User Friendly DNS Diagnosing Dashboard

H. K. M. U. I. K. Kavisekara

2020

User Friendly DNS Diagnosing Dashboard

A dissertation submitted for the Degree of Master of Information Technology

H. K. M. U. I. K. Kavisekara
University of Colombo School of Computing
2020

Abstract

LK Domain Registry is the organization which handles the domain registration service for .lk country code top level domain in Sri Lanka. From various tasks with regards to domain registration that is handled by the technical division of LK Domain Registry, handling customer inquiries related to the functionality of their network services such as website and email is another task that they need to attend daily with increasing number of domain names each day.

When handling above mentioned customer inquiries, a certain process involve in diagnosing the issue to check for any propagation delay or misconfigurations that can provide by the customer. The efficiency and effectiveness of this diagnose process depends on the knowledge and experience of the system engineer since he/she must follow various Linux commands and use third party tools such as intoDNS or MX toolbox to check how the propagation of the change in resource records have reflected on authoritative or recursive servers. This diagnose process may be difficult for a newcomer to adapt and learn quickly. As they use different tools there is no history maintained for these issues which otherwise would have provide more information that is useful for a system engineer with the use of past records.

Considering the issues that arise in the traditional way of DNS diagnose procedure makes this project to initiate. Apart from this, understanding the practical context of DNS and the ability to study the actual work flow of a domain registry can be described as another motive for selection of this project.

A software was designed and developed with a single point of diagnose capability where a certain user need to provide only the domain name where the complete diagnose process executes and provide user friendly message along with additional details regarding resource records. DNS bulk lookup, resource record change monitoring and report generation are some of the functionalities that is additionally provide by the system.

The effort of implementation of the system is challenging yet rewarding at the same time. From the organizational perspective the employees will get the benefit of handling customer inquiries efficiently. From the software development and research perspective it will pave the path to develop more advanced and innovative solutions in the future.

Declaration

The thesis is my original work and has not been submitted previously for a degree at this or any other university/institute.
To the best of my knowledge it does not contain any material published or written by another person, except as acknowledged in the text.

Student Name: H. K. M. U. I. K. Kavisekara
Registration Number: 2017/MIT/038
Index Number: 17550382

Signature: Date:

This is to certify that this thesis is based on the work of Ms. H. K. M. U. I. K. Kavisekara under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by:
Supervisor Name: Dr. T. N. K. De Zoysa

Signature: Date:

Acknowledgement

In this activity, I would first like to thank Dr. Kasun De Zoysa for his supervisory role and guidance given to me throughout the project and for the valuable feedback and comments given to me in implementing the system. I ever appreciate him for directing me in the correct path towards a better outcome.

I also wish to acknowledge the guidance received during the project work from LK Domain Registry. A special thanks to Mr. Chamara Disanayake, Head of Technical Division of LK Domain Registry for his advisory role in the project and encouraging me to initiate the project while providing resources needed in deploying the system. Also I would like to pay my sincere thanks to Ms. Uththara Wijayamanna, System Engineer of LK Domain Registry for her immense support in sharing valuable information related to this project work and for evaluating the system from the start of the project.

With more respect I recall my memories here, with more gratefully for my parents, family members and friends and the supportive institutes all who helped me throughout the years of completing of this academic program.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

LKDR	LK Domain Registry
IP	Internet Protocol
URL	Uniform Resource Locator
DNS	Domain Name System
TLD	Top Level Domain
gTLD	Generic Top Level Domain
ccTLD	Country Code Top Level Domain
ICANN	Internet Corporation for Assigned Names and Numbers
IETF	Internet Engineering Task Force
RR	Resource Records
A Record	Address Mapping Record
CNAME Record	Canonical Name Record
NS	Name Server
MX	Mail Exchanger
DNSSEC	Domain Name System Security Extensions
dig	Domain Information Groper
MVC	Model-View-Controller
ER Diagram	entity–relationship Diagram

Chapter 1 Introduction

1.1 Problem statement

Information sharing and connectivity has spread beyond geographical boundaries globally with online purchasing, social media, e-banking, freelancing, etc… where every organization and individual try to mark their online presence on cyberspace. In such an era where Internet is growing at an exponential rate with ever increasing number of websites each day, domain name plays a major role in ensuring the identity of each of these websites. Among various types of identities, a domain name can be categorized according to the country it represents.

LK Domain Registry (LKDR) is the organization which handles the domain registration service for .lk country code top level domain (ccTLD) in Sri Lanka [1]. The technical division of LKDR mainly handles the task related to TLD (Top Level Domain) zone insert and updates that is require after domain registration process. Apart from this, handling customer inquiries related to the functionality of their network services such as website and email is also included in the activities which undertake by the technical division [2].

When handling customer inquiries mentioned above, the initial step is to check the DNS (Domain Name System) functionality by querying DNS servers. By doing this, they check whether the resource records (RR) such as Address Mapping record (A Record), IP Version 6 Address record (AAAA Record), Canonical Name record (CNAME Record), etc… are correctly configured in the DNS servers and if those records are correct, then check whether the DNS propagation (the time period taken by the Internet Service Provider nodes across the world to update their caches with new name server records of the domain [3]) has occurred properly by using third party tools such as intoDNS[4], MX toolbox[5], etc…[2].

When following the above process, members of the technical division use different tools and queries to solve the DNS issue where it is difficult for a newcomer to adapt and learn quickly. As they use different tools there is no history maintained for these issues which otherwise would have provide more information that is useful for a system administrator with the use of past records.

1.2 Theoretical framework of Domain Name System

DNS can be identified as the *process of mapping the domain name to the corresponding IP address of that domain name* with the use of a directory which maintain list of domain names and IP addresses.

DNS undergo a hierarchically structured procedure to resolve domain names. This structure can be identified as a tree where the *root* is at the top and *top level domains* such as .com, .edu, .net, .lk, .nz, etc… comes at the next level of the tree. Fig.1.1 below shows this tree structure in more details.

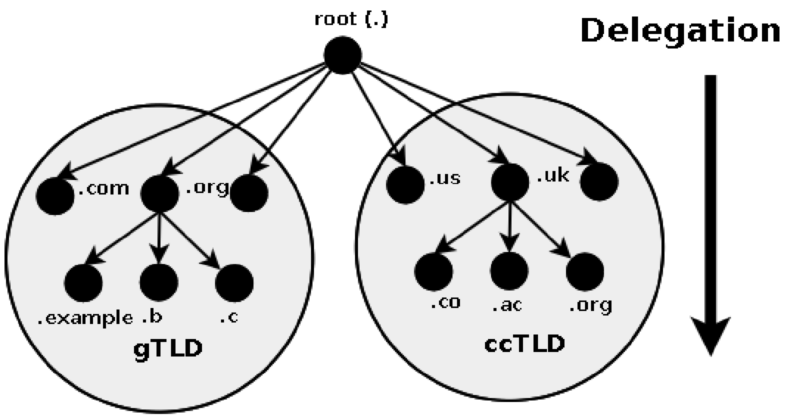


Figure 1.1 DNS tree hierarchy

According to figure 1.1 above, the top level domains are mainly categorized into two sections as:

1. gTLD (Generic Top Level Domains)
2. ccTLD (County Code Top Level Domains)

The gTLDs include top level domains such as .com, .org, .net, etc… which mainly administered by ICANN (Internet Corporation for Assigned Names and Numbers). The ccTLDs include two letter code which represent a country such as .ca (TLD for Canada), .nz (TLD for New Zealand), .jp (TLD for Japan). Since ccTLDs represent a country, it is mainly handles by a certain registry or the government of that country. As such .lk is the TLD for Sri Lanka and the responsible registry which handles this ccTLD is LK Domain Registry [1].

Each layer in the DNS tree mentioned above may *delegate* the *authoritative* control to the next lower level. So authority is delegated from a parent to a child. This delegation can also identify as a *zone* which is an *administrative space* responsible for portion of a domain's name space. Figure 1.2 depicts zones and delegations within .net name space.

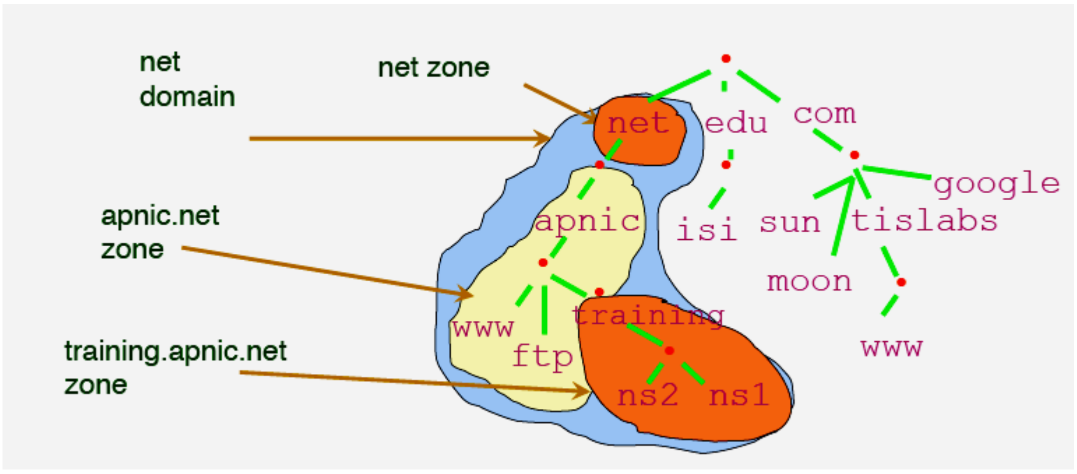


Figure 1.2 Zones and Delegations

As fig.1.2 depicts, each of these zones contains a *zone file* written in IETF standard to be read by any DNS software and it often include *Resource Records* (RR). The resource records that are commonly used is given in table 1.1 below.

RR Type	Name	Functions
A	Address record	Maps domain name to IP address `www.apnic.net. IN A 203.176.189.99`
AAAA	IPv6 address record	Maps domain name to an IPv6 address `www.apnic.net. IN AAAA 2001:db8::1`
NS	Name server record	Used for delegating zone to a nameserver `apnic.net. IN NS ns1.apnic.net.`
PTR	Pointer record	Maps an IP address to a domain name `99.189.176.203.in-addr.arpa. IN PTR www.apnic.net.`
CNAME	Canonical name	Maps an alias to a hostname `web IN CNAME www.apnic.net.`

MX	Mail Exchanger	Defines where to deliver mail for user @ domain
		`apnic.net. IN MX 10 mail01.apnic.net.`
		`IN MX 20 mail02.apnic.net.`

Table 1.1 Common resource records used in a zone file

As described in table 1.1 above each of the above RR points a domain name to a certain IP address. So by inserting or updating these zone files helps navigate the given domain name to its corresponding IP address. The zone file updates are often done by a relevant authority such as a domain registry which handles the given TLD.

1.3. Motivation of the project

Considering the issues that arise in the traditional way of DNS diagnose procedure that was discussed under problem state above makes this project to initiate. Since it is the responsibility of the domain registry organization to ensure that DNS process works as intended once the domain get registered successfully, it is important that these issues will be handled in an effective and efficient manner.

So the problem of how to develop a user friendly diagnosing system with existing tools and technologies will be expecting to address with this project at the outset.

Apart from this, understanding the practical context of DNS and the ability to study the actual work flow of a domain registry can be described as another motive for selection of this project.

1.4. Introduction to project, objective and scope

The project aims at implementing a user friendly system to diagnose DNS issues where a system administrator can enter a domain name and as a result the system is able to inform the user with the issue that is occurring with a single query. This will eliminate the problem of using different tools to DNS and even helps a newly employed system administrator to easily carry out a diagnosing procedure.

The objectives of the project can be identified as:

1. To develop a user friendly system to diagnose DNS issues for efficient customer inquiry handle.
2. To maintain DNS issue history to generate important reports based on past records.
3. To increase the effectiveness of DNS issue diagnosing.

The scope of the project can be identified as:

1. The primary focus of the system is to diagnose DNS issues related to domain names of LKDR.
2. The system is used by both the internal technical staff of LKDR and external users.
3. There are 2 user roles as system engineer and head of technical division from internal technical staff of LKDR.
4. External user can be any person who is interested in resolving a DNS issue of a certain domain name.
5. External users will get a limited functionality through a given public URL to access the system and diagnose DNS issues upon giving a domain name.
6. The system should provide a functionality to query any DNS and generate an output with a user friendly message for the system engineer and external user upon giving the domain name.
7. System should be able to perform DNS bulk lookups when it is needed.
8. System should be able to maintain DNS issue history and generate useful reports from that.
9. System should provide a dashboard to the manager with relevant graphs and charts generated based on DNS issues history.
10. System should be able to monitor DNS record changes (A, TXT, NS,…) of selected domains according to a given schedule (weekly, monthly, …) and alert if a change exists.
11. System should provide a user management module where each system user will have a separate login.

The background of the project will be discussed in the next chapter where a comprehensive description about the requirement analysis and a review of the similar systems will be explained in detail.

Chapter 2 Background

2.1. Existing Diagnose Process

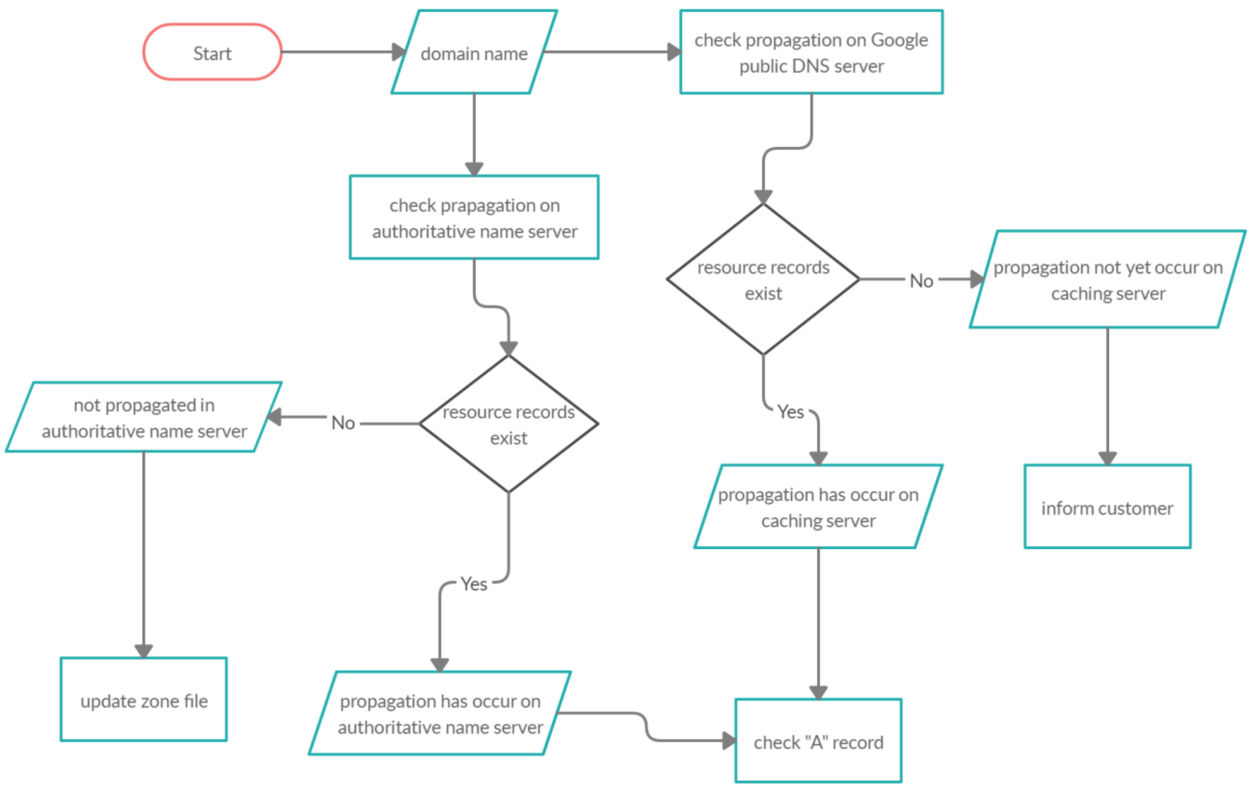
As discussed in section 1.2 “Theoretical framework of Domain Name System”, resource records such as ‘A’ record, ‘TXT’ record, ‘NS’ record of each domain name are included in a zone file. So the name server which has this original zone records can be identified as an *authoritative name server* [6]. Any change done in resource records needs to be available in authoritative name server that has to propagate where it may take from 24 to 72 hours.
Based on the interviews carried out with systems engineer of LKDR, there is mainly website loading issues that comes from customer inquiries. This may be due to the changes made to resource records through online portal [7] by the customer where that change has not reflected on authoritative name server due to propagation delay.
Another possible reason for website loading issue can be propagation delays that can occur on recursive name servers. Recursive name servers will respond to the user with cached data or else send a request to other name servers including authoritative name server [8]. In this scenario, the resource record change has not reflected on recursive name servers even though it has reflected to authoritative name server. Google public DNS can be identified as one of the recursive resolver with high performance [9]. According to the system engineer, they query from Google public DNS to check propagation has occur on recursive name servers [10].
Apart from propagation delays on name servers, wrong web server configurations given by the customer can be identified as another common reason for this issue.

The steps involved in diagnosing procedure for website loading issue can be identified as follows:

1. Check propagation on caching name server by querying Google public DNS.
2. Check propagation on authoritative name server by querying LKDR name server.
3. Check propagation of “A” record on caching name server.
4. Check propagation of “A” record on authoritative name server.
5. Check for resource record updates using third party tool such as IntoDNS.
6. If there is no issue with any resource record, then load website on the browser by providing the URL and IP address respectively. If website is only loading for IP

address and not for the URL or website is not loading for both URL and IP address then customer has given incorrect web server configurations.

A simple flow chart can be described for the above diagnose process as below in figure 2.1 which depict a clear view of the process.



Start
domain name
check propagation on Google public DNS server
check prapagation on authoritative name server
resource records exist
No
propagation not yet occur on caching server
not propagated in authoritative name server
No
resource records exist
Yes
propagation has occur on caching server
inform customer
Yes
update zone file
propagation has occur on authoritative name server
check "A" record

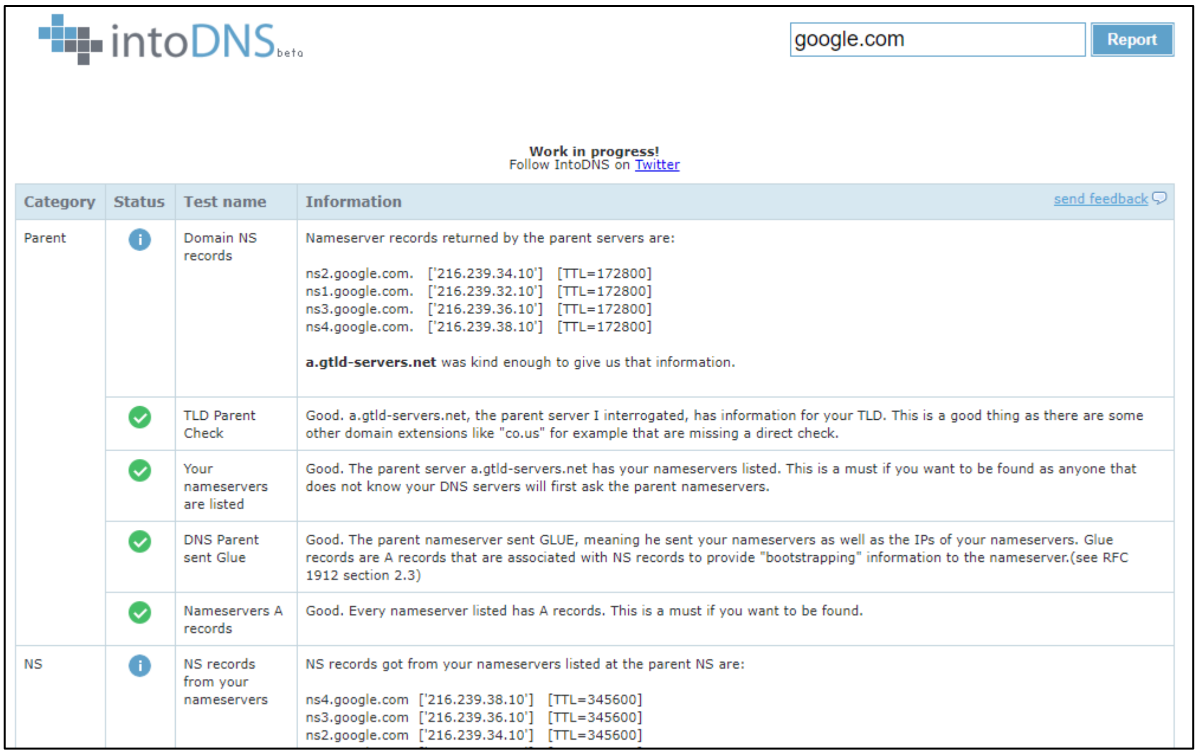
Figure 2.1 Flow chart for existing DNS issue diagnosing process

2.2. Review of Similar Systems

There are some online tools that are available to troubleshoot various DNS related issues and invalid configurations.

