

FOOD NUTRITION PROGRAM REPORTING SYSTEM

by

SRAVYA BOGGAVARAPU

B.Tech., Jawaharlal Nehru Technological University, India, 2008

A REPORT

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2012

Approved by:

Major Professor
Dr. Daniel Andresen

ABSTRACT

FNPRS program offers nutrition education all over the counties in Kansas. It is necessary to keep track of the budget expenses for the program, resources used and many other parameters involved in the program. There exist number of commodities and products in the process of educating people. Research is needed to determine which value-added products or processes are economically possible and what percentage of it is accepted by people. For these issues, it is very important to maintain this information in a database and generate reports accordingly.

The aim of the project is to create a web interface for users to enter the program information regarding the various programs conducted by Family Nutrition Program. The various kinds of data include information about the budget for the program, information about the various collaborating agencies, various kinds of resources used, services provided, proposed equipment and travel funds etc. Users for this application are county agents who take the responsibility of conducting the program and managing their data. Creating a web interface provides a solution to facilitate the agents to manage their data more efficiently and to monitor their records on a day to day basis.

It also aims for generating reports for Family Nutrition Program in order to keep a check over their advancements in the program. This project involves handling of various kinds of information such as FNP Proposals, Agent information, FNP Funds, Collaborating Agencies.

Database maintenance is made simple thereby allowing the administrators to add as much as data possible and manage accordingly.

TABLE OF CONTENTS

TABLE OF CONTENTS.....	iii
LIST OF FIGURES	iv
LIST OF TABLES	v
ACKNOWLEDGEMENTS	vi
CHAPTER 1: INTRODUCTION	1
CHAPTER 2: RELATED WORK.....	3
CHAPTER 3: IMPLEMENTATION	5
3.1 System Design	5
3.2 System Architecture.....	8
3.3 Modules	17
3.3.1 Intended Users	17
3.3.2 Requirements	17
3.3.3 Inputs and Outputs:	17
3.4 Screenshots	20
CHAPTER 4: EVALUATION	27
4.1. Unit Testing	27
4.2 Performance Testing.....	31
4.2.1 Tool for performance testing.....	32
4.2.2 System Configuration	32
4.2.3 Test plans Results and Evaluation	32
CHAPTER 5: CONCLUSION AND FUTURE WORK	39
5.1 Conclusion	39
5.2 Over all Experience	39
5.3 Future Enhancements to the project.....	40
CHAPTER 6: REFERENCES	41

LIST OF FIGURES

Figure 1: Class diagram from the Normal user's point of view	5
Figure 2: Class diagram from the Admin user's point of view.....	7
Figure 3: Three - tier architecture	8
Figure 4: Average Response Time – Login Page	35
Figure 5: Average Response Time - Main Page	38

LIST OF TABLES

Table 1: Test cases and results.....	29
Table 2: System configuration.....	32

ACKNOWLEDGEMENTS

This acknowledgement transcends the reality of formality when I would express deep gratitude and respect to all those people behind the screen who guided and inspired me for the completion of my project work.

My special thanks to my major professor Dr. Daniel Andresen for giving me timely advice, encouragement, and guidance throughout the project.

I would also like to thank Dr. Gurdip Singh and Dr. Mitchell Neilsen for graciously accepting to serve on my committee. Furthermore, I would like to thank Mr. Gamage P Dissanayake, my Project Manager at Department of Communications for the great support he provided me in learning new technologies and for the successful completion of my project

I would like to thank the administrative and technical support staff of the department of CIS for their support throughout my graduate study.

CHAPTER 1: INTRODUCTION

The current project mainly develops a website for Kansas State Research and Extension group in order to manage information regarding the Food Nutrition Program. It allows users to enter the Program information. Users here are representatives of the counties who manage the program usually identified as agents. The program description is stored on yearly basis for every county as worksheets. Reports are generated in order to keep track of the budget expenses and the resources needed. Administrators can manage user counties such as adding and deleting agents from counties. The data entered into the database can also be edited according to the changes made in the program. Privileges are varied between administrators and normal agents who also have the privilege to delete/modify certain information whereas normal agents are restricted to few portions of the website.

It also has to maintain and store data in varied ways such as the budget expenses for the program, information about the various collaborating agencies, various kinds of resources used, services provided, proposed equipment and travel funds etc. It also needs to generate reports in order to keep in track the progress of the program and also calculate the needs of the program. If it has to be done manually it requires lot of effort and also maintaining paper work for huge data is irksome.

The solution for handling the above mentioned issues is to maintain an online system in order to reduce the task of the agents manually. A website should be designed in an orderly manner which is helpful for the users/agents to enter the data, store in the system and edit them accordingly whenever needed. The website developed is hence user friendly and also do not take much of the process time when generating or reading huge data from the database. This has been handled using the performance metrics.

User should be able to view the information on a yearly basis for every county in Kansas. The application allows selecting a year and also county in order to view and add the program information. Every agent is either a program assistant or nutrition assistant who are given salaries and benefits according to their FTE (Full Time Equivalent). The application calculates the wages given to the agent according to the position which is a part of the budget proposal.

The application also allows entering the justification for every proposal entered into the system which counts for making advancements and necessary changes in the program. Summary of the whole program can be generated and viewed once the agent enters the complete program information.

The major concern of the Family Nutrition Program was the heavy data it handles. It had to keep a record of the various kinds of data such as work done, the budget, wages, travel, resources etc. If this has to be done manually it requires lot of tedious effort from the agents. Database is designed in order to store these data in an orderly manner from which it can retrieve information quickly without taking much time of the users. In the absence of an online system it is highly impractical to maintain data.

ASP.NET with C# is used for designing the website. HTML, CSS and JQuery are used to enhance the features of the website. Oracle 11g database management system has been used to store all the information for Kansas State Research and Extension.

The rest of this paper first discusses related work in chapter 2, and then describes our implementation in chapter 3. Chapter 4 describes how we evaluated our system and presents the results. Chapter 5 presents our conclusions and describes future work.

CHAPTER 2: RELATED WORK

Website can be designed in any number of ways. Choosing a methodology of developing a website depends on the requirements, performance and ease of working on an application. In a broader sense, there can be many approaches in developing a website. It is not necessary that every web application should have a proper architecture which contributes in performance. Structure of the application is important for greater performance. While considering the web application, developing a website is quite easy with ASP.NET framework as it reduces the amount of code required to build large applications. It is complemented with a rich toolbox and designer in the Visual Studio integrated development environment. The implementation of website which is proposed for developing FNPRS is beneficial.

ASP.net is preferred for developing this website for several factors. It is purely server side technology. ASP code executes on the server before it is sent to the browser. The pages are written with HTML and C# code together and hence easier to maintain. Also offers built in security features through windows authentication. Maintenance of program makes easier as the content and program logic are separated. It also makes easy deployment and needs no registration of components because the configuration information is built in with ASP.Net. Web Server continuously monitors the pages, components and applications running on it. If there are any memory leaks, infinite loops and any other illegal activities, it immediately destroys those activities and restarts itself.

FNPRS has abundant information stored in the database and requires ADO.Net to handle data binding. ASP.net works easy with ADO.Net which runs faster and counters large volumes of users without having performance problems. It provides database access and database design tools from within Visual Studio .Net. It supports cascading style sheets (CSS) and also JavaScript for developing enhanced features. FNPRS is developed with C# programming language which provides a type safe, object oriented version of the C programming language. In JSP deployment of servlet is a pain and there is a long deployment process in restarting the server whereas ASP.Net has automatic deployment.

IDE used is Visual Studio 2008 developed by Microsoft, which helps to build various ASP.NET web applications and XML Web services by providing various development tools. It

also helps in building console as well as GUI applications that can be in native code combined with managed code for all platforms that can be supported by Windows, .Net framework etc. There are several built in tools which include a forms designer which can also be used to build GUI applications, web/ class/ database designer etc.

FNPRS GUI is developed using HTML, ASP Controls, CSS and jQuery have been used. jQuery UI provides abstractions for low-level interaction and animation, advanced effects and high-level, theme able widgets, built on top of the jQuery JavaScript Library, that you can use to build highly interactive web applications.

Designing database is crucial for developing web applications. Many applications use SQL Server, MS Access or Oracle for database management. Challenging part here is to select a database suitable for the application. All these vendors made enhanced manageability of their database server. Despite the addition of new features in the area of manageability in SQL Server 2008, it still lags behind Oracle Database 11g. Oracle 11g is a sophisticated self-managing database that automatically monitors adapts and fixes itself. The self-managing solutions found in the Oracle database allow DBAs to be more productive. Oracle Enterprise manager is the GUI tool provided by Oracle for the management and maintenance of their systems. It provides the ability to seamlessly manage hundreds and thousands of systems located across countries from a single point of control. Looking from a database-centric point of view, Enterprise Manager significantly simplifies day-to-day database administration.

Database design counts to the performance of the request for every user. All the relations defined in the FNPRS are in third normal form, which eliminates redundancy in database and makes it consistent. FNPRS needs to update program information on day to day basis. It requires database that can handle large amounts of data. To handle these issues, Oracle is chosen for FNPRS which best performance and scalability.

CHAPTER 3: IMPLEMENTATION

3.1 System Design

Based on the requirements, the different components, modules, interfaces and interactions of the system have been designed. Various entities have been identified and the relations between these entities are described in the following class diagrams.

Class Diagram:

A class diagram showcases the basic types being built in the system. It forms a prototype for the application being developed. It encompasses the system's classes, attributes, operations and the relationship between these classes. The main structure of the web application developed during this project can be represented by the following class diagram.

Class diagrams have been designed each for different roles Administrators, Normal users.

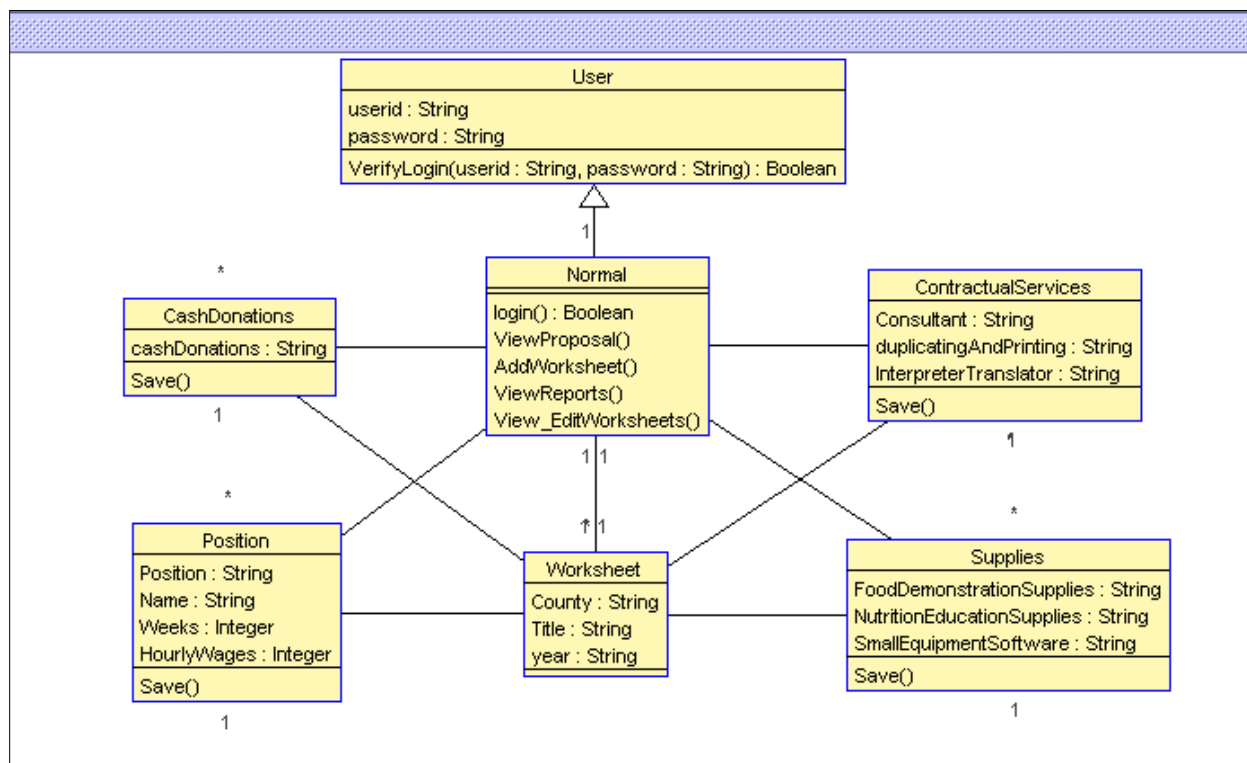


Figure 1: Class diagram from the Normal user's point of view

The above class diagram describes the important operations involved in each component. Normal user can add work sheets with all the details mentioned as classes such as Cash donations, Contractual services, Position and Supplies.

