Community-based management of hypertension in Nepal (COBMIN)

by

Arjun Khanal

B.E, Kantipur Engineering College, Tribhuvan University, 2008

A REPORT

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Computer Science
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2017

Approved by:

Major Professor
Dr. Daniel Andresen

Copyright

Abstract

In COBMIN project, we explore two popular software architectural pattern to implement WHO (World Health Organization) STEPS Instrument for Non-Communication Diseases Risk Factor Surveillance for Nepal. COBMIN web application implements Model-View-Controller(MVC) pattern, which divides the application into three interconnected parts - model, view, and controller. Similarly, COBMIN mobile application utilizes one of the most popular Model-View-ViewModel (MVVM) software architectural pattern which isolates the development of graphical user interface from the business logic and data models. We implement above-mentioned pattern using new .Net Core technologies and a cross-platform mobile development API Xamarin.Forms.

The COBMIN project comprises of a web application, a web API, and a mobile application to facilitate community-based management of hypertension in Nepal. The primary purpose of the web application is to manage demographic, behavioral, and physical information of the members of the community who are part of the study group. The mobile application is used to expedite the process of data acquisition from the members. The Web API defines request-response message system for mobile client application employing the central data storage.

Table of Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank members of the NEDS organization for giving this opportunity to work on their project. Special thanks to Dinesh Neupane, Ph.D. student at Aarhus University, Denmark, who helped me throughout the project for requirement gathering and analysis.

I would like to thank my major professor Dr. Daniel Andresen for graciously accepting my project proposal and supporting, guiding and overseeing me throughout the project. I am grateful to Dr. Mitchell Neilsen and my advisor Dr. Torben Amtoft for accepting my request to be in advisory committee. Also, I am very thankful to Judy Dizon for guiding me and providing extra time to work on my project.

I would also like to thank my beloved wife Tulsi Paudel Khanal and adorable son Shreeyam Khanal for caring and supporting me throughout my study period.

Chapter 1 - Introduction

1.1 Project Description

There are many non-profit organization working in public health projects in the high hilly regions of the Nepal. The people who live in remote areas of Nepal have little access to health care because of the geographic location. One of the organization working in public heath area is Nepal Development Society (NEDS) http://www.nedsnepal.org/. NEDS is currently running the public health project called “community-based management of hypertension (COBMIN) based on WHO (World Health Organization) Steps Instrument survey for non-communicable diseases. The purposed application is designed to provide a solution for organization (NEDS) to manage their projects. This application addresses the few common problems faced by NEDS. First, it facilitates to keep records of all the members participating in the project eliminating the paper-records of those members. Second, it helps to manage multiple survey projects eliminating the need for separate application. Third, it expedites the survey process by eliminating the need to fill up the paper form. Fourth, it provides centralized database storage to analyze data.

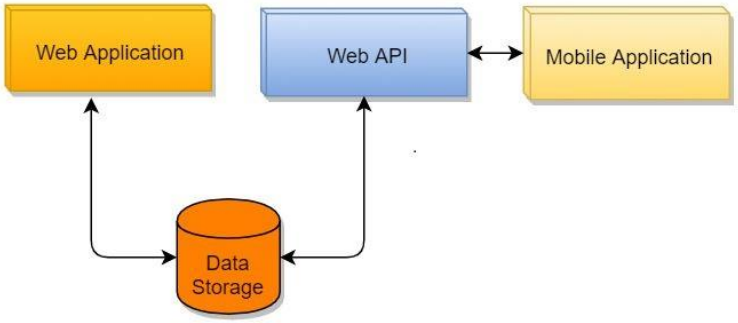


Figure 1-1: Components of COBMIN Project

There above figures illustrates the components of the COBMIN projects. There are three components involved in the project, web application, web API, and Mobile Application. The web application serves as setup and reporting tool. This interface can be used to setup project details and related institution, manage user and participants account, defined roles for the user, and generate administrative reports and so on. On the other hand, Web API defines the services and implements those services to facilitate the client application, for example, mobile application to consume the defined resources in the web API. Both web application and web API utilize the same central database storage. The third component is client application, android or iOS application. The application is primarily used to facilitate the user to collect the responses to the questions from the participants. Mobile application completely depends on web API component for the data. The mobile application works based on request-response with web API.

1.2 Motivation

NEDS works in diverse fields such as public health, environment, information technologies, agriculture, education, veterinary and social sciences. NEDS has a team of top experts from several fields with several years of professional experience. Currently NEDS has ongoing projects on farmers health and pesticide management in Chitwan and community based management of hypertension in Kaski district of Nepal. I recently joined this (NEDS) organization and though of helping them with their ongoing COBMIN projects by designing and developing the application which can keep track and manage the COBMIN project. So, I decided to develop the application using new technologies so that I will have opportunity to learn new technologies and contribute to the organization.

Chapter 2 - Related Work

2.1 Web Application Development

There exist numerous methodologies and development process for designing and implementing a web application. However, selecting a particular method greatly depends on the requirements. Also, user friendliness and the performance of the application are other two factors that influence the decision of choosing an appropriate development method.

Regarding web application, it is relatively easy with ASP.NET MVC framework since it is lightweight and integrated with existing ASP.NET features. Moreover, MVC design pattern segregates an application into three main components: the model, the view, and the controller.

I have selected ASP.NET MVC framework for this project for the following reasons. First, it separates the application task in three layers - input logic, business logic, UI logic. It allows the use of existing ASP.NET features such as forms authentication, Windows authentication, and membership-based authentication. Views are the components which display the application's user interface. Typically, it uses the model data and renders it to the user interface. Controllers handle the user interaction with the application; it processes the user request and selects a view to display to the user in the UI. It uses model data for validation and input processing. The model implements the logic for the application data domain.

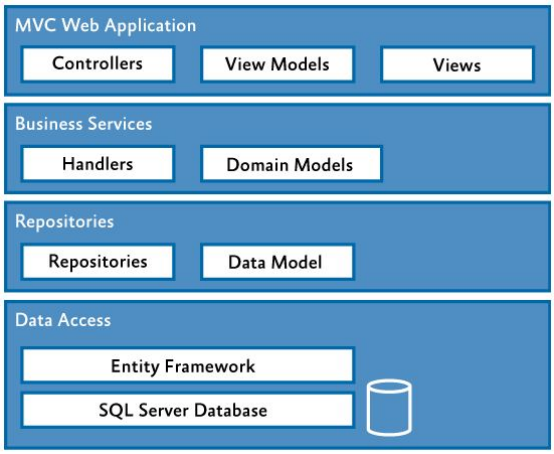


Figure 2-1: MVC framework application architecture with entity framework

COBMIN project has enough information stored in the database and requires an efficient way to handle data binding between the application and data storage. ADO.NET entity framework provides three ways to create data model: code-first, database-first, and model-first approach. In code-first, classes are used by entity framework to generate the database schema. In database-first, the entity framework creates the data model classes from the existing database. In model-first approach, entity data model (.edmx) can be used to generate the code and the database.

2.2 Mobile Application Development

Xamarin.Forms is a API library to build native apps for iOS, Android, and Windows using a single shared C# codebase. It has a great variety of cross-platform controls and layouts which are mapped to platform-specific native user interface at runtime. It allows to share code across different platform [9]. There are several mobile application development frameworks for both android and iOS application development. But, I preferred Xamarin because it provides an abstraction of the native API library and support code sharing and cross-platform controls.

2.3 Web API Development

Application Programming Interface(API) generally refers to the communication between computer systems over the network. The resources are hosted in the server and clients utilize the resources by request-response message system. In the server-side, programming interface is used facilitate the request-response system which is typically known as web services. The most commonly used design model for web service are SOAP and REST.

Simple Object Access Protocol (SOAP) defines the messaging framework based on Extensible Markup Language (XML). The main purpose of XML is store and transport data. It defines the request and response format for every operation defined in the programming interface based on XML. The request and response message contain all the parameters and its data types. The details of the input and output parameters are documented in WSDL (Web Service Description Language). The WSDL serves as a contract between the clients - the consumer of the web service- and provider.

Representation state transfer(REST) is an architectural style for developing RESTful web services which provides interoperability between computer Systems on the Internet. Rest utilizes existing HTTP protocol for communicating between the systems. In Restful web services, a request made to the resources is responded back in different format (XML, HTML, JSON). The operations available through Rest web services include HTTP get, post, put, and so on.

