1. INTRODUCTION

Query form is one of the most widely used user interfaces for querying databases. Traditional query forms are designed and predefined by developers or DBA in various information management systems. With the rapid development of web information and scientific databases, modern databases become very large and complex. In natural sciences, such as genomics and diseases, the databases have over hundreds of entities for chemical and biological data resources. Many web databases, such as Freebase and DBPedia, typically have thousands of structured web entities. Therefore, it is difficult to design a set of static query forms to satisfy various ad-hoc database queries on those complex databases.

Many existing database management and development tools, such as EasyQuery, Cold Fusion, SAP and Microsoft Access, provide several mechanisms to let users create customized queries on databases. However, the creation of customized queries totally depends on users' manual editing. If a user is not familiar with the database schema in advance, those hundreds or thousands of data attributes would confuse him/her.

a. Problem Definition

Modern scientific databases and web databases maintain large and heterogeneous data. These real-world databases contain over hundreds or even thousands of relations and attributes. Traditional predefined query forms are not able to satisfy various ad-hoc queries from users on those databases. This paper proposes DQF, a novel database query form interface, which is able to dynamically generate query forms. The essence of DQF is to capture user's preference and rank query form components, assisting him/her to make decisions. The generation of a query form is an iterative process and is guided by the user. At each iteration, the system automatically generates ranking lists of form components and the user then adds the desired form components into the query form. The ranking of form components is based on the captured user preference. A user can also fill the query form and submit queries to view the query result at each iteration. In this way, a query form could be dynamically renewed till the user satisfies with the query results.

b. Aim and Objective of the Project

How to let non-expert users make use of the relational database is a challenging topic. A lot of research works focus on database interfaces which assist users to query the relational database without SQL. We propose a dynamic query form system which generates the query forms according to the user's desire at run time. The system provides a solution for the query interface in large and complex databases.

The system is proposed to have the following objectives along with functional requirements.

1. To generate dynamic query form
2. To enhance the efficiency of searching of records using keyword search, Query refinement and ranking algorithm.
3. The system should provide ranked components to assist user to make him/her decisions.
4. The system should provide authentication and authorization to the user databases and assign roles to users.
5. The user selected query must provide instantaneous results.

c. Scope and Limitations of the Project

This system in software requirement specification is designed to let a non-expert user interact with the database for retrieval of data according to his/her need and his/her preferences. While the proposed system will assist him/her to make decisions.

Our main objective is to provide query form interface for searching through database, so to preserve integrity and consistency any particular user will not be able to add his own attributes to database.

d. Timeline of the project

Process Model used: **Waterfall Model**

In waterfall model, phases are organized in linear order. In this model project begins with feasibility analysis. Upon successfully demonstrating the feasibility of project, the requirements analysis and project planning begins. The design starts after the requirement analysis is complete and coding begins after the design is complete. Once the programming is completed the code is integrated and testing is done. Upon successful completion of testing, the system is installed. After this, the regular operation and maintenance of the system takes place.

The following documents generally form a reasonable set that should be produced in each project:

- ✓ Requirements document
- ✓ Project plan
- ✓ Design documents (architecture, system, detailed)
- ✓ Test plan and test reports
- ✓ Final code
- ✓ Software manuals (e.g. user, installation)

	July	August	September	October	November	December	January	February	March
Topic Finalization									
Requirement Gathering									
Analysis									
Planning & design									
Module 1 [Login]									
Module 2[Query Execution]									
Integration & testing									

Fig. Project Timeline

e. Project Cost

Cost estimation based on COCOMO model

In this project the Cost Estimation based on COCOMO (Constructive Cost Model) the formula for the this Model is follows

Effort = Constant × (Size) scale factor× Effort Multiplier

– Effort in terms of person-months

– Constant: 2.45 in 1998 based on Organic Mode

– Size: Estimated Size in KLOC

– Scale Factor: combined process factors

– Effort Multiplier (EM): combined effort factors

The basic COCOMO equations take the form

Effort Applied (E) = $a_b(KLOC)^{b}_{b}$ [man-months]

Development Time (D) = $c_b(\text{Effort Applied})^{d}_{b}$ [months]

People required (P) = Effort Applied / Development Time [count]

Where, KLOC is the estimated number of delivered lines (expressed in thousands) of code for project. The coefficients a_b, b_b, c_b and d_b are given in the following table .

Software Project	a_b	b_b	c_b	d_b
Organic	2.4	1.05	2.5	0.38
Semidetached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

Organic Mode

1. Estimation of development effort

Effort = 2.4*(KLOC)*1.05 = 41.67

2. Estimation of development time

Development Time= 2.5 *(41.67)*0.38 = 10.12 Months

3. Number of people in the project

People required = 41.67/10.12 ~= 5 people

2. BACKGROUND STUDY

&

LITERATURE REVIEW

a. Investigation of Current Project and Related Work

How to let non-expert users make use of the relational database is a challenging topic. A lot of research works focus on database interfaces which assist users to query the relational database without SQL. QBE (Query-By-Example and Query Form are two most widely used database querying interfaces. At present, query forms have been utilized in most real-world business or scientific information systems. Current studies and works mainly focus on how to generate the query forms.

Customized Query Form:

Existing database clients and tools make great efforts to help developers design and generate the query forms, such as EasyQuery, Cold Fusion, SAP, Microsoft Access and so on. They provide visual interfaces for developers to create or customize query forms. The problem of those tools is that, they are provided for the professional developers who are familiar with their databases, not for end-users. They proposed a system which allows end-users to customize the existing query form at run time. However, an end-user may not be familiar with the database. If the database schema is very large, it is difficult for them to find appropriate database entities and attributes and to create desired query forms.

Automatic Static Query Form:

Recently, proposed automatic approaches to generate the database query forms without user participation. They presented a data-driven method. It first finds a set of data attributes, which are most likely queried based on the database schema and data instances. Then, the query forms are generated based on the selected attributes. It is a workload-driven method. It applies clustering algorithm on historical queries to find the representative queries. The query forms are then generated based on those representative queries. One problem of the aforementioned approaches is that, if the database schema is large and complex, user queries could be quite diverse. In that case, even if we generate lots of query forms in advance, there are still user queries that cannot be satisfied by any one of query forms. Another problem is that, when we generate a large number of query forms, how to let users find an appropriate and desired query form would be challenging. A solution that combines keyword search with query form generation. It automatically generates a lot of query forms in advance. The user inputs several keywords to find relevant query forms from a large number of pre generated query forms. It works well in the databases which have rich textual information in data tuples and schemas.

However, it is not appropriate when the user does not have concrete keywords to describe the queries at the beginning, especially for the numeric attributes.

Auto completion for Database Queries:

In this, novel user interfaces have been developed to assist the user to type the database queries based on the query workload, the data distribution and the database schema. Different from our work which focuses on query forms, the queries in their work are in the forms of SQL and keywords.

Query Refinement:

Query refinement is a common practical technique used by most information retrieval systems. It recommends new terms related to the query or modifies the terms according to the navigation path of the user in the search engine. But for the database query form, a database query is a structured relational query, not just a set of terms.

Dynamic Faceted Search:

Dynamic faceted search is a type of search engines where relevant facets are presented for the users according to their navigation paths. Dynamic faceted search engines are similar to our dynamic query forms if we only consider *Selection* components in a query. However, besides *Selections*, a database query form has other important components, such as *Projection* components. *Projection* components control the output of the query form and cannot be ignored. Moreover, designs of *Selection* and *Projection* have inherent influences to each other.

Database Query Recommendation:

Recent studies introduce collaborative approaches to recommend database query components for database exploration. They treat SQL queries as items in the collaborative filtering approach, and recommend similar queries to related users. However, they do not consider the goodness of the query results. They proposes a method to recommend an alternative database query based on results of a query. The difference from our work is that, their recommendation is a complete query and our recommendation is a query component for each iteration.

Dynamic Data Entry Form:

They develop an adaptive forms system for data entry, which can be dynamically changed according to the previous data input by the user. Our work is different as we are dealing with database query forms instead of data-entry forms.

Active Feature Probing:

Zhu et al. develop the active featuring probing technique for automatically generating clarification questions to provide appropriate recommendations to users in database search. Different from their work which focuses on finding the appropriate questions to ask the user, DQF aims to select appropriate query components.

b. Literature Overview

A lot of research works focus on database interfaces which assist users to query the relational database without SQL. QBE (Query-By-Example) and Query Form are two most widely used database querying interfaces. Current studies and works mainly focus on how to generate the query forms.