Based on the requirements given, every user can add certain details into the database and hence these are divided into classes. In the above diagram User class had user id and password which are accepted from the user in the string format. The user class is defined with user id and password as attributes. The operations involved are verifying the login entered by the user.

Normal user is the generalization of a user which inherits the attributes from the User class. It has additional operations such as viewing proposal, adding worksheets and also editing worksheets.

Every user can add a work sheet. Work sheet contains Contractual Services. To enter contractual services user has to enter the consultant who is managing the program. The expenses and the equipment required for duplicating and printing the records. The attributes are defined as Consultant, duplicatingAndPrinting accepted as string variables. After entering the equipment details the user has to save the data into the database. The operation of contractual services is considered as save.

Proposal details also consider Supplies information. Supplies information includes food demonstration supplies, Nutrition education supplies and small equipment software. These specifics are included in Supplies class as attributes. Save is an operation for Supplies class to save the information into database.

Every program has agents/persons who manages the program or responsible for taking classes. These agents are classified as certain positions in order to pay their salaries accordingly. The name and position are accepted as strings from the user. The number of weeks and the rate at which the agent works on hourly basis are accepted as integers from the user. These details are saved in the database and salaries are calculated based on this information.

There are also cash donations taking place for every program. The cash donations attribute are accepted as string, then converted into double and saved into the database.

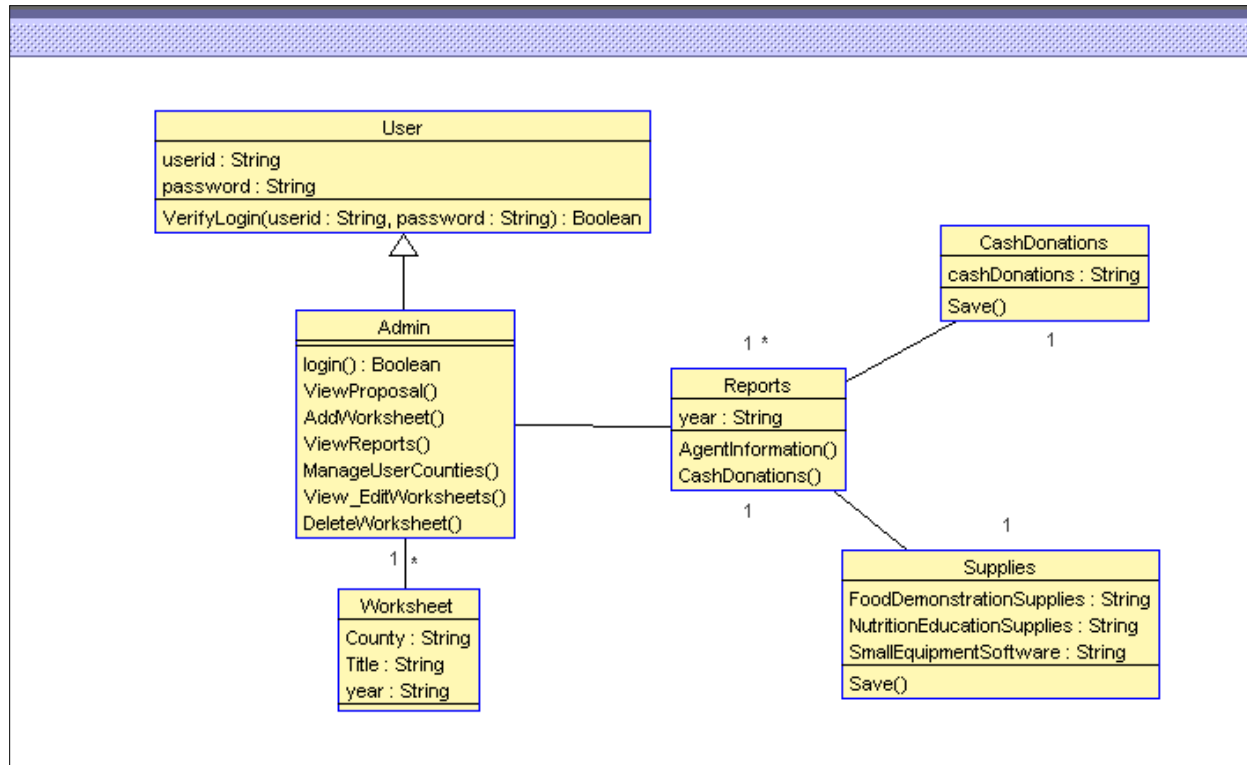


Figure 2: Class diagram from the Admin user's point of view

The class diagram in the above figure describes the important operations and attributes involved in each component. Admin user can view and generate the reports for Cash Donations and also Supplies entered by the user.

Admin is the generalization class of a user, which inherits the attributes from the User class. It has additional operations such as viewing proposal, adding worksheets and also editing worksheets. Viewing reports and User management are the additional operation which an Admin can perform.

Admin also can add worksheets as already mentioned for the normal user. For every worksheet user has to enter the county where the program is taken place and also the title of the program. Also the year in which the program is taken place is entered to keep in track the proposal for certain year.

Reports class is to generate reports for the details entered by the user. The operations here are Agent information and cash donations. Supplies and Contractual services operate same as a normal user.

3.2 System Architecture

FNPRS is developed using 3- tier architecture. Three tier architecture follows the divide and rule policy and is divided into three logical layers.

- Presentation Layer
- Business Logic Layer
- Data Access Layer

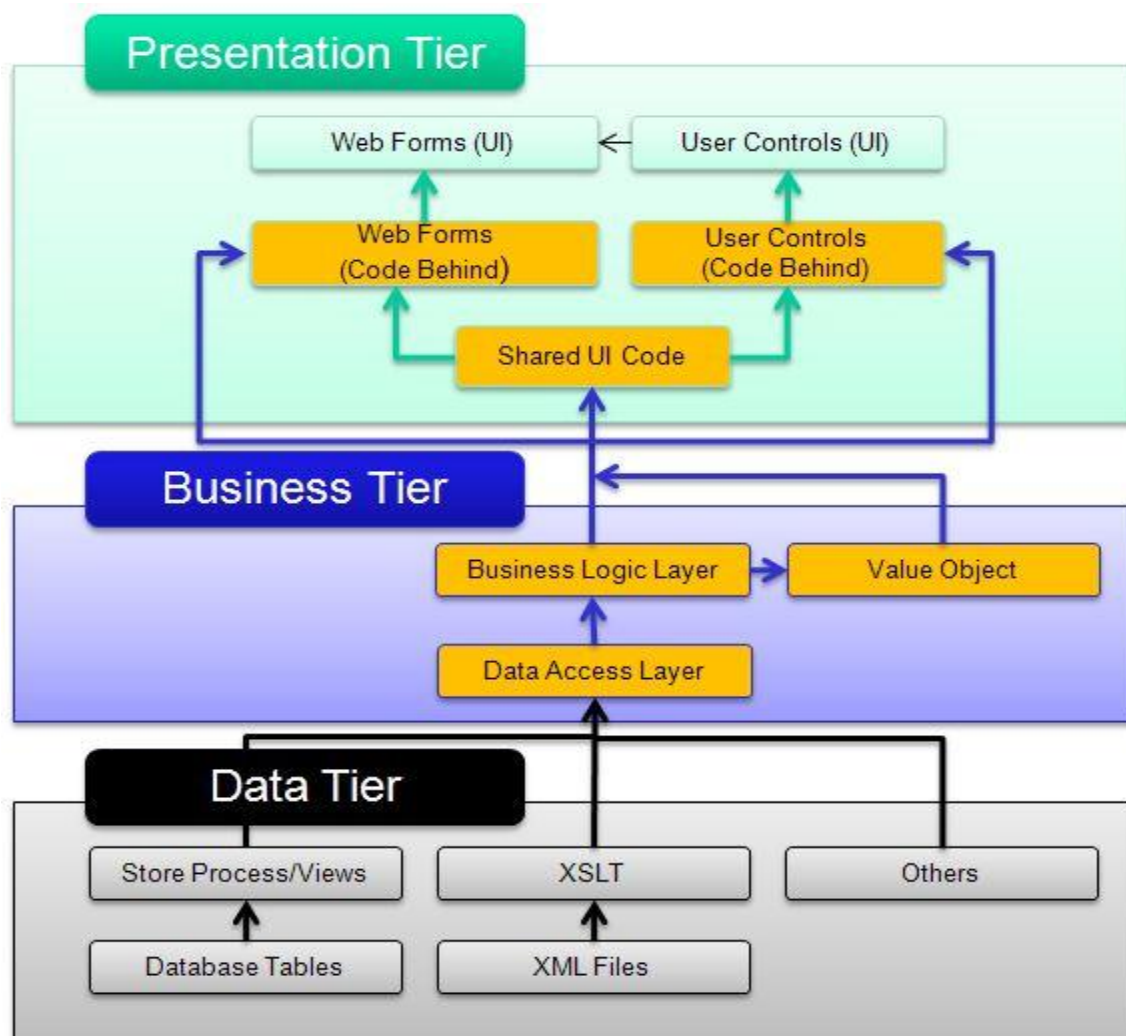


Figure 3: Three - tier architecture

<http://shivannavk.blogspot.com/2010/03/3-tier-architecture.html>

Each layer ideally specializes in one or a handful of functionalities that service the upper layer. Each of these layers is designed such that the layer above does not need to understand nor know the implementation details of any of the layers below it. This has been accomplished by providing well defined layers of abstraction that are not visible to the layers above it.

The advantage of this approach is that even if implementation has been changed in any of the layers, it does not affect the other layers. 3-tier architecture saves hours of development time. In the divide and conquer strategy each tier is easy to develop. Here each bit is coded only once with powerful re usage. Three simple parts are always better than single complex code.

Presentation Layer:

The presentation layer here refers to the Graphical User Interface of the website. It is responsible for retrieving the data that has been rendered by the other layers. The data is manipulated and presented to the user in an easy to consume format. It includes the following types of software components that perform specific tasks. The combination of below two types of components forms presentation layer. Presentation layer interoperates with the business and data access layers to form the overall solution.