- IntoDNS [4]

  This is a web based tool which is freely available to check the health of the domain name and to find any issues with name server configurations.



Work in progress!
Follow IntoDNS on Twitter

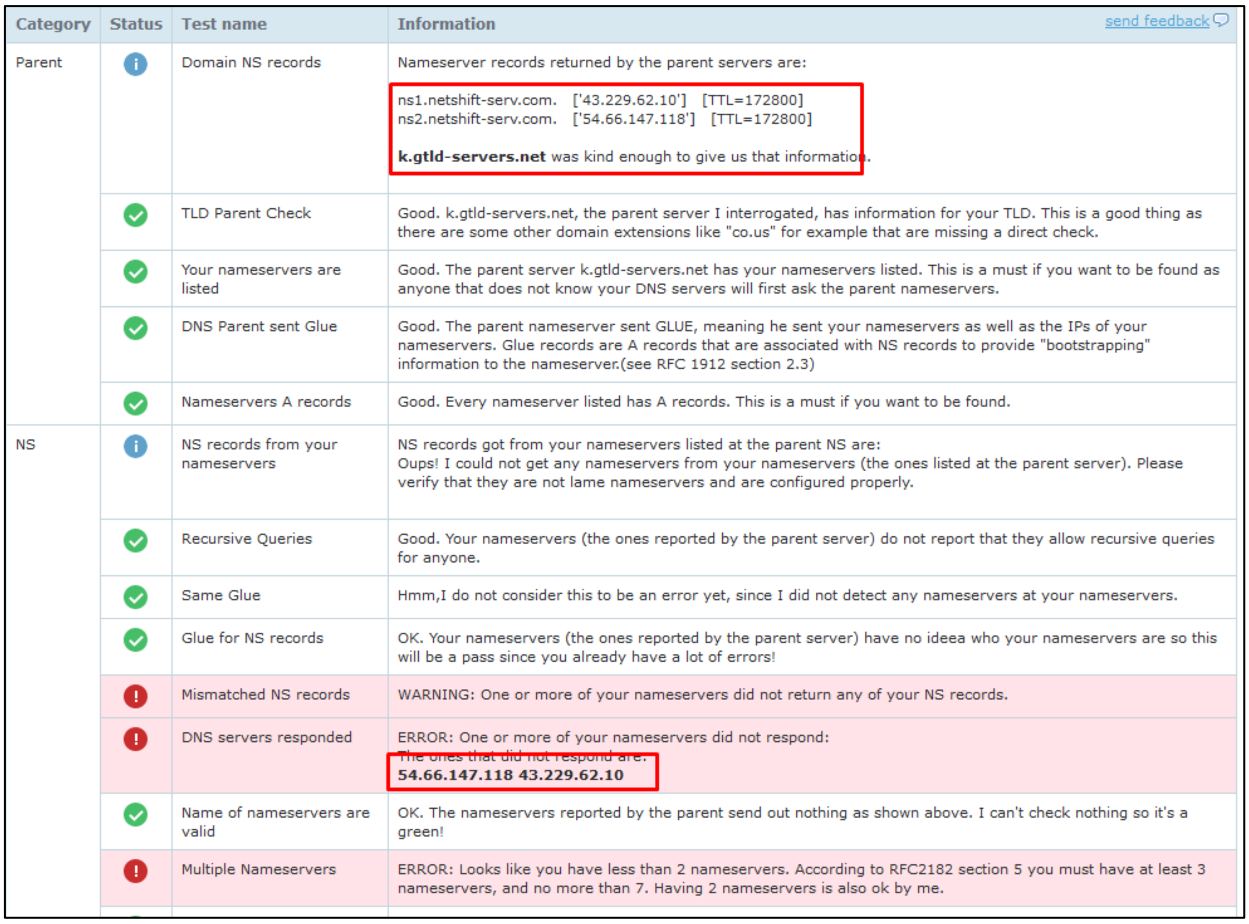
Category	Status	Test name	Information
Parent	ℹ	Domain NS records	Nameserver records returned by the parent servers are: ns2.google.com. ['216.239.34.10'] [TTL=172800] ns1.google.com. ['216.239.32.10'] [TTL=172800] ns3.google.com. ['216.239.36.10'] [TTL=172800] ns4.google.com. ['216.239.38.10'] [TTL=172800] **a.gtld-servers.net** was kind enough to give us that information.
	✓	TLD Parent Check	Good. a.gtld-servers.net, the parent server I interrogated, has information for your TLD. This is a good thing as there are some other domain extensions like "co.us" for example that are missing a direct check.
	✓	Your nameservers are listed	Good. The parent server a.gtld-servers.net has your nameservers listed. This is a must if you want to be found as anyone that does not know your DNS servers will first ask the parent nameservers.
	✓	DNS Parent sent Glue	Good. The parent nameserver sent GLUE, meaning he sent your nameservers as well as the IPs of your nameservers. Glue records are A records that are associated with NS records to provide "bootstrapping" information to the nameserver.(see RFC 1912 section 2.3)
	✓	Nameservers A records	Good. Every nameserver listed has A records. This is a must if you want to be found.
NS	ℹ	NS records from your nameservers	NS records got from your nameservers listed at the parent NS are: ns4.google.com ['216.239.38.10'] [TTL=345600] ns3.google.com ['216.239.36.10'] [TTL=345600] ns2.google.com ['216.239.34.10'] [TTL=345600]

send feedback

Figure 2.2 Generated report of IntoDNS tool

According to figure 2.2 above, a comprehensive report specifying authoritative name servers for the given domain and details about resource records such as A record, CNAME, etc… will be generated.

So if there is an incorrectly configured resource record, then it will highlight that with errors or warnings.



Category	Status	Test name	Information
Parent	ℹ	Domain NS records	Nameserver records returned by the parent servers are: ns1.netshift-serv.com. ['43.229.62.10'] [TTL=172800] ns2.netshift-serv.com. ['54.66.147.118'] [TTL=172800] **k.gtld-servers.net** was kind enough to give us that information.
	✓	TLD Parent Check	Good. k.gtld-servers.net, the parent server I interrogated, has information for your TLD. This is a good thing as there are some other domain extensions like "co.us" for example that are missing a direct check.
	✓	Your nameservers are listed	Good. The parent server k.gtld-servers.net has your nameservers listed. This is a must if you want to be found as anyone that does not know your DNS servers will first ask the parent nameservers.
	✓	DNS Parent sent Glue	Good. The parent nameserver sent GLUE, meaning he sent your nameservers as well as the IPs of your nameservers. Glue records are A records that are associated with NS records to provide "bootstrapping" information to the nameserver.(see RFC 1912 section 2.3)
	✓	Nameservers A records	Good. Every nameserver listed has A records. This is a must if you want to be found.
NS	ℹ	NS records from your nameservers	NS records got from your nameservers listed at the parent NS are: Oups! I could not get any nameservers from your nameservers (the ones listed at the parent server). Please verify that they are not lame nameservers and are configured properly.
	✓	Recursive Queries	Good. Your nameservers (the ones reported by the parent server) do not report that they allow recursive queries for anyone.
	✓	Same Glue	Hmm,I do not consider this to be an error yet, since I did not detect any nameservers at your nameservers.
	✓	Glue for NS records	OK. Your nameservers (the ones reported by the parent server) have no ideea who your nameservers are so this will be a pass since you already have a lot of errors!
	❗	Mismatched NS records	WARNING: One or more of your nameservers did not return any of your NS records.
	❗	DNS servers responded	ERROR: One or more of your nameservers did not respond: The ones that did not respond are: **54.66.147.118 43.229.62.10**
	✓	Name of nameservers are valid	OK. The nameservers reported by the parent send out nothing as shown above. I can't check nothing so it's a green!
	❗	Multiple Nameservers	ERROR: Looks like you have less than 2 nameservers. According to RFC2182 section 5 you must have at least 3 nameservers, and no more than 7. Having 2 nameservers is also ok by me.

Figure 2.3 IntoDNS report that indicate a mismatch in NS records

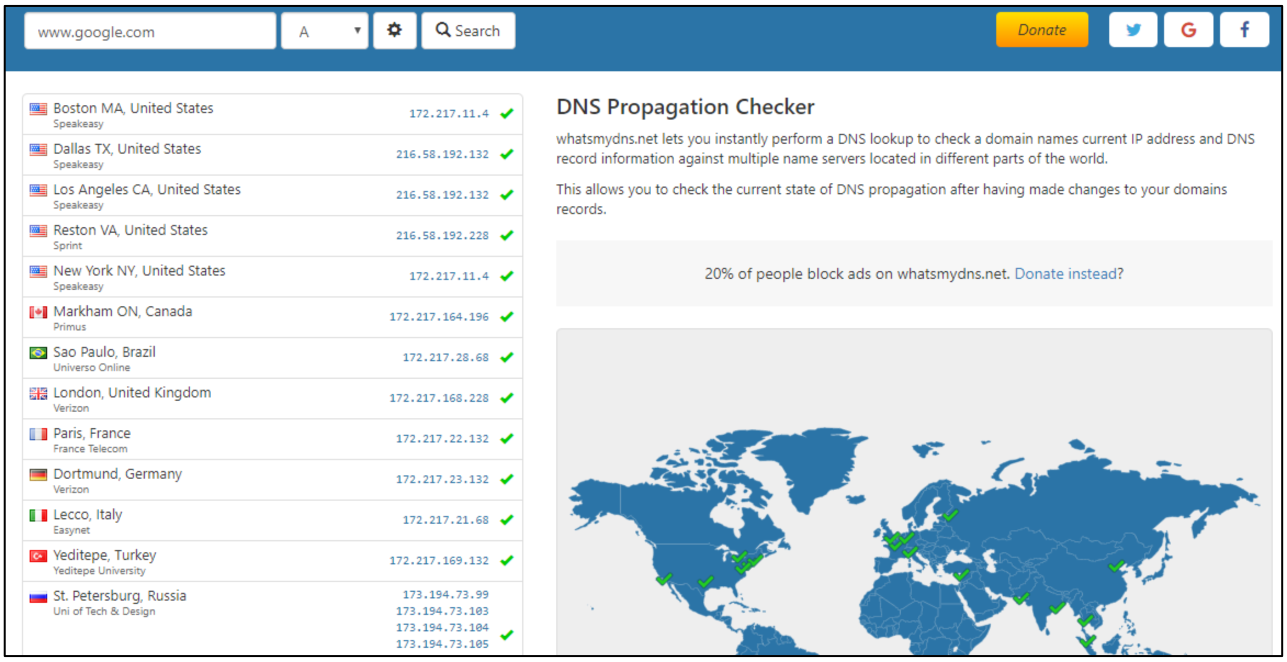
As the figure 2.3 point out, the tool is capable of indicating any mismatches available in NS records which is a frequent error. In a situation where the NS records of a website may be given correctly in the CDN (Content Delivery Network) provider but may be wrong in the domain registrar’s control panel [11].

Apart from basic DNS lookups this tool is also optimized to check email settings as well.

- Whatsmydns [12]

  There are multiple name servers that is located in different parts of the world. So in a situation where there has been any update to DNS records of a certain domain, it has to be reflected in all those servers across the world and it takes certain time duration which often identified as propagation time.

So this online tool provide a convenient way of checking the current IP address and DNS record information of a given domain name in multiple name servers that is located in different parts of the world and produce a distinct output.



Boston MA, United States Speakeasy	172.217.11.4	✓
Dallas TX, United States Speakeasy	216.58.192.132	✓
Los Angeles CA, United States Speakeasy	216.58.192.132	✓
Reston VA, United States Sprint	216.58.192.228	✓
New York NY, United States Speakeasy	172.217.11.4	✓
Markham ON, Canada Primus	172.217.164.196	✓
Sao Paulo, Brazil Universo Online	172.217.28.68	✓
London, United Kingdom Verizon	172.217.168.228	✓
Paris, France France Telecom	172.217.22.132	✓
Dortmund, Germany Verizon	172.217.23.132	✓
Lecco, Italy Easynet	172.217.21.68	✓
Yeditepe, Turkey Yeditepe University	172.217.169.132	✓
St. Petersburg, Russia Uni of Tech & Design	173.194.73.99 173.194.73.103 173.194.73.104 173.194.73.105	✓

DNS Propagation Checker

whatsmydns.net lets you instantly perform a DNS lookup to check a domain names current IP address and DNS record information against multiple name servers located in different parts of the world.

This allows you to check the current state of DNS propagation after having made changes to your domains records.

20% of people block ads on whatsmydns.net. Donate instead?

Figure 2.4 DNS propagation of www.google.com domain

As the figure 2.3 demonstrate above, the tool has perform a check on "A" record of the given domain name and provide a user friendly output indicating list of servers with update status so that a user can see at a glance in case a propagation has not taken place on a certain server.

- MXtoolbox [5]

This tool is capable of performing various DNS lookup and network diagnose including email related DNS records status check.

It is important to check MX records (mail exchanger record) that are responsible in delivery of email to the given email address under a certain domain name.

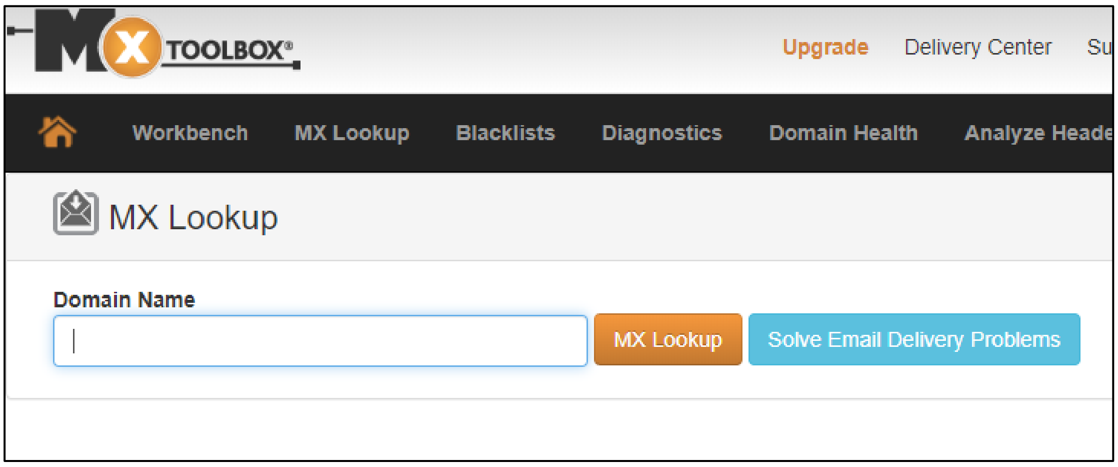
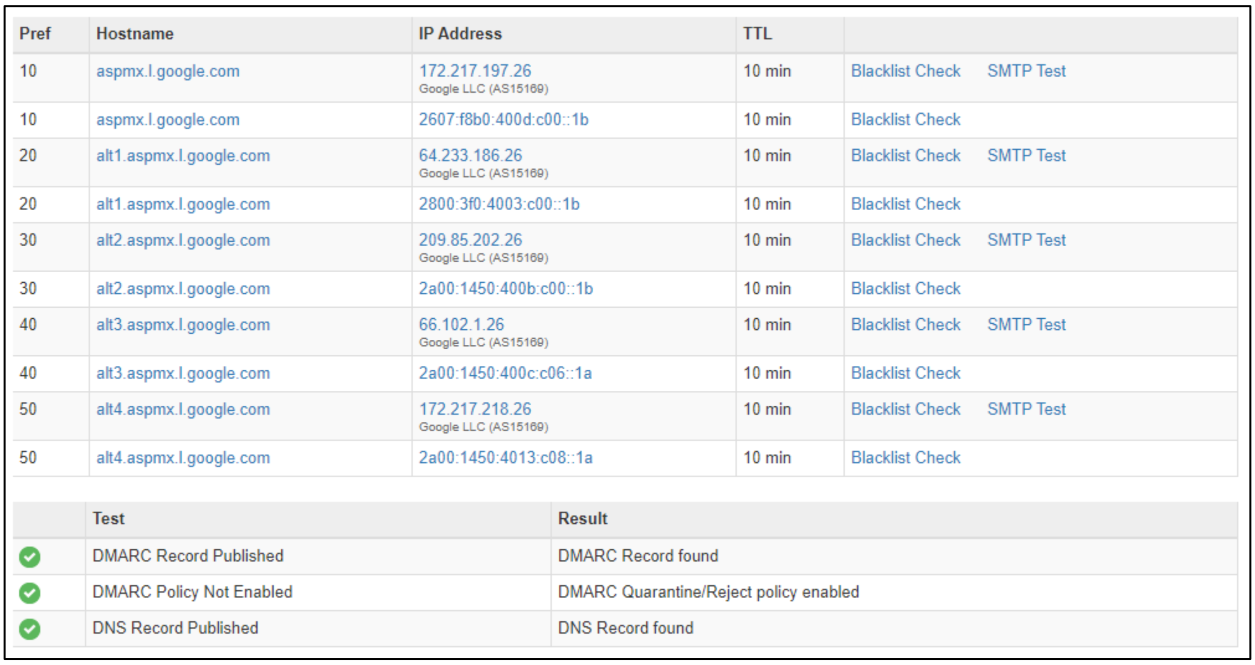


Figure 2.5. MXtoolbox main page

With the use of MXtoolbox the user only needs to provide the domain name to perform a check on relevant MX records. Figure 2.4 above shows the main page of the online tool.



