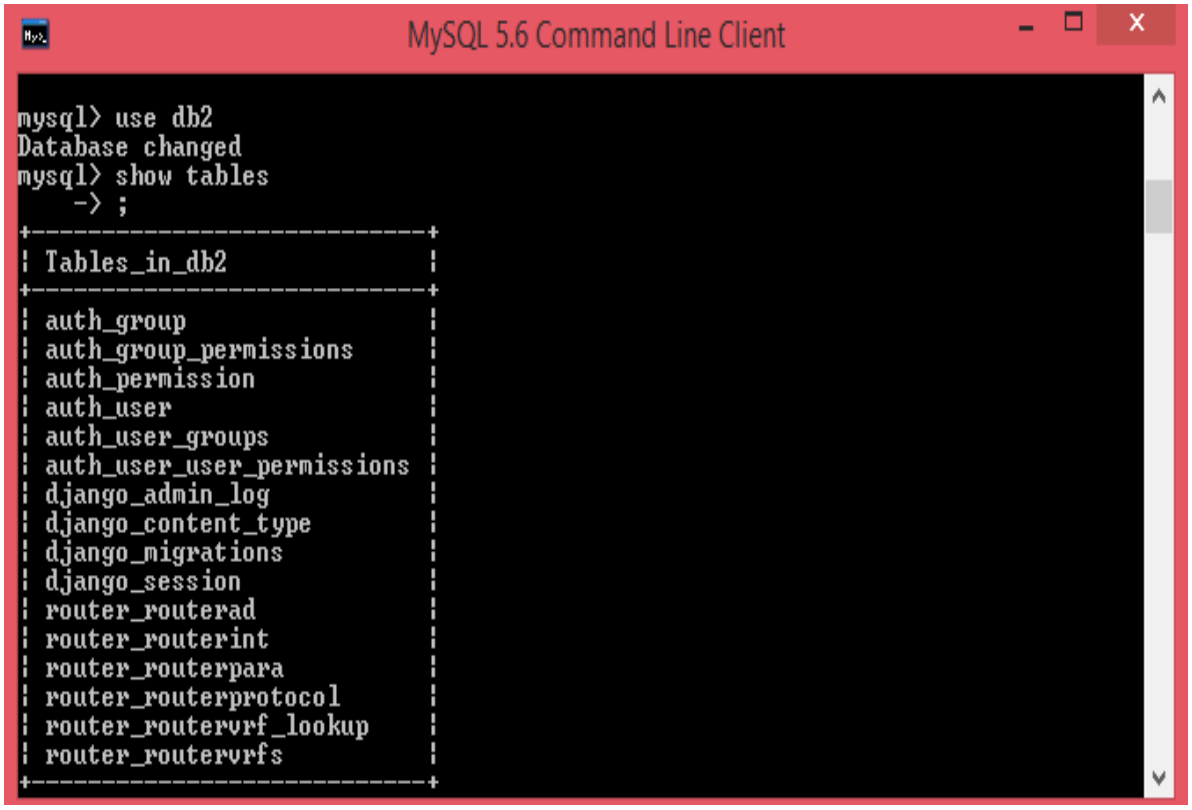


Figure 35: Table Entries

5.2.2 Router Host Table

Figure 36: Host Table

This above information required to SSH into the router. I logged into the PE3 router to show the table entries in MySQL. The outputs shown below in Figure 37,38,39,40 are belong to PE3.

5.2.3 Router interface Table

```
mysql> select * from  router_routerint;
+----+-------------+-----------------+---------------+-----------------+------
-------+---------------------+
| id | hostip      | interface_name  | ip_addr       | admin_up        | proto
col_up | t_stamp             |
+----+-------------+-----------------+---------------+-----------------+------
-------+---------------------+
| 19 | 172.168.16.4 | FastEthernet2/1 | 192.168.1.14  | up              | up
       | 2016-03-04 02:50:02 |
| 20 | 172.168.16.4 | FastEthernet2/0 | 172.168.16.4  | up              | up
       | 2016-03-04 02:50:02 |
| 21 | 172.168.16.4 | FastEthernet1/0 | 172.16.2.9    | up              | up
       | 2016-03-04 02:50:02 |
| 22 | 172.168.16.4 | FastEthernet1/1 | unassignedPE3 | administratively | down
       | 2016-03-04 02:50:02 |
| 23 | 172.168.16.4 | FastEthernet0/1 | 172.16.2.6    | up              | up
       | 2016-03-04 02:50:02 |
| 24 | 172.168.16.4 | FastEthernet0/0 | 172.16.2.2    | up              | up
       | 2016-03-04 02:50:02 |
| 25 | 172.168.16.4 | Loopback0       | 3.3.3.3       | up              | up
       | 2016-03-04 02:50:02 |
| 26 | 172.168.16.4 | FastEthernet3/0 | unassignedPE3 | administratively | down
       | 2016-03-04 02:50:02 |
+----+-------------+-----------------+---------------+-----------------+------
-------+---------------------+
8 rows in set (0.00 sec)

mysql>
```