User Interface Components: These components make up user interface of the component. Users interact with these components. The main purpose of this layer is to acquire data from the user and render data to the user. These are also used to validate user login, input masking of the password and using appropriate controls to collect the input from the user such as dropdowns for selecting countries and textboxes to enter the input etc.

This layer also involves managing the visual layouts, styles and general appearance of the application. Also navigation from one page to other is being involved here. Formatting and displaying data in different visual styles. Browsing and organizing displayed data is also done in the presentation layer.

User interface process components: These elements control user interaction. Interface process components are not seen by the user but they perform vital supportive role to user interface components.

These are mainly used for managing the control flow through user interface components. Separating the conceptual user interaction flow from the implementation or device where it occurs. Keeping track of the progress of a user in a certain process. ASP.net pages contain the pages which are viewable by the user. They are responsible for separation for application logic and user interface. ASP.net pages have the extension as .aspx. These pages contain the web controls of the application and also the User Controls. User controls are customized controls which can be created specific to the application and are added to the .aspx pages.

Business Logic Layer:

This layer separates the business logic from other modules such as data access layer and user interface. This can be advantageous as the business logic of an application can often withstand modifications or replacements of the other two tiers. In detail, in our application we have properly separated business logic layer and data access layer; the data access layer could be rewritten to retrieve data from a different database, without affecting any of the business logic. This helped us while developing the application as we have split the layers and worked simultaneously.

The sample code written in the business logic is, if there is a Boolean value which displays it to the user as true and the value of it is stored in the database as 1, the conversion from the value 1 to Boolean value true is being done in this business logic layer. Also computing scores of the users are done in the business logic layer.

Data Access Layer:

This layer provides simplified access to data stored in persistent storage as in Oracle 11g database. Data access layer returns a reference to object complete with its attributes instead of row of fields from a database table. This helped the client modules to be created with higher level of abstraction.

This is handled by creating a class of data access methods that directly reference a corresponding set of database stored procedures. This helps to hide the complexity of the underlying data store from the external world.

Instead of using commands such as insert, delete and update to access a specific table in a database, a class and a few stored procedures could be created in the database. The procedures would be called from a method inside the class, which would return an object containing the requested values. The insert, delete and update commands are executed within simple functions like InsertnewUser etc. stored within the data access layer.

Business logic methods are mapped to the database access layer. Instead of making a query into a database to fetch all users from different tables the application can call a single method from a DAL which abstracts those database calls.

In FNPRS application, sabw_data is the data access layer which is a class file. It contains methods to all database queries.

Few of the methods are

- Get_counties: This method returns all the counties present in the Kansas. Counties are displayed according to the user id selected. Administrators and normal users have registered into certain counties by the administrators. These are managed in the User counties module.
- Get_Sabw_items: Every worksheet in the program is stored in the database with an id. Sabw_id is considered as an identity for every worksheet. When Sabw_id is given as input to get_sabw_items method it returns all the position details entered by the user such as hourly wages, annual salary, working hours etc.
- Insert_general_update: This method inserts the worksheet details into the database when the user enters program information. It inserts into the table SB_Data, worksheet title, county details and also the user who entered the details.
- Update_Worksheet: This method updates the SB_Data table whenever the user wants to modify the details previously entered. It can modify all the information such as county, worksheet title and the year which it has been entered.
- Insert_agent_update: This method inserts the position details which is one of the program assistant, nutrition assistant, student worker or graduate students. It enters FTE, weeks and Annual Salary of the agent. Agent represents the county who manages the data.

- **Insert_desc_update:** This method inserts the program description for the program such as Curriculum information, targeted audience, types of partnering agencies/organizations. Summary of the program is also entered in the program description table. User/Agent can modify these details accordingly.
- **Delete_agent:** This method deletes an agent if an agent is out of the program. This deletes from the positions table, the details entered for a particular agent such as FTE, annual salary etc.
- **Insert_justification:** This method inserts justification for the details entered by the user. Every detail entered by the user is given a justification to keep in track the amounts entered by the user for a program
- **Insert_travel:** This method enters the funds which are allocated for the travel purpose. These amounts are entered by the user and reports are also generated accordingly. Any changes needed in the program are made in assistance to the reports generated.

A database connection is created in all these methods, which has the below format.

```
OracleConnection myconnection = new
OracleConnection(ConfigurationSettings.AppSettings["connectionString"]);
```

ConnectionString is defined in the web.config file which contains the database connection information. It is defined by adding a key in the web.config file.

```
<add key="ConnectionString" value="Data Source=KSREDB2;Persist Security
Info=True;User ID=fnprs;Password=trlgg3r;Unicode=True"/>
```

Myconnection defined in the above statement acts as a variable for performing the database operations. OracleCommand is written to create a command that sends a command to the database.

```
OracleCommand oCmd = new
OracleCommand("PKG_SABW.insert_jus_supplies",myconnection);
```

This command calls the insert_jus_supplies stored procedure which is contained in the PKG_SABW package. oCmd is the variable that creates a command with the above declared connection.

```
myconnection.Open();
oCmd.CommandType = CommandType.StoredProcedure;
```

Myconnection.Open() opens the connection with the database specified in the web.config file. oCmd sends a command as stored procedure to the database. Stored procedures may have input and output parameters which has to be sent from the server.

```
OracleParameter parameterprogid= new
OracleParameter("sabw_id_in",System.Data.OracleClient.OracleType.VarChar,50);
parameterprogid.Direction = System.Data.ParameterDirection.Input;
parameterprogid.Value = sb_data_id;
oCmd.Parameters.Add(parameterprogid);
```

OracleParameter defines the parameter which has to be passed to the stored procedure. Input or output parameter can be specified by mentioning the direction for the parameter. Value defines the parameter value.

```
oCmd.ExecuteNonQuery();
```

ExecuteNonQuery is used for a query that does not return any rows(an update, delete or insert) but instead returns an integer indicating the number of rows affected by the query.

There are also ExecuteReader and ExecuteScalar methods. ExecuteReader is used when the query returns a reader instance that is connected to the resulting rowset within the database, allowing the rows to be retrieved. ExecuteScalar is used for query returning a single value, usually takes the first column of the first row of the resulting rowset, and discards all the other values. The DataTable class is used to retrieve rowsets or tables stored within Dataset class.

```
myconnection.Close();
```

The above statement closes the connection after the operation is performed.

All the methods defined in the data access layer is called from the business logic layer. In the business logic layer an instance of data access layer is created.

- `sabw_data Sabw_Data = new sabw_data();`

Using this instance, all the methods mentioned above is being called from the data access layer in the business layer.

Sample method:

```
public DataTable insert_jus_supplies(string sb_data_id, string
JUSTIFICATION_SUPPLIES)
{
    return sabw_data.insert_jus_supplies(sb_data_id,
JUSTIFICATION_SUPPLIES);
}
```

This mentions that it returns the DataTable which means all the values retrieved from the database are in the form of datatable. It reads the datatable and displays it to the user which can be modified accordingly in the Presentation Layer.

All the parameters are sent through the business logic layer to the data layer from which it talks to the database and performs the functions accordingly.

User Controls:

The extension of user controls are .ascx pages. In our application we developed few features using user controls. User controls are easily added to the web pages. Main usage of the user control is to modularize different functionalities by using simple links. This finally results to the upgrading of performance. Also makes the code manageable. User control can work as widgets and can be made completely independent of the parent page. Speeds up the development process- development is modularized which means you have different functionalities available by using just a simple tag in your web page.

Code for including the user control:

```
<%@ Register TagPrefix="uc1" TagName="budget_sheet"
Src="usercontrols/budget_sheet.ascx" %>
```

Register tag registers the usercontrol budget_sheet on to the aspx page.

```
<uc1:top_panel id="Top_panel1" runat="server"></uc1:top_panel>
```

The above tag places the user control on to the specified div in the aspx page.

Database:

As mentioned earlier, we have used Oracle 11g database. It integrates well with visual studio and provides good performance. Tables are physical storage for the data stored in the database. All the operations utilizes database via stored procedures.

Stored Procedures are used to manipulate or operate data between tables. For example, if there are three different queries we need to operate on three different tables, they can be accommodated in one single stored procedure. Stored procedures are always advantageous over in line queries as stored procedures are compiled when created, but when a query is used, there is a database hit, which may degrade the performance.

And also because of modularity issues, if a same query has to be used many number of times in the code, a stored procedure can be used. If there is an error in the query or any change has to be made, then the fix has to be done in number of places and also the code has to be deployed which may use lot of resources. Instead if we are using a stored procedure, we can overcome the above issues.

Stored procedures also provide a measure of security as all parameters passed to a stored procedure are type checked. Stored procedures in Oracle are saved in packages and package bodies.

Structure of the database:

Tables used:

- **SB_DATA:** This table stores the details of the worksheet. Sb_data_id is an auto identity column considered as primary key. It is created with the trigger before insert on the sb-data table. It saves the county and year in which the program has been created. It also saves the title of the worksheet.
- **SB_DATA_ITEMS:** This table stores all the budget details of the program. Item_Id is the primary key which is an auto identity column. It has a foreign key constraint for the

sb_data database on sb_data_id column. Match_Type column in the table represents the FNP of county.

- **SB_POSITIONS_LIST:** This table saves all the positions of the agents entered for every worksheet. The Name and position details are stored in this table. Id is the foreign key constraint with the sb_data table.
- **SB_PROPOSED_EQUIPMENT:** This table contains the equipment information entered by the agent for every program. The estimated equipment is entered and also reports are generated on this table. Equip_id is the primary key and Sb_data_id is the foreign key relation with sb_data table.
- **SB_SURVEY_QUESTIONS:** This table contains the questions for the survey which are managed by the program officials. They can be manipulated by the program administrators.
- **SB_SURVEY_RESPONSES:** This table saves the responses entered by the agents for the questions provided in the sb_survey_questions table. It has foreign key relationship with both sb_data table and sb_survey_questions table, which means that the questions from sb_survey_questions table are answered for every worksheet.
- **PAS_COUNTIES:** This table contains all the counties in the kansas. This table can be managed by administrators who have the privileges for adding and deleting the counties. This table is used by sb_data to get the county text.

Stored Procedures used:

- **Get_sabw:** It returns the list of worksheets with their ids, modified dates, modified user and in which county the program is taking place.
- **Get_counties:** It returns the list of counties stored in the table for kansas state.
- **Get_sabw_items:** It returns the list of positions entered for every worksheet/program. It also returns the details for the positions entered by the user/agent.
- **Update_title:** It updates the title for the worksheet in the sb_data table.
- **Insert_justification:** Inserts justification entered by the user. This stored procedure is used to store all the justification details entered by the user for every amount entered in the database.

3.3 Modules

As already mentioned, Kansas state Research and Extension group runs a nutrition program in order to enhance the knowledge on food and nutrition for all ages of people. They need to handle the budget expenses, travel and equipment needed for the program. As this program is delivered in all the counties across Kansas, there is a need for them to maintain a record for their expenses. The solution for it is creating a web interface in order to enter all these details and save them in the database for future use.

3.3.1 Intended Users

Agents or employees from the Kansas State Research and Extension group who are responsible to arrange the program and manage the resources required for the program. Every county is assigned an agent who manages the program for the particular county.

3.3.2 Requirements

The website design is targeted towards the employees working for Kansas State Research team. Main purpose is to provide agents an easy, simple and efficient manner of managing data related to the program and calculating the budgets required for the program. The user interface developed for this website help the agents to enter information based on the county and year.