Pref	Hostname	IP Address	TTL		
10	aspmx.l.google.com	172.217.197.26 Google LLC (AS15169)	10 min	Blacklist Check	SMTP Test
10	aspmx.l.google.com	2607:f8b0:400d:c00::1b	10 min	Blacklist Check	
20	alt1.aspmx.l.google.com	64.233.186.26 Google LLC (AS15169)	10 min	Blacklist Check	SMTP Test
20	alt1.aspmx.l.google.com	2800:3f0:4003:c00::1b	10 min	Blacklist Check	
30	alt2.aspmx.l.google.com	209.85.202.26 Google LLC (AS15169)	10 min	Blacklist Check	SMTP Test
30	alt2.aspmx.l.google.com	2a00:1450:400b:c00::1b	10 min	Blacklist Check	
40	alt3.aspmx.l.google.com	66.102.1.26 Google LLC (AS15169)	10 min	Blacklist Check	SMTP Test
40	alt3.aspmx.l.google.com	2a00:1450:400c:c06::1a	10 min	Blacklist Check	
50	alt4.aspmx.l.google.com	172.217.218.26 Google LLC (AS15169)	10 min	Blacklist Check	SMTP Test
50	alt4.aspmx.l.google.com	2a00:1450:4013:c08::1a	10 min	Blacklist Check	

	Test	Result
✓	DMARC Record Published	DMARC Record found
✓	DMARC Policy Not Enabled	DMARC Quarantine/Reject policy enabled
✓	DNS Record Published	DNS Record found

Figure 2.6. MX records list of google.com domain

As shown in figure 2.5 above, MXToolbox will list the MX records in priority order along with other related useful tests upon giving the domain name. User can perform either a blacklist check or SMTP check for each of these MX records. In blacklist check it will perform a test from these mail server IP addresses against number of email blacklists.

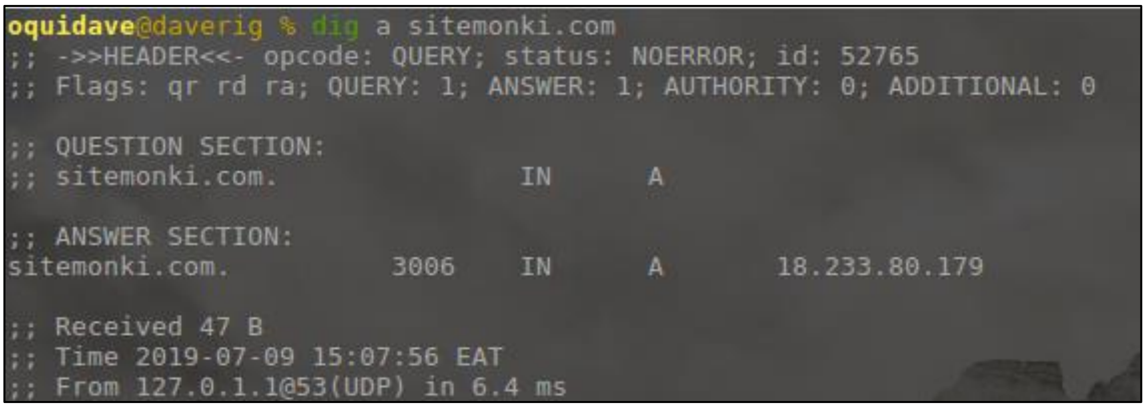
Among other tests that perform by MXToolbox, DMARC (Domain-based Message Authentication, Reporting, and Conformance) record test is useful in current context of email marketing. If a non-aligned email received from a certain domain, DMARC record will specify what to perform by the receiving servers which have an impact on email deliverability. If there is no record setup for DMARC, MXtoolbox is also capable of providing the facility to generate a DMARC record for a given domain as well.

Apart from diagnosing email delivery issues MXtoolbox is also capable of other checks such as domain health checks, DNSSEC, whois lookup, etc…

Apart from the online tools that mentioned above, there are command-line tools as well to query DNS.

- "dig" (Domain Information Groper) command

With "dig" command a user can perform many queries that is important in generating various results that is useful in DNS diagnose process. It is native to Linux and Unix operating systems where it is utilize to perform quick DNS queries from the local machine.



```
oquidave@daverig % dig a sitemonki.com
;; ->>HEADER<<- opcode: QUERY; status: NOERROR; id: 52765
;; Flags: qr rd ra; QUERY: 1; ANSWER: 1; AUTHORITY: 0; ADDITIONAL: 0

;; QUESTION SECTION:
;; sitemonki.com.               IN      A

;; ANSWER SECTION:
sitemonki.com.          3006    IN      A       18.233.80.179

;; Received 47 B
;; Time 2019-07-09 15:07:56 EAT
;; From 127.0.1.1@53(UDP) in 6.4 ms
```

Figure 2.7. details of 'A' record output by dig command

According to figure 2.6 above the given query (dig a sitemonki.com) causes the command to look up the "A" record for the given domain name. As such a common list of commands can be identified as follows [13].

Command	Description
dig example.com any	Query for any type of resource information (NS, A, SOA, …) for a given domain name
dig example.com	Query for ‘A’ record for the given domain name
dig example.com MX	Query to get list of mail servers for a given domain
dig example.com NS	Query to get authoritative DNS servers
dig example.com +trace	Query to trace the path taken
dig @ns.example.com www.example.com	Query using a specific name server

Table 2.1. Common list of 'dig' command

- nslookup command

  This command is native for both Windows and Linux operating systems to do DNS lookups.

  nslookup google.com

  Above query will perform look up for ‘A’ record of google.com domain along with IP address of DNS server. As such nslookup is also capable of finding other resource records such as NS, MX, etc…

When comparing the online tools and command line tools mentioned above, command line tools are used by technically sound people in most situations where they are comfortable with providing commands with their local machines for quick DNS query. But other online tools can be used by both technical and non-technical people with certain level of understanding about DNS. Online tools always provide user interfaces to work and provide output that can be understandable by an average user.

Among the online tools that was discussed above, MXToolbox provide many tools that is useful for diagnosing DNS issues and mail server issues all bundled in one place and many of the tests are free which is very useful in diagnosing process. When compared with IntoDNS, MXtoolbox has this advantage. But it was identified that for most simple DNS lookups many use IntoDNS and to check MX records and email delivery problems, MXtoolbox becomes the main tool.

As command line tool dig command is considered more powerful when compared with nslookup since nslookup has less features and consider as deprecated command.

2.3. Comparison of Alternative Design Strategies

When considering different types of design strategies for the proposed system, there are many approaches that are available.

One such design strategy is to implement the system as a standalone application. Since the DNS lookup module use “dig” command which is native to Linux and Unix environments use to query DNS servers, a standalone system would not be capable of executing these commands if deployed on a Windows environment. Or else the system may needs to bundle with third party tool such as BIND to execute these Linux commands. So to overcome these issues it was decided to implement the system as a web based application which is platform independent where it requires only a web browser for user to interact with the system.

Considering the future scalability of code and maintaining the flexibility of code, it was decided to precede the system implementation using MVC (Model View Controller) architecture so that it will separate the user interfaces from business logic and business logic from data access logic [14]. A customized framework was designed from the scratch [15] since the file structure is more simple and easy to understand when compared to frameworks such as CodeIgnitor, Larevel, Zend, etc…

Chapter 3 Methodology

In order to have a comprehensive understanding of the system modules, its functionalities, flow of events and object interaction with respect to time sequence the designing of the system was done with the use of various design diagrams. As such use case, activity and sequence diagrams was use to design the system which helped in mapping the functional requirements to the system which was unclear at the initial requirement gathering stage.

3.1. High level architecture of the system

The high level architecture of the system depict in figure 3.1 below explain how each module of the system communicate with each other to achieve system functionalities.

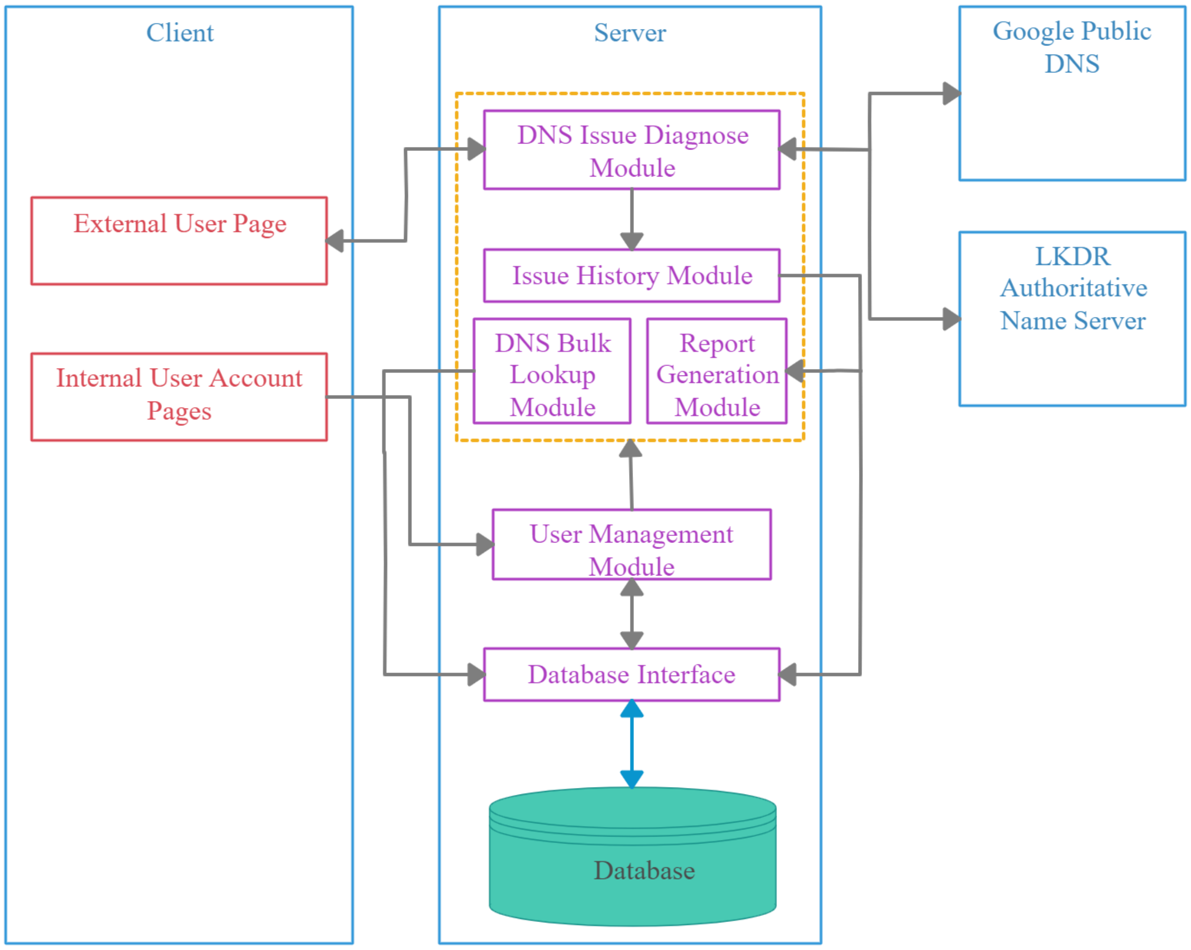


Figure 3.1. High level architecture of the system

According to fig.3.1 above, client tier corresponds with server which hold the application logic. In order to query the propagation, the server communicates with Google public DNS server and authoritative server of LKDR.

The client tier can be mainly categorized in to two views as internal and external user pages which act as the entry point to the system. External users are allowed only to diagnose DNS issues without login into the system. So external user page can directly communicate with DNS issue diagnose module as in fig.3.1 above. Internal user pages needs to go through user management module to access system functionalities. User management module will authenticate and evaluate the user level before redirecting each user to access other modules. Once authenticate, internal users are allowed to access any module in the system. Apart from this the user management module is responsible of maintaining user profiles and managing user roles.

DNS issue diagnose module is responsible of validating given domain name and execute the Linux “dig” command where it will query from the specified server (Google public DNS or LKDR authoritative name server) based on the given query. Once it receive the response from each server it will process the received data, perform diagnose and output a user friendly message to the client tier. Upon completing the diagnose process diagnose module sends diagnose details to issue history module where it perform the insert functionality and add to database by communicating with database interface as depict in fig.3.1.

DNS bulk lookup module will perform lookup functionality to list resource records for multiple domains and monitor any changes made for resource records in a given list of domain names on a given schedule.

Generating reports based on past data and draw graphs based on those data are being done through report generation module.

3.2. Detail Description of Functional Requirements

To understand the flow of events, use case narratives was used as the initial stage of design. Table 3.1 lists all the use cases available and grouping them into a possible use case classes.

Class of use cases	**Use cases**	**Description of use cases**
Use case related to DNS issue diagnose and maintain	Execute DNS issue diagnose	Display user friendly message upon diagnosing the DNS issue
	Add issue to history	Insert details regarding the issue to the history
	Diagnose DNS issue via public URL	Facility for external users to diagnose DNS issues and display user friendly message
Use case related to DNS lookup	Perform DNS bulk lookup	DNS lookup for multiple domain names to list resource records
	Monitor resource record changes	Monitor changes done in RRs of given list of domains on a given schedule
Use case related to profile maintain	Insert profile details	Insert information to profile
	Update profile details	Update existing information in profile
	Delete profile details	Remove existing information in profile
Use case related to dashboard display	View dashboard	Display information of domain health and overall summary of issue diagnose
Use case related to maintain system users	Insert new user role	Insert new user role to the system
	Update user role	Update existing user role with new privileges
	Deactivate user role	Deactivate existing user role from the system
Use case related to generating reports	Generate reports	Generate reports related to DNS issue diagnose summary

Table 3.1. Use cases of DNS diagnose system

Expanded version of use case narratives can be given as follows for each of the above use cases that provide detailed description of functional requirements. Considering the main functionalities of the system, use case classes “DNS issue diagnose and maintain” and “DNS lookup” will be explained in this chapter.

Use case related to DNS issue diagnose and maintain

Use case 1: execute DNS issue diagnose

Primary actor: system engineer

Pre condition: user logged in

Main scenario:

1. User initiate issue diagnose functionality
2. System ask for domain name
3. User enters domain name
4. System validate domain name
5. System perform issue diagnose functionality
6. System generate user friendly message

Alternate scenario:

1. Invalid domain name provided
    a) System ask the user to re-enter domain name correctly
    b) User enters domain name in correct format
    c) System start issue diagnose functionality
2. Domain name does not exist
    a) Diagnose fails, display error message
3. Failure occur on authoritative name server
    a) Diagnose fails, display error message

Use case 2: Add DNS issue to history

Primary actor: system engineer

Pre condition: user needs to log into the system, ‘execute DNS issue diagnose’ use case must be executed before this use case execution

Main scenario:

1. User initiates adding issue to history
2. System gather data such as domain name, system generated diagnose message, current date, logged in user email
3. Display option to insert user comments
4. User insert other comments on issue
5. System perform adding DNS issue to history
6. Display success message

<u>Alternate scenario:</u>

1. User does not insert other comments
    a. System perform adding DNS issue to history
    b. Display success message

Use case 3: Diagnose DNS issue via public URL

<u>Primary actor:</u> external user

<u>Pre condition:</u> none

<u>Main scenario:</u>

1. User go to public URL
2. User initiate DNS issue diagnose functionality
3. System ask for domain name
4. User enter domain name
5. System validate domain name
6. System perform diagnose functionality
7. System display user friendly message

<u>Alternate scenario:</u>

1. Invalid domain name provided
    a. System asks to re-enter domain name correctly
    b. User enter domain name in correct format
    c. System starts issue diagnose functionality
2. Domain name does not exist
    a. Diagnose fail, display error message
3. Server failure occur
    a. Diagnose fail, display error message

<u>Use case related to DNS lookup</u>

Use case 1: perform DNS bulk lookups

<u>Primary actor:</u> system engineer

<u>Pre condition:</u> user needs to log into the system

<u>Main scenario:</u>

1. User initiates DNS bulk lookup
2. System ask to enter multiple domain names
3. User enter multiple domain names

4. System validate each domain name
5. System query for NS records from authoritative name server and Google public DNS
6. Display NS record details of each domain

<u>Alternate scenario:</u>

1. Invalid domain name provided
    a. System asks to re-enter domain names correctly
    b. User enter domain names in correct format
    c. System starts to query for RR
2. Domain name does not exist
    a. Fail to find RR, display error message
3. Server failure occur
    a. Fail to find RR, display error message

Use case 2: monitor resource record changes

<u>Primary actor:</u> system engineer

<u>Pre condition:</u> user needs to log into the system

<u>Main scenario:</u>

1. User initiates monitor functionality
2. System ask to enter domain names that needs to monitor and to select a schedule (weekly, monthly, etc…) to monitor
3. User enter domain name and a schedule
4. System validate each domain name
5. System query for current resource records from authoritative name server
6. System store new domains with their respective RR in monitoring table
7. System trigger the monitor on the given schedule and check for RR changes comparing the RR stored in monitoring table in DB
8. Send alert to user if a change exist in any of the RR of a particular domain name

<u>Alternate scenario:</u>

1. Invalid domain name provided
    a. System asks to re-enter domain names correctly
    b. User enter domain names in correct format
    c. System starts to query for RR
2. Domain name does not exist
    a. Fail to find RR, display error message

3. No change exist in RR
    a. Notify user that there are no changes in RR
    b. System wait until next trigger for monitor
4. Server failure occur
    a. Fail to find RR, display error message

The use case diagram along with other use case narratives are given in ‘Appendix A’ with their respective activity diagram. The scenarios “DNS issue diagnose”, “DNS bulk lookup” and “monitor resource record changes” are given in sequence diagrams which is also included in ‘Appendix A’.

3.3. Database design of the system

Since the functionalities such as maintaining DNS issue history and monitoring RR changes of domains require data to be stored in database tables, the database design was done by focusing on these scenarios.
As discussed in section 3.2 “Detail Description of Functional Requirements”, “add DNS issue to history” use case scenario explains a possible situation where the system engineer will insert the issue to history upon completing DNS issue diagnose process. In this scenario following data is important to be stored in the database with respect to DNS issues that was diagnosed.

- domain_name – the domain name which had the DNS issue that needs to be diagnosed
- diagnose_message – message explaining the propagation details that display after diagnose
- comments – comments given by user regarding the issue
- date – issue diagnosed date
- category_name – corresponding category where the issue belong (NS record issue, TXT record issue, MX issue, etc…)

Considering above attributes and applying database normalization techniques, as a result the ER diagram in fig.3.2 consist of two tables "DNS Issue" which hold the details of the issue and "Issue Category" to store the different issue categories with one-to-many relationship between two tables.

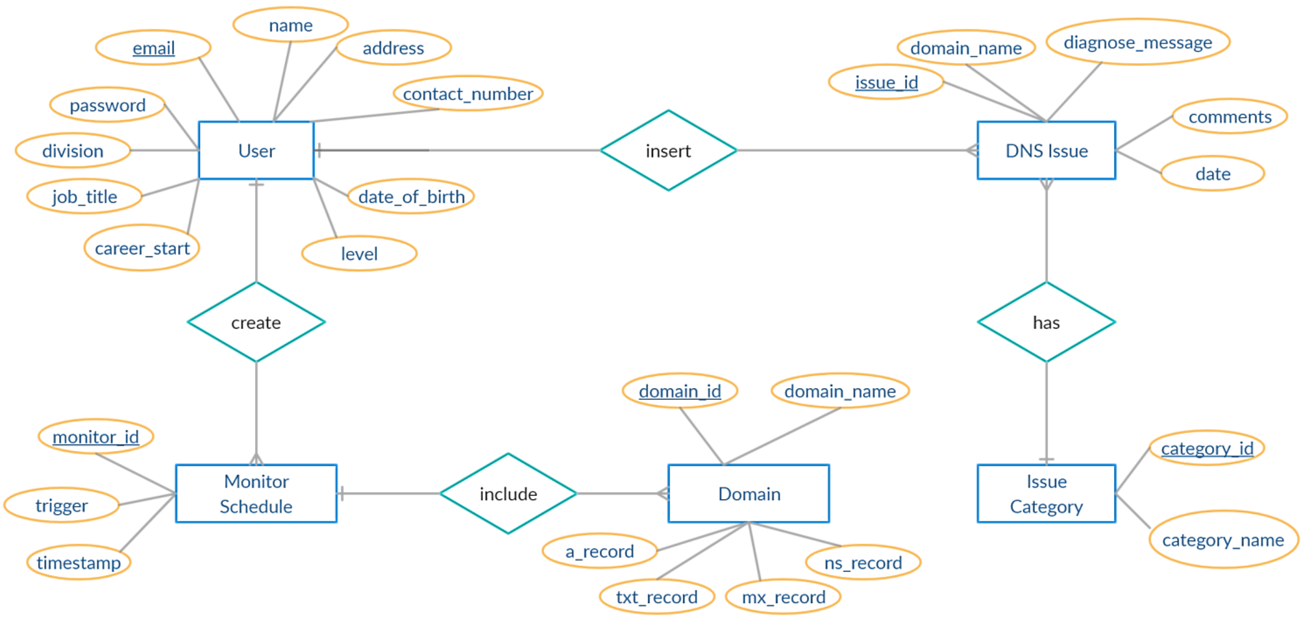


Figure 3.2. Database design of DNS issue diagnose system

As such, the use case scenario "monitor resource record changes" that was discussed in section 3.2 "Detail Description of Functional Requirements", following data was identified as important to be stored in database with respect to each monitoring of domains carried out by user.