I decided to use REST paradigm for COBMIN web service project for the following reasons. First, it eliminates the overhead for the client app to modify and recompile the code every time the WSDL contract definition changes. Second, REST is data driven and supports many machine-humane readable format, for example, JSON (JavaScript Object Notation). Third, REST is stateless. This eliminates the overhead to manage a session between the client and the

server. Once the user is authenticated, every time a user makes request to the server, it sends the token in the HTTP header which eliminates the need to maintain a session between the server and the client.

Chapter 3 - System Requirement Analysis

3.1 Intended Users

There are basically two categories of the users, administrator and users. The administrators are responsible for system setup – defining project and related questions - and users perform various operations on the system, for example, surveying participants.

The trained personnel from NEDS organization is responsible for surveying the participant. The users of the system are expected to be familiar with the system user interface and various operations. The use of this application is not intended by participants. Participant are subjects who take part in the projects indirectly.

3.2 Requirement

The primary requirement of the application is to manage the survey questions and conduct the study with the participant. The user interface developed will provide an easy and efficient way to store the participant information, define survey project and its related questions, and capture the responses to question in the database.

3.2.1 Web Application Requirements

The web application is expected to meet following functional requirements.

1. Application should have interface to create two types of users – administrator and users.
2. Application should have role and privilege management interface.
3. Administrator should be able to define new institution, update the institutions information as necessary.
4. Administrator should be able to create multiple survey projects under an institution.
5. Application should have interface to define the various sections, description and purpose of the section for a project.
6. Administrator should have interface to define multiple questions for a section.
7. Application should have interface to setup list of options available for a question.

8. Administrator should have interface to manage user accounts.

3.2.2 Web API Requirements

1. Web API should authenticate all the defined users in Web Application.
2. No user shall have access to the Web API except those defined in the systems.
3. API should have methods for the following operation:
    a) Project List – Client app should be able to get all the defined projects in the web application.
    b) User Authentication – API should have method to authenticate the mobile application user.
    c) Participant List – API should define a method to search participant based on name of the participant.
    d) Question details – API should have method to get details of a question
    e) Question Response – API should have method to save the responses to the questions.
    f) Password Change – Client user should be able to change their user account password using API methods.
    g) Question Option – API should define method to get all the available options for a question.

3.2.3 Mobile Application Requirements

Mobile Application primary purpose to facilitate user for the following operations:

1. Application should have user interface for login.
2. Application should have interface to search the participants by their name.
3. Application should have interface for navigating through all the question for a project and save the response from the participants.
4. Application should have interface to manage user profile.

Chapter 4 - Implementation

4.1 Web Application Design

After the completion of requirement analysis, various components, and modules have been identified and designed. All the participating entities and their relationship in the system have been defined and presented in the following E-R diagram.

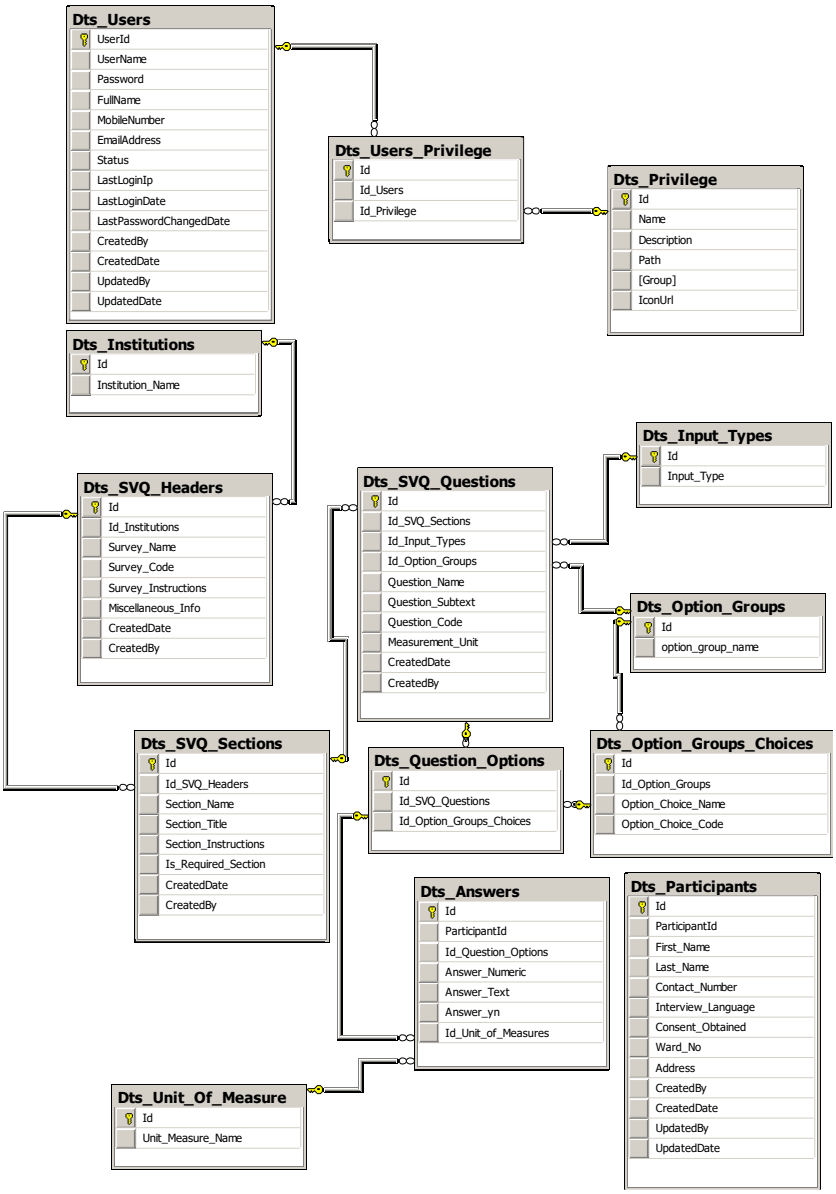


Figure 4-1: Entity Relationship Diagram

Dts_Users:

This entity stored the user login information along with system access and user status. The status represents whether the user has access to system or not.

Dts_Privilege:

This entity stores all the available privilege in the system. Dts_User_Privilege is used to map users to privilege. This helps to identify the privileges associated with a particular user. This is one to many relationships because a user can have multiple privileges.

Dts_SVQ_Headers:

This entity is used holds the all the survey project of an institution. Institution could create multiple survey project.

Dts_SVQ_Sections:

Sections contains section name and section title. The relationship between Dts_Headers to Dts_Sections is one to many.

Dts_SVQ_Questions:

This contains all the questions. Question_Name represents the actual question and Queston_subtext provides extra information about the question making it easier to answer the question. It contains primary keys from Dts_Input_Types and Dts_Option_Groups. The relationship between section to question is one to many.

Dts_Input_Types:

This entity contains all the answers type that could be use while answering the questions. Some of these include Dropdown box, Text, Checkbox, Radio Button etc.

Dts_Option_Groups:

This contains options group for generic question. For example, question related to gender, age range, level of education is grouped and options choices for those questions are stored in Dts_Option_Groups_Choices.

Dts_Question_Options:

This hold the question id and option choices id. Instead of keeping question id and its associated option choices id, Dts_Question_Options id is recorded as the answer to the question.

Dts_Answers:

This entity captures the actual response from the user. It stores participant, question, and answer along with unit of measure.

Dts_Participants:

It contains all the demographic information of the participant. In this application, participants do not directly use the application. They provide their consent to volunteers who perform survey and store all the answers received from the participant.

4.2 Web Application Architecture

COBMIN is developed using MVC (model-view-controller) architecture. This primary MVC architecture emphasis is separation of concern allowing the application to be divided into three layers.