A. Modified Query Form:

The tools provided by the database clients make great efforts to help developers generate the query forms, such as Easy Query, Cold Fusion and so on. They provide visual interfaces for developers to create or customize query forms. The problem of those tools is that, they are provided for the professional developers. H.V. Jagadish proposed a system which allows end-users to customize the existing query form at run time. If the database schema is very large, it is difficult for end user to find appropriate database entities and attributes.

B. Automated Creation of Forms:

M. Jayapandian presented a data-driven method. It first finds a set of data attributes, which are most likely queried based on the database schema and data instances. Then, the query forms are generated based on the selected attributes.

C. Automating the design and construction of query forms:

H.V. Jagadish presented a workload-driven method. It applies clustering algorithm on historical queries to find the representative queries. The query forms are then generated based on those representative queries. One problem of the aforementioned approaches is that, if we generate lots of query forms in advance, there are still user queries that cannot be satisfied by any one of query forms. Another problem is that, when we generate a large number of query forms, how to let users find an appropriate query form would be challenging.

D. Combining keyword search and forms:

A solution for aforementioned approaches is proposed in it. It automatically generates a lot of query forms in advance. The user inputs several keywords to find relevant query forms from a large number of pre-generated query forms but it is not appropriate when the user does not have concrete keywords to describe the queries.

3. **REQUIREMENT ANALYSIS**

a. HARDWARE AND SOFTWARE REQUIREMENTS

- **Minimum Hardware Requirements:**

✓ Intel Core 2 Duo @2.83 GHz

✓ 80 GB HDD

✓ 1 GB RAM

- **Software Requirements:**

✓ Visual Studio 2012-15

✓ ASP .Net 4.5

✓ Microsoft SQL Server

✓ FontAwesome Library

✓ IIS server

✓ Web-Browser

b. External Interface:

Input:

The input will be in the form of keyword or selected attribute as the user preferences and his/her requirements. Also the keyword may be from ranked components as assisted by the system.

Output:

Output will be the result of query as refined by the user.

c. Functional Requirements:

The system is proposed to have the following modules along with functional requirements.

1. Query Form Enrichment
2. Query Execution
3. Customized Query Form
4. Database Query Recommendation

d. Performance Requirements:

The system is proposed to have the following performance requirements:

1. The system should provide ranked components to assist user to make him/her decisions.
2. The system should provide authentication and authorization to the user databases and assign roles to users.
3. The user selected query must provide instantaneous results.

e. Software system attributes:

There are a number of attributes of software that can serve as requirements. It is important that required attributes be specified so that their achievement can be objectively verified.

i) Reliability

The query results will be accurate and will be totally based on user preferences and ranking algorithm.

ii) Security

The database and its data retrieval will be available for authorized users only.

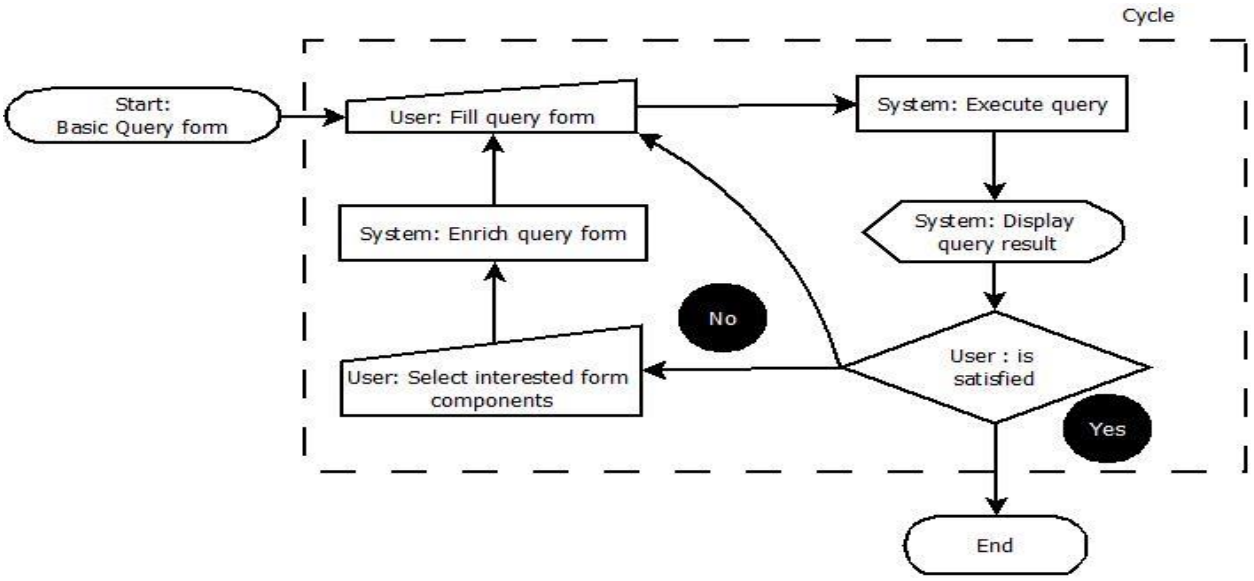
iii) Availability

The system will be available as a web-application.

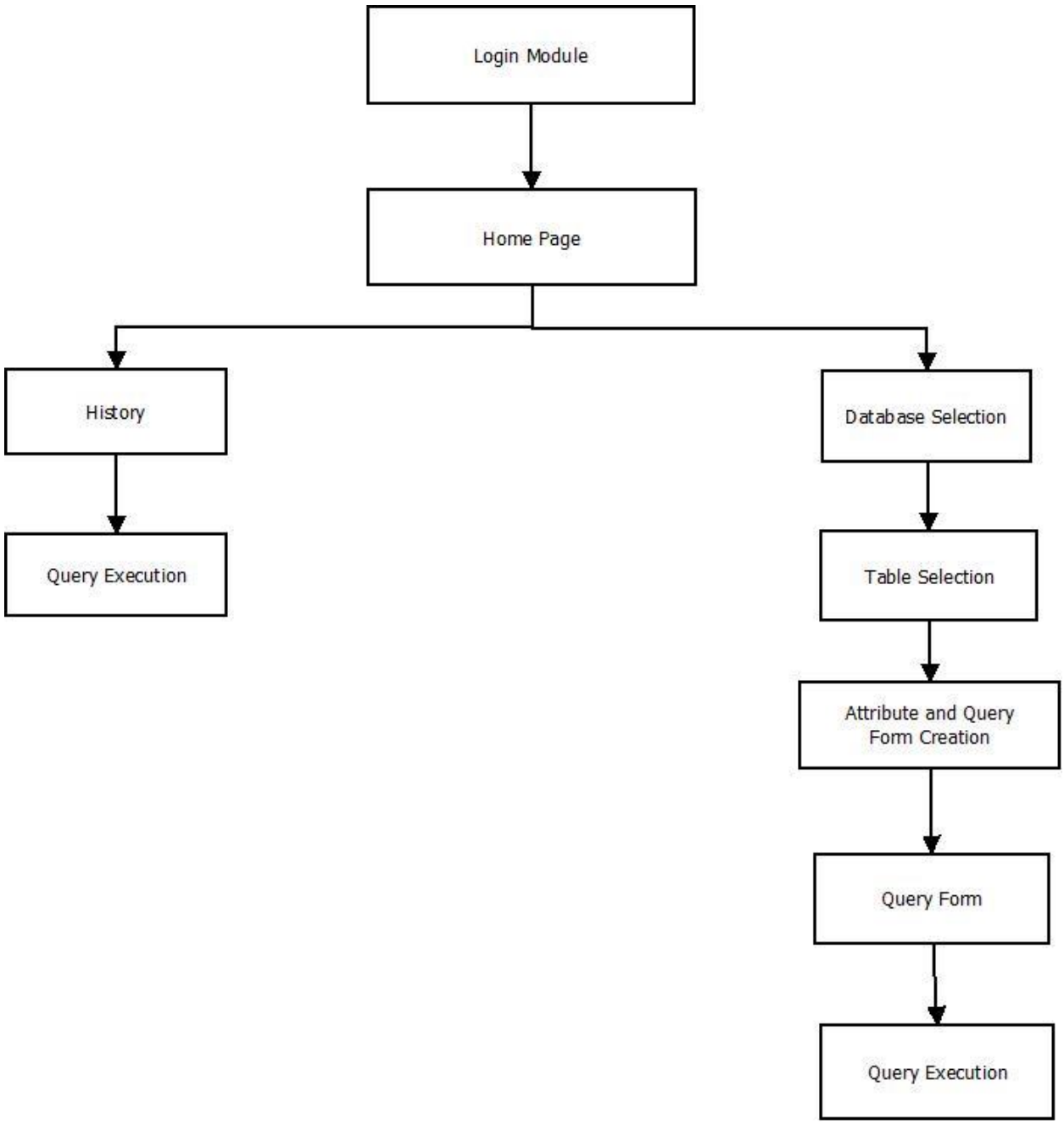
4. SYSTEM DESIGN

System design covers system architecture. System and the elements consist of design specification, for instance, use case diagram, activity diagram, state-chart diagram and so on, samples of those diagrams are included in the chapter and discussed briefly.