- trigger – time duration that the monitor needs to be triggered (monthly, weekly, etc…)
- timestamp – timestamp the user create the monitoring schedule
- domain_name – selected domain name to monitor RR changes
- a_record – initial 'A' record that was given
- txt_record – initial 'TXT' record that was given
- mx_record - initial 'MX' record that was given
- ns_record – initial 'NS' record that was given

since there can be multiple domains in a single monitoring process, two tables included as "Monitor Schedule" to store details of the monitor schedule details and "Domain" to store the domain names along with their respective RRs that was choose to monitor with one-to-many relationship between two tables as depicted in fig.3.2 above.

3.4. Pseudo code for DNS issue diagnose functionality

The most important part of the system is the diagnose functionality. Below pseudo code explain a possible algorithm to diagnose an issue by querying Google public DNS and authoritative name server.

```
Function diagnose(){
    domainName = post value from domain name field;
    IF(domainName is valid) THEN
        // query to get NS records from Google DNS
        googleNSOut = queryGoogleNS(domainName);
        IF(googleNSOut returned NS records) THEN
            print("NS records has propagated to Google public DNS");
            print array(googleNSOut);
        ELSE
            // query to get NS records from authoritative name server
            authNSOut = queryAuthNS(domainName);
            IF(authNSOut returned NS records) THEN
                print("NS records has propagated to authoritative name server");
                print array(authNSOut);
            ELSE
                // query to get 'A' records from Google DNS
                // if 'A' record exist then print results, else get 'A' record from
                        //authoritative server
                // if 'A' record exist then print results, else
                    //print message " 'A' record has not yet propagated to
                        authoritative server"
            END IF
        END IF
    ELSE
        print(error message);
    END IF
}
```

Sample pseudo code for *queryGoogleNS()* function is given below which explain how to extract ‘NS’ records and output user friendly message based on different levels of decisions. For a better understanding, a possible ‘dig’ command output is included in figure B.1 of Appendix B.

```
Function queryGoogleNS(domainName){
    shellOutNS = execute command via shell(“dig <domainName> NS @ 8.8.8.8”);

    answerStart = get starting position of answer section(shellOutNS);
    answerEnd = get end position of answer section(shellOutNS);

    IF(answerStart exist AND answerEnd exist) THEN
        shellAnswer = extract answer section from shellOutNS;
        IF(shellAnswer exist) THEN
            // extract NS recrods from shellAnswer
            regexNS = ‘/pattern to match NS record/’;
            nsRecords = ARRAY();
            START LOOP shellAnswer
                matchString = perform regular expression match(regexNS,
                                shellAnswer);
                IF(matchString is TRUE) THEN
                    nsRecords [] = matchString;
                END IF
            END LOOP
            return nsRecords;
        ELSE
```

```
                    message = “NS records not propagated to Google public DNS”;
                    return message;
          END IF
     ELSE
          message = “no response from Google public DNS”;
          return message;
     END IF
}
```

Chapter 4 Implementation

The implementation stage of the system was achieved in different levels of the development life cycle. At the initial requirement gathering phase in order to have a clear understanding of the requirements, a low fidelity prototype of the system was built. Apart from understanding requirements, identifying the most suitable technology for implementation and implementation environment was also achieved at initial stages of implementation work.

4.1. Details on implementation environment

As explained in section 3.4. “Pseudo code for DNS issue diagnose functionality”, it require the Linux ‘dig’ command to be executed in the program. The reason for executing the ‘dig’ command rather than using any built in function is due to the fact that those built in functions does not provide the facility to query for resource records on a given name server. Since the selected scripting language is PHP, the built in PHP function “dns_get_record()” does not have the ability to retrieve resource records from a given name server.

Due to this limitation with built in PHP function, it was decided to execute the exact command that is used in the existing diagnose process. For this, Linux command needs to be executed. Since current working environment is Windows and the development started using XAMPP local server where it was not possible to run Linux commands. As a result it was decided to precede the development of the system on a testing server.

NetBeans IDE [16] was used as the development environment by downloading with PHP extension to support for PHP programming within the IDE. Since it is free software and runs on Windows environment it was decided to use this IDE while configuring to run the project as a remote web site with FTP enabled and remote connection to testing server. This option is more advantageous since it avoid having to use third party FTP applications such as FileZilla to establish FTP connection remotely.

4.2. Implementation Details of the project

The project was developed according to MVC architecture. For this purpose, a simple framework was developed from the scratch [15] rather than using already available framework such as CodeIgnitor, Laravel, etc…

The file structure organized as depict in fig.4.1 below with a 'Model' to interact with database, 'View' to display relevant web pages to user while integrating data from Controller and 'Controller' to exchange data between Model and View.

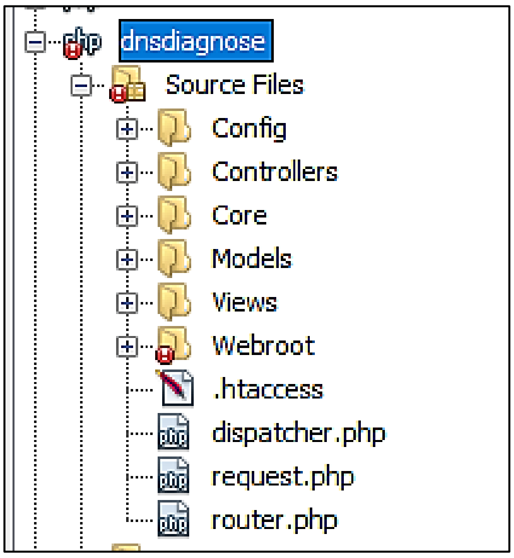
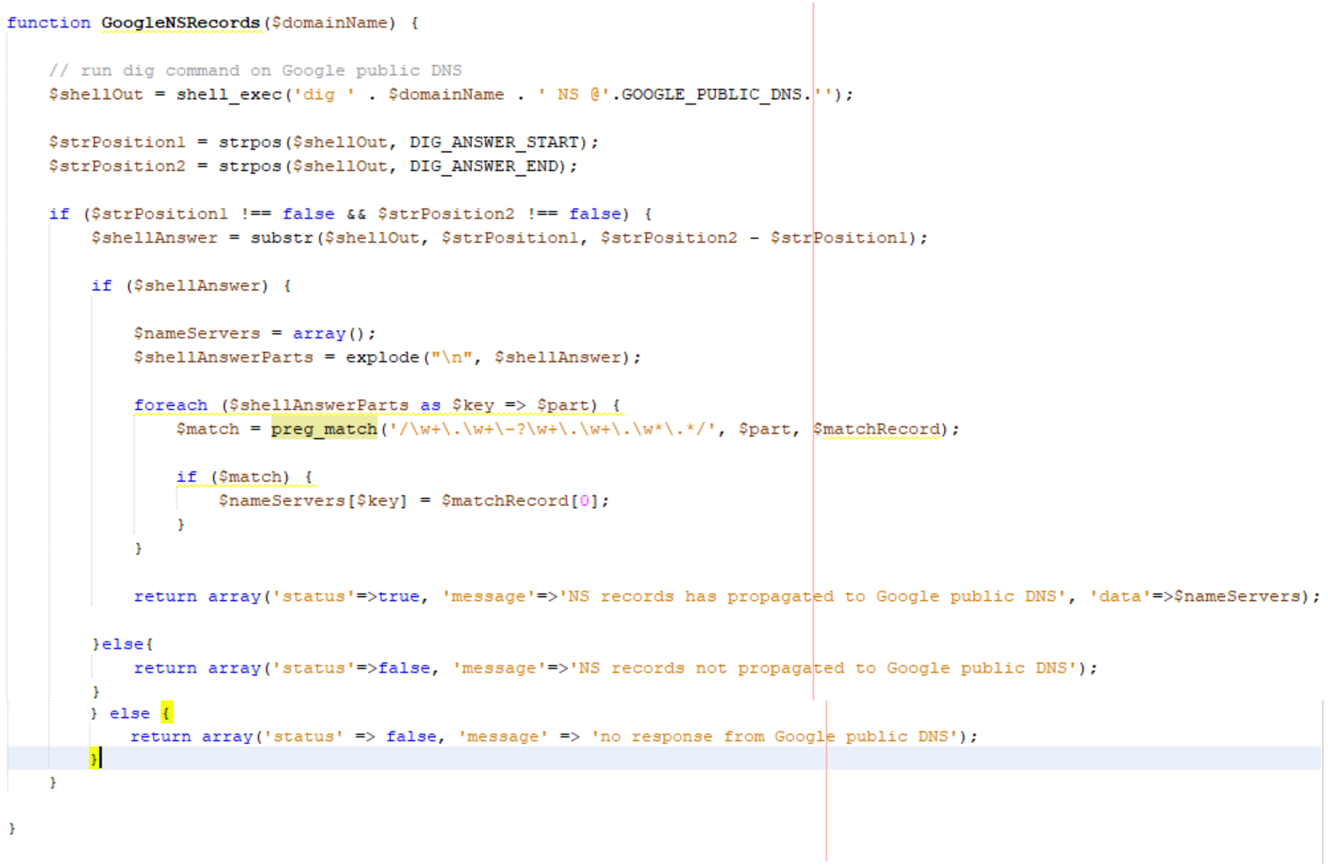


Figure 4.1. File structure of the system

The corresponding source code segment for the pseudo code discussed in section 3.4 "Pseudo code for DNS issue diagnose functionality" can be given in fig.4.2 below. The code segment shows how to extract 'NS' records from 'dig' command by querying Google public DNS and output user friendly message based on different levels of decisions.



```php
function GoogleNSRecords($domainName) {

    // run dig command on Google public DNS
    $shellOut = shell_exec('dig ' . $domainName . ' NS @'.GOOGLE_PUBLIC_DNS.'');

    $strPosition1 = strpos($shellOut, DIG_ANSWER_START);
    $strPosition2 = strpos($shellOut, DIG_ANSWER_END);

    if ($strPosition1 !== false && $strPosition2 !== false) {
        $shellAnswer = substr($shellOut, $strPosition1, $strPosition2 - $strPosition1);

        if ($shellAnswer) {

            $nameServers = array();
            $shellAnswerParts = explode("\n", $shellAnswer);

            foreach ($shellAnswerParts as $key => $part) {
                $match = preg_match('/\w+\.\w+\-?\w+\.\w+\.\w*\.*/', $part, $matchRecord);

                if ($match) {
                    $nameServers[$key] = $matchRecord[0];
                }
            }

            return array('status'=>true, 'message'=>'NS records has propagated to Google public DNS', 'data'=>$nameServers);

        }else{
            return array('status'=>false, 'message'=>'NS records not propagated to Google public DNS');
        }
        } else {
           return array('status' => false, 'message' => 'no response from Google public DNS');
        }
    }

}
```

Figure 4.2. Source code to query Google DNS and extract NS records

Chapter 5 Evaluation

5.1. User Evaluation

The objectives of the project can be identified as:

1. To develop a user friendly system to diagnose DNS issues for efficient customer inquiry handle.
2. To maintain DNS issue history to generate important reports based on past records.
3. To increase the effectiveness of DNS issue diagnosing.

In order to evaluate the achievement of above project objectives, an evaluation process carried out by allowing each user to work with the final system and acquiring their feedback through a questionnaire. As the main users, system engineer and head of technical division participated in the evaluation since they are using majority of system functionality.

At the beginning of the project the diagnose functionality was identified as a major requirement that needs to be fulfill by the system. So the priority was given to implement DNS diagnose functionality at the initial design and implementation stage. So the evaluation of DNS diagnose section was started soon after implementing that functionality. Based on that evaluation and considering other functionality of the finale system, the questionnaire attached in figure C.1 in Appendix C was completed by the system engineer.

According to figure C.1, first 4 questions which is the first part of the questionnaire were included to check the overall satisfactory level of the system in general. In the second part of the questionnaire depict in figure C.2, it focuses on evaluating the satisfactory level of system functionality that is specifically provided for the system engineer. As such, the satisfactory level of executing DNS issue diagnose, DNS bulk lookup, monitor resource record changes, etc… was evaluated in the second part of the questionnaire.

The same format was followed when designing the questionnaire for head of technical division by including the same set of questions that was included in questionnaire of system engineer. As given in figure C.3 of Appendix C, second part of the questionnaire held questions related to functionality that is provided for head of technical for evaluating those functions. As such the satisfactory level of system dashboard, user role management, report generation, etc… was evaluated in the second part of the questionnaire.

5.2. Test cases related to system functionalities

Following test cases were identified for each function of the system depict in table 5.1 below.

Test case ID	Test scenario	Test cases	Expected results
1	Check user login function	login with valid user name and password	successfully login to the system
2		login with invalid username and password	display invalid user name or password message
3		login with invalid username or password	display invalid user name or password message
4		login without entering username and password	mark empty required fields
5		login without entering username or password	mark empty required fields
		login as user level 1	redirect to the page to select dashboard view or diagnose view
		login as user level 2	redirect to dashboard view
		login as user level 3	redirect to diagnose view
6	Check DNS issue diagnose function	enter valid domain name	execute diagnose and output user friendly message
7		enter invalid domain name	display invalid domain name message
8		diagnose without entering domain name	mark empty required fields
9	Check diagnose DNS issue via public URL	enter valid domain name with correct captcha	execute diagnose and output user friendly message
10		enter invalid domain name and captcha	display invalid domain name and captcha message
11		enter invalid domain name or captcha	display invalid domain name or captcha message
12		diagnose without entering domain name or captcha	mark empty required fields
13	Check DNS bulk lookup function	enter maximum number of domain names that allow for bulk lookup	start bulk lookup and produce RR of each domain name
14		enter minimum number of domain names that allow for bulk lookup	start bulk lookup and produce RR of each domain name
15		execute bulk lookup without entering any domain name	mark empty required fields
16		insert one or more invalid domain names for bulk lookup	mark invalid domain names
17	Check resource record change	enter list of domain names and select monitoring schedule	start scheduled monitoring of RR change in given domain list

18	monitoring function	enter list of domain names without selecting monitoring schedule	mark empty required fields
19		select a monitoring schedule without entering domain names	mark empty required fields
20		insert one or more invalid domain names to monitor	mark invalid domain names
21		change RR in one domain that is in a monitoring list	alert the change on the scheduled date
22		change RR in multiple domains that is in a monitoring list	alert the change of each domain on the scheduled date
23	Check user profile maintain functionality	enter all the required fields and save	successfully save user data
24		empty required fields and submit	mark empty required fields
25		enter invalid data and submit	mark invalid data fields
26	Check maintain user role functionality	enter valid email and select user level	successfully insert user and send email to user with auto generated password
27		save user without email and user level	mark empty required fields
28		save user without email or user level	mark empty required fields
29		search user by valid email	display user details
30		search user by invalid email	display invalid email message

Table 5.1. Test cases related to system functions

5.3. Testing and user evaluation results

According to the evaluation carried out with system engineer and head of technical division, there was a positive feedback provided by system engineer for the overall system features provided to diagnose DNS issues. The head of technical division had a positive impression on the system since it fulfill the major requirement they were expecting, which is the diagnose functionality.

The response given for the questionnaire by the system engineer is given in Appendix C figure C.4, Answered questionnaire of system engineer and response of the head of technical division is given in figure C.5, Answered questionnaire of head of technical division.

As described in section 5.1. User evaluation, both questionnaires that was given to system engineer and head of technical division included same set of questions to check the overall satisfactory level of the system in general. So based on the responses given to these questions by both users, a summary can be given as follows (responses were gathered by providing the likert scale where 1 is strongly disagree and 5 is strongly agree).

Question	**1**	**2**	**3**	**4**	**5**
easiness to learn the system				50%	50%
straightforwardness of task accomplishment by using the system				50%	50%
clearness in user interface elements				100%	
clearness in sequence of interfaces					100%

Table 5.2. Response summary of overall satisfactory level in user interfaces

Based on the response summary given in Table 5.2 above, the average satisfactory level of learning the system and straightforwardness of task accomplishment by the system have a strong agreement by the respondents. The average satisfactory level of user interface clearness also has a strong agreement level.

As for the individual evaluation of each user, figure C.4 and C.5 provide the evidence that the system receive a positive satisfactory level in majority of functionalities. According to head of technical division, the system was able to provide expected administrative functionality including the diagnose functionalities according to the given requirements. He suggest that the system can be improved by examine online tools such as MXToolbox which is a powerful tool in diagnosing mainly email delivery issues. Higher satisfactory level was given by the system engineer for DNS issue diagnose functionality due to the system's ability to perform single point diagnose just by providing the domain name which otherwise could have being a procedure where she has to execute several commands to diagnose the issue.

Testing of the system was done in two phases. In the first phase, several failures were identified and fixed in each module. In the second phase, 100% pass rate was achieved. The results of the test cases can be given in Table 5.3 as below.

Test case ID	Test scenario	Test cases	Expected results	Test result	
				Phase 1	Phase 2
1	Check user login function	login with valid user name and password	successfully login to the system	pass	pass
2		login with invalid username and password	display invalid user name or password message	pass	pass
3		login with invalid username or password	display invalid user name or password message	pass	pass
4		login without entering username and password	mark empty required fields	pass	pass
5		login without entering username or password	mark empty required fields	pass	pass
		login as user level 1	redirect to the page to select dashboard view or diagnose view	fail (page loaded with incorrect layout)	pass
		login as user level 2	redirect to dashboard view	pass	pass
		login as user level 3	redirect to diagnose view	pass	pass
6	Check DNS issue diagnose function	enter valid domain name	execute diagnose and output user friendly message	pass	pass
7		enter invalid domain name	display invalid domain name message	fail (error message did not displayed)	pass
8		diagnose without entering domain name	mark empty required fields	pass	pass
9	Check diagnose DNS issue via public URL	enter valid domain name with correct captcha	execute diagnose and output user friendly message	pass	pass
10		enter invalid domain name and captcha	display invalid domain name and captcha message	pass	pass
11		enter invalid domain name or captcha	display invalid domain name or captcha message	pass	pass
12		diagnose without entering domain name or captcha	mark empty required fields	pass	pass

13	Check DNS bulk lookup function	enter maximum number of domain names that allow for bulk lookup	start bulk lookup and produce RR of each domain name	pass	pass
14		enter minimum number of domain names that allow for bulk lookup	start bulk lookup and produce RR of each domain name	fail (additional table row added at the end)	pass
15		execute bulk lookup without entering any domain name	mark empty required fields	pass	pass
16		insert one or more invalid domain names for bulk lookup	display invalid domain name message	faile (error message did not displayed)	pass
17	Check resource record change monitoring function	enter list of domain names and select monitoring schedule	start scheduled monitoring of RR change in given domain list	faile (monitoring did not run according to the given schedule)	pass
18		enter list of domain names without selecting monitoring schedule	mark empty required fields	pass	pass
19		select a monitoring schedule without entering domain names	mark empty required fields	pass	pass
20		insert one or more invalid domain names to monitor	mark invalid domain names	pass	pass
21		change RR in one domain that is in a monitoring list	alert the change on the scheduled date	pass	pass
22		change RR in multiple domains that is in a monitoring list	alert the change of each domain on the scheduled date	fail (error in alert message)	pass

23	Check user profile maintain functionality	enter all the required fields and save	successfully save user data	pass	pass
24		empty required fields and submit	mark empty required fields	pass	pass
25		enter invalid data and submit	mark invalid data fields	pass	pass
26	Check maintain user role functionality	enter valid email and select user level to create new user	successfully insert user and send email to user with user account activation link	fail (email not sent)	pass
27		save user without email and user level	mark empty required fields	pass	pass
28		save user without email or user level	mark empty required fields	pass	pass
29		search user by valid email	display user details	pass	pass
30		search user by invalid email	display invalid email message	pass	pass

.

