DOMAIN NAME SYSTEM

**A PROJECT REPORT**

*Submitted by*

Shweta Gupta(16MCA0119)

Animisha Parmar(16MCA0184)

Prakash dubey(16MCA0169)

Course Code: ITA5003

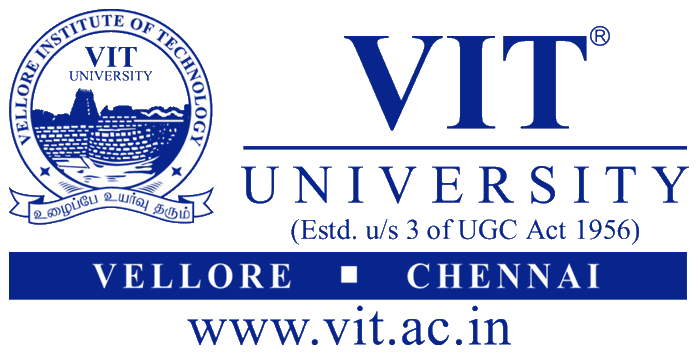
Course Title: Data Communication and Networking

Under the guidance of

**Dr. Asis Kumar Tripathy**

**Associate Professor, SITE,**

**VIT University, Vellore.**



**DEPARTMENT OF COMPUTER APPLICATIONS & CREATIVE MEDIA**

**SCHOOL OF INFORMATION TECHNOLOGY AND ENGINEERING**

**NOVEMBER 2016**

Table of Contents

**Revisions iii**

**1** **Introduction 1**

1.1 Document Purpose 1

1.2 Product Scope 1

1.3 Intended Audience and Document Overview 1

1.4 Definitions, Acronyms and Abbreviations 1

1.5 Document Conventions 1

1.6 References and Acknowledgments 2

**2** **Overall Description 3**

2.1 Product Perspective 3

2.2 Product Functionality 3

2.3 Users and Characteristics 3

2.4 Operating Environment 3

2.5 Design and Implementation Constraints 4

2.6 User Documentation 4

2.7 Assumptions and Dependencies 4

**3** **Specific Requirements 5**

3.1 External Interface Requirements 5

3.2 Functional Requirements 6

3.3 Behaviour Requirements 6

**4** **Other Non-functional Requirements 7**

4.1 Performance Requirements 7

4.2 Safety and Security Requirements 7

4.3 Software Quality Attributes 7

**Appendix A – Data Dictionary 9**

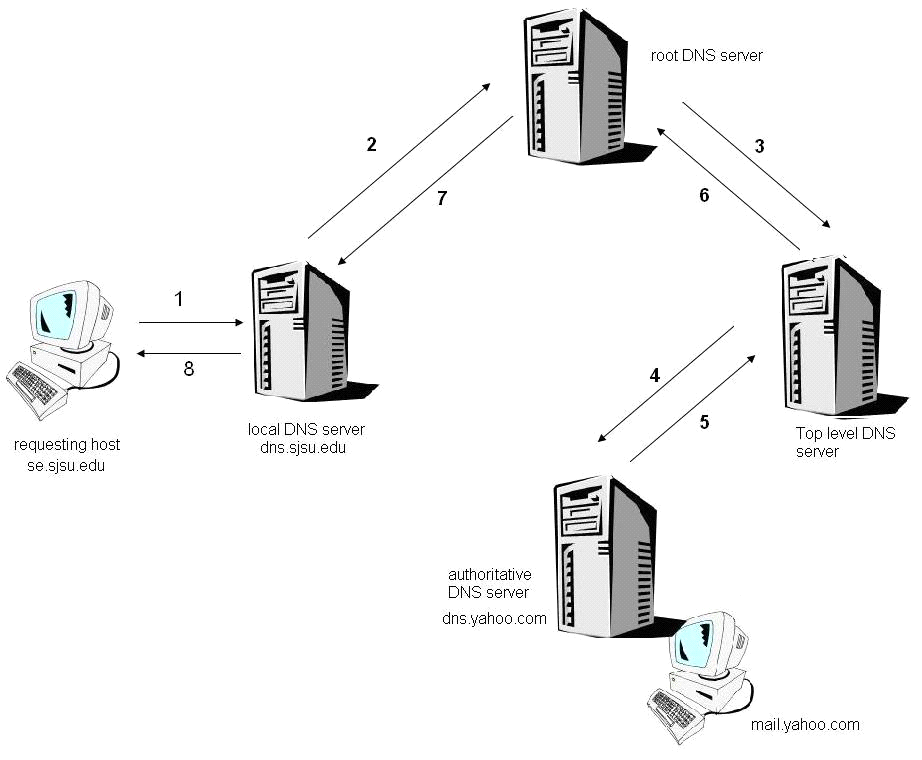
**Appendix B - Group Log 10**

**1. Introduction**

Domain Name System (DNS) enables us to use hierarchical, friendly names to easily locate computers and other resources on an IP network.It is a distributed database that contains mappings of DNS domain names to data. It is also a protocol for Transmission Control Protocol/Internet Protocol (TCP/IP) networks, defined by the Requests for Comments (RFCs) that pertain to DNS. DNS defines the following:

* Mechanism for querying and updating the database.
* Mechanism for replicating the information in the database among servers.
* Schema for the database.

Working of DNS:-



**1.1 Document Purpose**

The purpose of this Software Requirement Document (SRS) is to provide detailed overview of our project called open dynDNS.

**1.2 Product Scope**

* This project aims to provide open-source DNS solution for end-users who often find themselves in changing network environment (Dynamic IP address, hostname etc)
* These users would like to dynamically configure their own DNS server and make their servers accessible from the said network

**1.3 Intended Audience and Document Overview.**

**Intended Audience**

Clients :The user of the system will get a clear idea of the software and hardware requirements to be engaged.

Developers:Project developers have an advantage of quickly understanding the methodology enabled and personalizing the project.

the clients are suggested to go through the requirement section thoroughly before installing the software.

**Document Overview**

The following parts of this document provide a general description of the project.