3.3.3 Inputs and Outputs:

Input from the user: The user has to select a year from the dropdown list provided. Year should be selected between 2008 to 2013

Expected output: The output for the given input is the list of records which are displayed based on the year selected from the dropdown list.

The list is displayed with following columns:

- County
- Title
- Option to view, edit and delete.

Input: User has to click on view link

Expected output: It redirects to the View sheet page. It displays the details entered previously by the user. The list displayed in accordion.

Input: Click on edit link

Expected Output: It displays all the text boxes and text fields filled with the information for the selected proposal. The range of the output should be only the values entered by the user previously.

Input: Click on delete

Expected Output: It deletes the selected proposal and displays the rest of the records.

Input: In the positions screen, Annual salary has some constraints on its value. Select Student worker from radio button list.

Expected output: Percentage of benefits has default value 1.3. It is displayed in the text box associated with percentage of benefits.

Input: In the positions screen, Annual salary has some constraints on its value. Select Program assistant from radio button list.

Expected output: Percentage of benefits should be between 9.3 and 35. It displays an error message when the value outside the given range is entered.

Input: In the positions screen, Annual salary has some constraints on its value. Select Nutrition assistant from radio button list.

Expected output: Percentage of benefits should be between 9.3 and 43.8. It displays an error message when the value outside the given range is entered.

Input: In the positions screen, Annual salary has some constraints on its value. Select Graduate students from radio button list.

Expected output: Percentage of benefits has default value 5.6. It is displayed in the text box associated with percentage of benefits.

Input: In the positions screen, FTE should be entered in certain format.

Expected output: FTE textbox displays an error message if entered in different format other than specified format.

The requirements mentioned by the KSRE team is being divided into modules and developed accordingly. They are described as follows:

1. Add Worksheets: This module is responsible for enabling the user to add new worksheets for a particular county in the database after selecting the year. In this module, the user is allowed to enter a worksheet with new details such as position description, Justification Salary Benefits, Contractual Services, In State Travel, Supplies, Proposed Equipment, etc. The administrator can also take various kinds of surveys on this page.

2. View Worksheets: This module displays all the information that has been entered through worksheet module for a particular user and a particular county. jQuery is used to display the information using accordion.

3. Edit Worksheets: This module allows a logged in user to edit the details entered via worksheet module. Similar to worksheet module, the user is allowed to take various kinds of surveys in this module as well. All these components are developed using user controls.

4. User Management: This module is used to added new users to a particular county in the database.

5. User Counties: This module allows the administrator to add new counties to the database

6. PAS: This module allows the administrator to add new Program Activity Series for a particular month in the database.

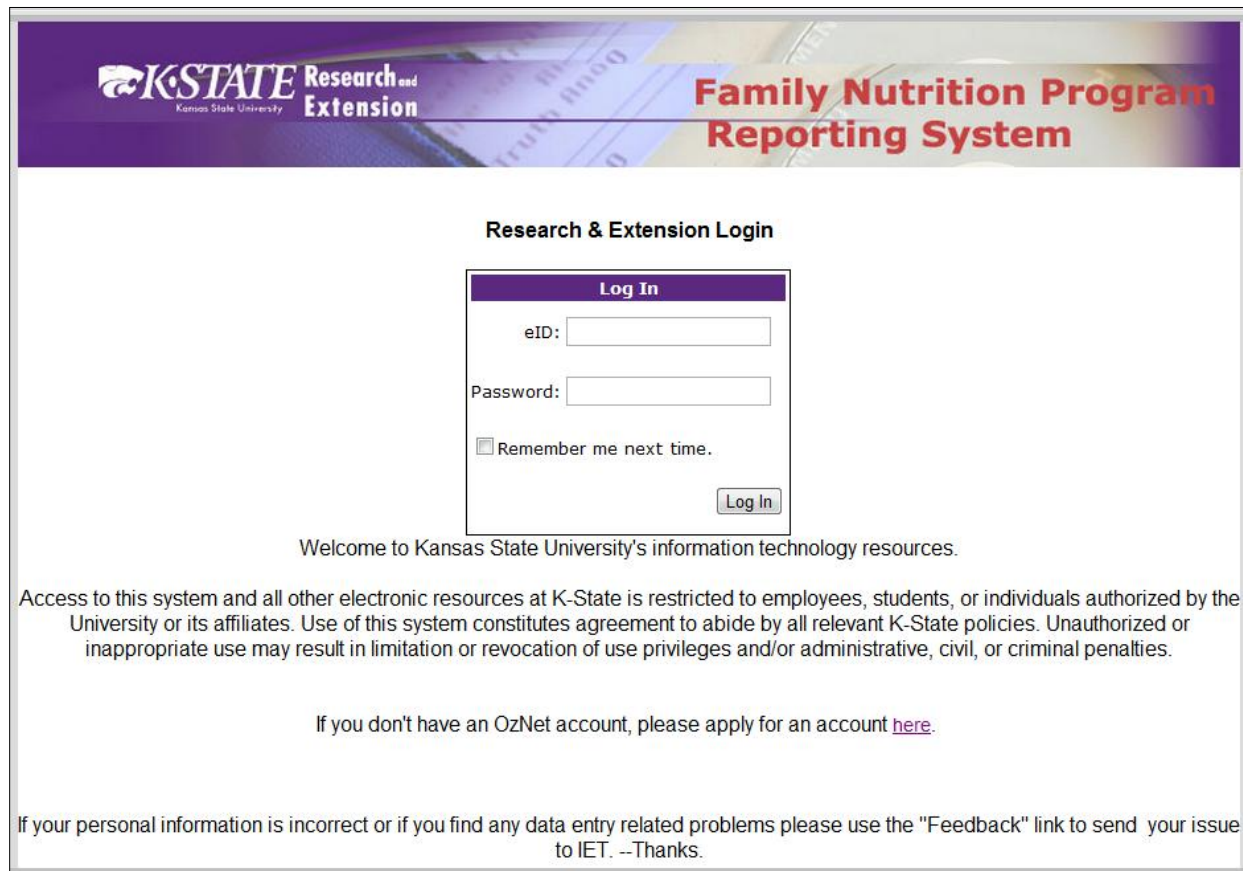
7. Report Home: In this module, all the data that has been enter through the previous modules is split into several short reports and various calculations and estimations are made for all the users at a county. Apart from that reports can also be exported to Microsoft Excel Sheets and these reports can also be printed.

This module enables the administrator to view various kinds of reports such as:

Agent Information, FNP Funds Contractual Services, FNP & County In-State Travel, FNP Funds Supplies, County Match Contractual Services, County Match Supplies, FNP Funds Totals, County Match Total, Volunteer Match & Cash Donations, Proposed Equipment Information.

3.4 Screenshots

1. Login Page: Every user accessing the website has a Kansas eid and password. They use their login id and password to enter into the system. It displays an error message if the login id or password is entered wrong. Once the login is validated, it redirects to the page where all the proposals are displayed.



The screenshot shows the login page for the Family Nutrition Program Reporting System. At the top, there is a purple banner with the K-State Research and Extension logo on the left and the text "Family Nutrition Program Reporting System" in red on the right. Below the banner, the page is titled "Research & Extension Login". In the center, there is a "Log In" box with a purple header. Inside the box, there are two input fields: "eID:" and "Password:". Below these fields is a checkbox labeled "Remember me next time." and a "Log In" button. Below the login box, there is a welcome message: "Welcome to Kansas State University's information technology resources." followed by a paragraph of terms and conditions: "Access to this system and all other electronic resources at K-State is restricted to employees, students, or individuals authorized by the University or its affiliates. Use of this system constitutes agreement to abide by all relevant K-State policies. Unauthorized or inappropriate use may result in limitation or revocation of use privileges and/or administrative, civil, or criminal penalties." Below this, there is a link: "If you don't have an OzNet account, please apply for an account [here](#)." At the bottom, there is a feedback message: "If your personal information is incorrect or if you find any data entry related problems please use the 'Feedback' link to send your issue to IET. --Thanks."

2. Main page: This page displays the proposal for a certain year. There is a drop down for all the possible years. Once the user selects the year, the proposals are displayed which were conducted in the selected year. There is an add worksheet button, when clicking on the button it redirects to an add proposal page. The styles for the links and buttons are given using jquery class. For eg: the class used to style worksheet button is:

```
class="ui-state-default ui-corner-all ui-button-text-only"
```

KSRE- Online FNP Proposal Home PAS Home User Counties Report Home	2012-2013 <input type="button" value="Add Worksheet"/>				
	County	Title	View	Edit	Delete
	Ford	Ford County 2013	View	Edit	Delete
	Finney - Addison	Finney County	View	Edit	Delete
	Harvey	FNP Grant	View	Edit	Delete
	River Valley District	Family Nutriton Program Grant	View	Edit	Delete
	Pottawatomie	FNP Proposal	View	Edit	Delete
	Miami	Miami FNP	View	Edit	Delete
	Meade	Family Nutrition Program	View	Edit	Delete
	Walnut Creek	Walnut Creek District FNP	View	Edit	Delete
	Johnson	Johnson County FY13	View	Edit	Delete
	McPherson	McPherson Co FNP	View	Edit	Delete
	Sedgwick	SEDGWICK CO	View	Edit	Delete
	Southwind	FNP Proposal	View	Edit	Delete
	Frontier	Frontier Extension District	View	Edit	Delete
	Ellis	Ellis County FNP Application	View	Edit	Delete
	Grant	FNP Proposal-Grant County	View	Edit	Delete
	Linn	Linn County	View	Edit	Delete
	Lyon	Lyon County FNP	View	Edit	Delete
	Kickapoo Nation	FNP Proposal	View	Edit	Delete
	Douglas	Douglas Co FNP	View	Edit	Delete
	Cherokee	FNP Prposal 2013	View	Edit	Delete
	Meadowlark District	Adult Audience Surveys	View	Edit	Delete
	Wichita	Wichita County FNP	View	Edit	Delete
	Barber	Barber County	View	Edit	Delete
	Anderson	Anderson County	View	Edit	Delete
	NW Area	Northwest Area	View	Edit	Delete
	Demo	sdfds	View	Edit	Delete

3. Add Worksheet: Proposals are categorized according to the year and county. They are recognized with the title. Hence before adding a worksheet, title, year and county are entered by the user. Click Save and continue in order to load the next user control.

General Information	
Please enter the following information.	
Worksheet Title	<input type="text"/>
Fiscal Year	2012-2013 <input type="button" value="v"/>
County/District	4-H <input type="button" value="v"/>
Save & Continue	
Back to FNP Proposal Application Home	

Once the proposal worksheet is added, the next crucial part of the proposal is to add agents or employees who manage the program. The salaries and benefits given to the agents are described. The agents are categories into four categories which are displayed as a radio button list. One of the radio button list is selected. Every category has a default percentage of benefits value that is displayed in the textbox. There are some required formats for the numbers to be entered which are displayed beside the textbox.

[ENP Proposal Home](#) | [View All](#) | [View Report](#) | [View paid employee Worksheet](#) | [View Budget](#) | [View Budget Narrative](#) |

County/District: Demo

Position

[Add Position](#)

☐ Program Assistant ☐ Nutrition Assistant
☒ Student Worker ☐ Graduate Students

Name:

FTE: (Required Format: 0.##)

Weeks:

Hourly Wage:

Percentage Of Benefits: (Required Format: xx.x)

[Save](#) [Continue](#)

[Back to FNP Proposal Application Home](#)

After all the details are entered, click on save. It displays in a grid view in the below format. These positions can be edited and deleted according to the future changes.

