GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

ONLINE QUIZ SYSTEM

A Project Report of Capstone Project – 2

Submitted by

NAME OF THE CANDIDATE(S)

(1713104060/ 17SCSE104062)

in partial fulfillment for the award of the degree

of

BACHELOR’S OF COMPUTER APPLICATIONS

IN

COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Under the Supervision of

Mr. Sachin Minocha Assistant Professor

May 2020

GALGOTIAS
UNIVERSITY

SCHOOL OF COMPUTING AND SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report “ONLINE QUIZ SYSTEM” is the bonafide work of “RUBY SHARMA(1713104060)” who carried out the project work under my supervision.

SIGNATURE OF HEAD	SIGNATURE OF SUPERVISOR
Mr. MUNISH SABARWAL	Mr. SACHIN MINOCHA
Professor and Dean	Assistant Professor
School of Computing Science and Engineering .	School of Computing Science and Engineering

Abstract:

Being a web-based project, the proposed questionnaire aims to provide a complete online platform for the management of questionnaires and team participants. Users of the program include managers, industry, institutions, organizations, academics and others.

The project provides access to admins with a valid user id and password. The user id and password are required for the administrator to "log in" to the system.

In the Online Quiz system project, an 'N' number of teams can be created and they can participate in a tournament. Also, the 'N' number of rounds can be played at the end to declare the team a winner. Teams with lower grades are eliminated, and the remaining teams compete at the next level. This process continues in the next round until the winner is announced.

Participants are automatically asked questions, and this is subject to a time limit. If a participant is unable to answer a question, a negative mark is given. There is also a provision for rapid fire rotation.

Scores earned by each group are automatically updated. Negative marking management function is performed by the administrator. In addition, viewing and deleting the query results and auto-delete groups based on their performance can be performed.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

FIGURE **PAGE**

1. Introduction:

This web-based intelligence-choice-quiz is a web-based survey, called the Online Quiz System, for assessing students. It is a system where students can come into question from anywhere in the world where there is no communication between pencil and paper rather than the computer-to-person communication. We use the word smart here as the system generates questions intelligently. There are many definitions of intelligence but we use the word intelligence in the sense that the system produces a new quiz query according to the result of the last answer query and the same query for something produces different queries for different readers. Questions vary from student to student of the same quiz while they sit the Questions at the same time. Teachers can use OIQS to evaluate students effectively, efficiently and effectively.

Any university, college, school or educational institution can use this program in their organization to conduct inquiries. Today they are effective and effective ways to test remote students. One of the main benefits of our program is automatic marking, that is, teachers do not have to scan the text for answers as they do for written questions. It saves valuable time for teachers. On the other hand, students can get points depending on their level of achievement and will give feedback about the student's weaknesses.

In recent years, the use of online questionnaire systems has become increasingly popular due to the pressure of increasing class sizes and the need for more efficient and effective means of screening remote students. It is because of the popularity of distance education.

And now institutions of higher learning have been using the book system to make inquiries. In recent years a few lecturers at some institutions have used the web-based M.C.Q. questionnaire. In fact, it depends on how much website technology a country makes or uses. For example, in Pakistan it is rare to use a web-based questionnaire. But in the case of the U.S.A. It is a common offense to use a web-based questionnaire

1.1 Manual Quiz System

In the early days it was the most popular means of assessing a student. By now the program has become very popular with both students and teachers. In this system there are a few problems we deal with in the usual way. Some of these problems are:

Literature program requires pens / pencils and paper.

The teacher needs to spend time exploring the text.

The student needs to wait for his or her results until the teacher has finished examining the script.

These are the most common problems with a book-based questionnaire for each query period created. For these reasons the popularity of the book system is decreasing day by day and the online quiz system is replacing the book list.

1.2 Web- Based Quiz System

Now a day the web-based web application is becoming faster. As technology spreads around the world the automated system will also replace parts of the manual. Currently large institutions are working with their online questionnaire system at a very effective rate. As time goes on, the online quiz is creating interest in students' skills. In big cities and organizations people understand how an online query system is better and more efficient. Not long ago the program would be staged in this technological world. But the system also has some shortcomings. If the program had a smart part, it would be perfect.