Table 5.3. Results of the test cases in phase 1 and 2

Chapter 6 Conclusion and Further Work

The aim of the project was to develop a user friendly system which is capable of diagnosing DNS issues related to .lk domains handled by LK Domain Registry. According to various user evaluations and testing carried out at different stages in the development lifecycle, it was identified that it is more efficient and effective to have a fully functioning system with a single point of diagnosing capability that can use instead of the existing diagnose procedure. As a domain registry organization, implementing a customized system to query internal servers is very useful in many aspects. Considering the possible future changes for the system and ability to control the functionalities provided by the system makes implementation of a customize system more important.

One of the challenges that arise was the ambiguity of the system functionality that may have affected the possibility of actual implementation. So the implementation of each working module was carried out one after the other. This helped in giving a positive incentive that implementation of certain functionality is achievable. Existence of various technologies to achieve one task was another challenge in selecting the technology that best suits the product. For an example initially it was decided to use PHP built in method "dns_get_record()" to query DNS servers but face with the problem of inability to specify name server through this method.

Throughout the implementation process of the system, the project came across various difficulties. As such, inactiveness of LKDR authoritative name server in some occasions, failures in remote connection, difficulties came across in arranging the meetings with system users due to their tight schedules were some problems beyond the control. So it can be considered as a great achievement to be able to develop the system fulfilling all the project objectives even though there were these problems and challenges.

When compared with online DNS diagnose tools such as IntoDNS and MXToolbox, the implemented system still lacks functionalities such as diagnosing email delivery issues and identifying blacklist emails. MXToolbox is providing an API for implementing the functionality to identify email delivery issues for a certain domain. So integrating such API in the system can be identified as a possible further work for the system.

Finally it can be conclude that implementation of a system to diagnose DNS issues including functionalities related to diagnose process helps in efficient and effective DNS issue handling with flexibility in customizing the system for further enhancements.

References

[1] 2020. About Us. LK Domain Registry. [Online] 2020. http://www.nic.lk/.

[2] Dissanayake, Chamara. 2019. Tasks handled by technical division and issues they face. 2019.

[3] 2018. DNS Propagation Explained. namecheap. [Online] August 11, 2018. https://www.namecheap.com/support/knowledgebase/article.aspx/9622/10/dns-propagation--explained.

[4] Hosterion. intoDNS: checks DNS and mail servers health. intoDNS. [Online] Hosterion - web hosting. https://intodns.com/.

[5] 2020. MXtoolbox: MX Lookup Tool - Check your DNS MX Records online - MxToolbox. MXtoolbox. [Online] 2020. https://mxtoolbox.com/.

[6] Pramatarov, Martin. 2018. What is Authoritative DNS server? ClouDNS Blog. [Online] ClouDNS, January 12, 2018. https://www.cloudns.net/blog/authoritative-dns-server/.

[7] My Account. LK Domain Registry. [Online] LK Domain Registry. [Cited: May 1, 2020.] https://www.domains.lk/myaccountlogin.

[8] Cloudflare. DNS Server Types. Cloudflare. [Online] [Cited: May 1, 2020.] https://www.cloudflare.com/learning/dns/dns-server-types/.

[9] 2018. Introduction to Google Public DNS. Google public DNS. [Online] Google, September 25, 2018. [Cited: May 1, 2020.] https://developers.google.com/speed/public-dns/docs/intro

[10] Wijayamanna, Uththara. 2019. DNS issue diagnose process. 2019.

[11] 2019. Cloudflare. Nameservers not resolving to correct IP addresses for one domain name. [Online] June 2019. [Cited: May 1, 2020.] https://community.cloudflare.com/t/nameservers-not-resolving-to-correct-ip-addresses-for-one-domain-name/94302/1.

[12] DNS Propagation Checker. whatsmydns.net. [Online] [Cited: May 1, 2020.] https://www.whatsmydns.net/.

[13] UNDERSTANDING THE DIG COMMAND. Media Temple. [Online] Media Temple. [Cited: May 4, 2020.] https://mediatemple.net/community/products/dv/204644130/understanding-the-dig-command

[14] 2019. Advantage and disadvantage of MVC architecture. crack your interview. [Online] March 20, 2019. [Cited: May 4, 2020.] http://crackyourinterview.com/Ads-Advantage-and-disadvantage-of-MVC-architecture.aspx.

[15] Gouirhate, Noufel. 2017. Create your own MVC framework in PHP. Medium. [Online] December 18, 2017. [Cited: May 4, 2020.] https://medium.com/@noufel.gouirhate/create-your-own-mvc-framework-in-php-af7bd1f0ca19.

[16] NetBeans. Welcome to NetBeans. NetBeans. [Online] [Cited: May 6, 2020.] https://netbeans.org/.

Appendix A Design Documentation

A detailed description of each use case of the diagram depict in figure A.1 below is explained in section 3.2, “Detail Description of Functional Requirements”.

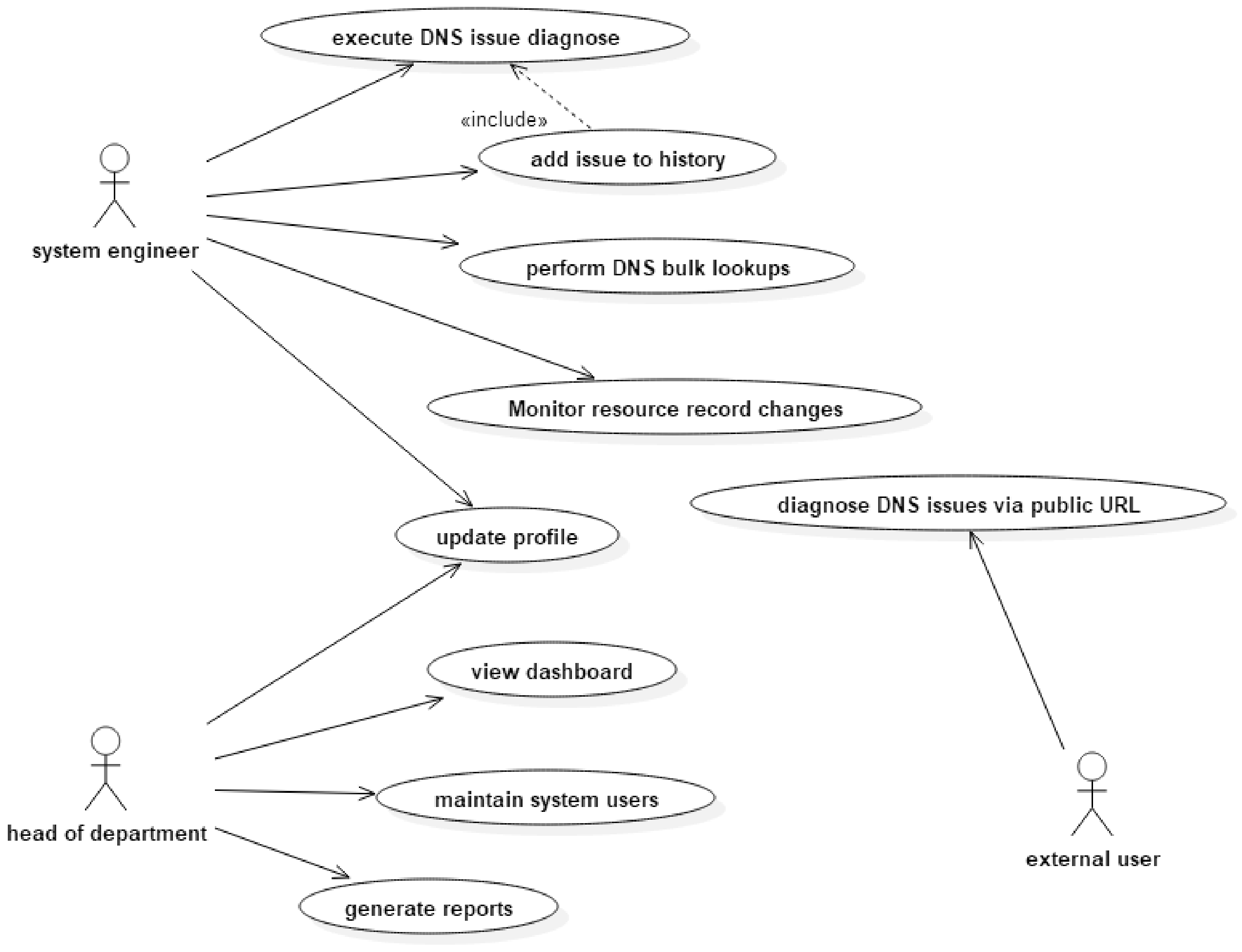
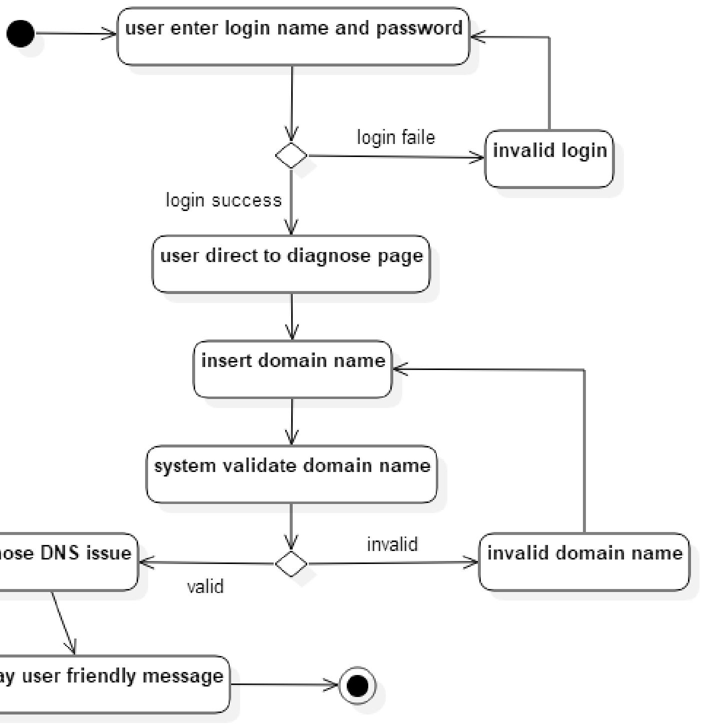


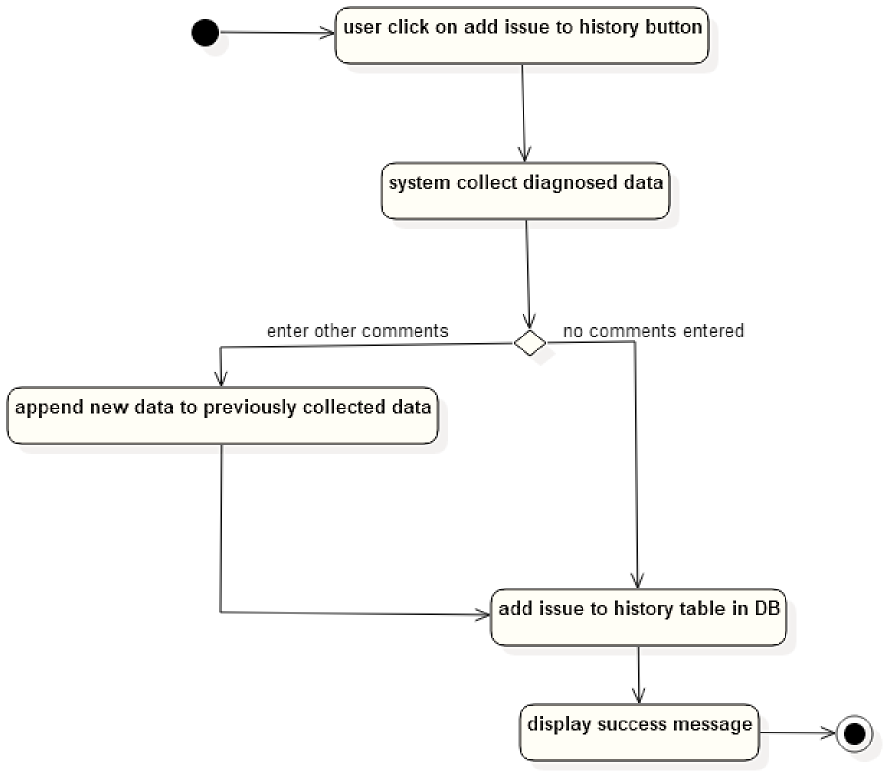
Figure A. 1. Use case diagram of DNS issue diagnose system

For the use case classes “DNS issue diagnose and maintain” and “DNS lookup” that was discussed in section 3.2, “Detail Description of Functional Requirements”, the corresponding activity diagrams can be given as shown in figure A.2 to A.6 below.



user enter login name and password
login faile
invalid login
login success
user direct to diagnose page
insert domain name
system validate domain name
invalid
invalid domain name
diagnose DNS issue
valid
display user friendly message

Figure A. 2. Activity diagram for execute DNS issue diagnose



user click on add issue to history button
system collect diagnosed data
enter other comments
no comments entered
append new data to previously collected data
add issue to history table in DB
display success message

Figure A. 3. Activity diagram for add DNS issue to history

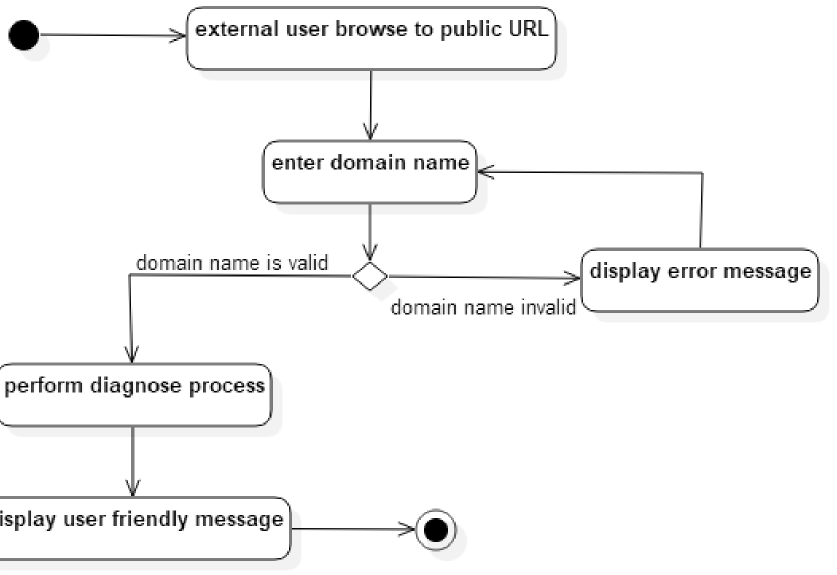
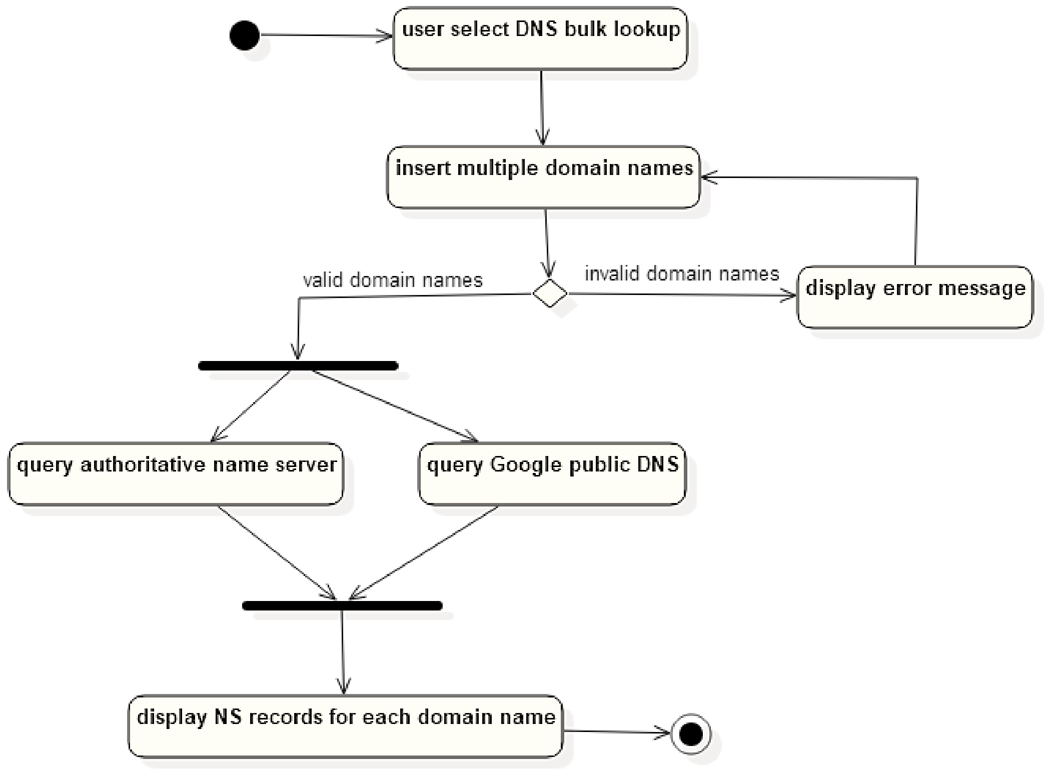
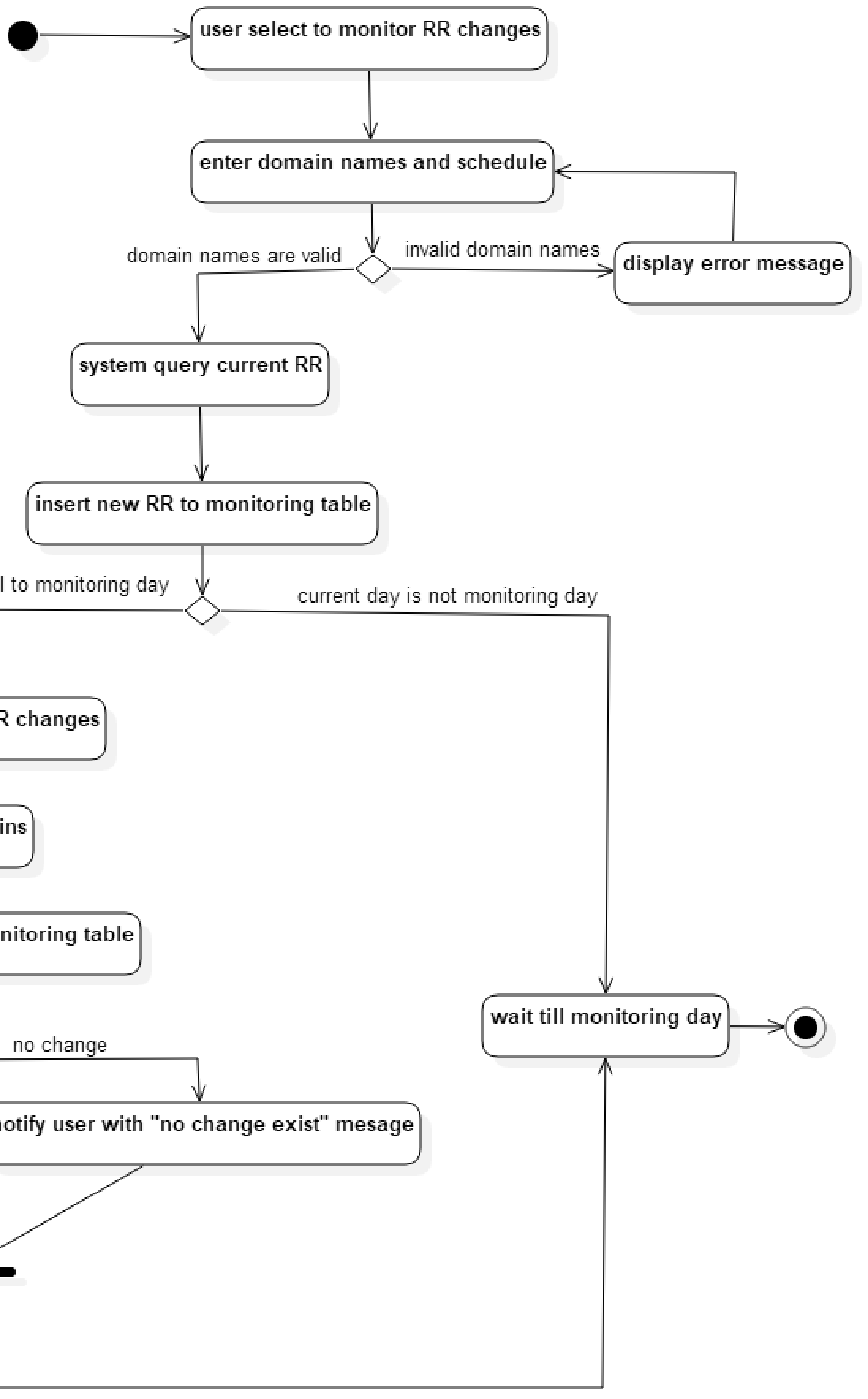