1. Presentation Layer
2. Business Logic (BI) Layer
3. Data Access Layer (DAL)

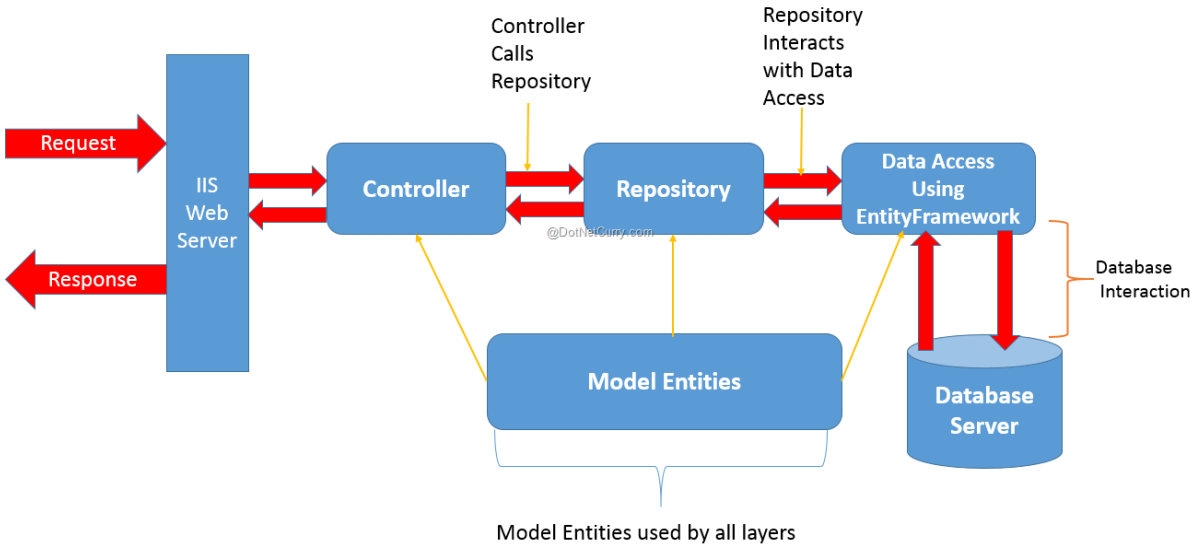


Figure 4-2: Model-View-Controller (MVC) Architecture

4.2.1 Presentation Layer

The presentation layer presents the user the interface required to interact with the application. It provides the interface to render the data to the user and allows the user to submit the user request to the application. In MVC application, views are used for rendering the content of the web page. Presentation layer interoperates with the business and data access layer to receive the content for display.

4.2.2 Business Logic Layer

Business logic layer form the middle layer in 3-tier architecture. It separates the presentation and data access layer and contains all the business logic of the application. This allows us to make changes in the business logic without having to change in presentation and data access layer. In this application, Razor view engine is used to in presentation layer views,

the presentation layer could be rewritten using web form view engine without affecting the business logic. This separation of layers allows us to work separately in each layer.

4.2.3 Data Access Layer

In this application, MS-SQL database is used to store all the information in the application. Data layer provides a mechanism to query against the database and get the results. Entity framework has been used to create data access layer. It reduces writing and managing a massive amount of ADO.NET code. It is an enhancement to ADO.NET to automate mechanism for accessing and storing the data in the database.

To perform CRUD operation (insert, update, delete) into the database to access a table, stored procedure could be created in the database. All the data manipulation operations are written in the stored procedure. Entity framework provides a mechanism to call stored procedure by calling functions. It maps the stored procedure results to import functions making it easy to query the database.

In COBMIN application, COBMIN.DAL is the data access layer which contains a CobinDB.edmx file. Below are the few functions defined as part of import?

PlsUserAuthentication:

This procedure is used to authenticate the user. It uses username and password as to authenticate the user.

PlsUser:

This method is used for user management, for example, creation of new user, editing the existing user, finding out all existing user in the system.

PlsParticipant:

Like PlsUser method, it manages the information for all the participants in the system.

PlsSVQHeaders:

This function is used to setup new project in the system. This represent the title of the survey.

PlsSVQSections:

Sections belong to headers. We could define multiple sections for a particular survey header. One survey may include multiple sections. This method allows us to create new section, update the existing section and so on.

PlsSVQQuestions:

Under each section, there could be multiple questions. This method is used to retrieve all the questions are that created as part of survey title. This allows CRUD operation related to all the questions.

PlsInterviewQuestions:

This is the main method used to conduct the survey. This method is called to fetch all the questions related to a survey in a sequential manner. Also, once the response has been recorded from the participant, the method is invoked to save the information in the database.

4.3 Use Case Diagram

UML use case diagrams provide pictorial view of the system users/actors who interact with the system. Use case diagram represents set of actions known as use cases that the actors or users of the system can perform. It describes the behavior of the system. The use case diagram represented in figure 4-3 depicts two roles – administrator and user. The administrator performs following actions in the system – creating new user accounts and managing their accounts, new institution setup and defining projects under institution, setting up question sections and questions, and viewing reports. The administrator actor inherits all the actions that a user actor

can perform in the system. The user actor can perform following actions in the system – login into the system, view participant list, search participants, and interview the participants.

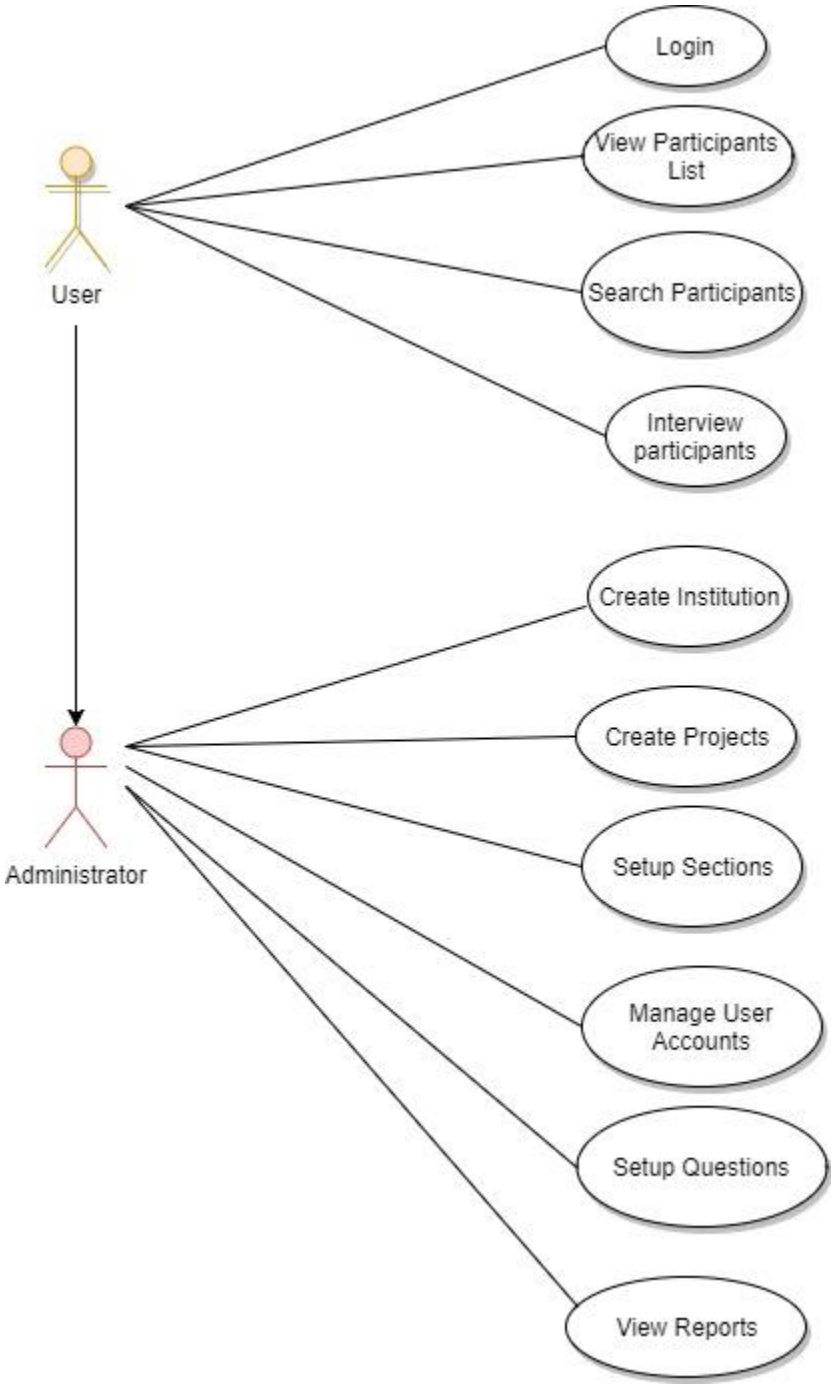


Figure 4-3: Web Application Use Case Diagram

4.4 Web API Architecture

The figure below depicts the architecture used in web API. It has four components, web API controller, repository classes, business layer and data access layer. The web API component of the application acts as an HTTP based interface and handles request from client application, for example, mobile application and performs operations on data. The repository classes provide a mechanism to decouple the business and data access layer from the controller. Web API uses inbuilt dependency injection (DI) to inject required dependency in object. Data access, business layer, repository can be registered with dependency injection and can be injected to each other. Repository classes define the method to access the data access layer by defining interfaces. Data access layer is responsible for CRUD (Create, Read, Update, Delete) operation.