Figure 37: Interface Table

5.2.4 Router Performance Table

```
mysql> select * from  router_routerpara;
+----+-----------+-------------+-------------+--------------+--------------+-----
------+-----------+----------+----------+
| id | host_name | vendor_name | model_name | image_name   | version_name | cpu_
usage | cpu_speed | mem_tot  | mem_used |
+----+-----------+-------------+-------------+--------------+--------------+-----
------+-----------+----------+----------+
|  2 | PE4       | Cisco       | 7206VXR    | C7200-JK9S-M | 12.4(13b)    | 2
      | 150       | 16777216 | 6468552  |
|  3 | PE3       | Cisco       | 7206VXR    | C7200-JK9S-M | 12.4(13b)    | 3
      | 150       | 16777216 | 5530640  |
+----+-----------+-------------+-------------+--------------+--------------+-----
------+-----------+----------+----------+
2 rows in set (0.00 sec)

mysql>
```

Figure 38: Router Performance

5.2.5 Router VRF Lookup

```
mysql> select * from  router_routervrf_lookup;
+----+--------------+---------------+--------------+-----------+-------------+-
--------------------+
| id | router_id    | customer_name | network      | protocol  | nexthop     |
timestamp           |
+----+--------------+---------------+--------------+-----------+-------------+-
--------------------+
| 34 | 172.168.16.4 | Site_B        | 172.16.2.4/30 | Connected |             |
2016-03-04 02:50:05 |
| 35 | 172.168.16.4 | Site_B        | 10.2.0.0/16   | BGP       | 1.1.1.1     |
2016-03-04 02:50:05 |
| 36 | 172.168.16.4 | Site_B        | 10.3.0.0/16   | BGP       | 1.1.1.1     |
2016-03-04 02:50:05 |
| 37 | 172.168.16.4 | Site_B        | 10.1.0.0/16   | BGP       | 1.1.1.1     |
2016-03-04 02:50:05 |
| 38 | 172.168.16.4 | Site_B        | 11.3.0.0/16   | BGP       | 4.4.4.4     |
2016-03-04 02:50:05 |
| 39 | 172.168.16.4 | Site_B        | 11.2.0.0/16   | BGP       | 4.4.4.4     |
2016-03-04 02:50:05 |
| 40 | 172.168.16.4 | Site_B        | 11.1.0.0/16   | BGP       | 4.4.4.4     |
2016-03-04 02:50:05 |
| 41 | 172.168.16.4 | Site_B        | 14.2.0.0/24   | BGP       | 172.16.2.5  |
2016-03-04 02:50:05 |
| 42 | 172.168.16.4 | Site_B        | 14.3.0.0/24   | BGP       | 172.16.2.5  |
2016-03-04 02:50:05 |
| 43 | 172.168.16.4 | Site_B        | 14.1.0.0/24   | BGP       | 172.16.2.5  |
2016-03-04 02:50:05 |
| 44 | 172.168.16.4 | Site_C        | 172.16.2.8/30 | Connected |             |
2016-03-04 02:50:05 |
| 45 | 172.168.16.4 | Site_C        | 10.2.0.0/16   | BGP       | 1.1.1.1     |
2016-03-04 02:50:05 |
| 46 | 172.168.16.4 | Site_C        | 10.3.0.0/16   | BGP       | 1.1.1.1     |
2016-03-04 02:50:05 |
| 47 | 172.168.16.4 | Site_C        | 10.1.0.0/16   | BGP       | 1.1.1.1     |
2016-03-04 02:50:05 |
| 48 | 172.168.16.4 | Site_C        | 11.3.0.0/16   | BGP       | 4.4.4.4     |
2016-03-04 02:50:05 |
| 49 | 172.168.16.4 | Site_C        | 11.2.0.0/16   | BGP       | 4.4.4.4     |
2016-03-04 02:50:05 |
| 50 | 172.168.16.4 | Site_C        | 11.1.0.0/16   | BGP       | 4.4.4.4     |
2016-03-04 02:50:05 |
| 51 | 172.168.16.4 | Site_C        | 15.3.0.0/24   | BGP       | 172.16.2.10 |
2016-03-04 02:50:05 |
| 52 | 172.168.16.4 | Site_C        | 15.2.0.0/24   | BGP       | 172.16.2.10 |
2016-03-04 02:50:05 |
| 53 | 172.168.16.4 | Site_C        | 15.1.0.0/24   | BGP       | 172.16.2.10 |
2016-03-04 02:50:05 |
| 54 | 172.168.16.4 | Site_A        | 172.16.2.0/30 | Connected |             |
2016-03-04 02:50:05 |
| 55 | 172.168.16.4 | Site_A        | 10.2.0.0/16   | BGP       | 1.1.1.1     |
2016-03-04 02:50:05 |
```