Figure A. 4. Activity diagram for diagnose DNS issue via public URL



user select DNS bulk lookup
insert multiple domain names
valid domain names
invalid domain names
display error message
query authoritative name server
query Google public DNS
display NS records for each domain name

Figure A. 5. Activity diagram for perform DNS bulk lookups



user select to monitor RR changes
enter domain names and schedule
domain names are valid
invalid domain names
display error message
system query current RR
insert new RR to monitoring table
current day is equal to monitoring day
current day is not monitoring day
system trigger monitoring RR changes
get new RR of domains
compare new RR with RR in monitoring table
change exist
no change
send alert to user
notify user with "no change exist" mesage
wait till monitoring day

Figure A. 6. Activity diagram for monitor resource record changes

The use case classes related to user profile maintain is given in next sections along with their respective activity diagram.

Use case related to profile maintain

Use case 1: insert profile details

Primary actor: head of department, system engineer

Pre condition: user log into the system, 'insert new user role' use case must be executed before this use case execution

Main scenario:

1. User initiate insert profile details functionality
2. System ask for employee name, division, job title, career start date, contact number, current address, date of birth
3. User insert required details
4. System validate data
5. System perform insert profile details functionality
6. System display success message

Alternate scenario:

1. User does not insert required details
    a. System ask to re-enter required fields
    b. User enter required details
    c. System perform insert profile details functionality
2. Invalid data provided
    a. System mark invalid data fields
    b. Display error message

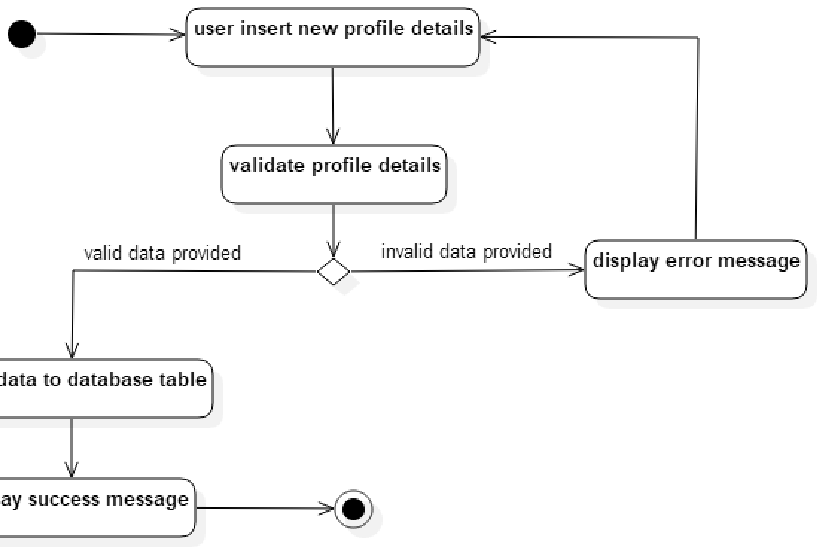


Figure A. 7. Activity diagram for insert profile details

Use case 2: update profile details

Primary actor: head of department, system engineer

Pre condition: user log into the system, 'insert new user role' and 'insert profile details' use cases must be executed before this use case execution

Main scenario:

1. User initiate update profile details functionality
2. System display existing profile details
3. User change existing details
4. User confirm the changes
5. System validate data
6. System perform update profile functionality
7. Display success message

Alternate scenario:

1. User cancel the changes
    a. System terminate profile change functionality
2. Invalid data provided
    a. System mark invalid data fields
    b. Display error message

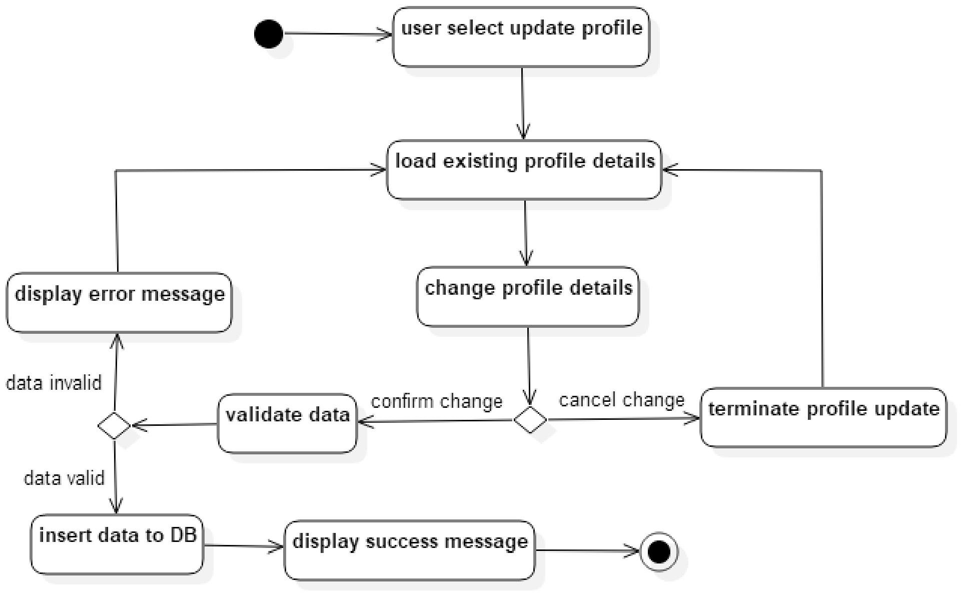


Figure A. 8. Activity diagram for update profile details

Use case 3: delete profile details

Primary actor: head of department, system engineer

Pre condition: user log into the system, 'insert new user role' and 'insert profile details' use cases must be executed before this use case execution

Main scenario:

1. System display existing profile details
2. User removes details
3. User confirm delete
4. System validate data removal
5. System perform delete profile details functionality
6. Display success message

Alternate scenario:

1. User try to empty required data
    a. System marks required data fields
    b. Display error message
2. Invalid data provided
    a. System marks invalid data fields
    b. Display error message

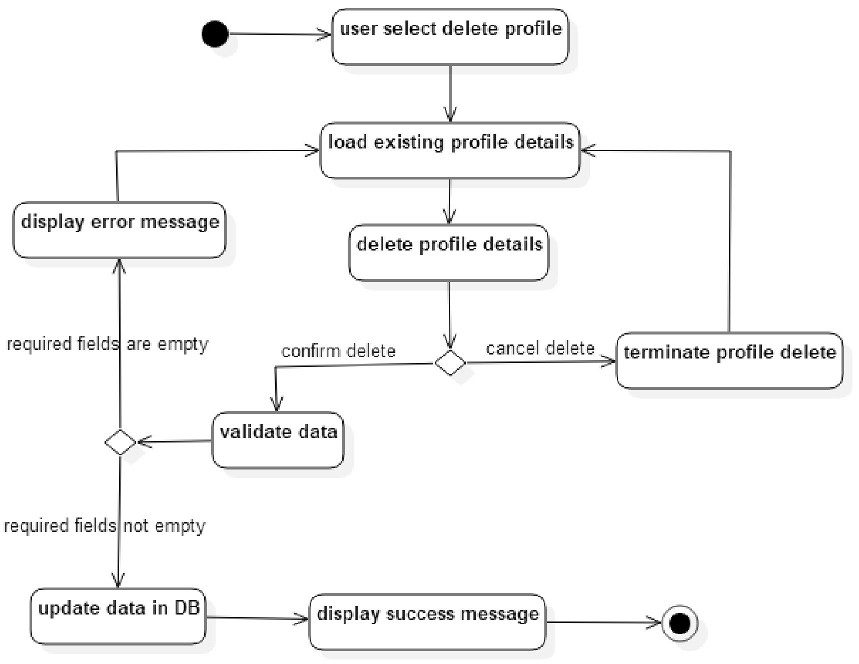


Figure A. 9. Activity diagram for delete profile details

Use case related to dashboard display

Use case 1: view dashboard

Primary actor: head of department

Pre condition: user log into the system

Main scenario:

1. User initiate view dashboard functionality
2. System query the database to collect data with related to diagnose history
3. Data available in history tables
4. System perform graph draw functionality
5. System display a summary of diagnose details with graphs and charts

Alternate scenario:

1. No data available to display in diagnosing history tables
    a. System prompt an information that no data available
    b. Terminate graph draw functionality

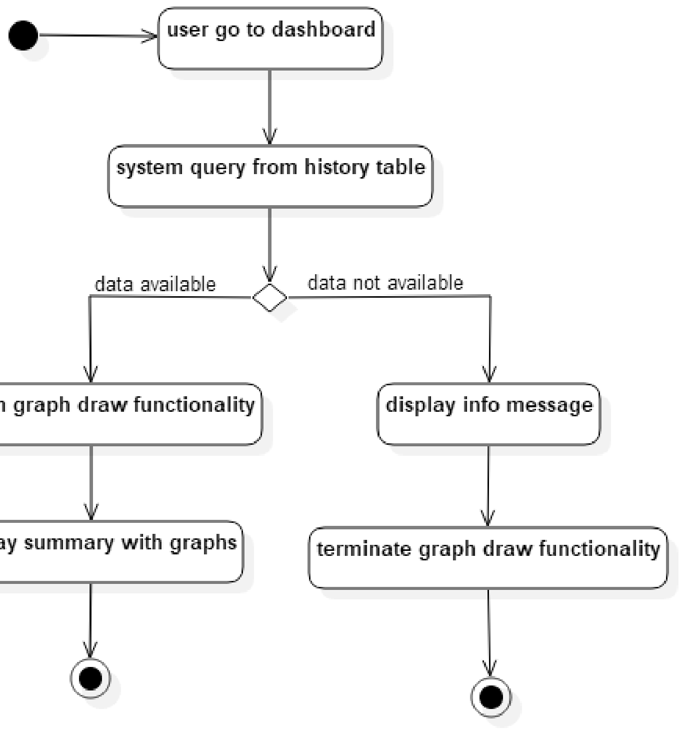

user go to dashboard
system query from history table
data available
data not available
perform graph draw functionality
display info message
display summary with graphs
terminate graph draw functionality

Figure A. 10. Activity diagram for view dashboard

Use case related to maintain system users

Use case 1: insert new user role

Primary actor: head of department

Pre condition: user log into the system

Main scenario:

1. User create new second level user giving his/her email
2. System validate email
3. System perform create user role functionality by auto generating password and inserting new user details to user table along with user type as 2
4. Display success message

Alternate scenario:

1. Invalid email provided
    a. System ask user to re-enter email
    b. User re-enter email
    c. System starts inserting new user role functionality
2. Email already exist
    a. Display error message
    b. Re-enter data

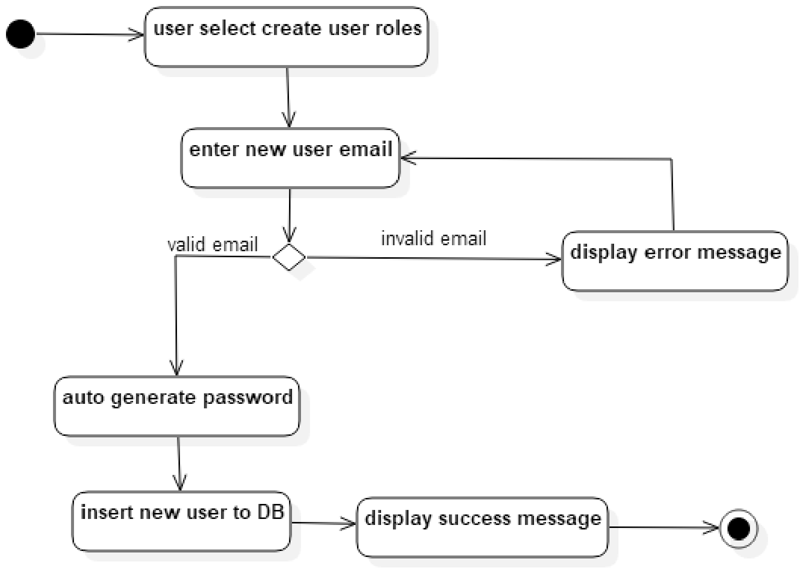


Figure A. 11. Activity diagram for insert new user role

Use case 2: update user role

Primary actor: head of department

Pre condition: user log into the system, 'insert new user role' use case must be executed before this use case execution

Main scenario:

1. User search employee by email
2. System validate email
3. System perform search employee functionality
4. Search successful, display employee details
5. User change the user level of employee
6. System perform update user role functionality
7. Display success message

Alternate scenario:

1. Invalid email provided
    a. Display error message
    b. Re-enter email and search
2. No employee exist under given email
    a. System inform user with a message

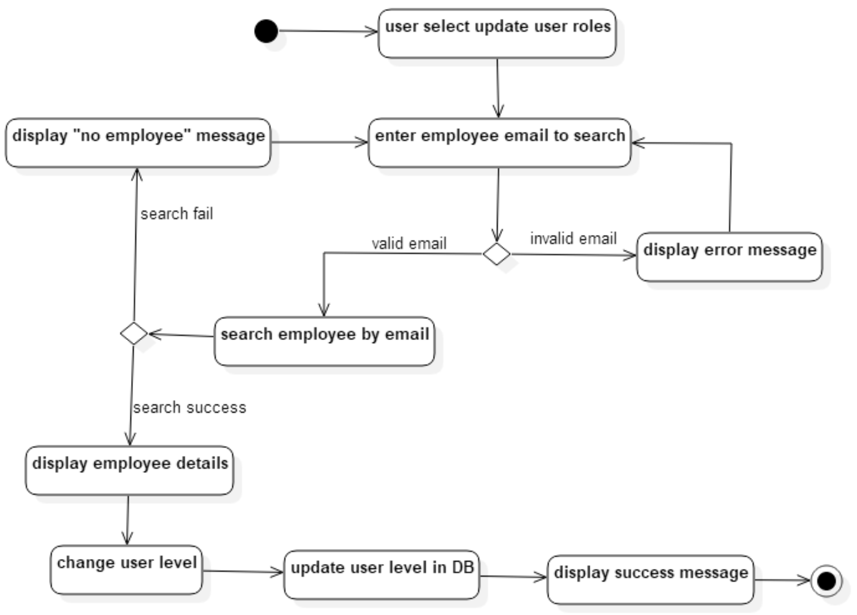


Figure A. 12. Activity diagram for update user role

Use case 3: deactivate user role

Primary actor: department head

Pre condition: user log into the system, ‘insert new user role’ use case must be executed before this use case execution

Main scenario:

1. User search employee by email
2. System validate email
3. System perform search employee functionality
4. Search successful, display employee details
5. User deactivate the employee
6. System ask for deactivation confirm
7. User confirm deactivation
8. System perform deactivation functionality
9. Display success message

Alternate scenario:

1. Invalid email provided
    a. Display error message
    b. Re-enter email and search
2. No employee exist under given email
    a. System inform user with a message
3. User does not confirm deactivation
    a. System terminate deactivation
    b. Display message

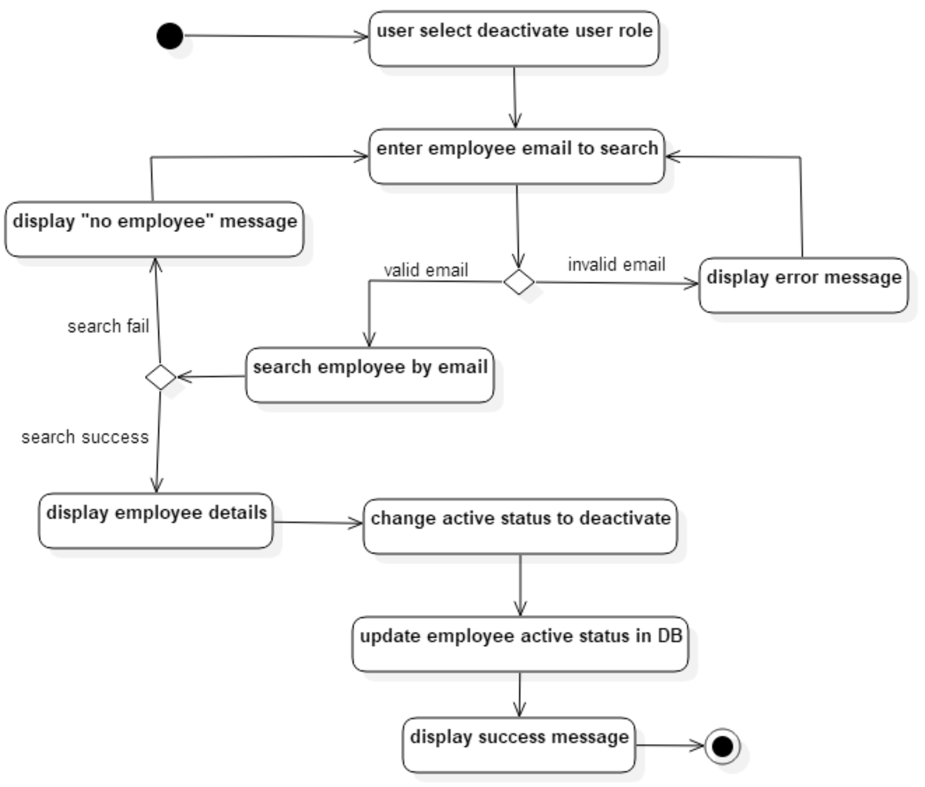

user select deactivate user role
enter employee email to search
display "no employee" message
valid email
invalid email
display error message
search fail
search employee by email
search success
display employee details
change active status to deactivate
update employee active status in DB
display success message

Figure A. 13. Activity diagram for deactivate user role

Use case related to generate reports

Use case 1: generate report

Primary actor: head of department

Pre condition: log into the system

Main scenario:

1. User select report type to generate
2. System perform report generation functionality
3. Display report
4. User initiate print report
5. System perform print report functionality

Alternate scenario:

1. User does not initiate print report functionality
    a. User close report view

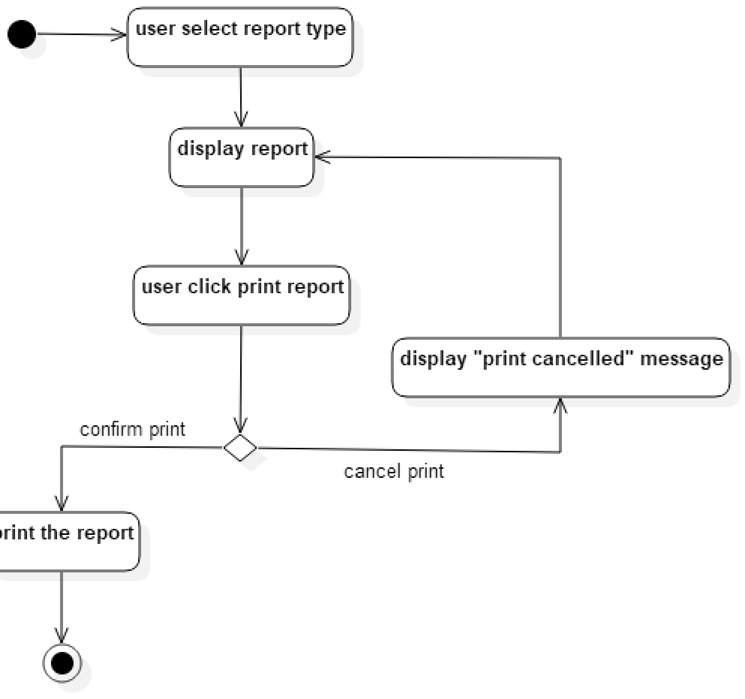


Figure A. 14. Activity diagram for generate report

The scenarios “DNS issue diagnose”, “DNS bulk lookup” and “monitor resource record changes” are given in sequence diagrams below from figure A.15 to A.17.