Possible problems with a web-based questionnaire:

Can't properly evaluate the student.

Can't make the correct answer for students.

1.3 Functional Requirements

1. Teacher

1. Can create queries after logging in!

2. You can enter a course and enter a query with its options and answer during the query creation.

3. 10 Each question mark needed to be completed.

2. Learner

1. Can search Questions in their interest.

2. Click on the query id and be ready to start it by clicking the button.

3. After completing all queries, the result will be displayed automatically.

1.4. Non-Functional Requirements

1. Secure access to confidential information (user information). SSL can be used.

2. Availability of 24 7 7

3. Browser testing and support for IE, NN, Mozilla, and Firefox

4. Reports sent to .XLS, .PDF

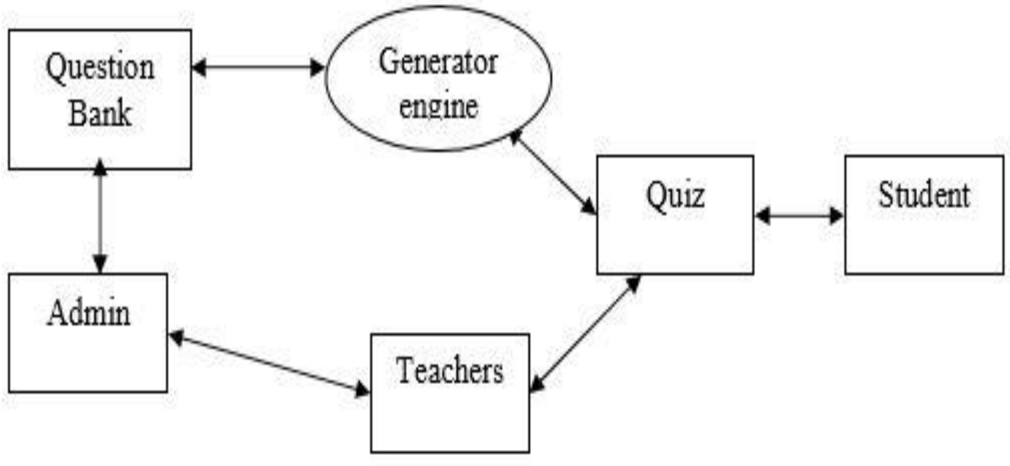
5. Create a detailed diagram of the UML (Part, Sequence, Class) of the program and its sub-components

2. System Overview

ONLINE Intelligent Quiz SYSTEM is a Web based M.C.Q. Quiz System for testing students. The program is developed through the focus of educational institutions such as the university, the college and the school. The student can attend a question from anywhere in the world by using the program.

This administrative source is responsible for the registration of both teachers and students. Teachers explain the number of questions, the duration of the questions, the schedule of questions etc. According to this information questions will be made for students. Teachers are responsible for adding and editing questions from the question bank. They can add a question to a question bank at a different level depending on their level of difficulty. Generator engine generates queries through the Question bank. To generate the next query, the query constructor checks the previous query response submitted

by the student and depending on the result the Question General determines the level of difficulty in the next question and produces it. It also provides feedback to students. The student may appear for questions when his / her time is up. However, before attending a registration ceremony, he or she must be enrolled in the course by the centre manager. At the end of the question they can see the correct answers and their performance. Teachers can assess the student at the end of the question.



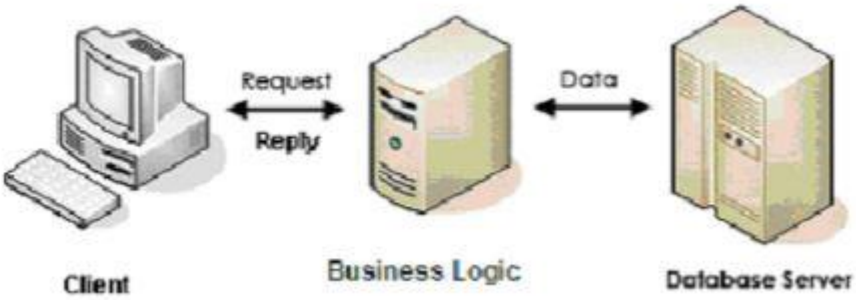
With the same text generator questionnaire, you can generate a different query for a different reader. That is a question that a student may have to ask for the same questions while appearing to the questions at the same time.