Position									
Name	FTE	Hours	Weeks	Hourly Wage	Benefits	Annual Salary	Type	Edit	Delete
Toni Bryant	0.40	16	4	04.00	03.33	8,320.00	SW	Edit	Delete

Every number entered about the agents' salary and their benefits should be given a justification. Hence there is a text field to enter the justification for salaries and benefits.

[ENP Proposal Home](#) | [View All](#) | [View Report](#) | [View paid employee Worksheet](#) | [View Budget](#) | [View Budget Narrative](#) |

County/District: Demo

Justification Salaries and Benefits

[Save & Continue](#)

[Back to FNP Proposal Application Home](#)

Clicking save and continue on the previous screen allows you to redirect to the contractual services. The contractual services include advertising for hire, consultant fees, and amount for duplicating and printing and below displayed services. Each textbox is allotted for subscribing all the amounts from the user. The total contractual services are calculated in the backend code. Justification should be given for the amounts entered. There is an error message displayed if the user does not enter justification and try to continue to next screen.

FNP Proposal Home View All View Report View paid employee Worksheet View Budget View Budget Narrative	
County/District: Demo	
Contractual Services	
Description	FNP Funds(\$)
Advertising for Hire	<input type="text"/>
Consultant	<input type="text"/>
Duplicating and Printing	<input type="text"/>
Interpreter/Translator	<input type="text"/>
Phone	<input type="text"/>
Postage	<input type="text"/>
Priority Support for Food Processor (Continuing Service)	<input type="text"/>
Space	<input type="text"/>
Total Contractual Services	0.00
Justification Contractual Services	
Enter Contractual Services Justification	
Save & Continue	

Click on save and continue in the Contractual services redirects to In-State Travel page. It collects the funds required for the In-state travel. Justification is mandatory for every amount entered. It displays an error message if user tries to continue without entering the Justification In-state travel.

FNP Proposal Home View All View Report View paid employee Worksheet View Budget View Budget Narrative	
County/District: Demo	
In-State Travel	
Description	FNP Funds(\$)
In-State Travel	<input type="text"/>
Total Travel	0.00
Justification Instate Travel	
<div style="border: 1px solid black; height: 100px; width: 100%;"></div>	
Save & Continue	
Back to FNP Proposal Application Home	

The below screen is the space for collecting Supplies information. The Supplies information includes Food Demonstration Supplies, Nutrition education Supplies, Office Supplies and expenses related to Small equipment and software. Every amount is entered in a text box which is then converted from string to double value. The value for total supplies is calculated from the above entered values internally and displayed when user tries to edit the proposal.

The value entered in the Small equipment and software is carried to the next page using a query string in order to limit the cost of equipment for the program.

FNP Proposal Home View All View Report View paid employee Worksheet View Budget View Budget_Narrative	
County/District: Demo	
Supplies	
Description	FNP Funds(\$)
Food Demonstration Supplies	<input type="text"/>
Nutrition Education Supplies	<input type="text"/>
Office Supplies	<input type="text"/>
Small Equipment and Software(<\$4999)	<input type="text"/>
Total Supplies	0.00
Justification Supplies	
Save & Continue	

The proposed equipment should be entered with the equipment used, cost of the equipment and justification. Once entered click on add equipment. There is a constraint here that the equipment should be equal to the amount entered in the Small equipment and software. When all the equipment and their costs are entered, their summed up value should be equal to that amount which is displayed as red in the page. The amount should not be greater or lesser than the given amount; else it shows an error message.

FNP Proposal Home View All View Report View paid employee Worksheet View Budget View Budget_Narrative	
County/District: Demo	
Proposed Equipment < \$4999	
Total Costs of all proposed equipment should be \$500	
Equipment	Cost
Justification	
<input type="text"/>	\$ <input type="text"/>
<input type="text"/>	
Add Equipment	
Continue	
Back to FNP Proposal Application Home	

After clicking on add equipment link it adds and display in a grid view in the below format. If the total equipment costs of all the equipment entered is greater than the given cost, user can edit or delete certain records of the displayed equipment. Editing the record gives the details while reading the identity of the equipment and allows the user to edit the information.

FNP Proposal Home	View All	View Report	View paid employee Worksheet	View Budget	View Budget_Narrative
County/District: Demo					
Proposed Equipment < \$4999					
Total Costs of all proposed equipment should be \$500					
Equipment	Cost(\$)	Justification			
Equipment	500	dfds		Edit	Delete
Equipment	Cost	Justification			
<input type="text"/>	\$ <input type="text"/>	<input type="text"/>		Add Equipment	
Continue					
Back to FNP Proposal Application Home					

Cash donations can be made in the program. There is a textbox to enter the cash donation. It is not mandatory and it does not count in the budget expenses while creating a report for the budget.

FNP Proposal Home	View All	View Report	View paid employee Worksheet	View Budget	View Budget_Narrative
County/District: Demo					
Cash Donations					
Cash Donations	<input type="text"/>				
Save & Continue					
Back to FNP Proposal Application Home					

CHAPTER 4: EVALUATION

Software testing is the process of executing a program to find the errors in the application. It is nothing but validating and verifying that a software program is working as expected and is meeting the technical as well as business requirements. This can be implemented at any time in the development process. There are many types of testing software ie. Unit testing, black box testing, performance testing, stress testing, regression testing etc. Unit testing and load testing are performed for FNPRS application.

4.1. Unit Testing

Unit testing refers to those tests that verify the functionality of a specific section of the code, usually at the function level. In an object oriented environment this is usually at the class level and the unit tests include the constructors and the destructors. The primary goal is to take the smallest piece of testable software, isolate it from the rest of the code and check if it is behaving exactly as expected. Each unit is tested separately before integrating them into modules to test the interfaces between modules.

This application has been tested individually for each view and made sure that each module is generating the desired output for the corresponding input. There are various modules involved in the application like viewing worksheet, editing worksheet, retrieving the profile information from the database etc.

Test cases and Results:

Each test case tests the functionality of a particular function. Following are the test cases that were designed and executed to check the functionality of the application.

	Test Module	Input	Test Case	Expected Result	Test Result
1	Login	Enter login id and password	Enter wrong login id and password	An error message should be displayed	Passed
2	Home Page	Select year from the	When a year is selected from the dropdown it	It displays number of worksheets for	Passed

		dropdown	should display all the worksheets.	the selected year.	
3	Home Page	Click on Add worksheet	It should redirect to the Add worksheet page.	It redirects to the page where user can enter worksheet information	Passed
4	Add Proposal Page	Click on Save and Continue	Do not enter the worksheet title and click on save and continue	An error message is displayed “Please enter a title for this worksheet”	Passed
5	Add Proposal Page	Enter title and county	Enter all the required fields and it should redirect to the Add positions page.	It redirects to the positions page	Passed
6	Add Position	Select one from the radio button list	Enter all the details in the position page, it should save the position details	It adds the position with the given details and display in the grid view	Passed
7	Justification salaries and benefits	Enter the justification	Enter justification for salaries and benefits and click the save button	It saves into the database and redirects to enter the contractual services	Passed
8	Contractual Services	Enter the contractual services	Enter contractual services and click the save button	It saves into the database and redirect to in-state travel	Passed
9	In sate travel	Enter Instate	Enter instate travel and	It saves into the	Passed

		travel details	justification for the travel funds. Click on the save button to continue to the next screen	database and redirects to enter the Supplies information	
10	Proposed Equipment	Enter Equipment	Enter equipment greater than the amount entered in the Small equipment and software field	It displays an error that the cost should be less than the entered amount for Small equipment and software field.	Passed

Table 1: Test cases and results

Unit testing for the database methods is also done, in order to check if the input passed to the method is proper or not. It also performs test on the output returned by the method. To perform this testing this application uses Visual Studio Unit testing feature which runs the tests and checks the failure condition.

Below is the sample test method.

```

/// <summary>
    ///A test for GET_TRAVEL_INFO
    ///</summary>
    // TODO: Ensure that the UrlToTest attribute specifies a URL to an
ASP.NET page (for example,
    // http://.../Default.aspx). This is necessary for the unit test to
be executed on the web server,
    // whether you are testing a page, web service, or a WCF service.
    [TestMethod()]
    [HostType("ASP.NET")]

    [AspNetDevelopmentServerHost("C:\\Sravya_Project\\sabw_040912_TestProject",
"/sabw_040912_TestProject")]
    [UrlToTest("http://localhost/sabw_040912_TestProject/sabw_add.aspx")]
    public void GET_TRAVEL_INFOTest()
    {
        report_data_Accessor target = new report_data_Accessor(); //
TODO: Initialize to an appropriate value
        string year = "2012-2013"; // TODO: Initialize to an appropriate
value
        int expected = 40; // TODO: Initialize to an appropriate value
        DataSet actual;

```

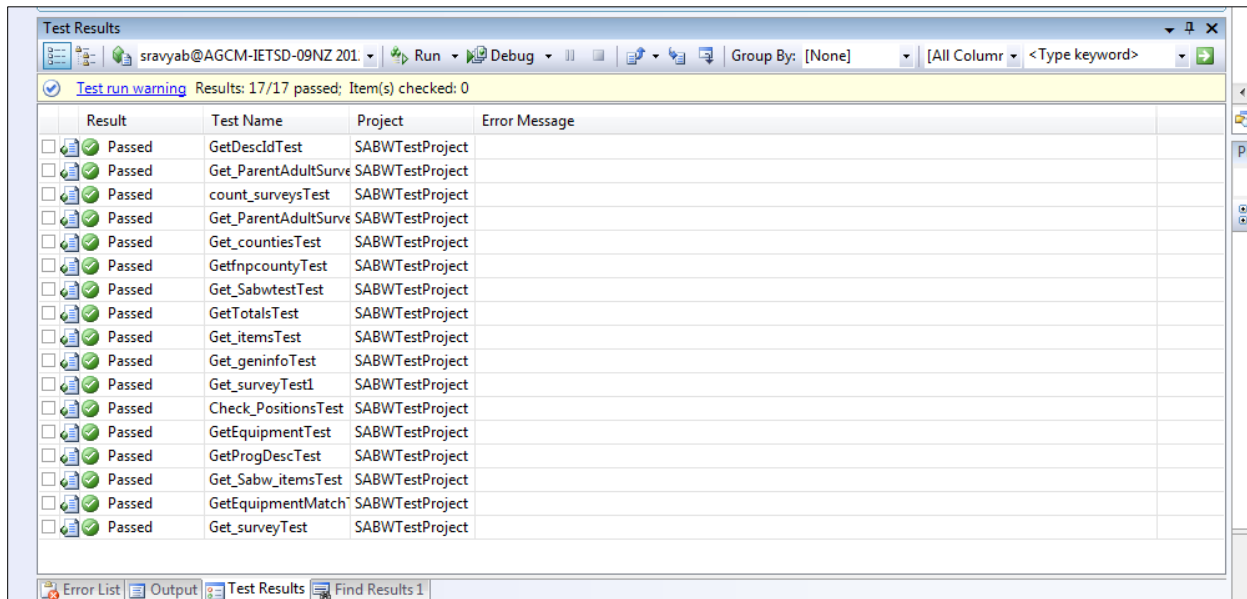
```

        actual = target.GET_TRAVEL_INFO(year);
        Assert.AreEqual(expected, actual.Tables[0].Rows.Count);
    }

```

The above test method tests the actual `get_travel_info` method by passing the year as input to the method. It returns the dataset of records for the given year. The test is performed to check if the number of records is equal to the expected records. If they are equal the test is passed or failed if they are not equal.