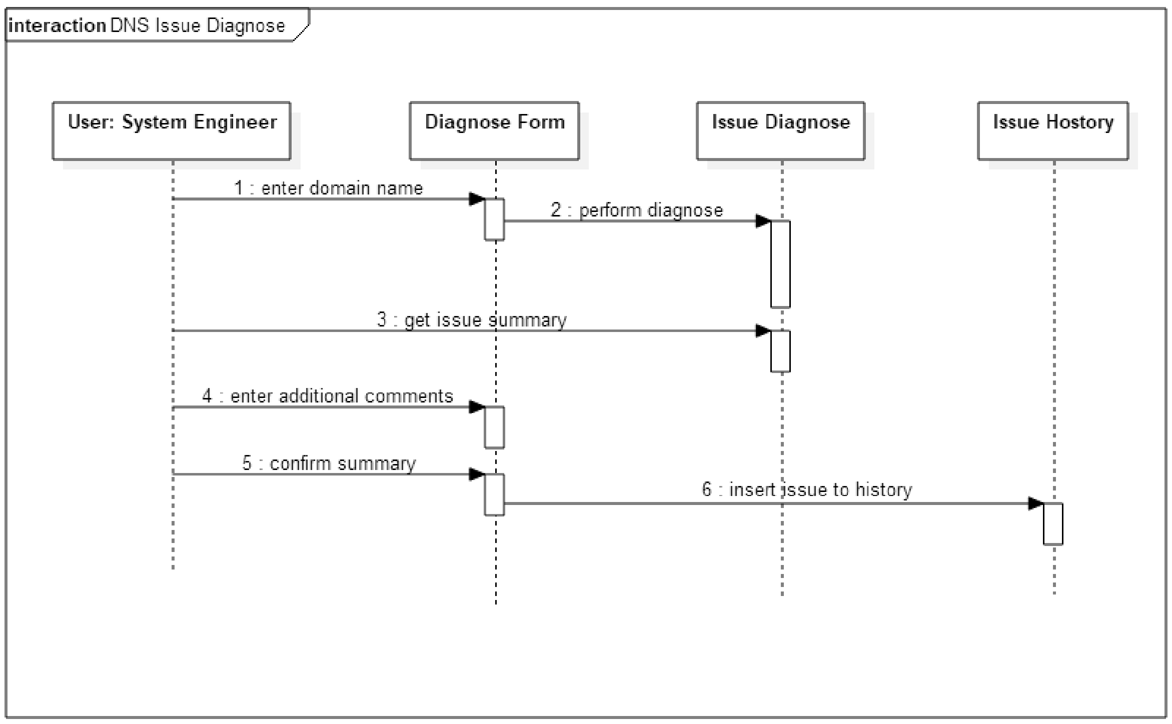


Figure A. 15. Sequence diagram for DNS issue diagnose

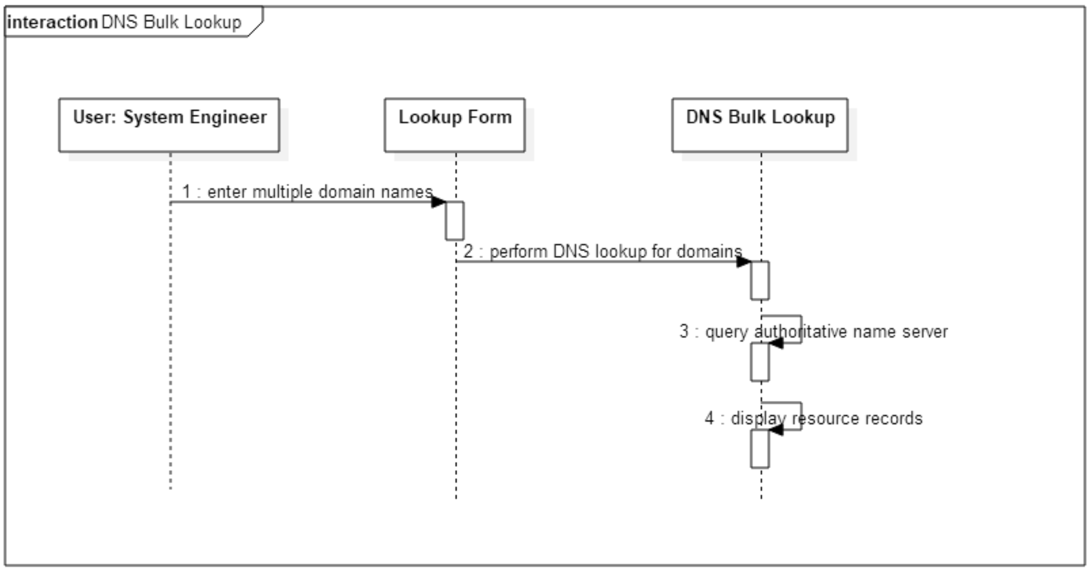

interaction DNS Bulk Lookup
User: System Engineer
Lookup Form
DNS Bulk Lookup
1 : enter multiple domain names
2 : perform DNS lookup for domains
3 : query authoritative name server
4 : display resource records

Figure A. 16. Sequence diagram for DNS bulk lookup

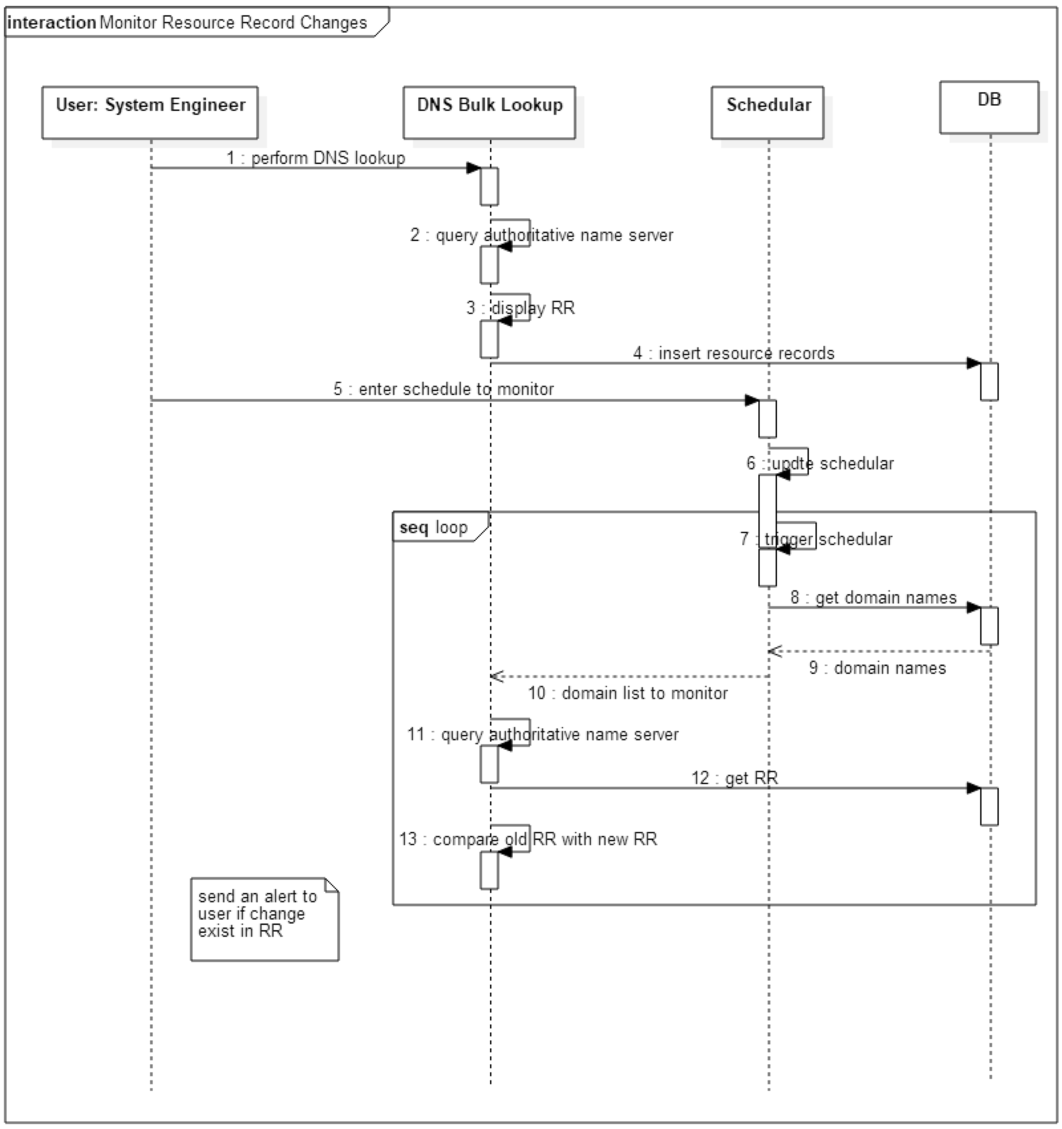
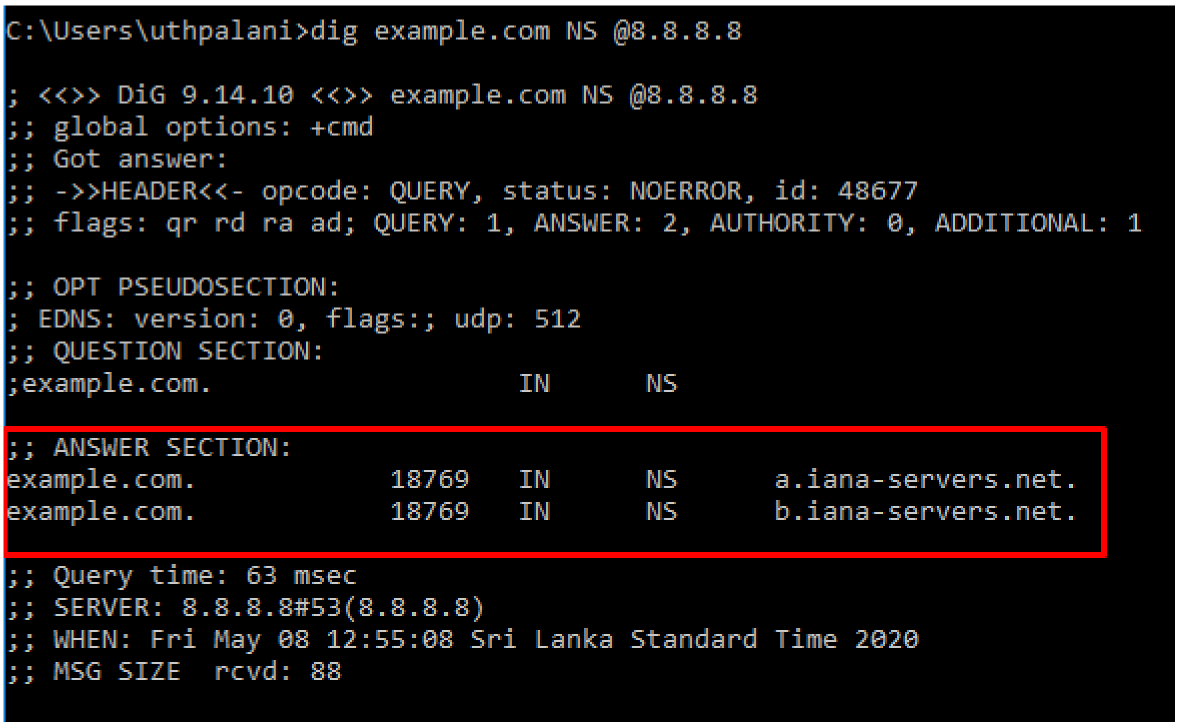

interaction Monitor Resource Record Changes
User: System Engineer
DNS Bulk Lookup
Schedular
DB
1 : perform DNS lookup
2 : query authoritative name server
3 : display RR
4 : insert resource records
5 : enter schedule to monitor
6 : updte schedular
seq loop
7 : trigger schedular
8 : get domain names
9 : domain names
10 : domain list to monitor
11 : query authoritative name server
12 : get RR
13 : compare old RR with new RR
send an alert to user if change exist in RR

Figure A. 17. Sequence diagram for monitor resource record changes

Appendix B Figures Related to Implementation

As discussed in section 3.4, “Pseudo code for DNS issue diagnose functionality” the function *queryGoogleNS()* try to extract ‘NS’ records from a given ‘dig’ command output like the output shown in figure B.1 below.

The area mark in red color of figure B.1 shows the answer section returned by Google public DNS (8.8.8.8) which is the same section that the function *queryGoogleNS()* try to extract with the use of various PHP string functions.

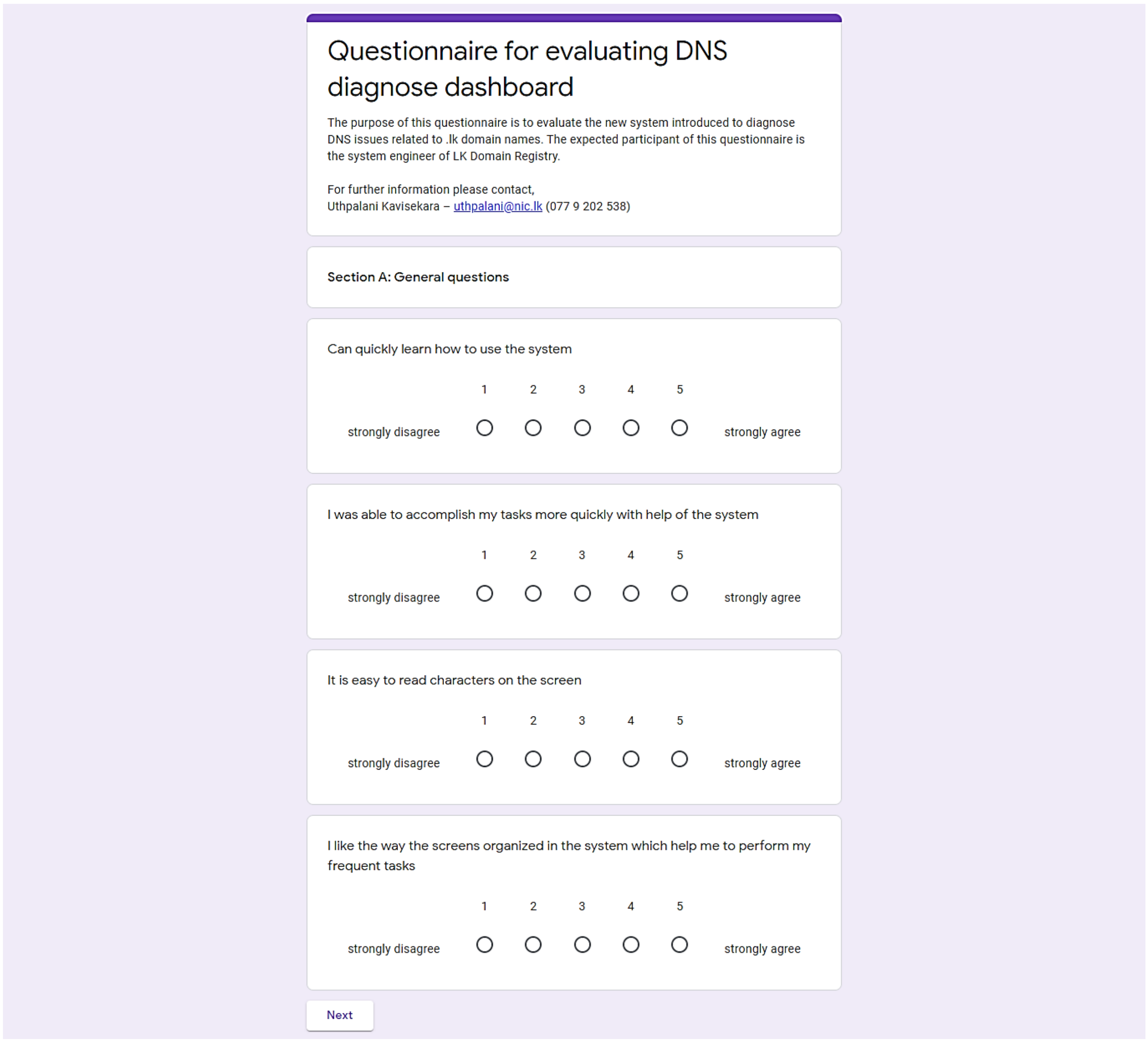


```
C:\Users\uthpalani>dig example.com NS @8.8.8.8

; <<>> DiG 9.14.10 <<>> example.com NS @8.8.8.8
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 48677
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;example.com.                   IN      NS

;; ANSWER SECTION:
example.com.            18769   IN      NS      a.iana-servers.net.
example.com.            18769   IN      NS      b.iana-servers.net.

;; Query time: 63 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Fri May 08 12:55:08 Sri Lanka Standard Time 2020
;; MSG SIZE  rcvd: 88
```

Figure B. 1. 'dig' command that output NS records of example.com domain

Appendix C Evaluation Documents

First part of the questionnaire that was designed to be completed by the system engineer is given in figure C.1 below.



Questionnaire for evaluating DNS diagnose dashboard

The purpose of this questionnaire is to evaluate the new system introduced to diagnose DNS issues related to .lk domain names. The expected participant of this questionnaire is the system engineer of LK Domain Registry.