Figure 39: VRF Routes Table

5.2.6 Router VRF

```
mysql> select * from  router_routervrfs;
+----+--------------+-----------+--------+----------+
| id | hostip       | vrf_name  | vrf_RD | vrf_int  |
+----+--------------+-----------+--------+----------+
|  7 | 172.168.16.4 | Site_B    | 2:6    | Fa0/1    |
|  8 | 172.168.16.4 | Site_C    | 3:6    | Fa1/0    |
|  9 | 172.168.16.4 | Site_A    | 1:6    | Fa0/0    |
+----+--------------+-----------+--------+----------+
3 rows in set (0.00 sec)

mysql>
```

Figure 40: IP VRF Table

When user logged out of the router, all the table runs a command Truncate to delete all the table entries. So, when user logged in with other SSH information, than that table entries will be available for user to view on the website.

Chapter 6

Conclusion and Future Work

In this project, MPLS, BGP and IS-IS is configured to achieve the network design and various sites has been configured through VRF to analyze the VPN scenario. Python scripts extract the network parameters and store it into the MySQL tables. CSS and JavaScript provide end user rich GUI as well as client side validation. The Web app functionality implemented in this project so far are:

- To Search an IP address in specific VRF.
- To Show VRF routes information by selecting dropdown menu.
- To Show router performance parameters.
- Show IP VRF
- Show IP interface brief
- Client side Validation(example IP address)

In this project, Python script is written for Cisco routers only .We can add more functionality by supporting more vendors such as juniper and Alcatel routers.

In the nutshell, python script can be written to show all the protocols and routes belong to show on the web. Information about the packet drops and security can be analyze .MPLS Traffic engineering concepts can be monitor through the GUI.

References

[1] http://www.tutorialspoint.com/ipv4/ipv4_address_classes.html

[2] http://www.tutorialspoint.com/ipv4/ipv4_vlsm.htm

[3] Ghein, Luc De. 2007. MPLS Fundamentals Cisco Systems. USA: Cisco Press

[4] IS-IS Configuration. Retrieved from Ciscopress: http://www.ciscopress.com/articles/article.asp?p=31319

[5] http://www.cisco.com/en/US/products/ps6599/products_white_paper09186a00800a3e6f.shtml

[6] http://www.ciscopress.com/articles/article.asp?p=31319

[7] http://www.dummies.com/how-to/content/isis-network-protocol-basics.html

[8] https://www.w3.org/standards/webdesign/htmlcss

[9] http://js-tutorial.com/parsley-ultimate-javascript-form-validation-library-507

[10] http://www.htmlgoodies.com/tutorials/forms/article.php/3888746/HTML-Forms-From-Basics-to-Style-Layouts.html

[11] http://stackoverflow.com/questions/9145994/django-and-css-and-a-really-simple-example-please

[12] https://css-tricks.com/fixing-tables-long-strings/

[13] https://css-tricks.com/complete-guide-table-element/

[14] https://www.youtube.com/watch?v=CFypO_LNmcc

[15] http://linfiniti.com/2011/12/django-development-with-eclipse-and-pydev/

[16] https://www.djangoproject.com/

[17] https://docs.djangoproject.com/en/1.9/ref/databases/

[18] https://realpython.com/learn/start-django/#django-18

[19] https://docs.djangoproject.com/en/dev/topics/http/sessions/

[20] http://stackoverflow.com/questions/5250276/how-to-render-an-ordered-dictionary-in-django-templates

[21] http://www.djangobook.com/en/2.0/chapter05.html

[22] https://docs.djangoproject.com/en/1.9/topics/db/sql/

[23] http://www.voidspace.org.uk/python/modules.shtml#pycrypto

[24] http://stackoverflow.com/questions/2890896/extract-ip-address-from-an-html-string-python

[25] https://www.youtube.com/watch?v=uyT3NSHhke8

[26] https://github.com/agiliq/django-parsley

[27] http://stackoverflow.com/questions/31201763/passing-the-value-of-selected-option-of-dropdown-value-to-django-database

[28] https://github.com/mccomasd/netmonsqlpython/blob/master/netmonsqlpython.py

[29] http://stackoverflow.com/questions/6893968/how-to-get-the-return-value-from-a-thread-in-python

[30] http://stackoverflow.com/questions/2988997/how-do-i-truncate-table-using-django-orm

[31] http://academy.gns3.com/courses/python-programming-for-real-life-networking-use/lectures/280631

[32] Previous Python Code from MINT709 Router Scripting vrf.py