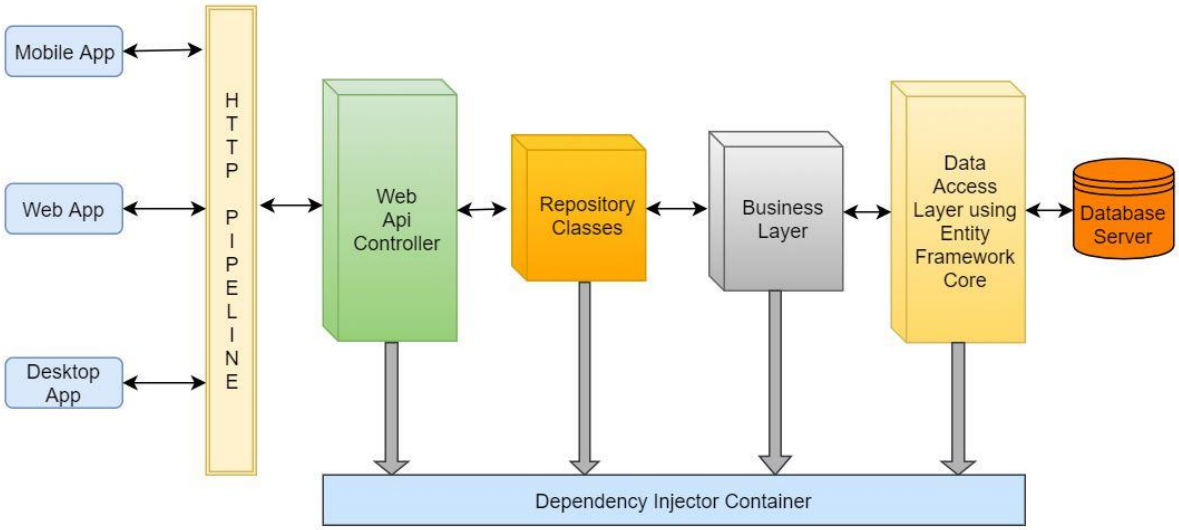


Figure 4-4: Web API Architecture

4.5 Mobile Application Architecture

The Model-View-ViewModel (MVVM) pattern is used in mobile application development to separate the user interface with business and presentation logic. The view represents user interface, view model implements properties and command to which view can bind data, model represents the data model. View is aware of the state of the view model and view model is aware of the model. However, the model is unaware of view model and view model is unaware of view. The view model separates view with model. The view is not directly dependent on the data model of the application. Whenever there is a change in model it sends back the notification to the view model and the view model sends back to the view using change notification events. The model relates to the view using data binding capability of the Xamarin.Forms.

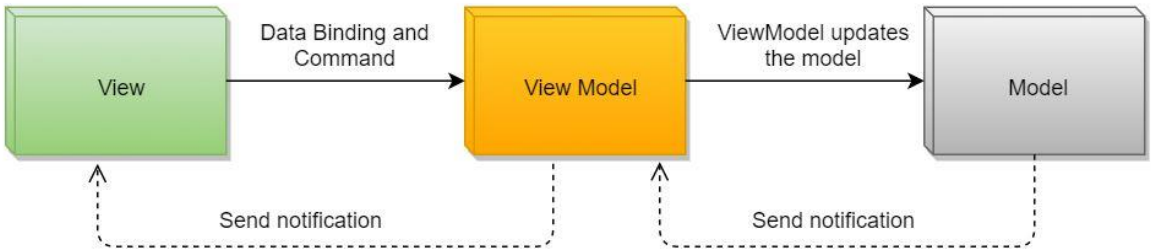


Figure 4-5: Mobile application with MVVM pattern

Chapter 5 - Evaluation

Evaluation of software system is a crucial part of the application development cycle. Software testing evaluates the application based on the requirement analysis and finds out the potential bugs in the system. It validates and verifies the software program is working as per the requirements and meets the technical and business requirement. Some of the conventional software testing methods are unit testing, black box testing, performance and stress testing etc. For COBMIN project, some of the standard operation test cases have been defined and tested.

Test cases and results:

Following test case test the functionality of the application. Following test cases were performed manually and verified.

Sn.	Test Module	Input	Test Case	Expected Result	Test Result
1	Login	Enter username and password	Enter wrong username and password.	A user-friendly user message should be displayed.	Passed
2	Setup Page	Navigate to the Project List Page.	It should display the list of projects.	It listed the project list from the database.	Passed
3	User Page	Navigate to Users List Page	It should display all the system users from the database.	It listed the available users from the database.	Passed
4	User List Page	Click on add new user icon.	It should redirect to new user creation page.	It redirected to the new user creation form where user can enter user	Passed.

				details and save user to database.	
5	Participant List Page	Navigate to Participant List.	It should list all the existing participants from the database.	It listed out the participant as expected.	Passed.
6	Participant List Page	Click on add new participant	It should redirect to page to enter new participant information.	It redirected to the new participant page and saved the data.	Passed
7	Participant List Page	Click on Edit Icon.	It should open an editable form with existing information of participant.	It redirected to edit form page and displayed associated information for the participant.	Passed.
8	Participant List Page	Click on delete Icon.	It should prompt the user before the participant is deleted.	It opened the confirmation dialogue box before deleting the participant.	Passed
9	Interview Page	Select Dropdown as COBMIN and enter participant Id	It should search the participant id and display the associated information before survey	It displayed the participant details if it was found in the database and displayed error message otherwise.	Passed

			questions starts.		
10	Interview Page	Click on start interview icon.	It should start the interview session and present the first question from the survey questions set.	It took to the question page and presented with the first question.	Passed

Table 5-1: Test Cases and Results

5.1 Web Application User Interface

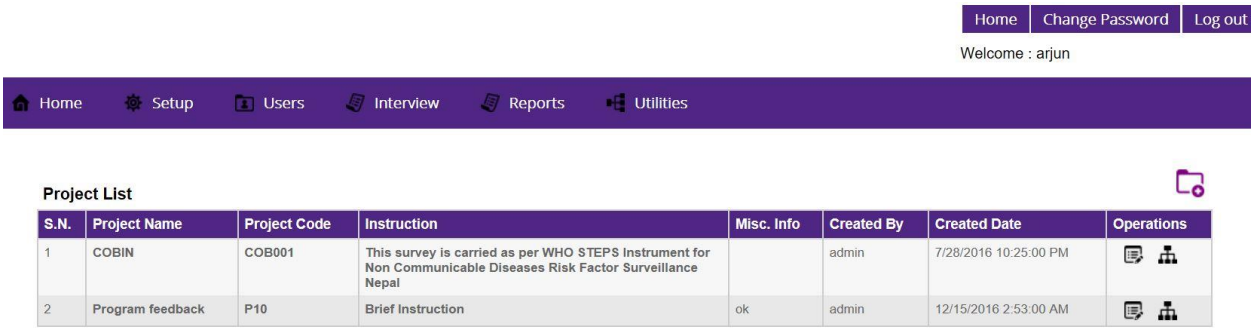
Login Interface:

Login Interface is used by administrators to setup the project and user accounts.

Figure 5-1: Web Application Login Interface

Project List Interface:

Project list interface is used to create new project, modify the existing one and navigate to the sections of a project. It has small add icon to add the new project, standard edit icon to edit, a branching icon to navigate to the section details.