For further information please contact,
Uthpalani Kavisekara – uthpalani@nic.lk (077 9 202 538)

Section A: General questions

Can quickly learn how to use the system

	1	2	3	4	5	
strongly disagree	○	○	○	○	○	strongly agree

I was able to accomplish my tasks more quickly with help of the system

	1	2	3	4	5	
strongly disagree	○	○	○	○	○	strongly agree

It is easy to read characters on the screen

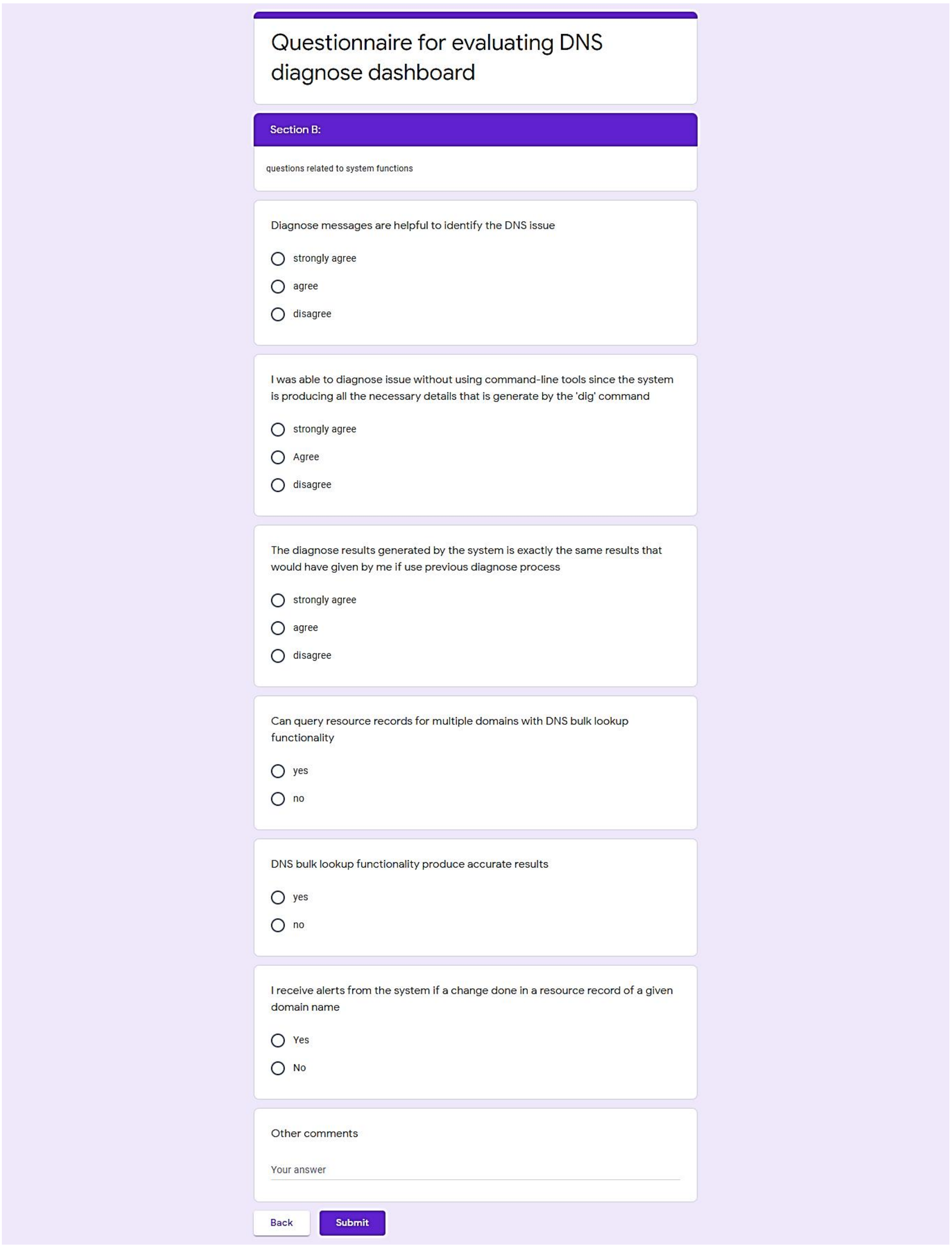
	1	2	3	4	5	
strongly disagree	○	○	○	○	○	strongly agree

I like the way the screens organized in the system which help me to perform my frequent tasks

	1	2	3	4	5	
strongly disagree	○	○	○	○	○	strongly agree

Figure C. 1. Questionnaire given to system engineer for evaluation (part A)

Second part of the questionnaire is given in figure C.2 which focuses on measuring the satisfactory level of the functionalities that provided for the system engineer.


Questionnaire for evaluating DNS diagnose dashboard

Section B:

questions related to system functions

Diagnose messages are helpful to identify the DNS issue

- strongly agree
- agree
- disagree

I was able to diagnose issue without using command-line tools since the system is producing all the necessary details that is generate by the 'dig' command

- strongly agree
- Agree
- disagree

The diagnose results generated by the system is exactly the same results that would have given by me if use previous diagnose process

- strongly agree
- agree
- disagree

Can query resource records for multiple domains with DNS bulk lookup functionality

- yes
- no

DNS bulk lookup functionality produce accurate results

- yes
- no

I receive alerts from the system if a change done in a resource record of a given domain name

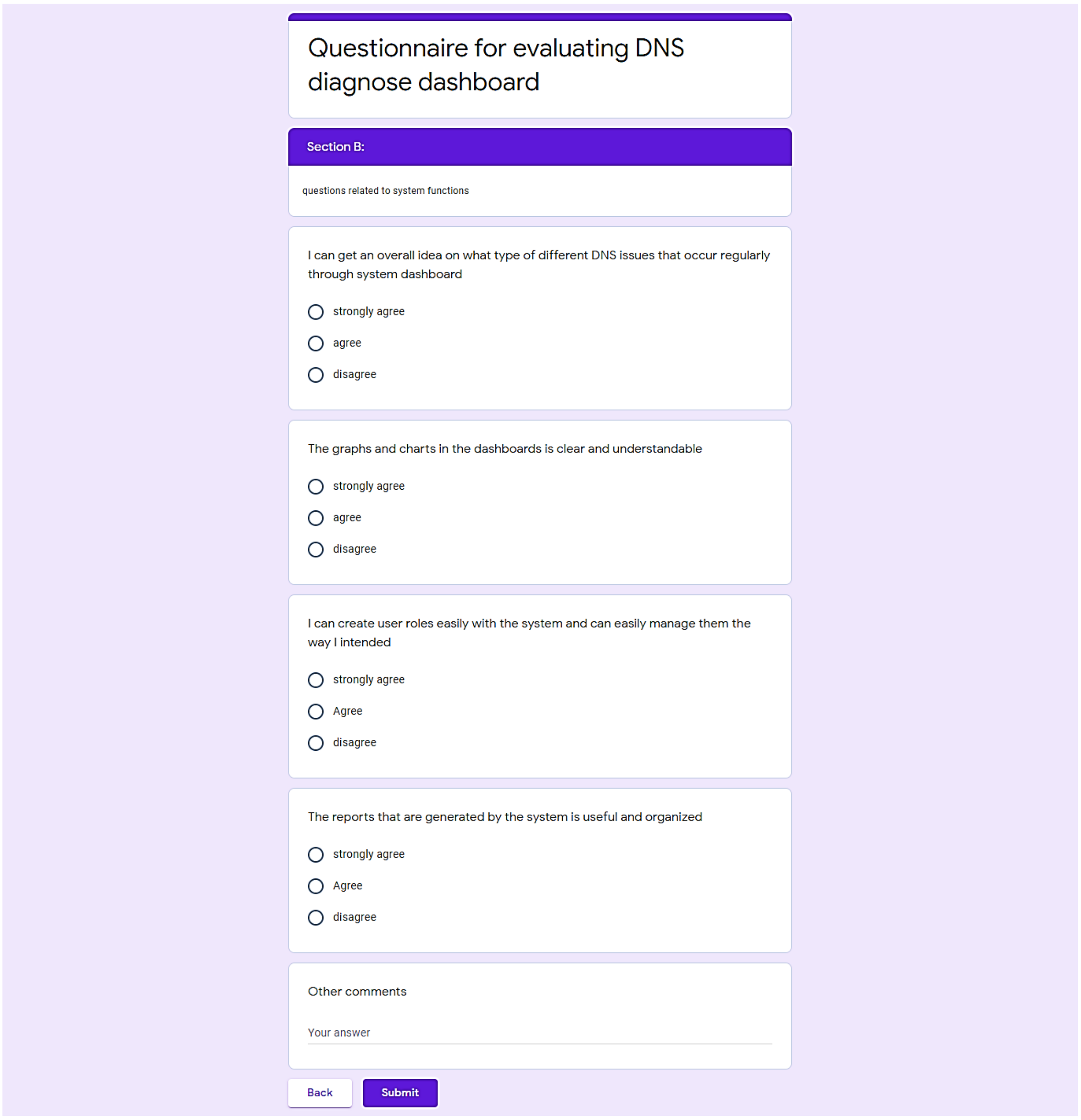
- Yes
- No

Other comments

Your answer

Figure C. 2. Questionnaire given to system engineer for evaluation (part B)

Second part of the questionnaire that was designed to be completed by the head of technical division of LKDR is given in figure C.3 below. It focuses on measuring the satisfactory level of the functionalities that provided for the head of technical division.



Questionnaire for evaluating DNS diagnose dashboard

Section B:

questions related to system functions

I can get an overall idea on what type of different DNS issues that occur regularly through system dashboard

- strongly agree
- agree
- disagree

The graphs and charts in the dashboards is clear and understandable

- strongly agree
- agree
- disagree

I can create user roles easily with the system and can easily manage them the way I intended

- strongly agree
- Agree
- disagree

The reports that are generated by the system is useful and organized

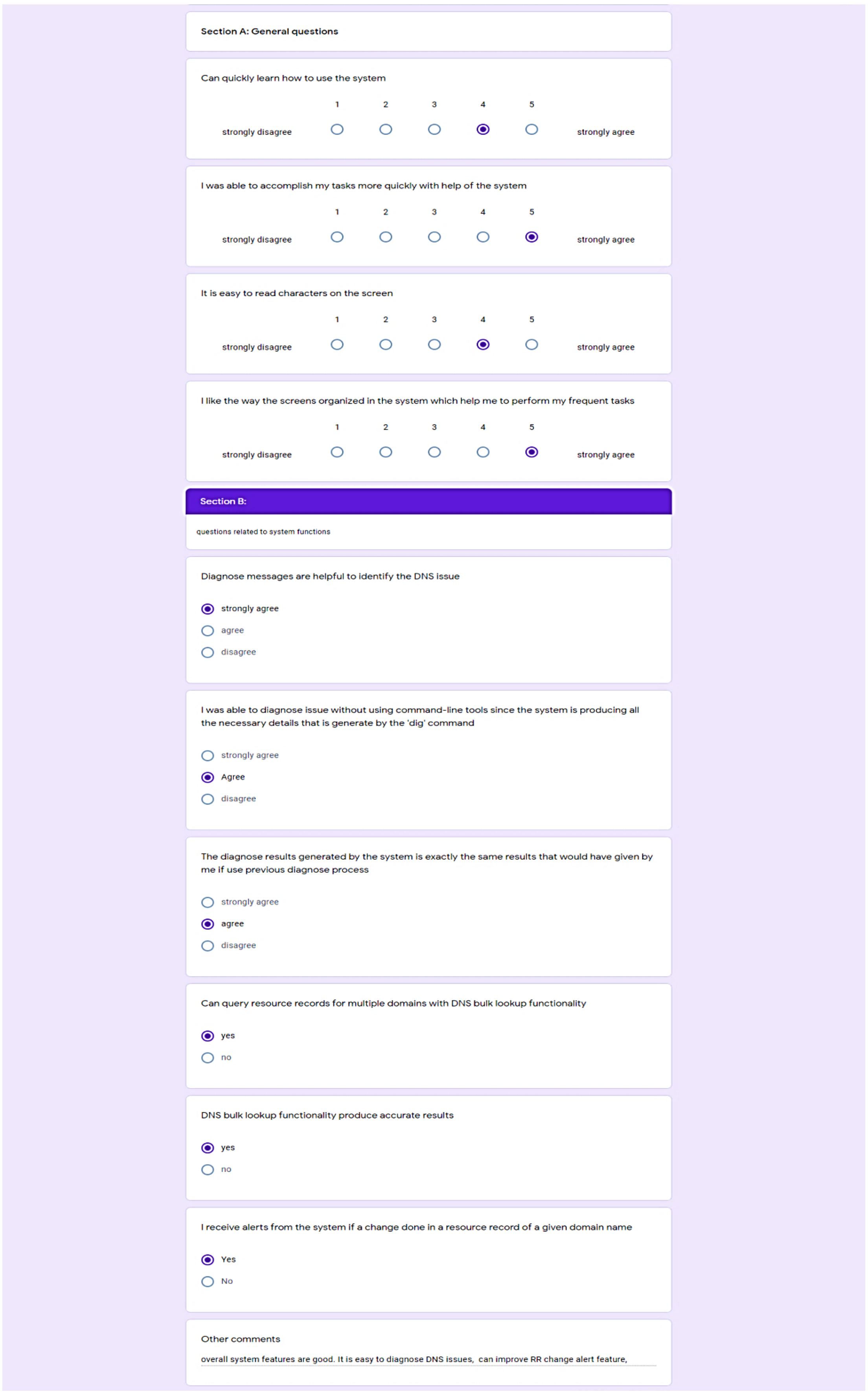
- strongly agree
- Agree
- disagree

Other comments

Your answer

Figure C. 3. Questionnaire given to head of technical division for evaluation (part B)

The responses given by the system engineer and head of technical division of LKDR is given below respectively.



Section A: General questions

Can quickly learn how to use the system

strongly disagree 1 2 3 [✓ 4] 5 strongly agree

I was able to accomplish my tasks more quickly with help of the system

strongly disagree 1 2 3 4 [✓ 5] strongly agree

It is easy to read characters on the screen

strongly disagree 1 2 3 [✓ 4] 5 strongly agree

I like the way the screens organized in the system which help me to perform my frequent tasks

strongly disagree 1 2 3 4 [✓ 5] strongly agree

Section B:

questions related to system functions

Diagnose messages are helpful to identify the DNS issue

[✓ strongly agree] agree disagree

I was able to diagnose issue without using command-line tools since the system is producing all the necessary details that is generate by the 'dig' command

strongly agree [✓ Agree] disagree

The diagnose results generated by the system is exactly the same results that would have given by me if use previous diagnose process

strongly agree [✓ agree] disagree

Can query resource records for multiple domains with DNS bulk lookup functionality

[✓ yes] no

DNS bulk lookup functionality produce accurate results

[✓ yes] no

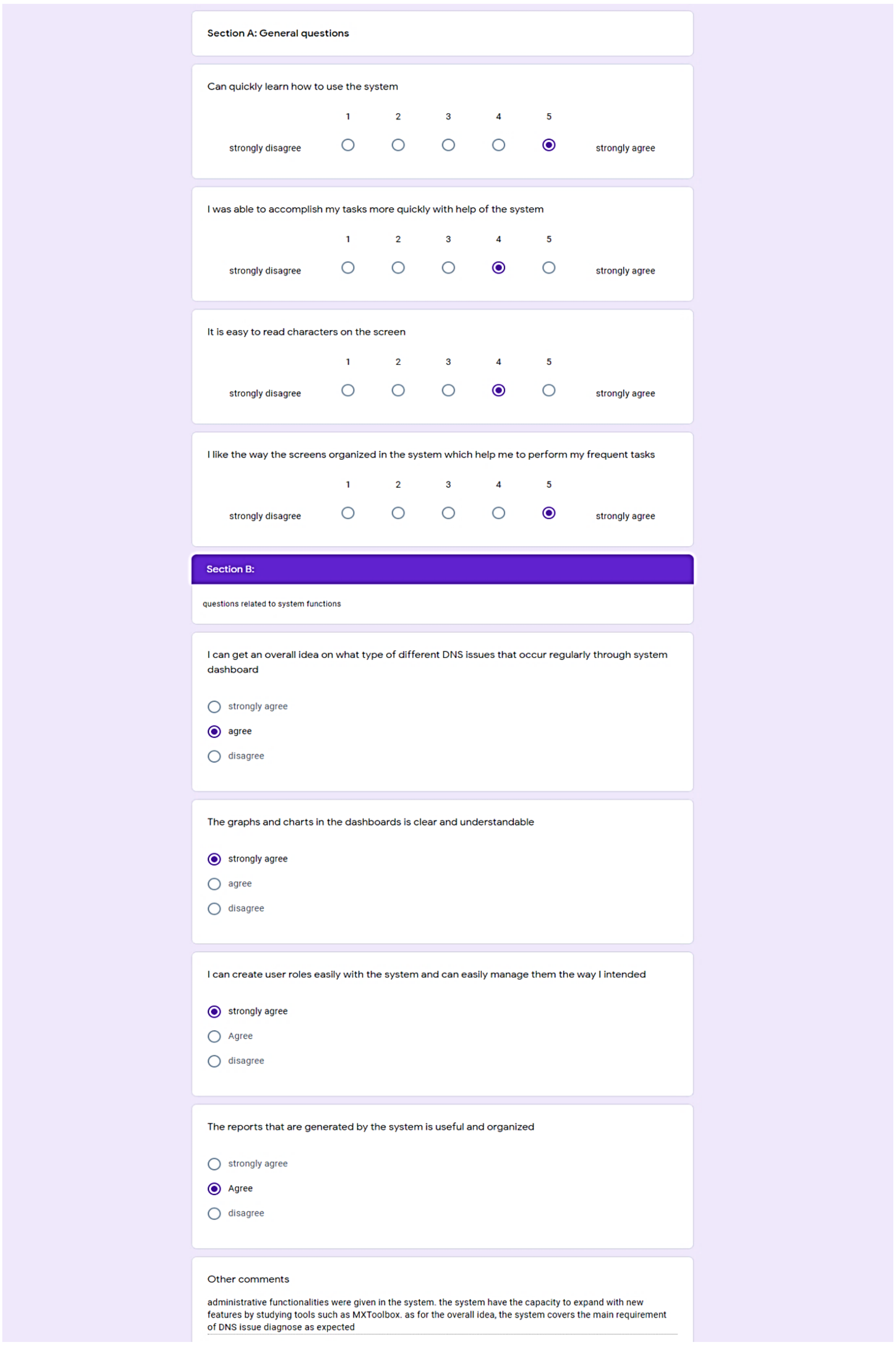
I receive alerts from the system if a change done in a resource record of a given domain name

[✓ Yes] No

Other comments

overall system features are good. It is easy to diagnose DNS issues, can improve RR change alert feature,

Figure C. 4. Answered questionnaire of the system engineer



Section A: General questions

Can quickly learn how to use the system

strongly disagree 1 2 3 4 [✓ 5] strongly agree

I was able to accomplish my tasks more quickly with help of the system

strongly disagree 1 2 3 [✓ 4] 5 strongly agree

It is easy to read characters on the screen

strongly disagree 1 2 3 [✓ 4] 5 strongly agree

I like the way the screens organized in the system which help me to perform my frequent tasks

strongly disagree 1 2 3 4 [✓ 5] strongly agree

Section B:

questions related to system functions

I can get an overall idea on what type of different DNS issues that occur regularly through system dashboard

- strongly agree
- [✓ agree]
- disagree

The graphs and charts in the dashboards is clear and understandable

- [✓ strongly agree]
- agree
- disagree

I can create user roles easily with the system and can easily manage them the way I intended

- [✓ strongly agree]
- Agree
- disagree

The reports that are generated by the system is useful and organized

- strongly agree
- [✓ Agree]
- disagree

Other comments

administrative functionalities were given in the system. the system have the capacity to expand with new features by studying tools such as MXToolbox. as for the overall idea, the system covers the main requirement of DNS issue diagnose as expected

Figure C. 5. Answered questionnaire of the head of technical division

Index