Section 2 describes the characteristics of the users of this application, some of the constraints encountered while working on the project and a few assumptions and dependencies made by the developers.

Section 3 provides primarily technical documentations for developers such as the hardware used for this application, the functional requirements, data requirements and constraints and assumptions made while designing and implementing the DNS solution.

Section 4 is for non specific requirements.

***1.4* Definitions, Acronyms and Abbreviations**

* DNS : Domain Name System
* REST : Representational state transfer
* MDNS : Multicast DNS
* JSON : JavaScript Object Notation

**1.5 Document Conventions**

When writing this SRS for Domain name system, the following terminologies are used:

In this document ,we use the following general typographic conventions :

**To describe We use the style**

1) URL,hostname,filename,or a new term or concept : Fixed Width

2) literal user input : Fixed Width BOLD

3) program output : Fixed Width

**1.6 References and Acknowledgments**

1 RFC 1034, Domain Names - Concepts and Facilities, P. Mockapetris, The Internet Society (November 1987) [1]

2 <http://dyn.com/blog/dns-why-its-important-how-it-works/>

3

**2 GENERAL DESCRIPTION**

This section of the SRS should describe the general factors that affect 'the product and its requirements. It should be made clear that this section does not state specific requirements; it only makes those requirements easier to understand.

**2.1 Product Perspective**

The domain/server couple should be capable of deploying into any traditional network.

For each version of the project :

* **Public**

Domain : Public wifi network.

Server : any public ISP subscription network.

The domain on the client side makes use of various IP detecting websites to determine the correct public IP address of itself.

* **Local**

-Domain and client : Any traditional network of a small company

mDNS clients are installed on each participating machine in the multicast network for the distributed protocol to function. It should be reminded that mDNS is installed above a standard internal DNS system and therefore the updates should be taken into account by the DNS server.

**2.2 Product Functionality**

This subsection of the SRS should provide a summary of the functions that each version of the software will perform :

* **Public**

The domain/server couple will communicate with each other to update periodically the dynamic IP address of the client-side domain. This update is stored into a webserver (JSON format) and also directly into the database files of the DNS server

* **Local**

When a newly arrived client or service server arrives on the network, it should be able to locate the DNS servers immediately and sends a REST update to the said DNS server with its new local IP address. From then onwards, the client running the domain will be accessible by anyone in the network.