S.N.	Project Name	Project Code	Instruction	Misc. Info	Created By	Created Date	Operations
1	COBIN	COB001	This survey is carried as per WHO STEPS Instrument for Non Communicable Diseases Risk Factor Surveillance Nepal		admin	7/28/2016 10:25:00 PM	
2	Program feedback	P10	Brief Instruction	ok	admin	12/15/2016 2:53:00 AM	

Figure 5-2: Web Application Project List Interface

Section Interface:

The branching icon in the project list interface will navigate the user to the sections of a project. Section Interface allows administrators to add new sections using small add icon, edit a section, delete the section and navigate to questions related to that section.

Section List

S.N.	Section Name	Section Title	Section Instruction	Is Required?	Created By	Created Date	Operations
1	Setp1 Behavioral Measurement	History of Diabetes			admin	8/28/2016 10:19:39 PM	
2	Step 1 Behavioral Measurements	Tobacco Use			admin	8/28/2016 10:19:39 PM	
3	Step 1 Behavioral Measurements	Alcohol Consumption			admin	8/28/2016 10:19:39 PM	
4	Step 1 Demographic Information	Demographic Information			admin	8/28/2016 10:19:39 PM	
5	Step1 Behavioral Measurement	Physical Activity	Next I am going to ask you about the time you spend doing different types of physical activity in a typical week. Please answer these questions even if you do not consider yourself to be a physically active person. Think first about the time you spend doing work. Think of work as the things that you have to do such as paid or unpaid work, study/training,household chores, harvesting food/crops, fishing or hunting for food, seeking employment, walking uphill/downhill for routing work. In answering the following questions ' vigorous-intensity activities' are activities that require hard physical effort and cause large increases in breathing or heart rate, 'moderate-intensity activities' are activities that require moderate physical effort and cause small increases in breathing or heart rate.		admin	8/28/2016 10:19:39 PM	
6	Step1 Behavioral Measurement	Indoor Air Pollution	The next questions ask about the indoor air pollution		admin	8/28/2016 10:19:39 PM	

Figure 5-3: Web Application Interface for Sections of a Project

Question Edit Interface:

The below screenshot shows the edit form interface for question. The same layout is used for user, participant, section, project edit form interface.

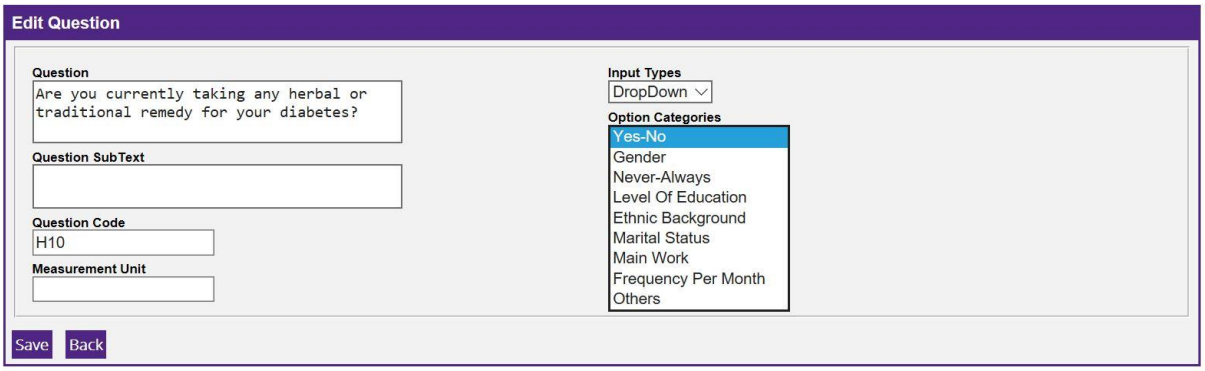


Figure 5-4: Web Application Question Edit Interface

Participant List Interface:

Participant list interface also uses the same navigation icons to modify the participant information. Also, similar interface is used for user account management.

S.N.	First Name	Last Name	Participant ID	Address	Ward No	Contact No	Consent Obtained	Operations
1	Arjun	Khanal	KL920348	Milanchowdk	6	9792838383	Yes	
2	Json	Adhikari	KL92034811	Tadi	2		Yes	
3	Abhishes	Lamsal	NA29340	Bharatpur	8	928348348	Yes	

Figure 5-5: Participant List Interface

5.2 Mobile Application User Interface

Login Interface:

The login interface for mobile users has standard username and password form for user authentication. The user is authenticated using Web API login method.

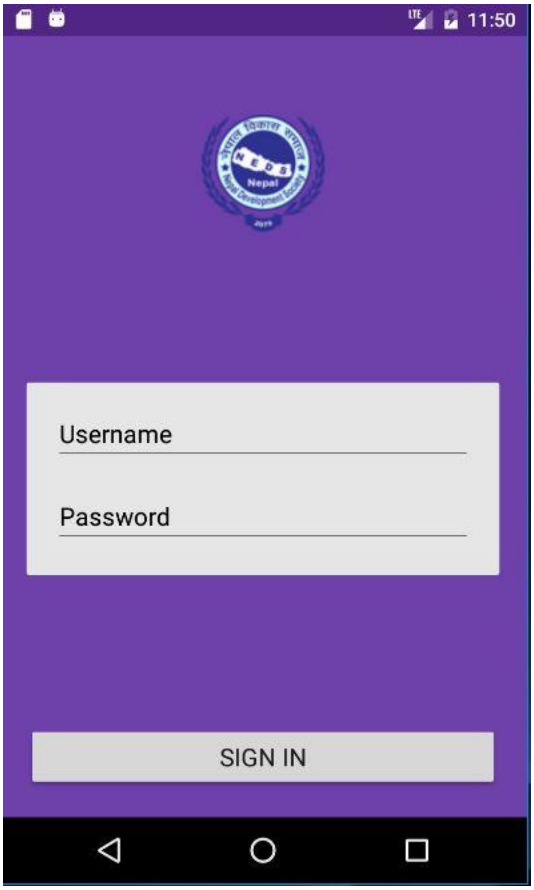


Figure 5-6: Mobile Application Login Interface

Action Menu Interface:

The below screenshot displays the action menu for mobile users. These actions require service from Web API. The following action are available:

- Current Projects: List all the available project in the system.
- Questions: Starts an interview session with the participant and follows next question once the first question is answered.
- Participants: This action menu allows to search participants.
- Change Password: Allows user to change current password and set to new password.
- Log Out: Allow user to go back to login page.

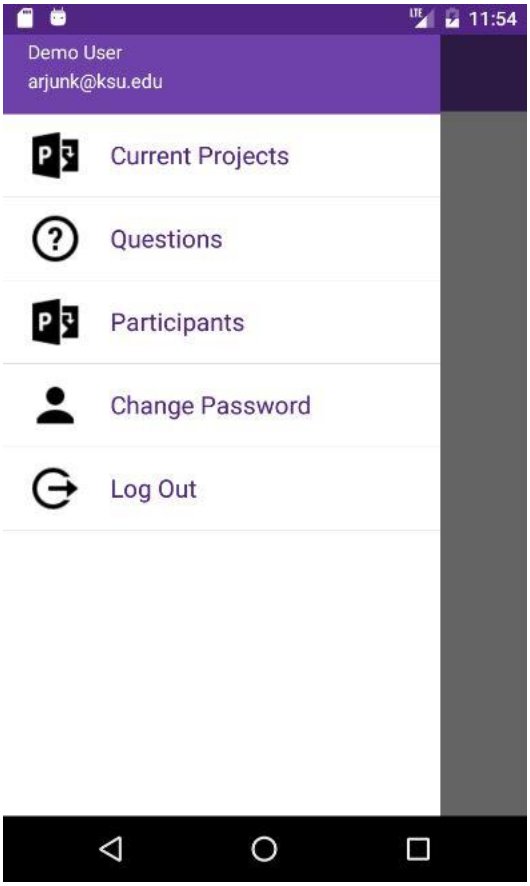


Figure 5-7: Mobile Application Action Menu Interface

5.2 Performance Testing

Performance testing is carried out to ascertain the effectiveness of the application especially when the number of users using the application increase. In Software application, it is done to estimate the response time and throughput of the application. These statistical measures give insight into how well the application is performing where number of user increase significantly.