a. Architectural Design

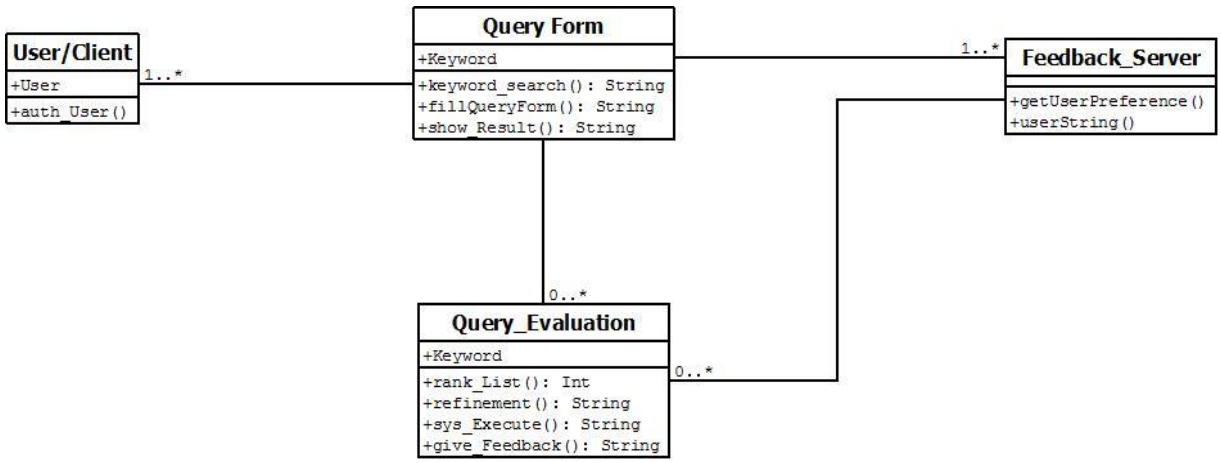


b. User Interface Design

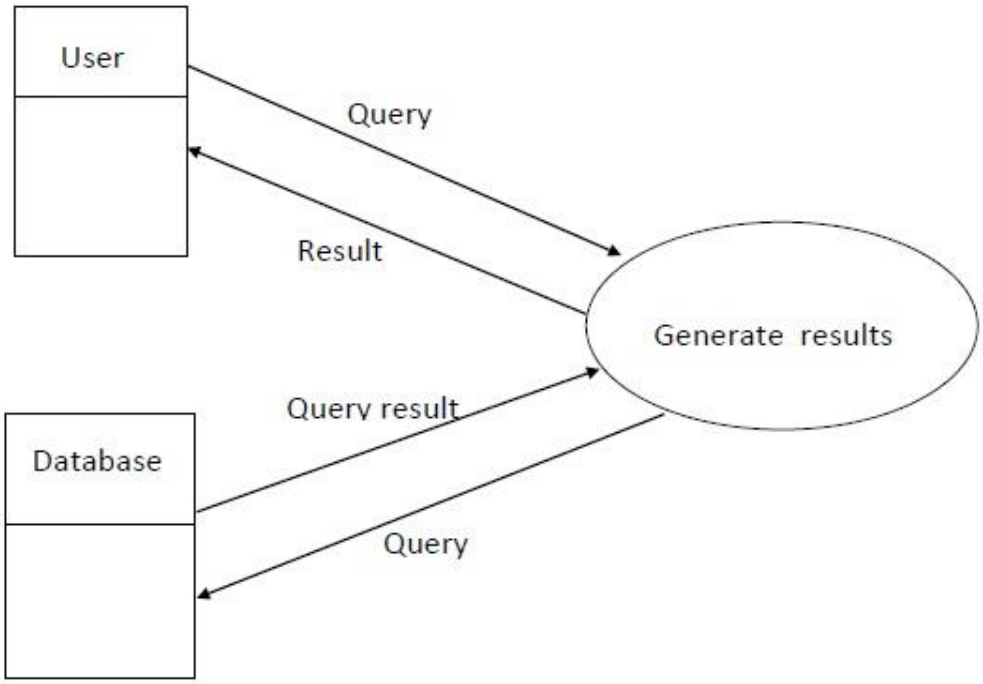


c. System Modeling

1. Class Diagram

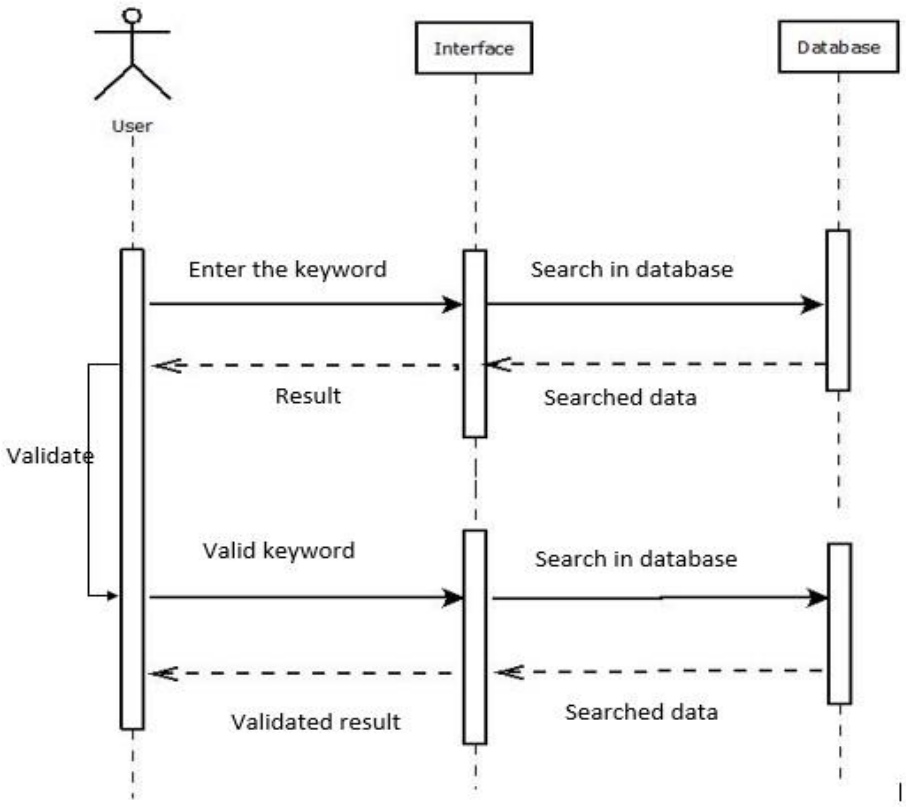


2. Dataflow Diagram



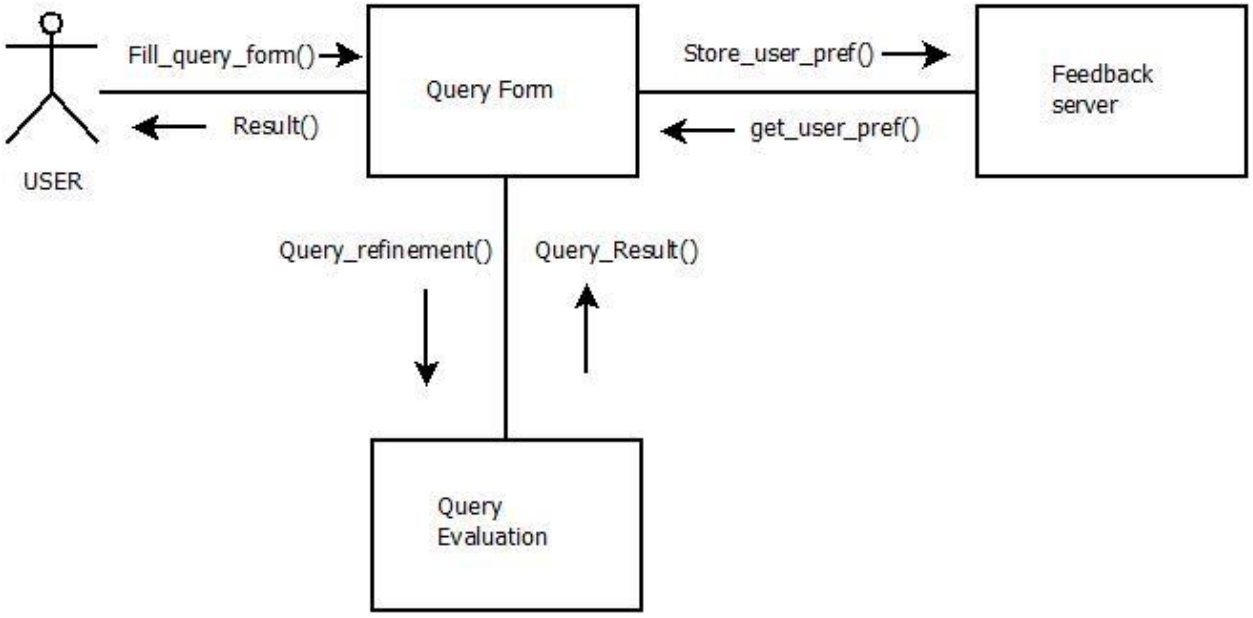
Data flow diagram

3. Sequence Diagram

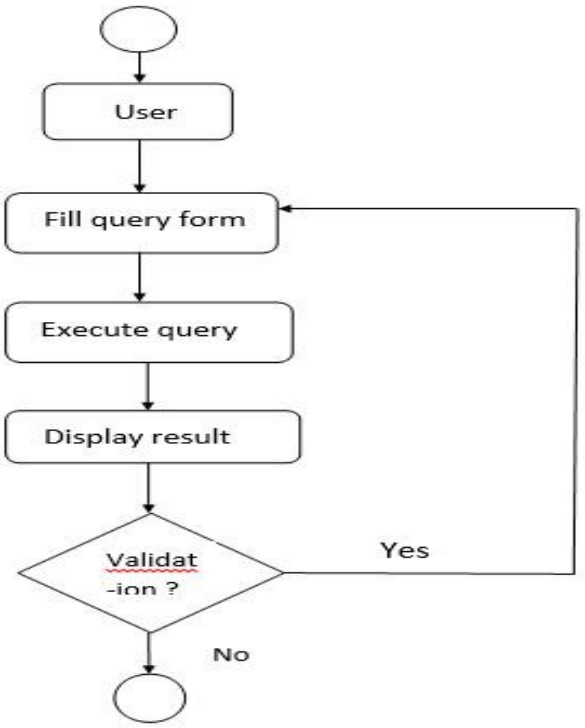


Sequence Diagram

4. Collaboration Diagram

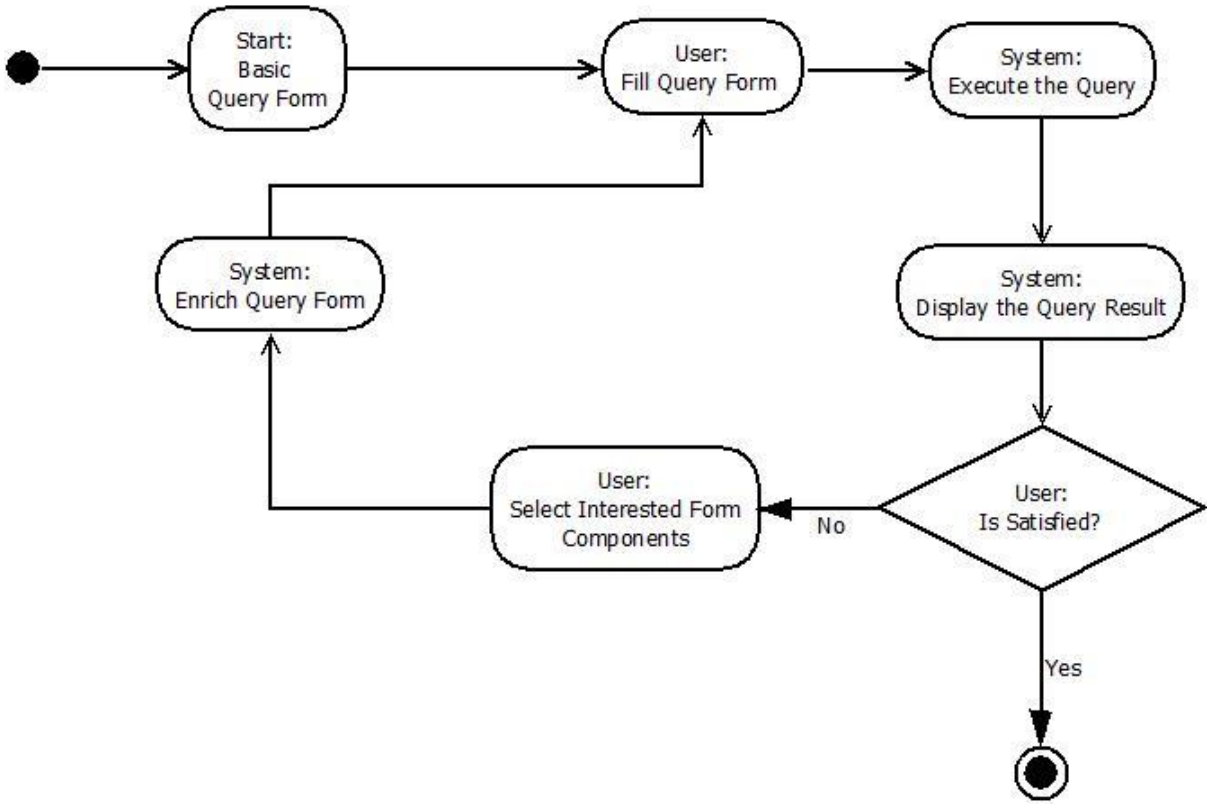


5. Use Case Diagram

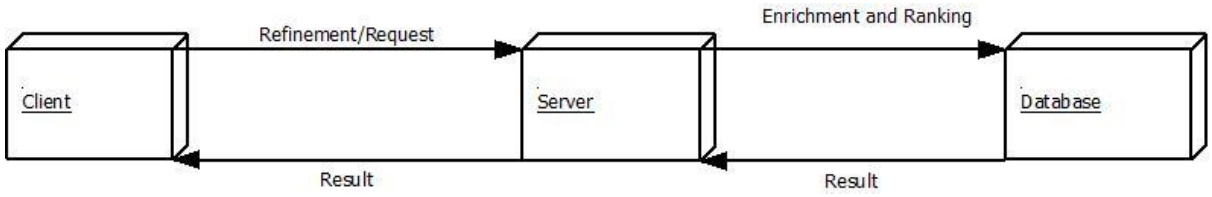


Use-case Diagram

6. Activity Diagram



7. Deployment Diagram



d. Algorithmic description of each modules

1. Login Module:

This will allow user to enter into DQF after authentication and authorization.

a. Authorization

Step 1 : Start
Step 2 : If user is already authenticated
Go to step 4
Else If
User is registered but yet to be authenticated by admin
Go to step 3
Else
I. Redirect to registeration page
II. Enter valid registration details
III. Send the user registration data to admin
Step 3 : Admin checks authenticity of user
Step 4 : If user is authenticated, allow user to login
Else
Deny login
Step 5 : Stop

b. Authentication

Step 1: Start
Step 2 : If user is not registered
Goto step 4
Else
if user is authenticated
if ISONo and Password are correct
I.Allow Login
II.Allocate session to user
III.Redirect to Home page
else
Show message ‘’invalid login details’’
else
Show message ‘’ user is not authorized by admin’’
Step 4 : Stop

2. Relevance Ranking Algorithm:

Step 1: Start.

Step 2: If user history and preferences are already present, go to step 6.

Step 3: Else, Proceed as Query form generation algorithm.

Step 4: Submit the query and get the result.

Step 5: Check whether query executed is already present in user session's history.

i) If yes, increment the query execution count by 1.

ii) If no, store the newly created query in the user session's history.