To design software, we use a 3-tier architecture. As a three-dimensional architecture divides the entire system into three parts, so designing a system we divide our system into three subsystems. These are the Client Tier / Client, Question Bank, Teachers, Student Quiz, Administrator, Generator Engine, Subdomain, Bus Logic Tier / Business Logic subsystem, and Database Tier / Database subsystem.

3. SYSTEM ARCHITECHTURE

The system refers to both Hardware and software. System architecture describes the collection of interconnected networking environments where the software will eventually operate. System architecture consists of

Hardware nodes, connections between locations, user locations, input and output.

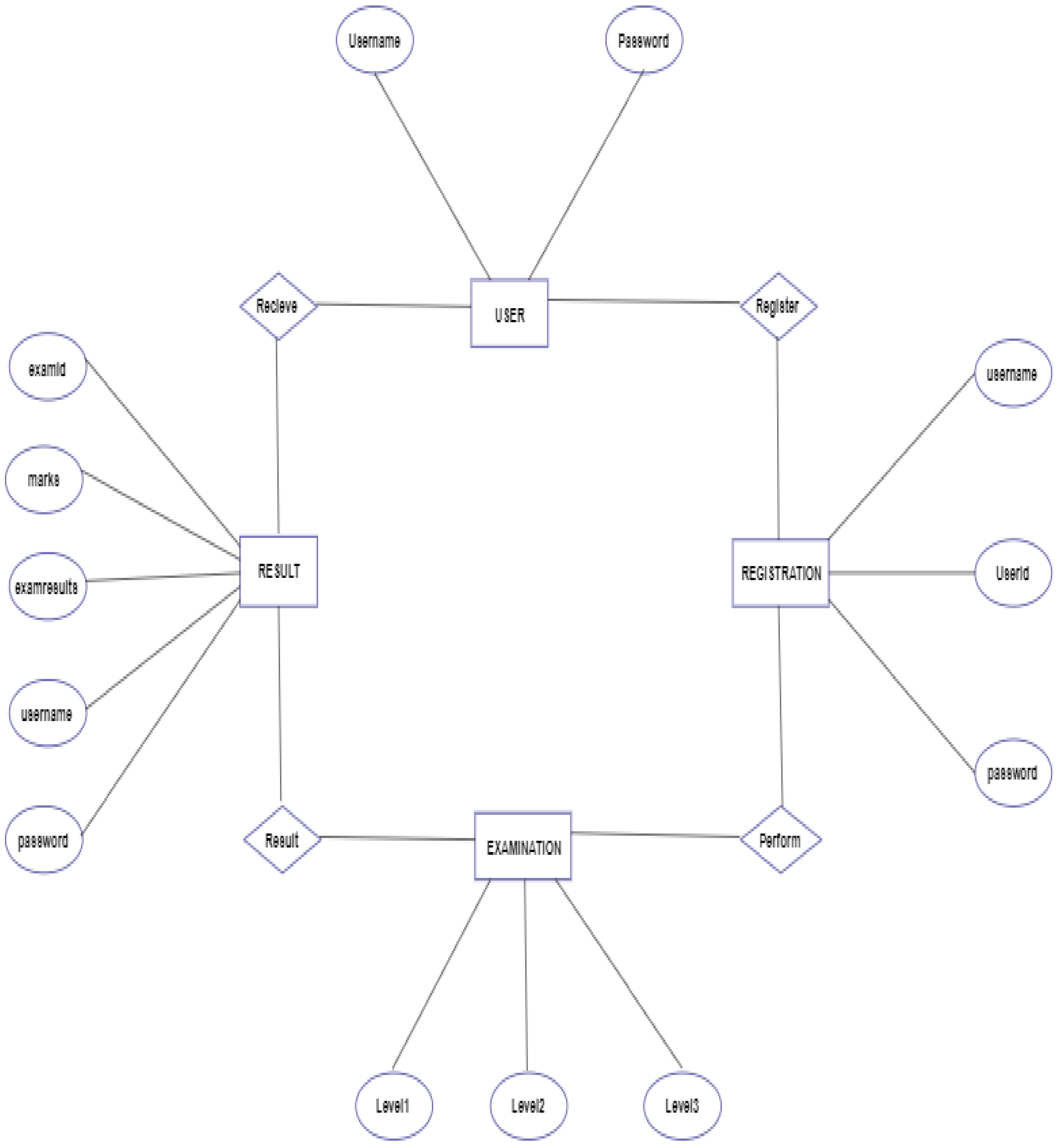


Software structure

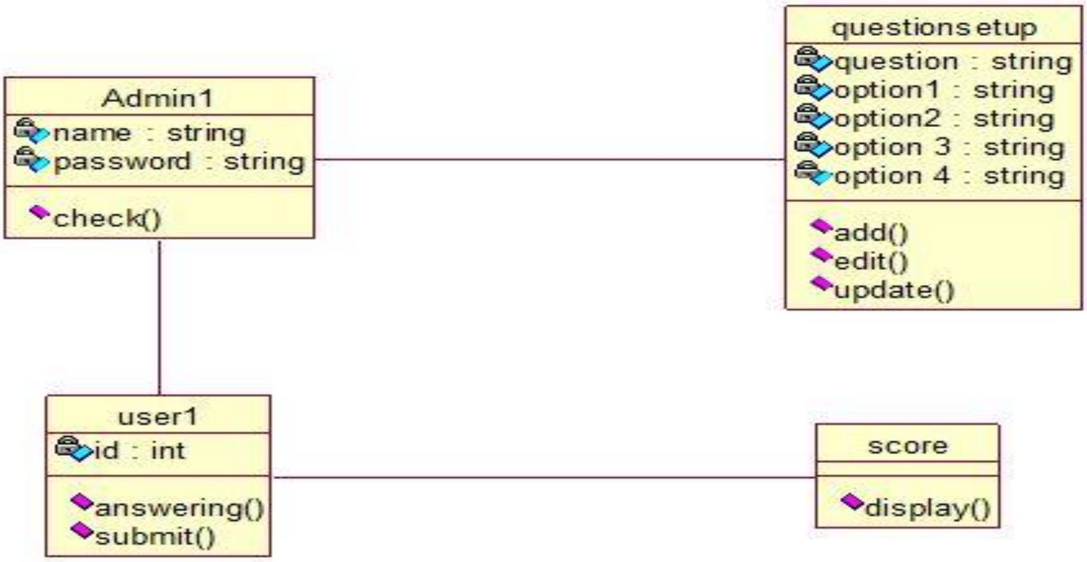
We use the three-dimensional architecture of our design. Because 3-tier Architecture enhances performance, flexibility, robustness, resilience, and robustness while hiding the complexity of shared processing for users / customers.