The client also automatically sends a REST update when it leaves the network.

Let it be known that the connection between server and client is protected with and SSL connection. Updates are sent securely on any network while the identity of the client will be always be verified.

If UPnP is enable on this network, any external user will be able to resolve DNS names to all the machines in the private multicast network.

**2.3 Users and Characteristics**

There are 2 main groups of user that interacts with the system :

**Administrator** Admins should have basic knowledge of DNS configurations and basic java understanding, as initial domain configurations are required in Bind9 and hosts should be manually declared beforehand in the REST webserver to prevent unwanted hosts adding themselves to the domain

**Clients** Clients should be able to install the domain on their respective devices such as tablets and laptops.

**2.4 Operating Environment**

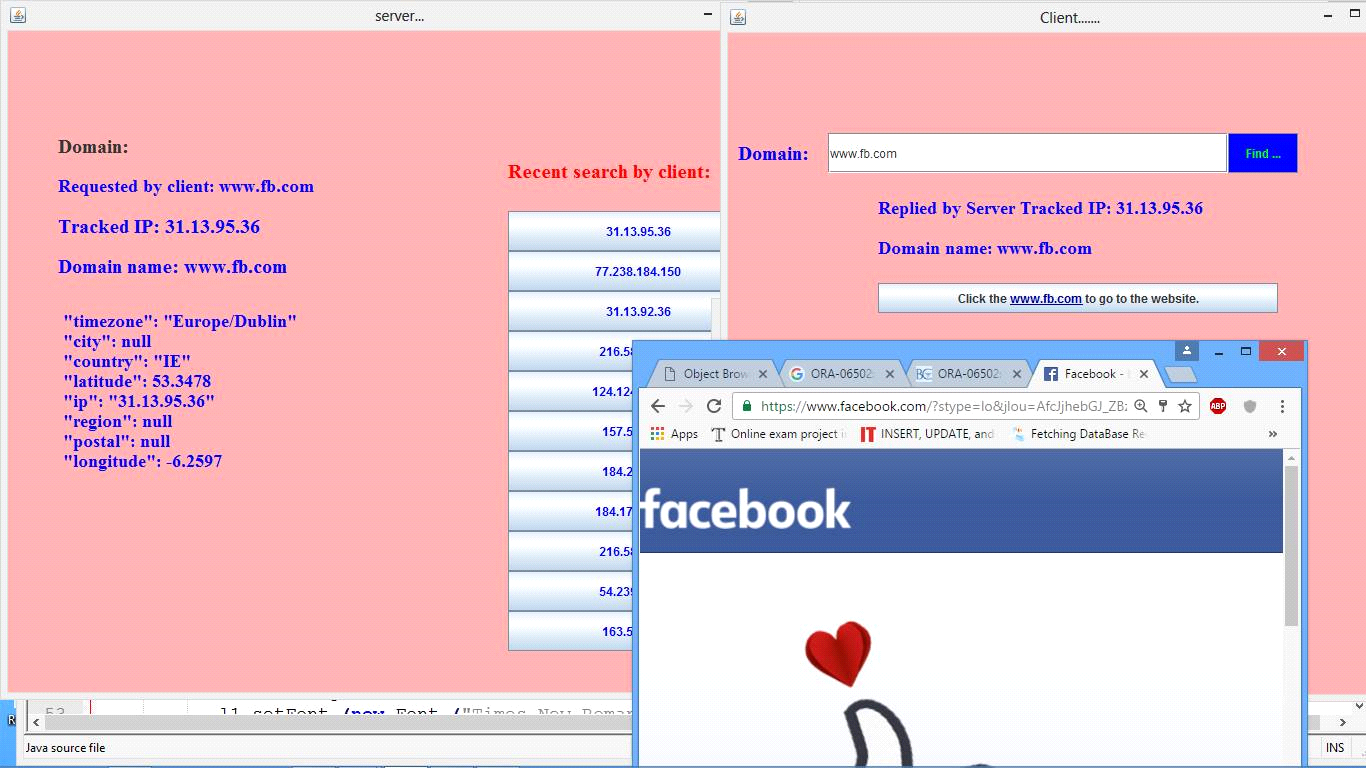
This Domain Name System should run on Operating Systems:Win2K / WinXP / Vista / Win7 / Win8. It is written in a windows-depended language so unfortunately it won’t run on a linux-based system or mac.