Step 6: Show new ranked history to the user

Step 7: If user wishes to execute more queries

i) From his history, go to step 6

ii) From new query form, go to step 3

Step 8: Stop.

3. Query form generation algorithm:

Step 1: Start.

Step 2: If user chooses keyword search then show ranked history filtered with his selected keyword and go to step 8.

Step 3: Else user chooses to create new query form.

Step 4: Select any one database from the set of databases.

Step 5: Select Tables from the earlier selected database.

Step 6: If condition attribute is present which is common to all selected tables

i) Select both the selected attributes and condition attributes.

Else

ii) Select the selected attributes.

Step 7: allow user to shortlist the attributes from chosen tables

Step 8: Check whether data required to user is present in the history.

i) If yes, then directly select it from history.

ii) Otherwise, make possible searches on selected and conditional attributes.

Step 9: Stop.

5. IMPLEMENTATION

a. Environmental Setting for Running the Project

For building the project the essential libraries like FontAwesome and Bootstrap are needed for UI design and for development purpose Visual Studio is required. After deployment only the Web-Browser is needed to access the query form.

First of all user must be registered to the DQF. Then that user is authorized by the admin later he/she can access the DQF. In order to register, User may have to navigate to the Register page inside the login module. After the registration and authorization user need to once again authenticate to access the DQF.

b. Detailed Description of Modules

1. Keyword search with Query form generation

In the future it creates good deal of query forms. The user has to provide some keywords get the query forms from the large databases. The database that contains information in the form of rich text as data schemas and tuples works well. Again we can understand one difficulty with this user doesn't have appropriate keywords to start.

2. Query Form Enrichment

1) DQF recommends a ranked list of query form components to the user.

2) The user selects the desired form components into the current query form.

3. Query execution

1) The user fills out the current query form and submit a query.

2) DQF executes the query and shows the results.

3) The user provides the feedback about the query results.

4. Customized Query Form

They provide visual interfaces for developers to create or customize query forms. The problem of those tools is that, they are provided for the professional developers who are familiar with their databases, not for end-users. If proposed a system which allows end-users to customize the existing query form at run time. However, an end-user may not be familiar with the database. If the database schema is very large, it is difficult for them to find appropriate database entities and attributes and to create desired query forms.

5. Database Query Recommendation

Recent studies introduce collaborative approaches to recommend database query components for database exploration. They treat SQL queries as items in the collaborative filtering approach, and recommend similar queries to related users.

6. INTEGRATION & TESTING

a. Description of the Integration Module

Integration Testing

Integration testing is any type of software testing that seeks to verify the interfaces between components against a software design. Software components may be integrated in an iterative way or all together ("big bang"). Normally the former is considered a better practice since it allows interface issues to be localized more quickly and fixed.

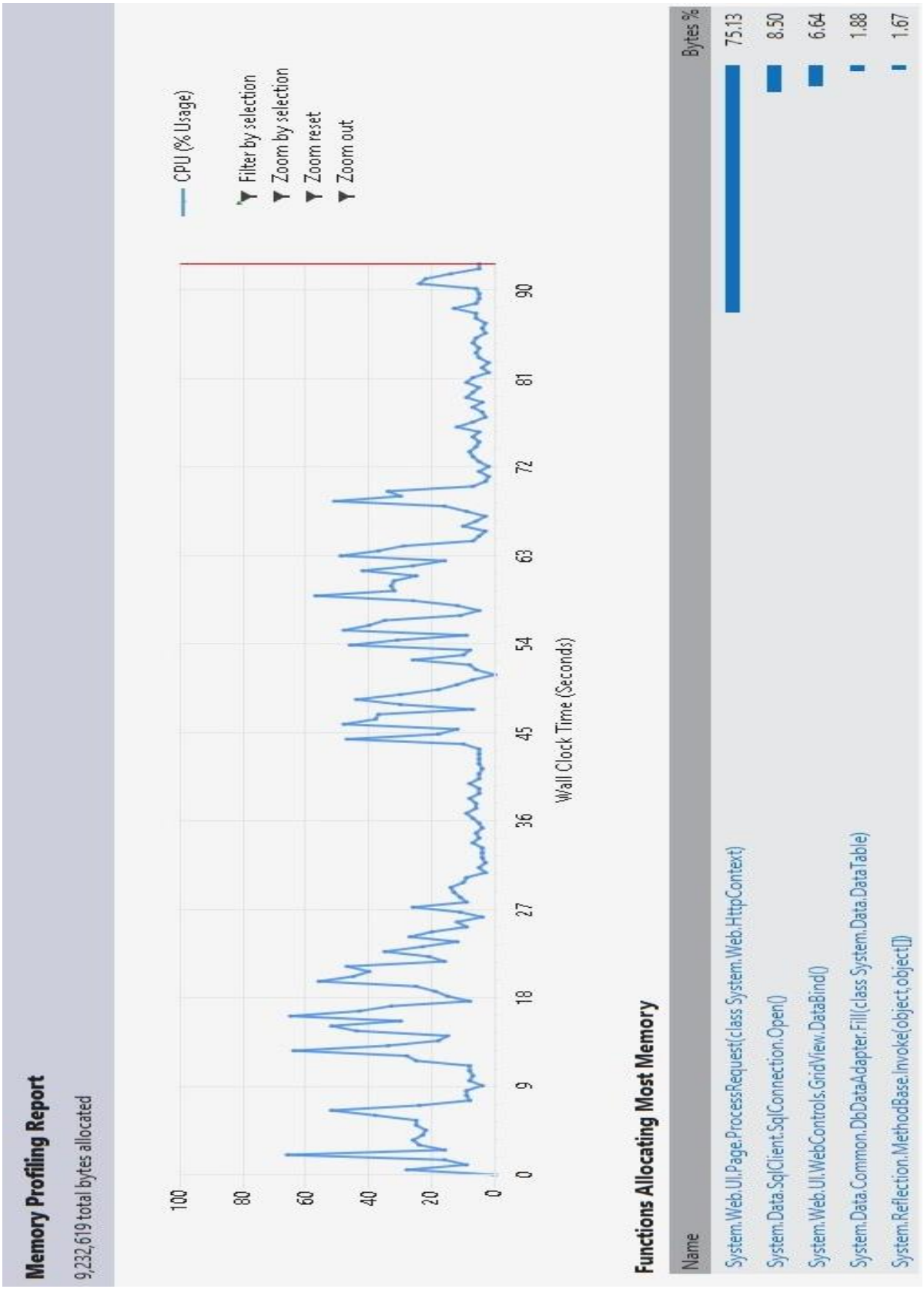
Integration testing works to expose defects in the interfaces and interaction between integrated components (modules). Progressively larger groups of tested software components corresponding to elements of the architectural design are integrated and tested until the software works as a system.

b. Testing - These are some sample test case examples which are performed in our project.

No.	Test case Title	Description	Expected Outcome	Result
1	Successful User Verification.	The login to the system should be tried with the login assigned by the admin and the correct password.	Login should be successful and the user should enter in to the system.	Passed
2	Unsuccessful User Verification due to wrong password.	Login to the system with a wrong password,	Login should fail with an error ‘Invalid Password’.	Passed

3	Authorization	After registration admin will authorize that particular user	Authorized user	Passed
4	Unsuccessful User Verification due to invalid login id.	Login to the system with an invalid login id.	Login should fail with an error 'Invalid user id'.	Passed
5	User have to fill query form or add keyword	Check whether the data present within the database.	Proper working of query form.	Passed
6	User get the result	Check whether the data is correct.	Proper working of keyword search and ranking algorithm.	Passed
7	User have to check for the valid result	To check whether the searched data is as per the need of user	Proper working of keyword search and ranking algorithm.	Passed
8	Validation	The data is not as per the user's need then user fill the form again	Valid result	Passed

7. PERFORMANCE ANALYSIS


Memory Profiling Report
9,232,619 total bytes allocated
CPU (% Usage)
Filter by selection
Zoom by selection
Zoom reset
Zoom out
100
80
60
40
20
0
0
9
18
27
36
45
54
63
72
81
90
Wall Clock Time (Seconds)
Functions Allocating Most Memory
Name
Bytes %
System.Web.UI.Page.ProcessRequest(class System.Web.HttpContext)
75.13
System.Data.SqlClient.SqlConnection.Open()
8.50
System.Web.UI.WebControls.GridView.DataBind()
6.64
System.Data.Common.DbDataAdapter.Fill(class System.Data.DataTable)
1.88
System.Reflection.MethodBase.Invoke(object,object[])
1.67

8. APPLICATION

- The essence of DQF is to provide query form interface to user. This form interface will be useful for the non-technical user like clerk, non-database users and naive users who are not able to write query to retrieve data from the database.