4. ARCHITECTUREAL DIAGRAMS

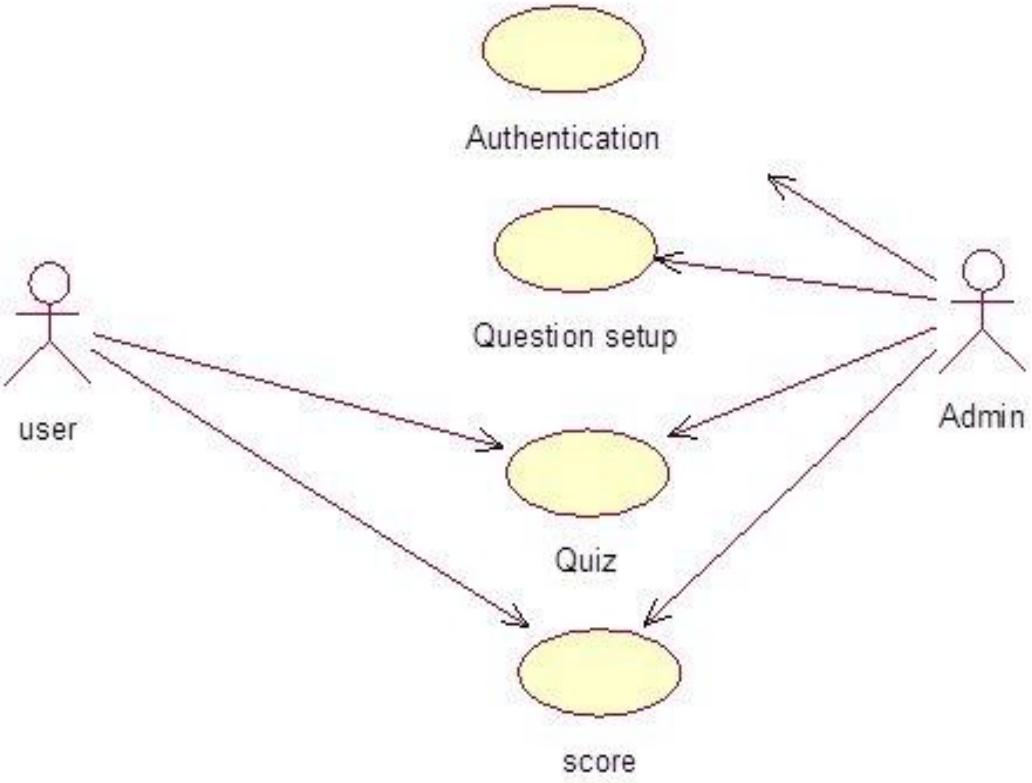
4.1. E-R Diagram



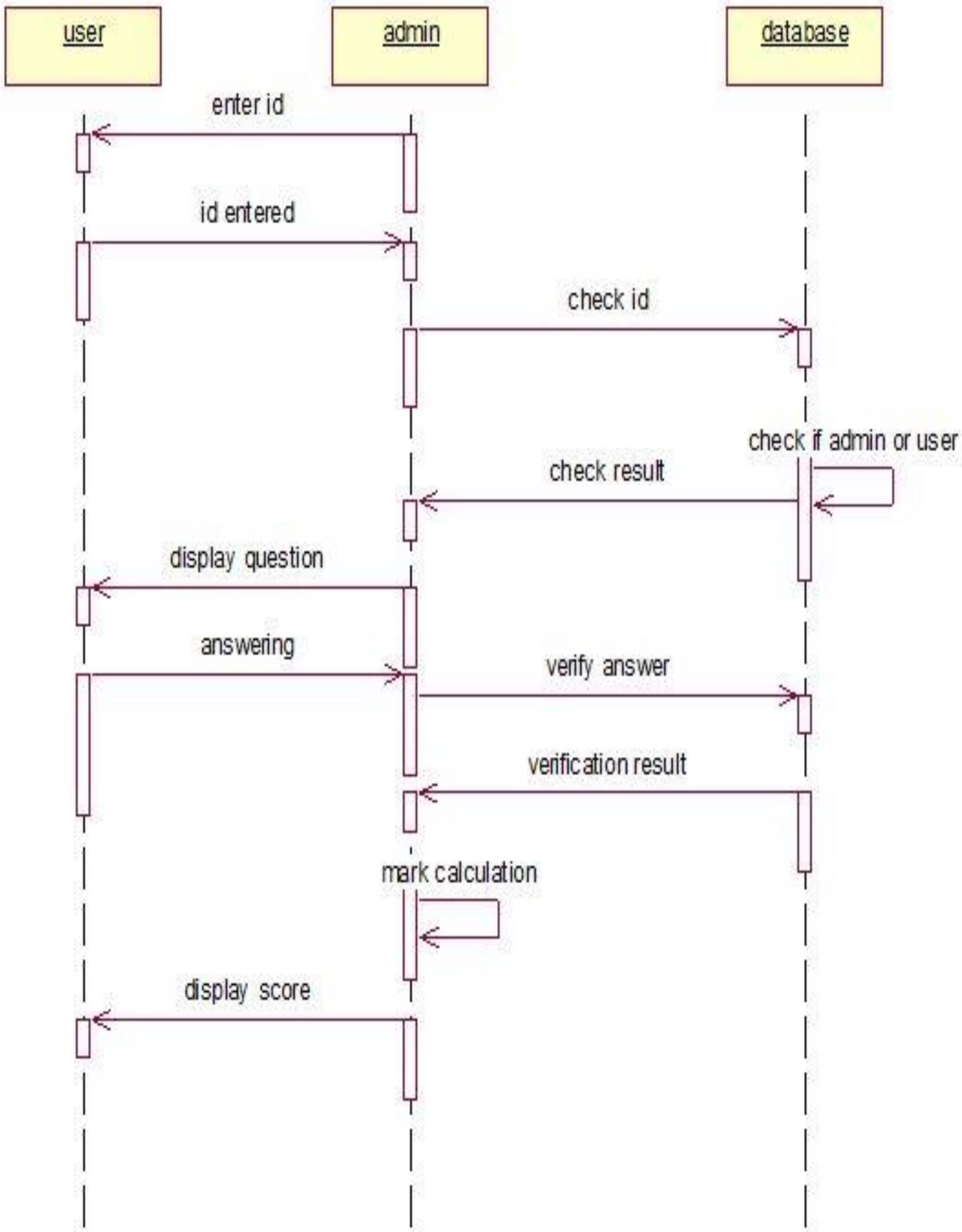
4.2. Class Diagram



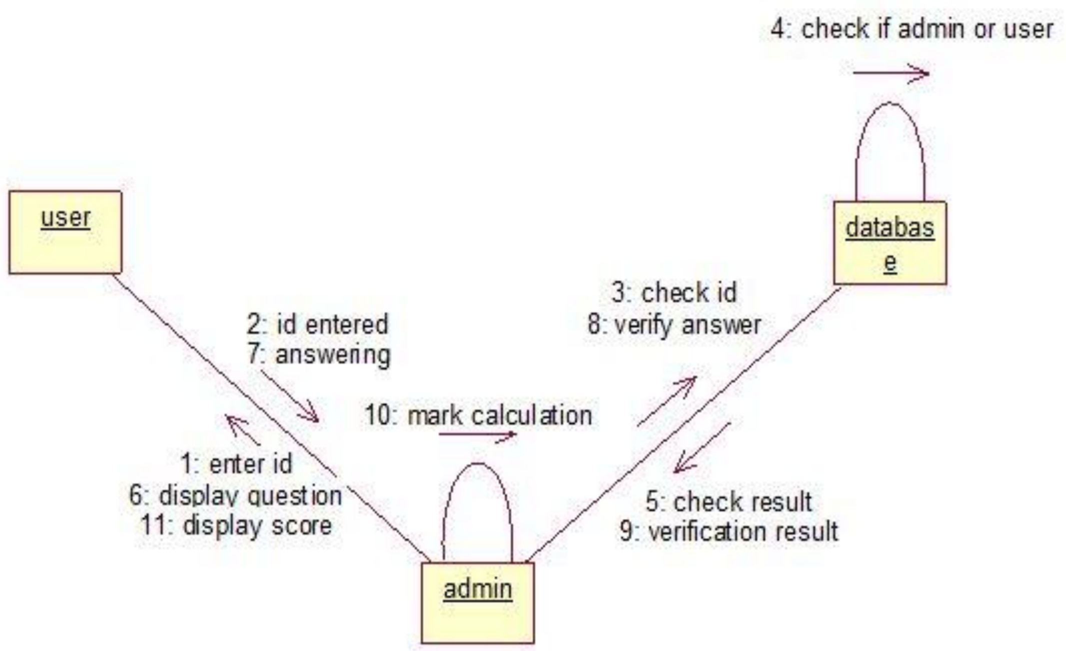
4.3. Use- Case Diagram



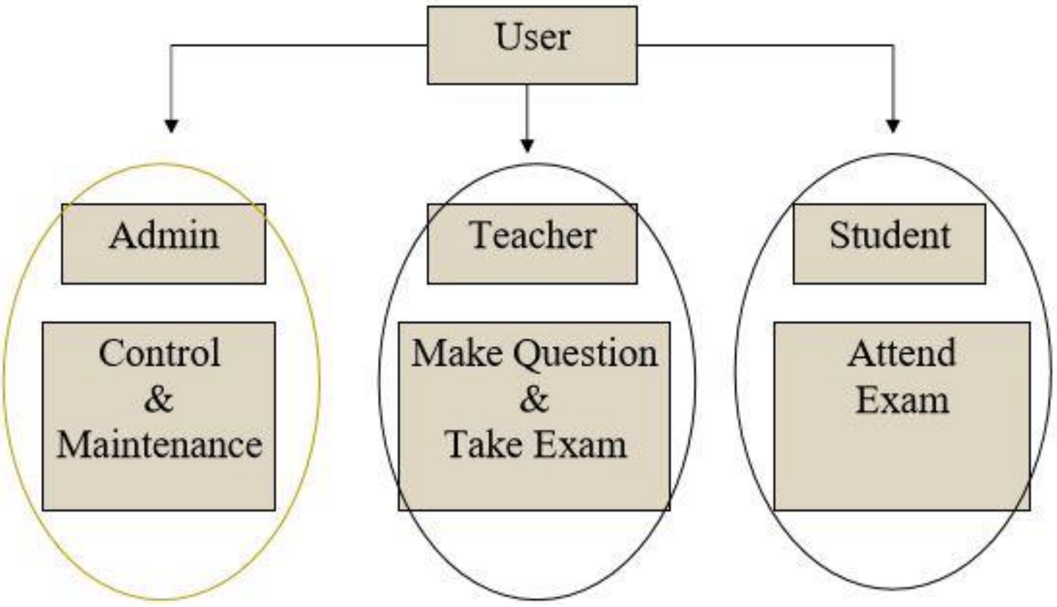
4.4. Sequence Diagram