5.2.1 Tool Used

The Apache JMeter, an open source Java software application, is used to measure the performance of the COBMIN application. This tool is very handy for simulating the high number of users and monitoring the response time of the application. It can invoke both HTTP get and post request which facilitates to test both static and dynamic web pages.

5.2.2 Performance Measurement Terminologies

Latency:

JMeter measures the latency from just before sending the request to just after the first response has been received [10].

Connect Time:

JMeter measures the time it took to establish the connection, including SSL handshake [10].

Throughput:

It is calculated as requests/unit of time. The time is calculated from the start of the first sample to the end of the last sample. This includes any intervals between samples, as it is supposed to represent the load on the server [10].

The formula is: Throughput = (number of requests) / (total time).

Standard Deviation:

Standard deviation is a measure of the variability of a data set. This is a standard statistical measure [10].

Median:

Median is a number which divides the samples into two equal halves. Half of the samples are smaller than the median, and half are larger [10].

Elapsed time:

JMeter measures the elapsed time from just before sending the request to just after the last response has been received [10].

5.3 Web Application Performance Test

The web application was tested reflecting the client-server scenario using workstation computers.

5.3.1 Web Application Test System Configuration

Operating System	Windows 8.1 – 64 bits
RAM	16 GB
Processor	Intel core i7
Processor Speed	3.4 GHz

Table 5-2: Web Application Client System Configuration

Operating System	Windows 10 – 64 bits
RAM	16 GB
Processor	Intel core i7
Processor Speed	2.9 GHz

Table 5-3: Web Application Server System Configuration

Test1: Login Page

I tested the login page with user login credentials supplied as a parameter for 100 users. The expected maximum number of the user of the web application is around 20. The average time per login request was less than two seconds.

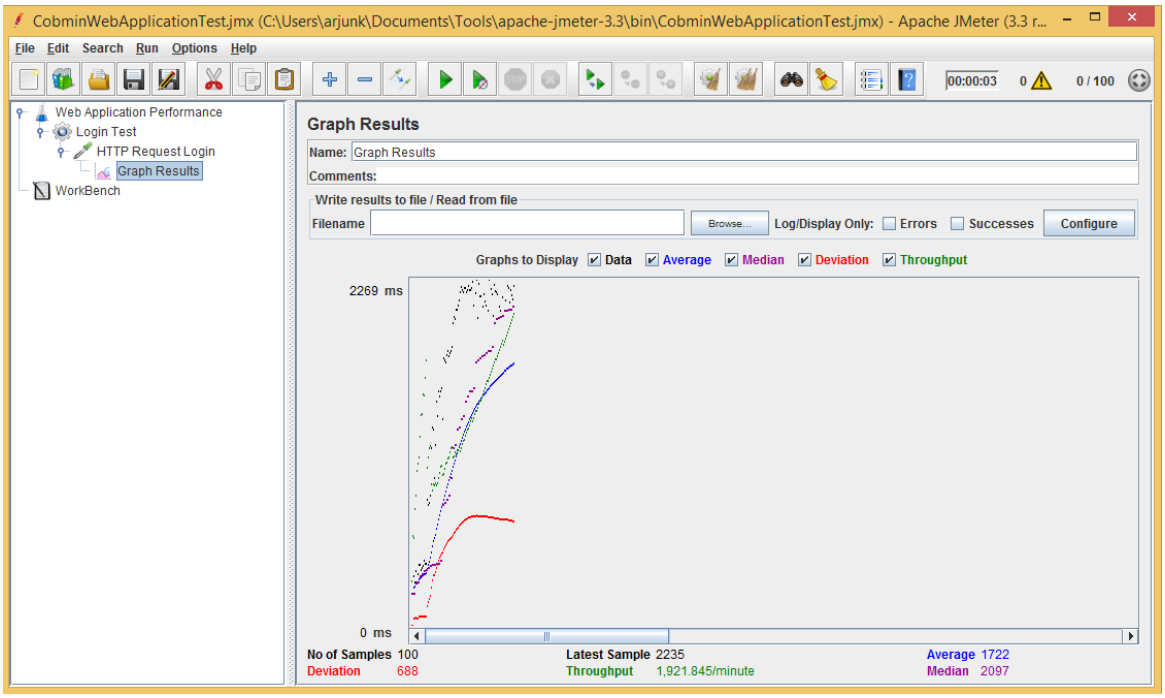


Figure 5-8: Web Application Login Test for 100 Users

Test2: Search Participants

I tested the search participant page with participant Id supplied as a parameter for 100 http request. The average response time for 100 http request was 543 milliseconds. The total number of participants in the system was 5.

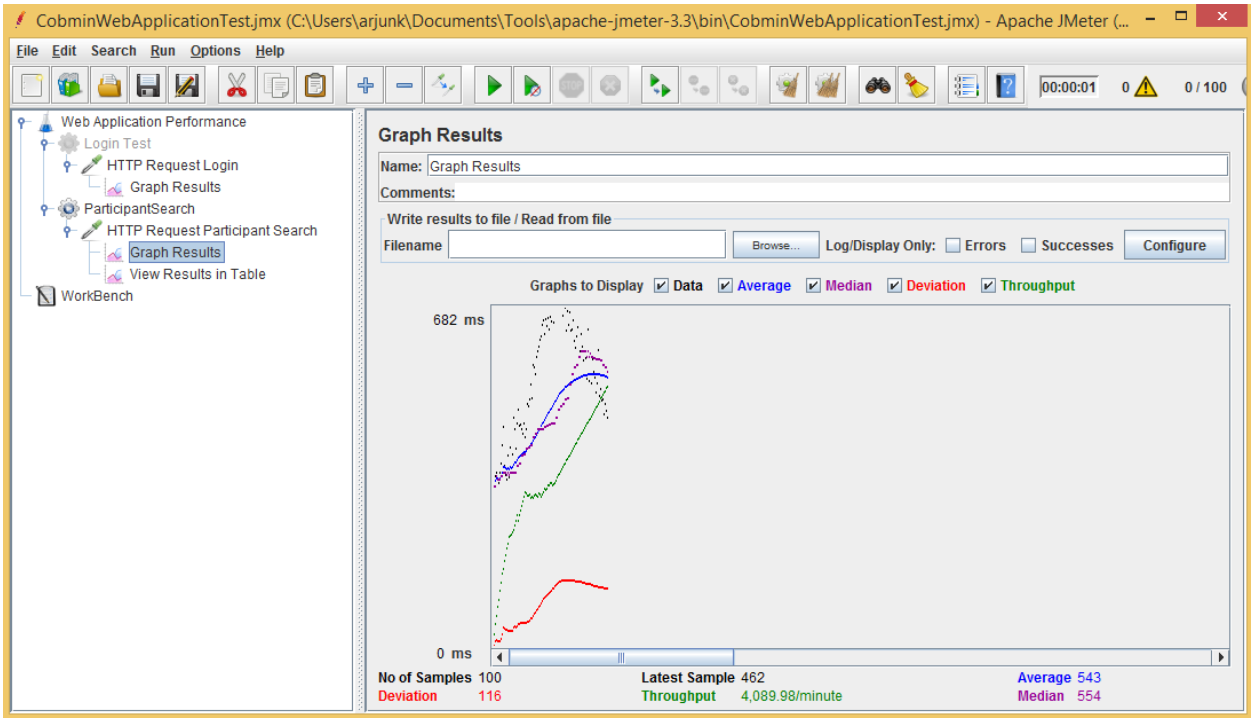


Figure 5-9: Web Application Participant Search with 100 Request

5.4 Web API Performance Test

The web API was tested reflecting the client-server scenario using exact same configuration provided for web application testing.

Test1: API Login Test

The login method in the Web API authenticates the user using username and password. If the user is successfully authenticated, the server creates new access token for that user and sends back the token to the user. The average response time for this test was around two seconds.