Below are the test results for the unit test cases.



Result	Test Name	Project	Error Message
Passed	GetDescIdTest	SABWTestProject	
Passed	Get_ParentAdultSurve	SABWTestProject	
Passed	count_surveysTest	SABWTestProject	
Passed	Get_ParentAdultSurve	SABWTestProject	
Passed	Get_countiesTest	SABWTestProject	
Passed	GetfnpcountyTest	SABWTestProject	
Passed	Get_SabwtestTest	SABWTestProject	
Passed	GetTotalsTest	SABWTestProject	
Passed	Get_itemsTest	SABWTestProject	
Passed	Get_geninfoTest	SABWTestProject	
Passed	Get_surveyTest1	SABWTestProject	
Passed	Check_PositionsTest	SABWTestProject	
Passed	GetEquipmentTest	SABWTestProject	
Passed	GetProgDescTest	SABWTestProject	
Passed	Get_Sabw_itemsTest	SABWTestProject	
Passed	GetEquipmentMatch	SABWTestProject	
Passed	Get_surveyTest	SABWTestProject	

Result	Test Name	Project	Error Message
Passed	GET_PTTTest	SABWTestProject	
Passed	GET_POSTTest	SABWTestProject	
Passed	GET_TRAVEL_INFOTe	SABWTestProject	
Passed	GET_SUPPLIESCA_INF	SABWTestProject	
Passed	GET_CANAME_TOTA	SABWTestProject	
Passed	GET_ADVTest	SABWTestProject	
Passed	GET_IDCTest	SABWTestProject	
Passed	get_parent_SurveyTes	SABWTestProject	
Passed	GET_TATest	SABWTestProject	
Passed	GET_CPCTest	SABWTestProject	
Passed	GET_HWB_CATest	SABWTestProject	
Passed	GET_DCTest	SABWTestProject	
Passed	GET_COUNTY_TOTAL	SABWTestProject	
Passed	GET_APCTest	SABWTestProject	
Passed	Get_CountyCSTest	SABWTestProject	
Passed	GET_CA_TOTALTest	SABWTestProject	
Passed	GET_COUNTY_HRSPT	SABWTestProject	
Passed	GET_SURVEY_TELL_US	SABWTestProject	
Passed	Get_FNPCSTest	SABWTestProject	
Passed	Get_Agent_InfoTest	SABWTestProject	
Passed	GET_NONMATCHTes	SABWTestProject	
Passed	GET_SUPPLIESCOUNT	SABWTestProject	
Passed	GET_ADV_TELL_USte	SABWTestProject	
Passed	GET_Partnering_Agen	SABWTestProject	
Passed	GET_EQUIPMENTTest	SABWTestProject	
Passed	GET_SUPPLIESFNP_IN	SABWTestProject	
Passed	GET_FNP_TOTALTest	SABWTestProject	
Passed	get_adult_SurveyTest	SABWTestProject	
Passed	GET_CSCA_INFOTest	SABWTestProject	
Passed	get_Survey_Questions	SABWTestProject	

4.2 Performance Testing

Performance test is used to determine the effectiveness of the application. Performance of the application effects when number of users are accessing the website. Hence performance test determines how fast some aspect of the system performs under a particular workload.

The performance of a website comes into picture when some hundreds or thousands of users are accessing the website and performance testing ensures that all the users are getting efficient results in less time. By performance testing we can estimate the maximum number of users accessing the web site simultaneously and by means of these testing results one can analyze the measures to further improve the performance of the application. The tool for performance testing and the testing results are described below.

4.2.1 Tool for performance testing

The website has been tested for its performance by the tool Apache JMeter, “which are open source software and a pure java based application. JMeter has been designed to load the test functional behavior and measure performance. This can be used to test performance on static and dynamic resources such as files, databases and queries and more. It can also be used to simulate heavy load on a server, network or to test the strength or to analyze the overall performance under different load types.

4.2.2 System Configuration

The details of the configuration of the system which has been used for testing the application are listed below:

Operating System	Windows 7 – 64 bit
RAM	4 GB
Processor	Intel core i7
Processor Speed	2.53 GHz

Table 2: System configuration

The below listed tests are conducted on the system with the above mentioned configuration.

4.2.3 Test plans Results and Evaluation

The results would show the response times as well as throughput, when increasing the number of users accessing the application at the same time.

This test considers the pages which use less database hits in the page and compared results with the pages which have more database content to be retrieved.

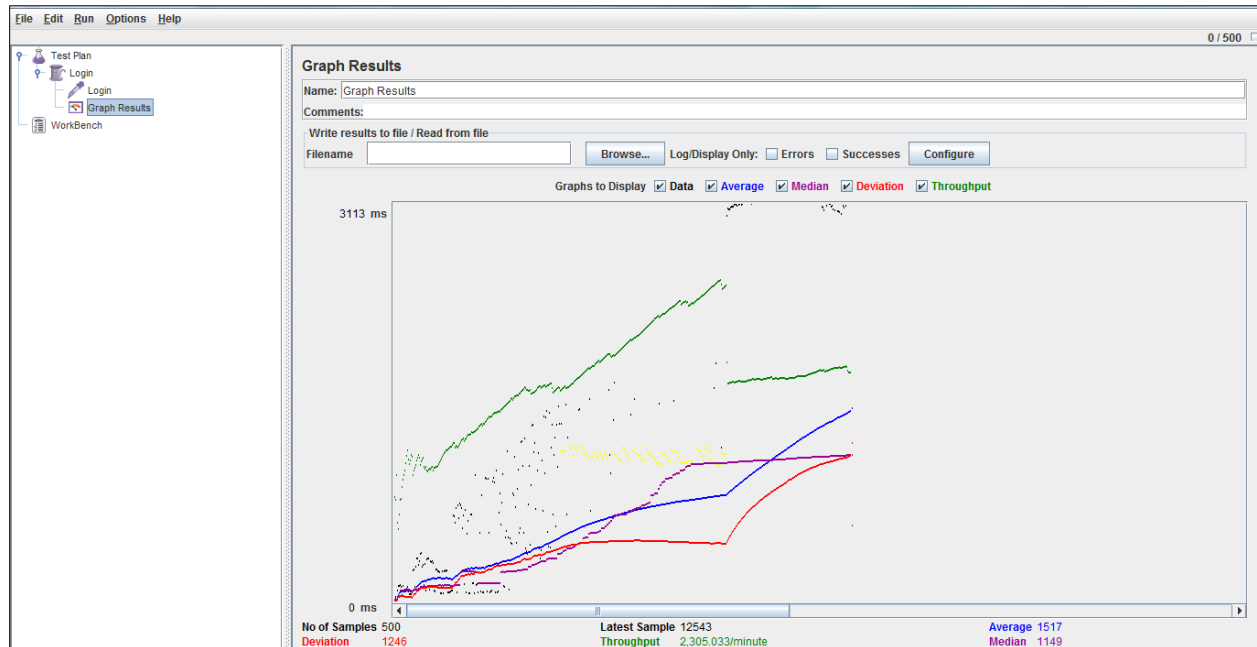
Test 1: JMeter test for Login page:

The Login page is tested to analyze its behavior and to get the idea about the performance of the application. The analysis would show the through put as well as response times for the web page when number of users is accessing it. This test plan is also used to get an approximate estimation of maximum number of users who can access it at the same time.

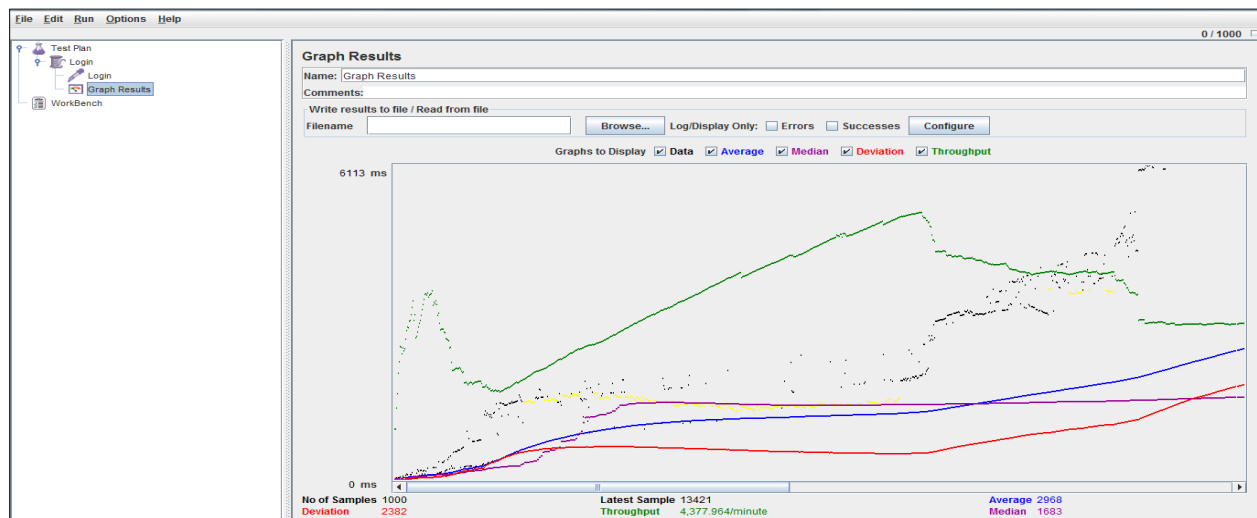
The number of users for the FNPRS application is around 1500 users. There are 1500 user ids entered in the database. On daily basis the users using the application may vary around

500. But considering the worst case, that is all the users using the application at single time the below threads are used to test the system.

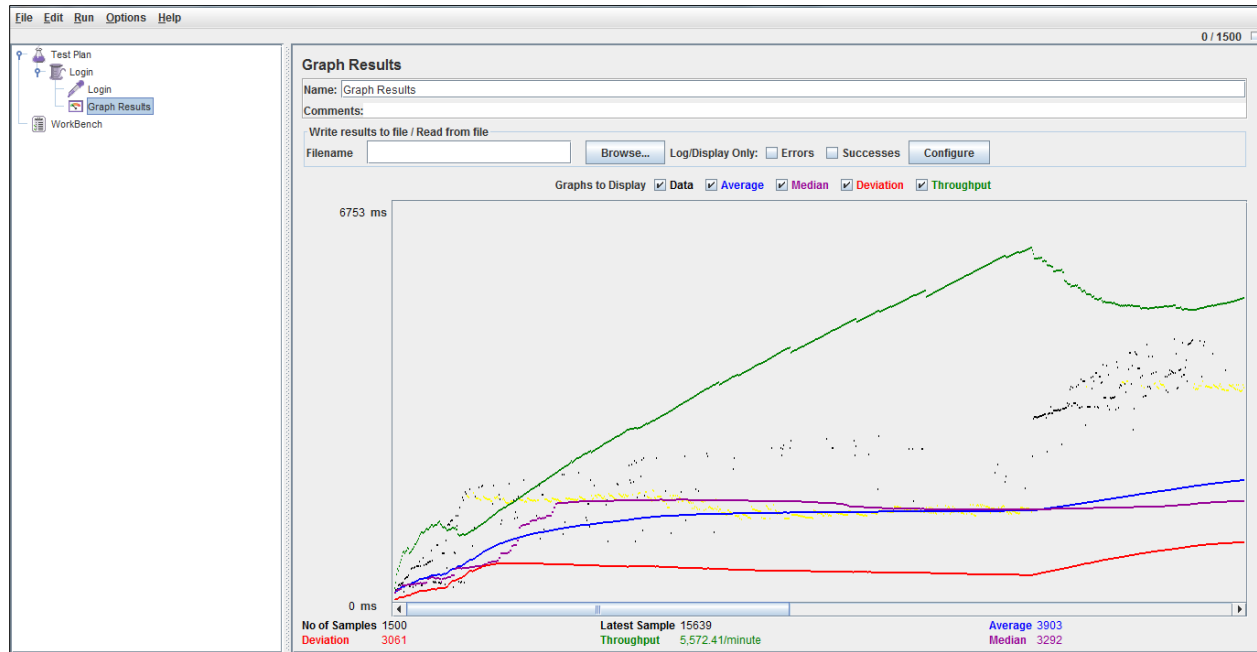
In order to test the Login page, it started with a total of 500 threads, simulating 500 simultaneous users, looping single time, with a ramp period of 1 sec, with a total of 500 requests.