- This DQF can also be used by database users where multiple relations are present and he/she need to retrieve any particular data quickly as possible.

- This query form interface can also be used in the organization where retrieval time for data from database has to be minimum.

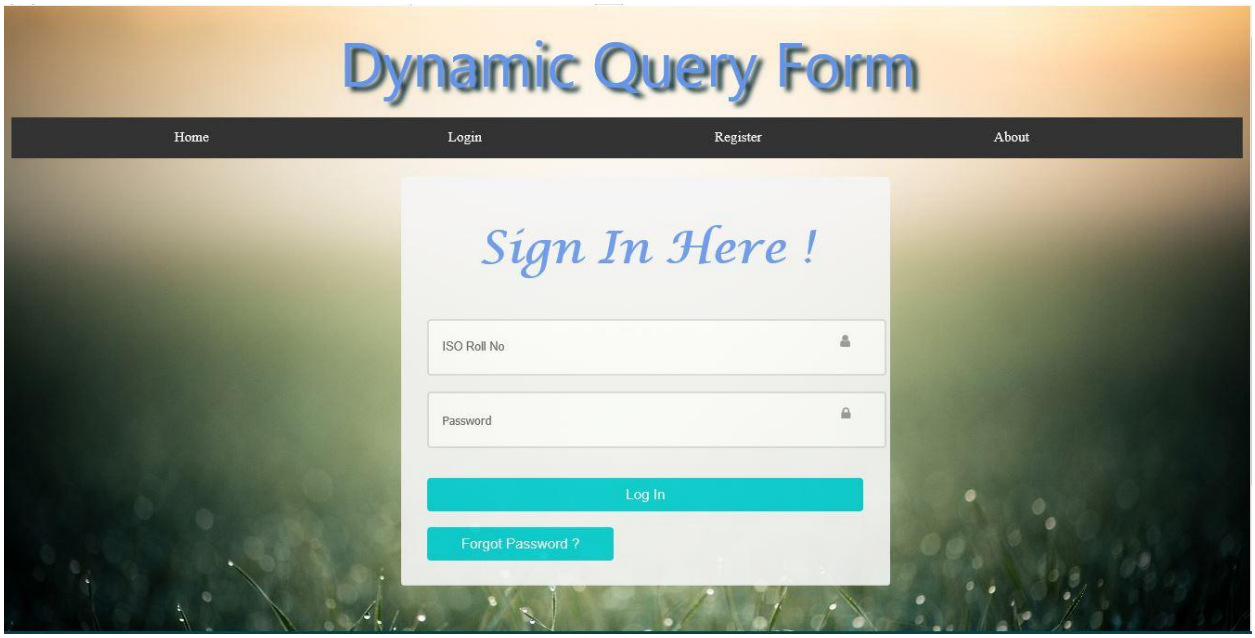
9. Installation Guide & User Manual

- The required databases must be added prior running the DQF to access those databases inside the DQF.
- The user must be registered and authorized to access the DQF.
- The authorization is done by admin after validating his/her details.
- After deployment of DQF only Web-Browser is needed to access the DQF no extra environmental settings are needed.

User Manual

This user manual consist of steps to access and interact with DQF.

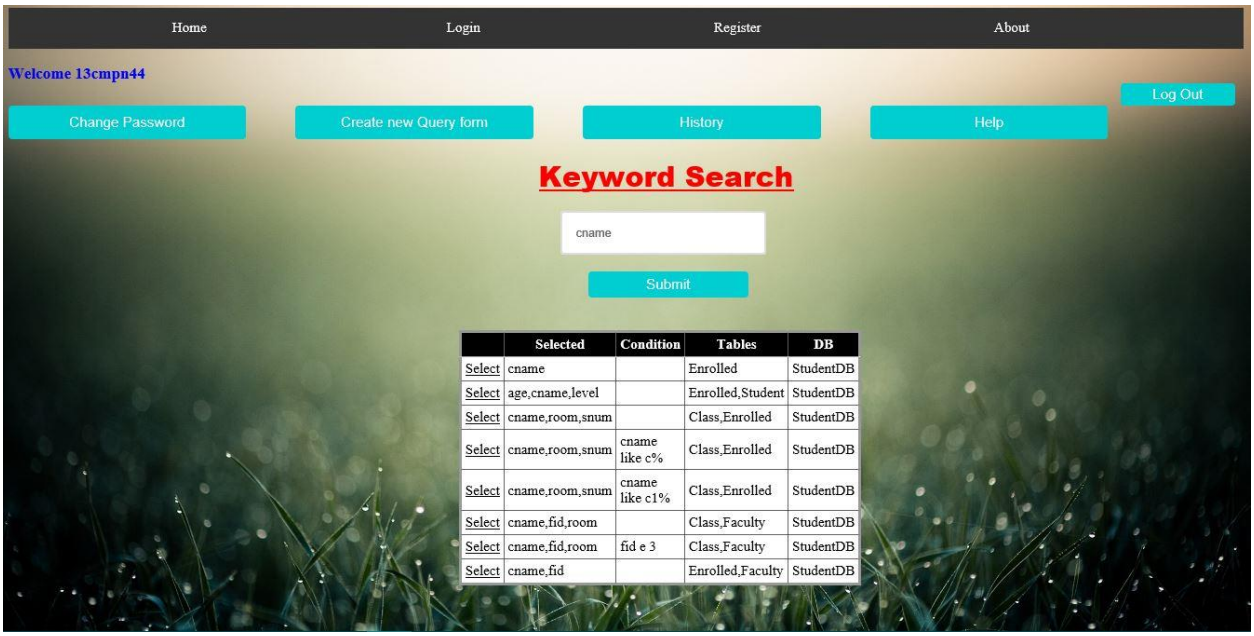
1. Login Page: Here authorized user need to enter valid username and password to access DQF



2. Register Page: Here user will have to fill his/her credentials to get registered.

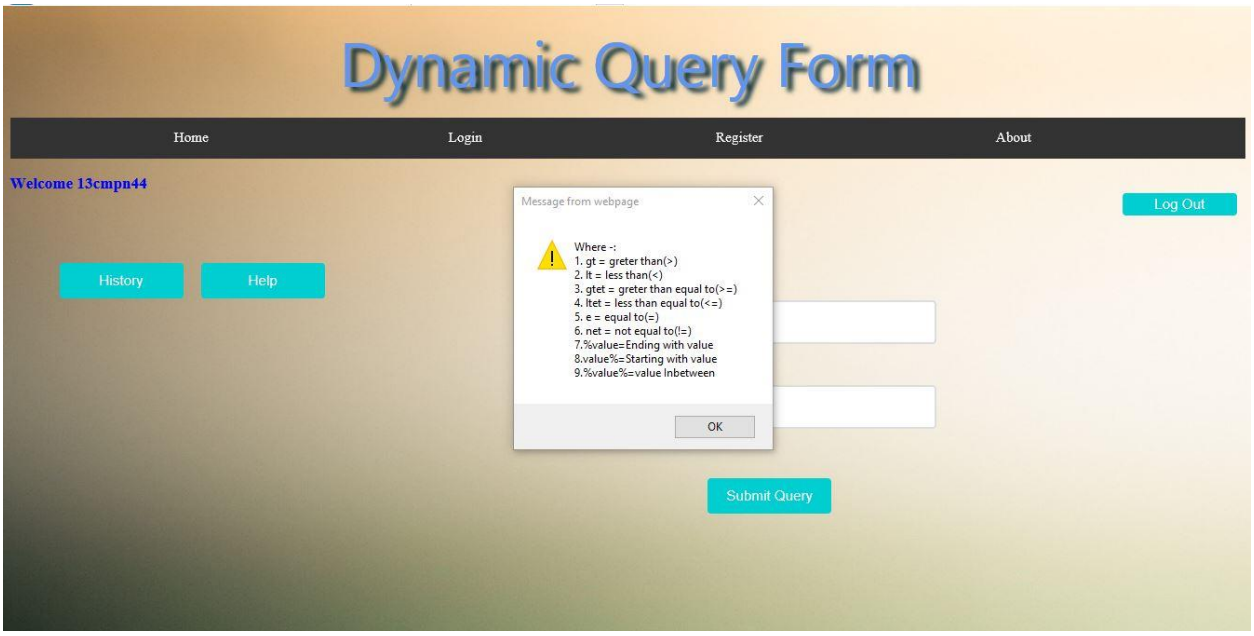
3. Home Page: This is home page for DQF which can be used to access the features of DQF