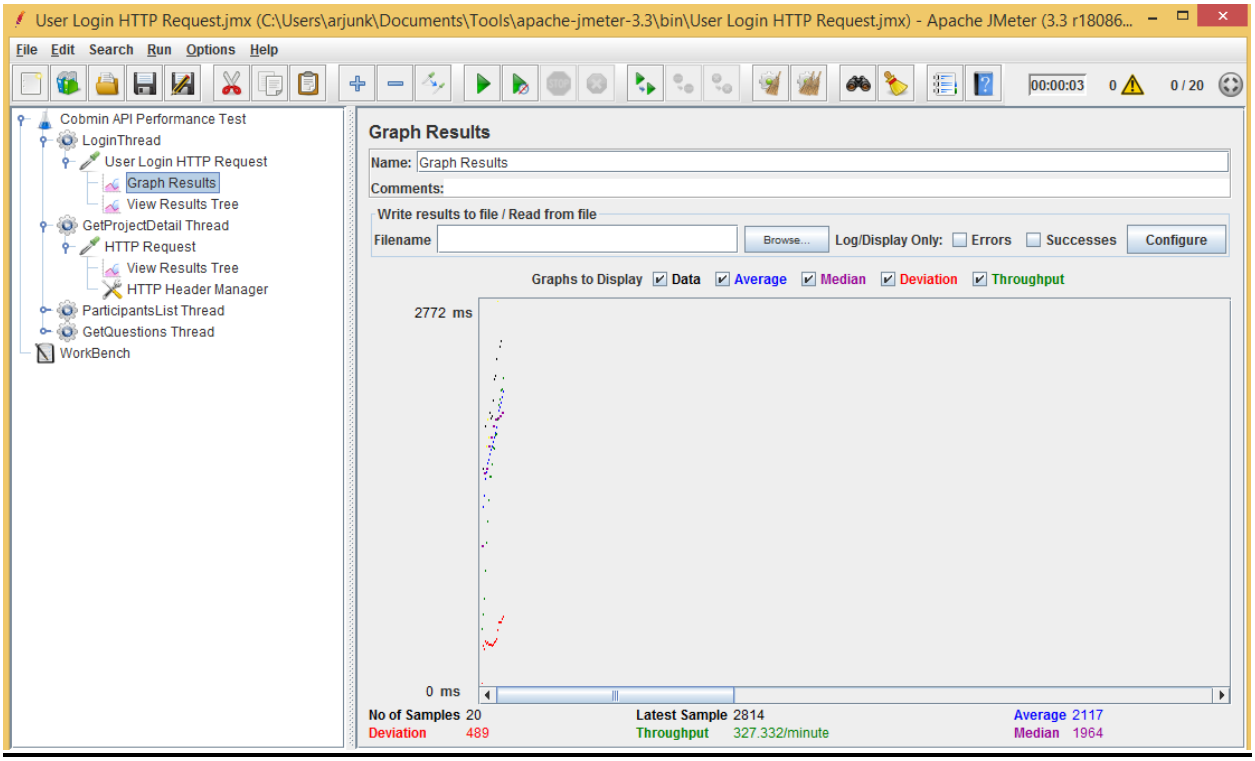


Figure 5-10: Web API Login Test with 20 Users

Test2: API Get Project Details

API Get Project Detail methods is used to request all the available projects in the system. This method validates the access token and if the token is valid it processes the request. I made 20 https request and the average response time for this method was around 4 seconds.

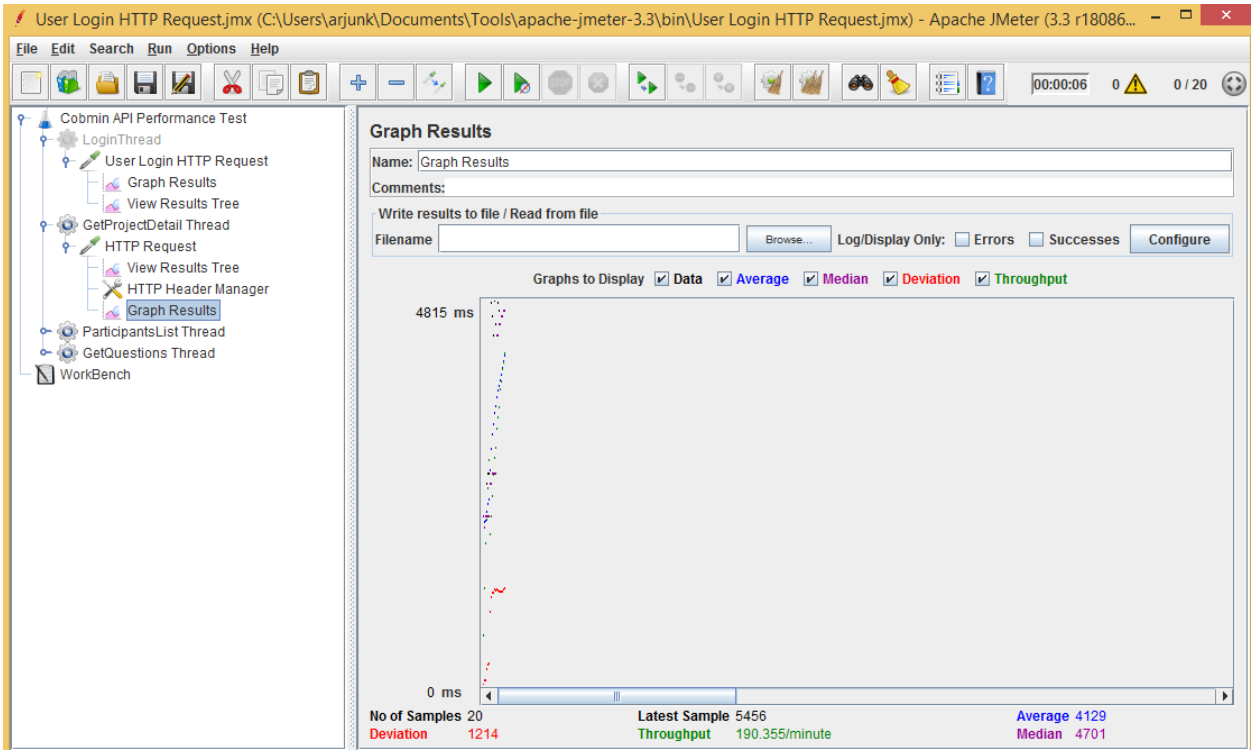


Figure 5-11: Web API Get Project Detail with 20 Http Request

Test3: API Participant Search

The response time for the search participant method was around 4 seconds as well.

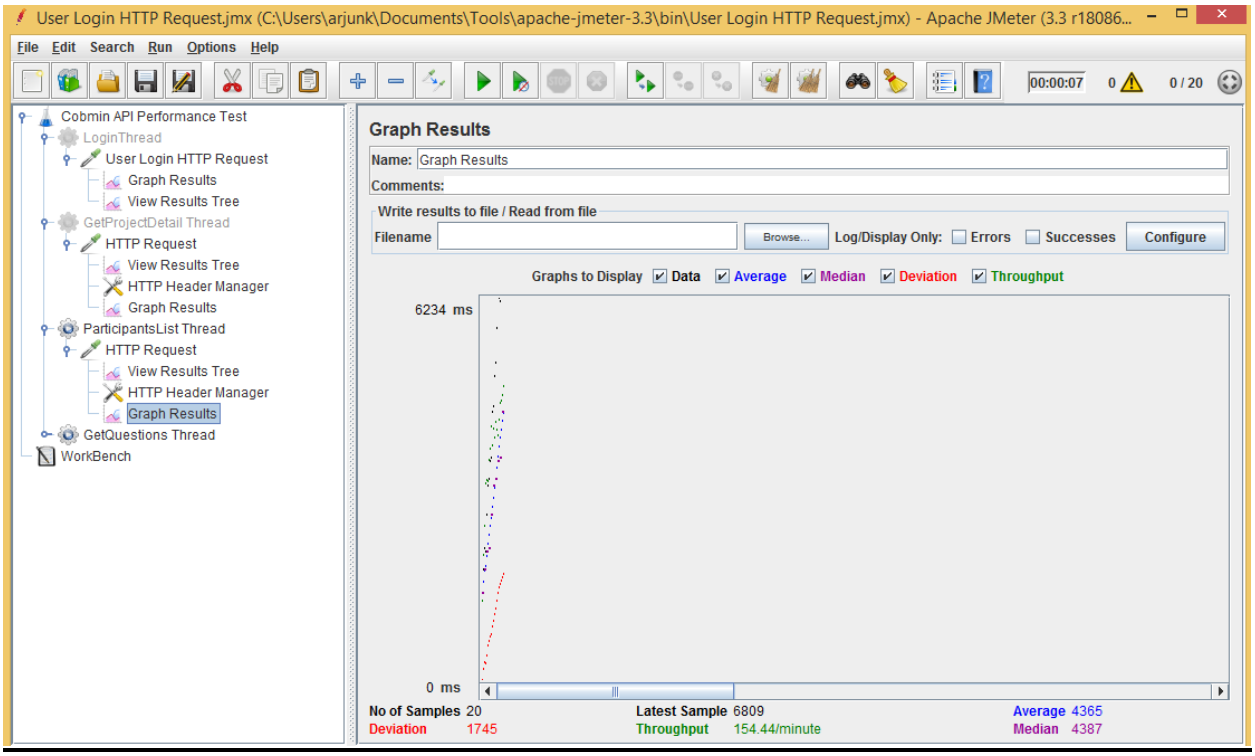


Figure 5-12: Web API Participant Search with 20 Http Request

Test4: API Get Question Details

This method will retrieve the questions for a project sequentially. It uses session id, question id to keep track of questions which are already responded to client. Moreover, this method saves the previous question response and gets the new questions for that session. The average response time for 20 http request was around 4 seconds.