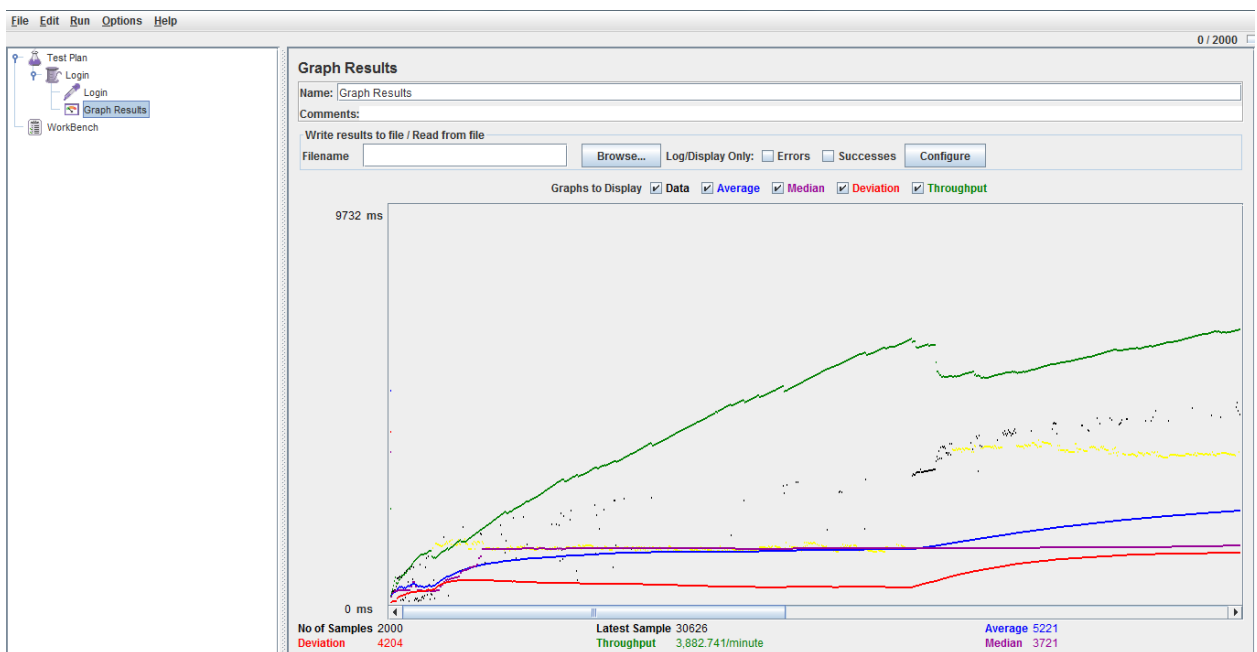
Then, increased the number of threads to 1000, which generates 1000 requests at a time. Below are the results for 1000 simultaneous users.



Generating the above test plan for 1500 users:



Generating the above test plan for 2000 users:



From the above graph, it can be seen that the throughput is stable for 500, 1000, 1500 users but the graph is not stable for 2000 users. Hence FNPRS application supports around 1500 users. The throughput is around 5000, which indicates that the website can handle 5000 requests per minute.

Bottle Neck for Login page: In this page it does not retrieve any information from the database. To consider it is kind of static page which only takes input from the user and sends to the database to check the login information.

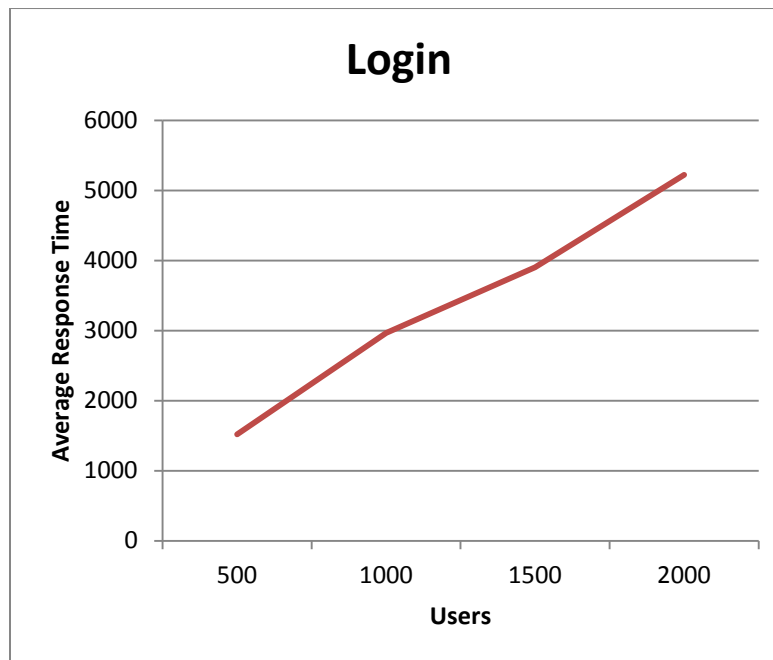


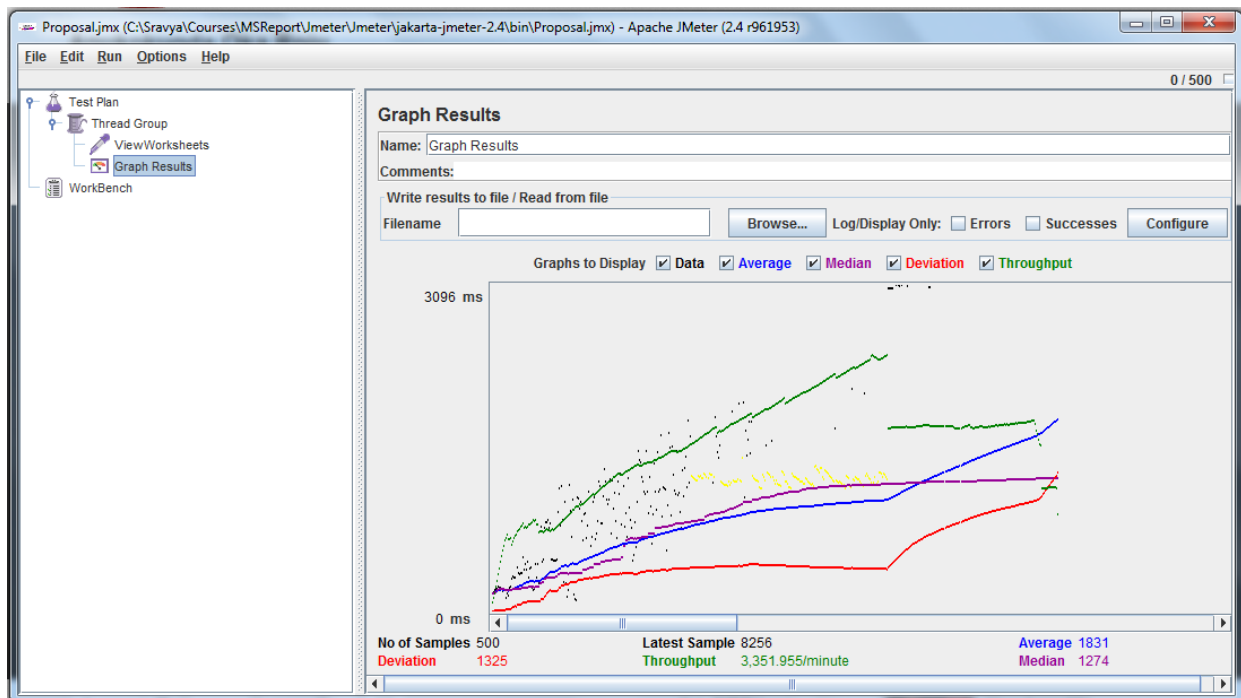
Figure 4: Average Response Time – Login Page

A graph is plotted with users and their response times for login page. The application is stable till around 1500 users. The average Response time is not stable after 1500 users. The throughput is decreased when the number of users increases, which is inversely proportional.

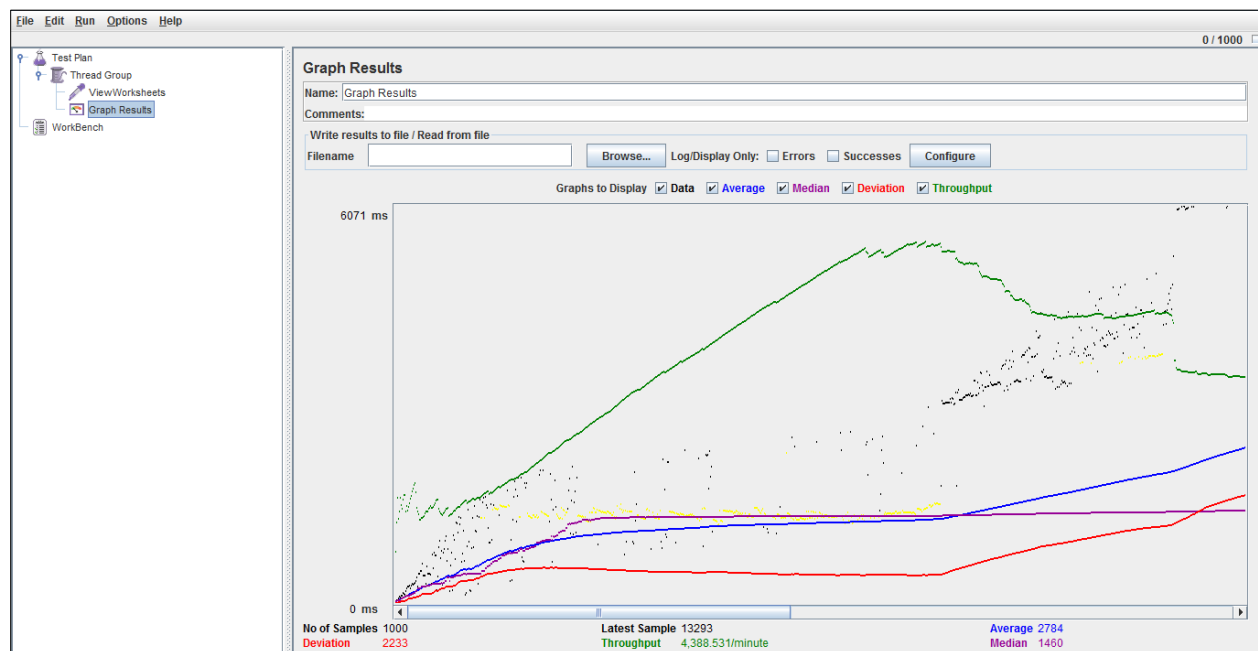
Test 2: JMeter test for Main page:

In order to test the Main page, this test uses a total of 500 threads, simulating 500 simultaneous users, looping single time, with a ramp period of 1 sec, with a total of 500 requests.