4. Keyword search: This will allow user to use keyword search on ranked history.

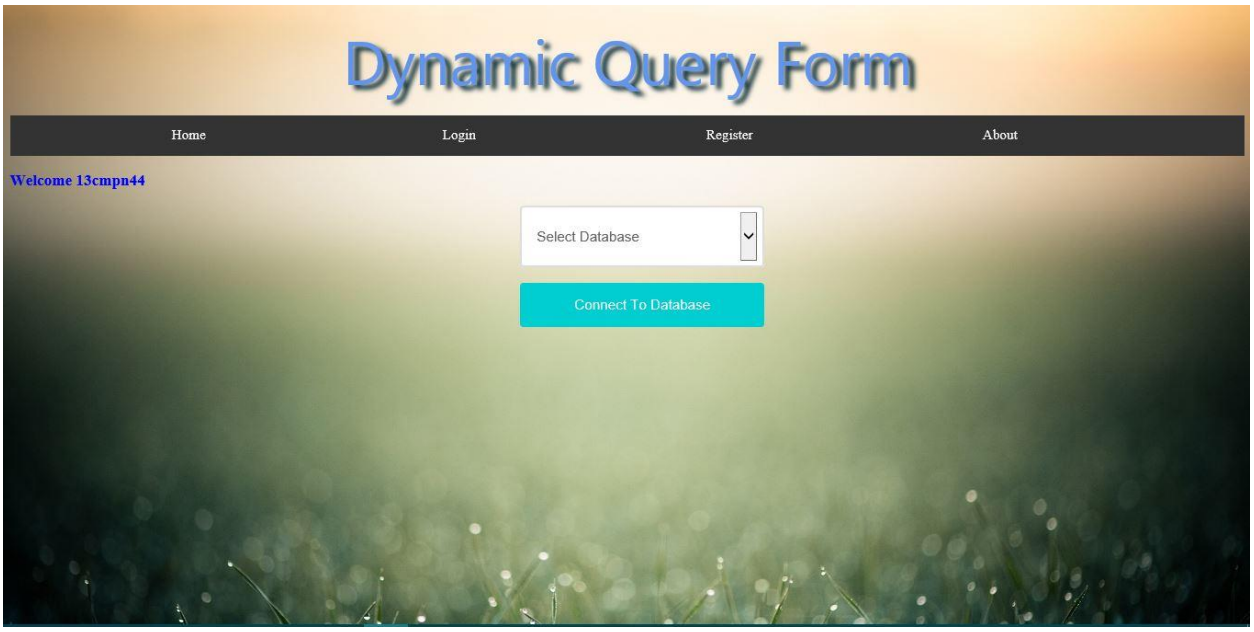


	Selected	Condition	Tables	DB
Select	cname		Enrolled	StudentDB
Select	age,cname,level		Enrolled,Student	StudentDB
Select	cname,room,snum		Class,Enrolled	StudentDB
Select	cname,room,snum	cname like c%	Class,Enrolled	StudentDB
Select	cname,room,snum	cname like c1%	Class,Enrolled	StudentDB
Select	cname,fid,room		Class,Faculty	StudentDB
Select	cname,fid,room	fid e 3	Class,Faculty	StudentDB
Select	cname,fid		Enrolled,Faculty	StudentDB

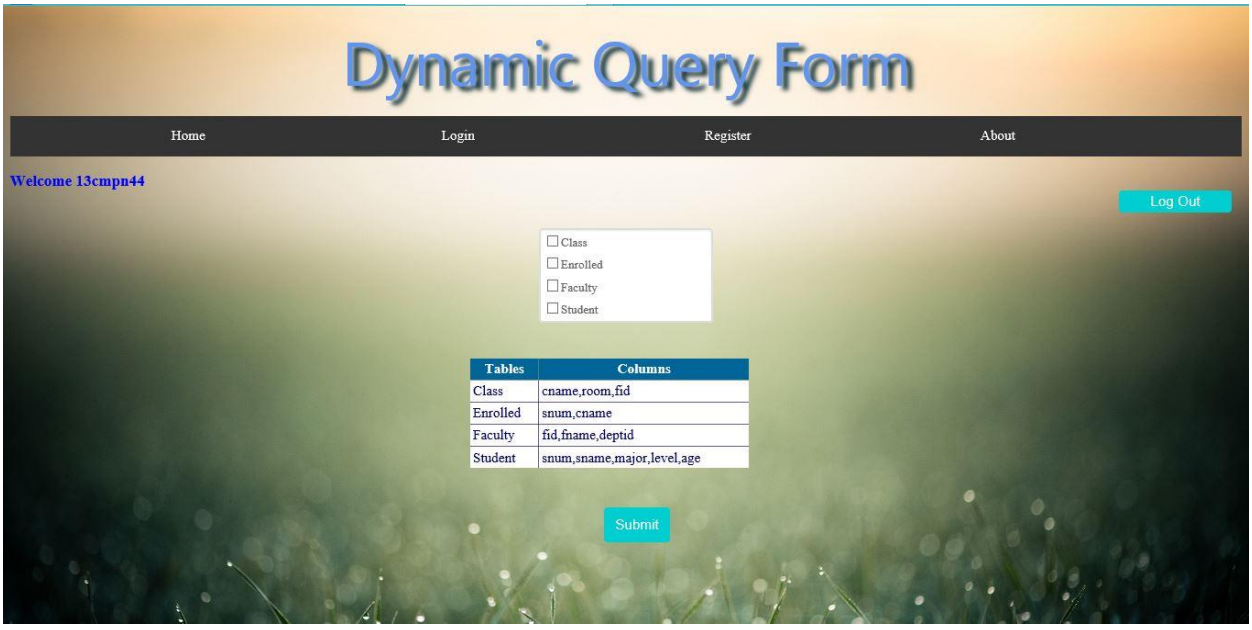
5. Help: Can be used to know the Database terms related to DQF.



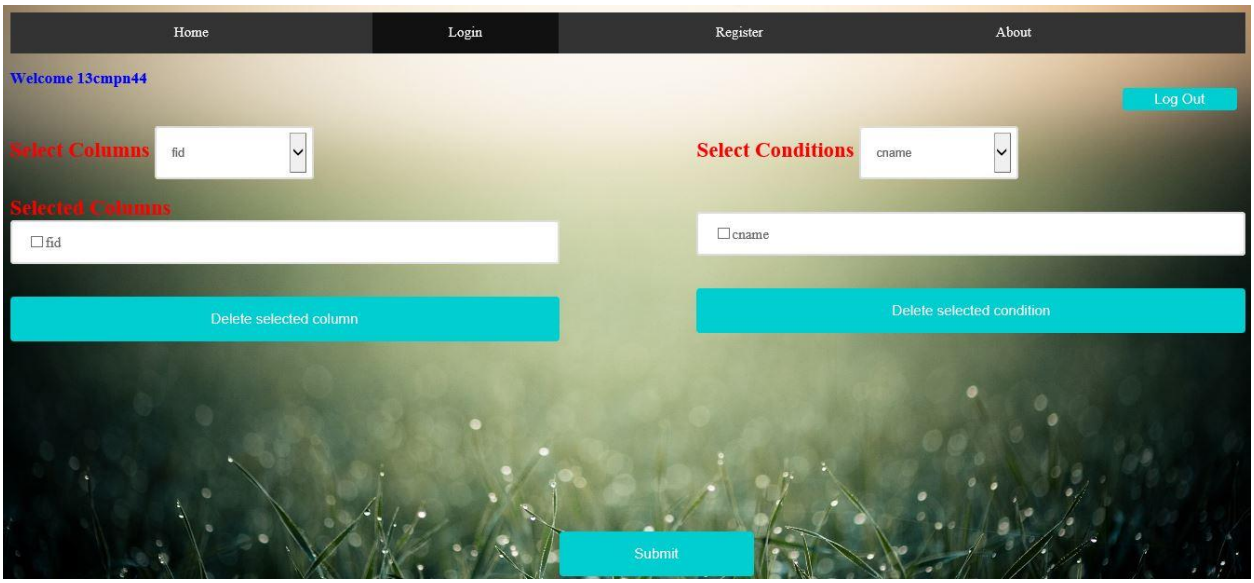
6. Navigator page: This page will allow user to select database to get results from.