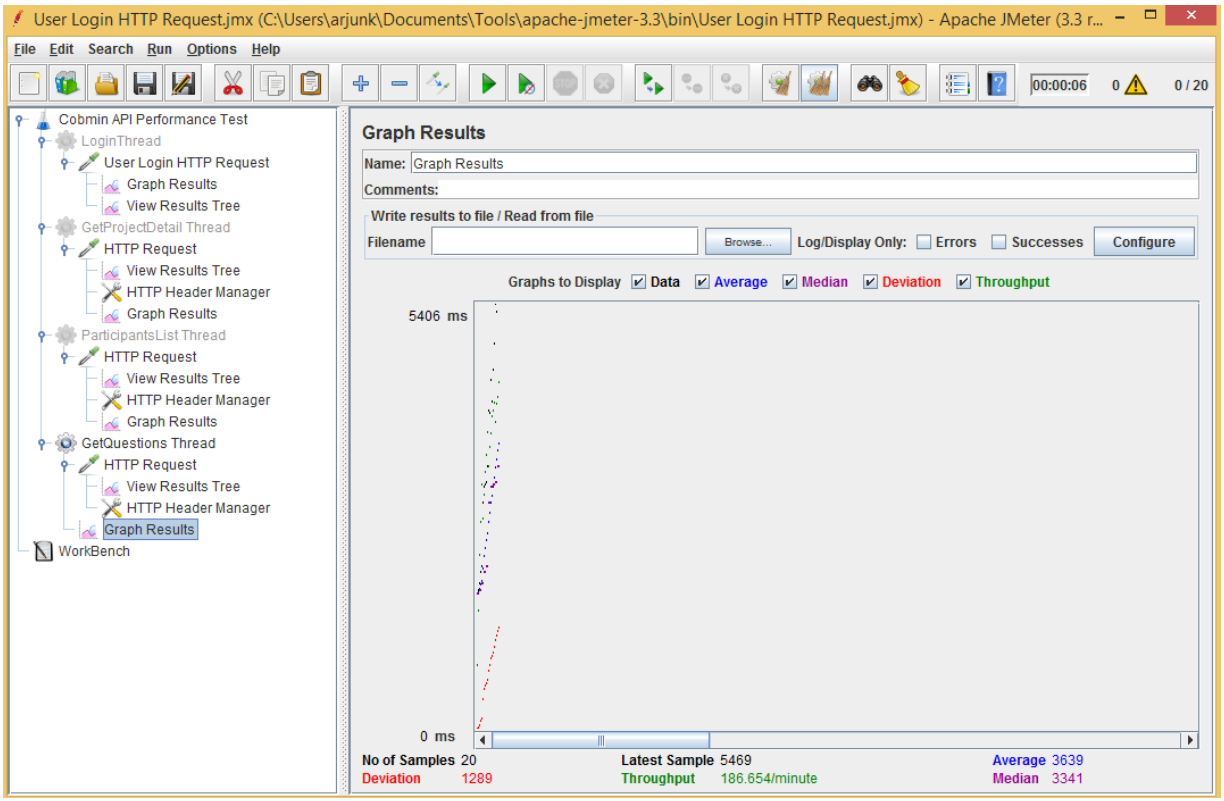


Figure 5-13: Web API Get Question Detail with 20 Http Request

5.5 System Security

Web applications are vulnerable to several security threats, for example, SQL injection, cross-site scripting attacks, and cross-site request forgery. The COBMIN application uses dynamic query and stored procedure to process the query and validates the requesting user. Also, it uses ValidateAntiForgeryToken to prevent falsification of requests. The application validates the request before it processes the request.

Likewise, the application utilizes encryption and decryption logic to conceal the information related to the participants in the database storage. In COBMIN application, the member's details are hidden, and functions which contain this logic are encryption itself making it impossible to decrypt the logic and view the participant information. This is required to protect the health information of the participants.

Chapter 6 - Conclusion

5.1 Conclusion

The COBMIN application is particularly developed for NEDS organization for expediting the survey process for Hypertension Management Program. The users of the system can perform various operations related to projects.

1. User Management: This module manages the application users. Admin can add a new user, edit and delete the existing users.
2. Participant Management: Users have the privilege to add new participant in the system and save demographic information in the system.
3. Survey questions Management: Admin can define survey sections and questions within each section.
4. Survey (Interview Session Management): Users can record the responses to each question in the survey from the participating member in the program.

5.2 Overall Experience

I designed and developed the COBMIN application using ASP.NET MVC and Entity framework. I have used MS-SQL, a relational database management system, as a backend for the storage system. This project helped me to enhance my knowledge of Model-View-Controller and Entity framework. I gained a good practical knowledge of these technologies by implementing these tools and techniques in the project.

Also, I learned database programming, for example, writing SQL queries in stored procedure. Stored procedure is database objects which can be used to manage multiple dynamic SQL queries and execute those queries based on the parameter passed to the procedure. Moreover, I learned to write user defined function as part of database programming, for example,

writing password encryption and decryption logic. These types of functions are very useful while working with sensitive data, for instance, the medical records of the persons.

I used visual studio 2010 for the development environment. I learned to configure the project and install entity framework from Nuget package manager. I configured the project in three parts COBMIN.Classes, COBMIN.DAL, COBMIN.Web. COBMIN.Classes contain membership and role provider classes along with some static method. COBMIN.DAL holds Entity Framework and functions for database operations. Similarly, COBMIN.Web contains views for presentation, controller for request processing, and model for managing business logic and rules.

In a nutshell, I gained significant knowledge about how to build the web application from scratch following the software development lifecycle, requirement analysis, system design, coding, testing, and release.

5.3 Future Work and Enhancements

Following are the certain aspects of the application which can be improved for the future.

1. The current scope of the application is limited to NEDS organization and the specific project they are working. In future, this can be extended to make it more general survey application.
2. It can be extended to make it applicable to other institutions for different types of survey projects.

Bibliography

[1] MVC pattern for database operation. http://www.dotnetcurry.com/aspnet-mvc/1155/aspnet-mvc-repository-pattern-perform-database-operations

[2] ASP.NET MVC Overview. https://msdn.microsoft.com/enus/library/dd381412(v=vs.108).aspx

[3] Implementing CRUD functionality with Entity Framework in ASP.NET MVC application. https://www.asp.net/mvc/overview/getting-started/getting-started-with-ef-using-mvc/implementing-basic-crud-functionality-with-the-entity-framework-in-asp-net-mvc-application

[4] Server-side Implementation. https://msdn.microsoft.com/en-us/library/hh404093.aspx

[5] Entity Framework. https://blog.magnusmontin.net/2013/05/30/generic-dal-using-entity-framework/

[6] Representational state transfer. https://en.wikipedia.org/wiki/Representational_state_transfer

[7] Xamarin.Forms. https://www.xamarin.com/forms

[8] Token Authentication. https://samueleresca.net/2016/12/developing-token-authentication-using-asp-net-core/

[9] Web API Architecture with Entity Framework http://www.dotnetcurry.com/entityframework/1348/ef-core-web-api-crud-operations

[10] Performance Measurement Terminologies Definition. http://jmeter.apache.org/usermanual/glossary.html