**2.5 Design and Implementation Constraints**

This program is created using java programming language and uses the Qt4 libraries for the main dnsClient and dnstServer modules. So a minimum PC having at least 64mb of RAM and CPU over 400mhz is required to run the program with good speed. Also the program uses at least 15 megabytes of hard disk space to store the program libraries.

For the connection stream Tcp-Ip is used as its the common gateway for internet applications.

**2.6 User Documentation**



**2.7 Assumptions and Dependencies**

* **Assumptions**

- HTTP requests (GET, PUT, POST) are enabled on most public wifi network and small private network of a company

- DNS queries are enabled on any network .

**3 Specific Requirements**

**3.1 External Interface Requirements**

**3.1.1 User Interfaces**

**3.1.2 Hardware Interfaces**

The minimum configuration required to run this project are:

* Processor – i3
* Hard Disk – 40 GB
* Memory – 260MB RAM
* Cache RAM -256 KB
* Clock speed – 550 MHZ
* System bus speed – 400MHZ

**3.1.3 Software Interfaces**

**Front end**: java, html

**Back end**: SQL server

**Software Used:**

* Operating system-Windows 7 and above
* Back end design-Microsoft SQL Server 2008
* Front end - Netbeans
* Language- JDK 1.5.

**3.1.4 Communications Interfaces**

Setting up the server into server mode requires that there will be open ports for accepting connections from the clients. The connection between the client and the server uses Connection-oriented communication,via TCP/IP—Transfer Control Protocol/Internet Protocol, implements reliable delivery of messages.

**3.2 Functional Requirements**

**Function**: To dynamically direct the client who enter the url,to perticular Ip-address using the internet.

**Description**: Implement an dynamic DNS server and to run on various device (PC, laptop, Android tablet etc)

**Inputs:** Dynamic IPs

**Source:** Users who host their websites or depositories behind a router/modem with dynamic IP

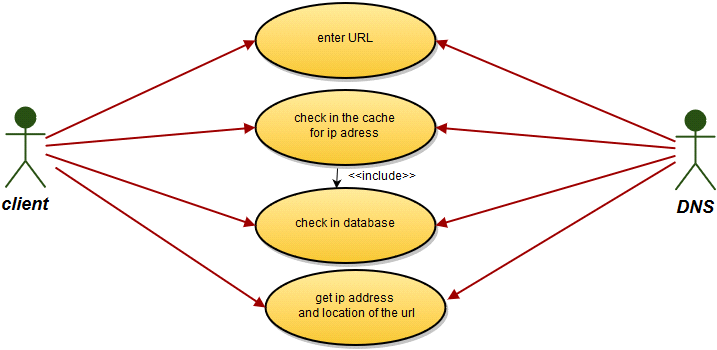
**Outputs**: Dynamic DNS updates

**Destination:** Open DynDNS server

**Action**:Open DynDNS client must notify DNS server whenever it a request for a url is generated

**3.3 Behaviour Requirements**

USE CASE VIEW



**4. Other Non-functional Requirements**

**4.1 Performance Requirements**

**Performance :**the system performs as what every user expects .So in every action-response of the system,there are no immediate delays.

**4.2 SAFETY AND SECURITY REQUIREMENTS**

This program uses object oriented mechanisms to protect its data passed using methods.Also there is no currently a security schema of this program. Thus the log files that are being created are readable using a simple text reader

**4.3 Software Quality Attributes**

**Availability**:Checking that the system always has something to funtion and always pop up error messages in case of component failure. In that case the error messages appear when something goes wrong so to prevail availability problems.

**Usability**:Checking that the system is easy to handle and navigates in the most expected way with no delays. In that case the system program reacts accordingly and transverses quickly between its states.

**Functionality**: When the user writes the websites name in the gui he should be directed to the that website using DNS.