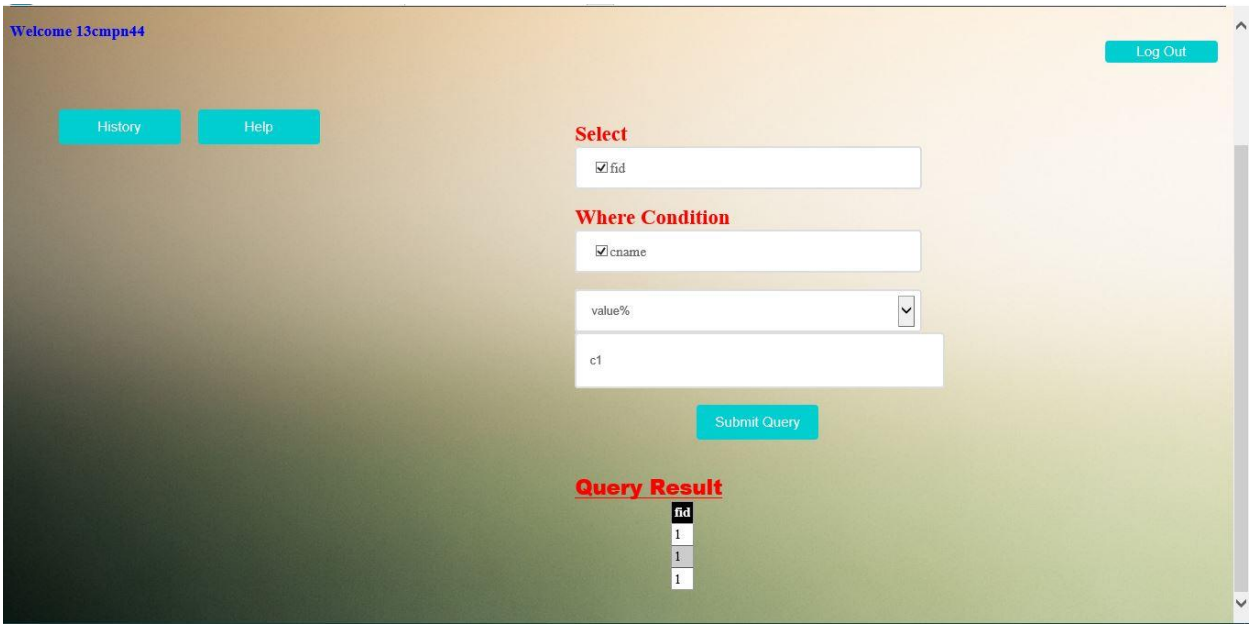
7. Selector Page: Here user will select particular tables from database selected earlier.



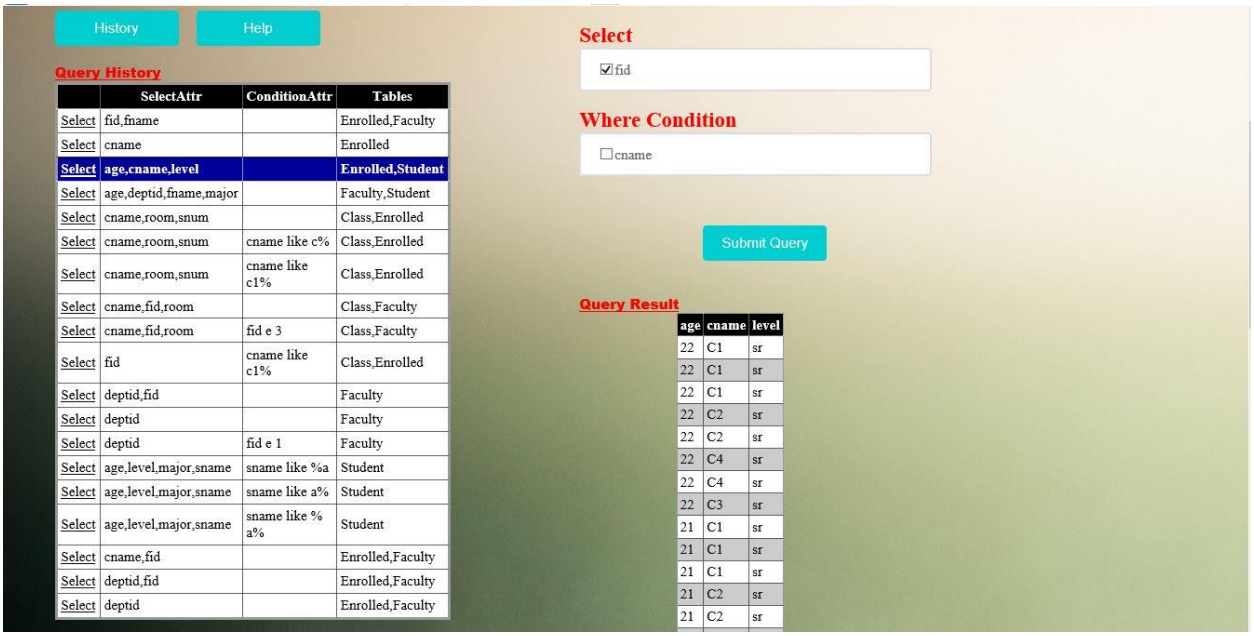
8. Create form page: Here user will select table attributes.



9. Query form: This will allow user to select attributes and enter conditions to generate query results.



10. History: This will allow user to get results directly from selecting queries from recent ranked history.



	SelectAttr	ConditionAttr	Tables
Select	fid,fname		Enrolled,Faculty
Select	cname		Enrolled
Select	age,cname,level		Enrolled,Student
Select	age,deptid,fname,major		Faculty,Student
Select	cname,room,snum		Class,Enrolled
Select	cname,room,snum	cname like c%	Class,Enrolled
Select	cname,room,snum	cname like c1%	Class,Enrolled
Select	cname,fid,room		Class,Faculty
Select	cname,fid,room	fid e 3	Class,Faculty
Select	fid	cname like c1%	Class,Enrolled
Select	deptid,fid		Faculty
Select	deptid		Faculty
Select	deptid	fid e 1	Faculty
Select	age,level,major,sname	sname like %a	Student
Select	age,level,major,sname	sname like a%	Student
Select	age,level,major,sname	sname like % a%	Student
Select	cname,fid		Enrolled,Faculty
Select	deptid,fid		Enrolled,Faculty
Select	deptid		Enrolled,Faculty

age	cname	level
22	C1	sr
22	C1	sr
22	C1	sr
22	C2	sr
22	C2	sr
22	C4	sr
22	C4	sr
22	C3	sr
21	C1	sr
21	C1	sr
21	C1	sr
21	C2	sr
21	C2	sr

10. DECLARATION OF ETHICS

As a computer Science and Engineering student, I believe it is unethical to,

1. Surf the internet for personal interest and non-class related purposes during classes.
2. Make a copy of software for personal or commercial use.
3. Make a copy of software for a friend.
4. Loan CDs of software to friends.
5. Download pirated software from the internet.
6. Distribute pirated software from the internet.
7. Buy software with a single user license and then install it on multiple Computers.
8. Share a pirated copy of software.
9. Install a pirated copy of software.

11. REFERENCES

1. M. Jayapandian and H. V. Jagadish, Automating the design and construction of query forms, IEEE Trans. Knowl. Data Eng., vol. 21, no. 10, Oct. 2009, pp. 13891402.

2. E. Chu, A. Baid, X. Chai, A. Doan, and J. F. Naughton, Combining keyword search and forms for ad hoc querying of databases, in Proc. ACM SIGMOD, Providence, RI, USA, Jun. 2009, pp. 349 360.

3. M. Jayapandian and H. V. Jagadish, Automated creation of a forms based database query interface, Proc. VLDB, vol. 1, no. 1, Aug. 2008, pp. 695709.

4. C. C. Aggarwal, J. Han, J.Wang, and P. S. Yu, A framework for clustering evolving data streams, in Proc. VLDB, Berlin, Germany, Sept. 2003, pp. 8192.

5. S. Agrawal, S. Chaudhuri, G. Das, and A. Gionis, Automated ranking of database query results, in Proc. CIDR, 2003, pp. 888899.