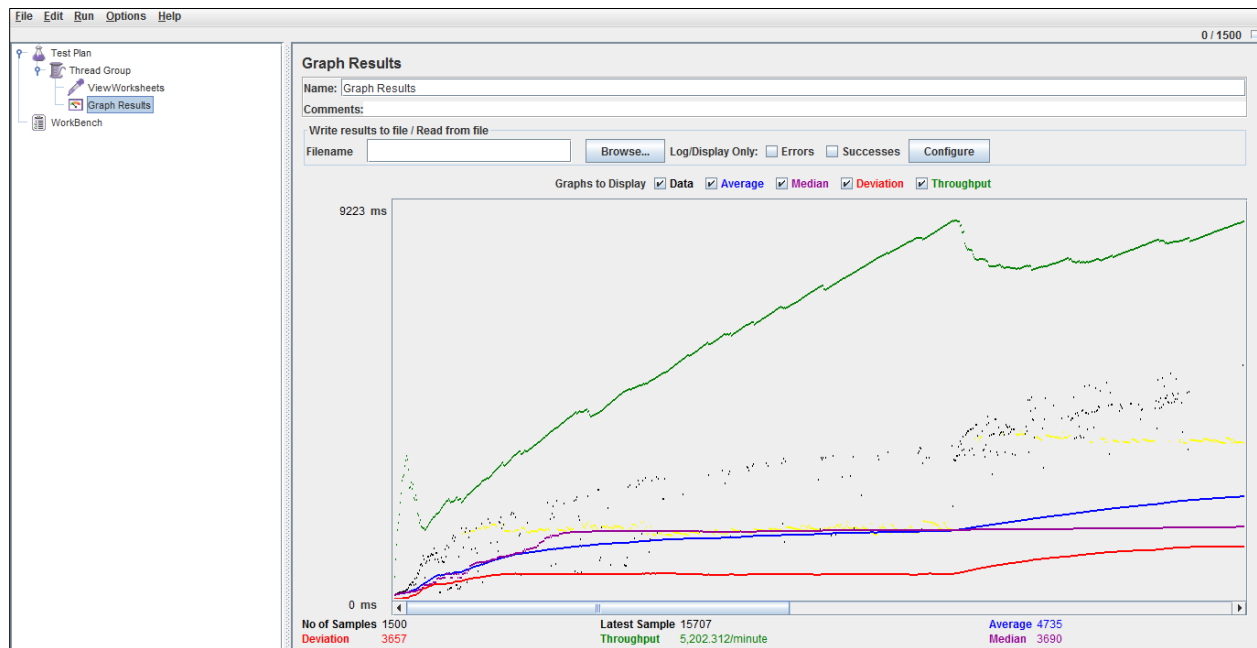
Below are the results of the above test plan.



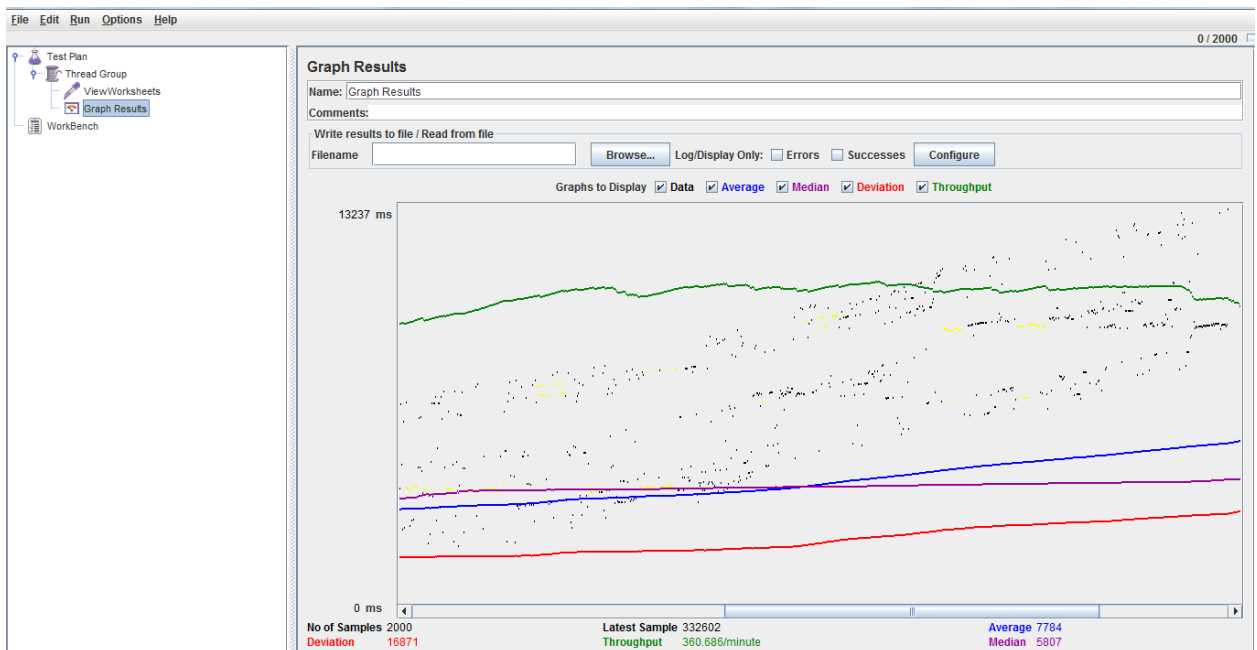
Then, increased the number of threads to 1000, which generates 1000 requests at a time. Below are the results for 1000 simultaneous users.



Generating the above test plan for 1500 users:



Generating the above test plan for 2000 users:



Bottle Neck for Main page:

- The information retrieval from the database is one hindrance for the less number of requests per minute and more average time
- CPU Utilization can be considered as a bottle neck when we have more than 5000 requests per minute.

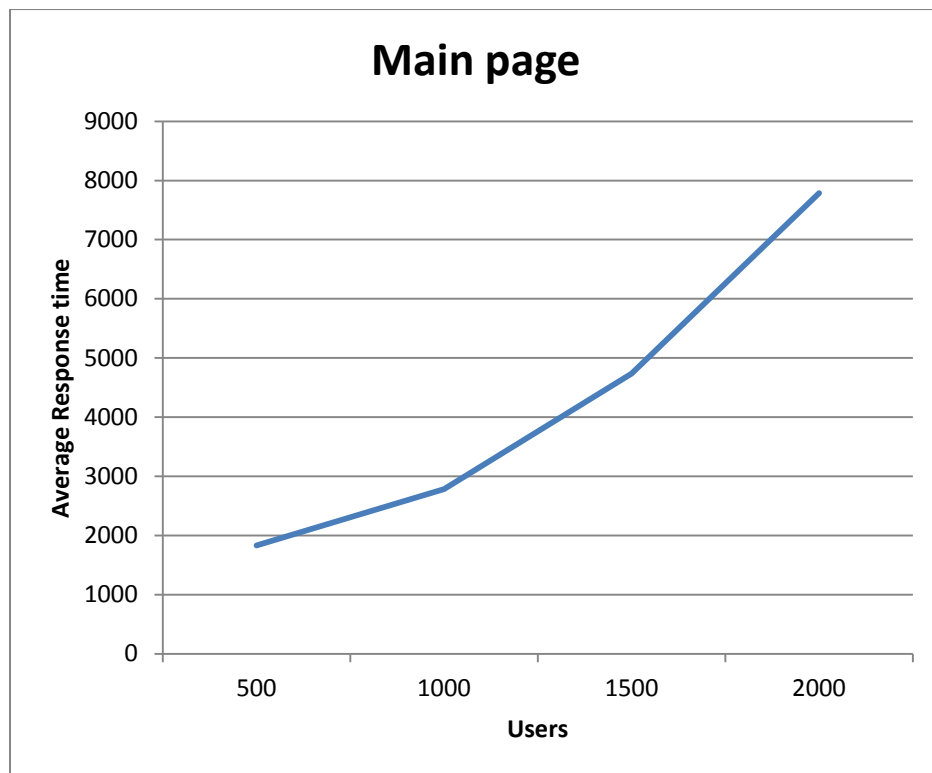


Figure 5: Average Response Time - Main Page

A graph is plotted with users and their average response times for Main page. The application is stable till around 1500 users. The average response time is highly increased after 1500 users. The throughput is decreased from around 5000 to 3000 when the number of users is increased from 1500 to 2000, which says that the application is not stable.

CHAPTER 5: CONCLUSION AND FUTURE WORK

5.1 Conclusion

The Agents from the Kansas State research and extension team can add proposals, view/edit proposals. The Admin and normal users have different privileges to manage the FNPRS website. The below are the operations which the agents can perform in the website.

1. Add worksheet: Adding proposal worksheets based on the year. Work sheets contain all the details which are required to be saved in the database by the agents.
2. Edit worksheet: Agent once entered the details and saved in the database, he/she can edit the information in future. If the program proposal has some budget entered by the agent and if there any differences later, the agents can edit the information
3. View worksheets: Every proposal can be summarized and viewed by the agent in order to be viewed at a time.
4. View Reports: All the budget expenses entered by the agent can be generated as reports.

5.2 Over all Experience

The current application, FNPRS which is a .Net web application has been developed using ASP.NET 3.5, Oracle and C# Programming language. jQuery for implementing UI .

It has been a great pleasure for me to work on this challenging project. This project provided me a good practical knowledge of various things like programming in C#.net web based application and writing SQL queries to retrieve information from Oracle database. It also provided me a good knowledge about the latest technologies in web based applications. This learning experience helps me a lot in providing guidance to working in my future projects involving these technologies.

This application also gave a deeper insight into ASP.Net controls such as Grid views, Repeater controls, data lists, managing huge database system and also building applications on visual Studio 2008.

5.3 Future Enhancements to the project

The following changes can be made to the application in future:

- The scope of the application is restricted to Kansas State. But this project can be extended to support various states in US other than Kansas.
- It can also be modified to handle data not only for Kansas State Research and Extension but also for various other groups at Kansas State University.
- The current project is developed in .Net 3.5 framework but can be moved to .Net 4 framework. It can also be developed using MVC architecture to add more enhanced features in easier way.

CHAPTER 6: REFERENCES

[1] Visual Studio

<http://msdn.microsoft.com/en-us/vstudio/cc136611>

[2] Three tier architecture

[http://msdn.microsoft.com/en-us/library/windows/desktop/ms685068\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/windows/desktop/ms685068(v=vs.85).aspx)

[3] Class Diagrams overview: Use tool

<http://www.objectmentor.com/resources/articles/umlClassDiagrams.pdf>

[4] [Matthew MacDonald](#), "ASP.Net: The Complete Reference", McGraw Hill, ISBN 978-0-0704-9536-4,

<http://www.firstload.com/?ir=1&fn=complete+reference+asp+net+pdf> (retrieved 2010-11-11), 2002