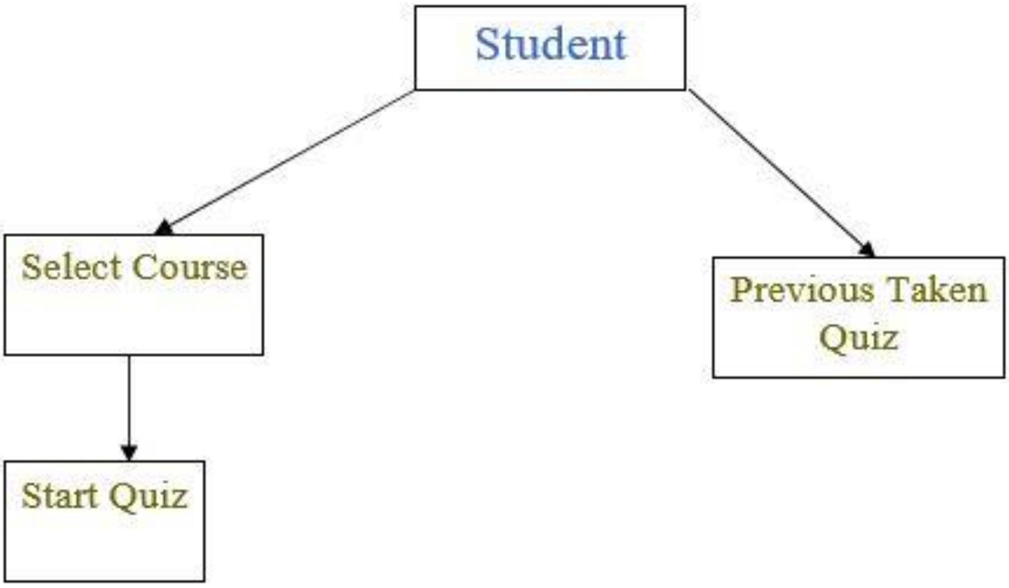
4.5. Collaboration Diagram



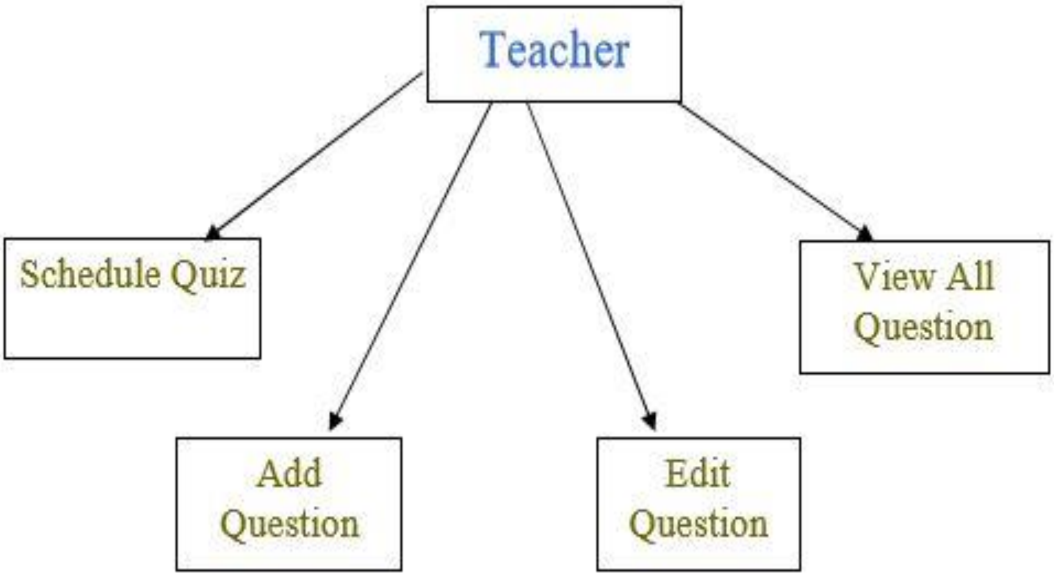
4.6. Tree Structure View



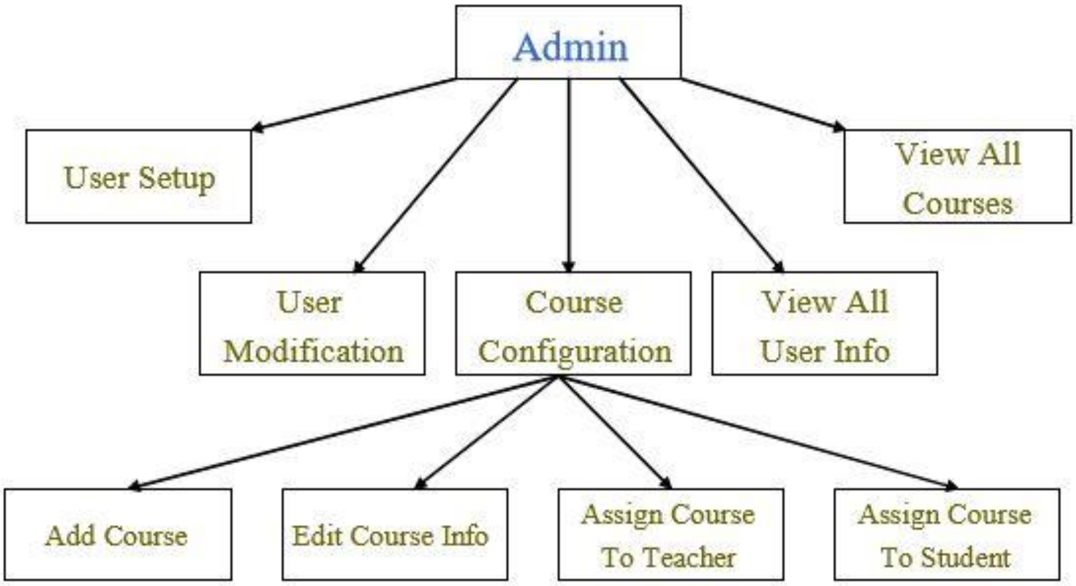
4.6.1. Tree Structure View of Student Module



4.6.2. Tree Structure View of Teacher Module



4.6.3. Tree Structure View of Admin Module



5. IMPLEMENTATION AND CODING

Creating Menu for Home Page

```
< div id='cssmenu'>
< ul>
   < li class=''>< a href='${pageContext.request.contextPath}'>< span>Home< /span>< /a>< /li>
   < li>< a href='${pageContext.request.contextPath}/login'>< span>Login< /span>< /a>< /li>
   < li>< a href='${pageContext.request.contextPath}/register'>< span>Register< /span>< /a>< /li>
   < li class='#'>< a href='#'>< span>Submit a Question< /span>< /a>< /li>
   < li class='#'>< a href='#'>< span>Feedback< /span>< /a>< /li>
   < li>< a href='#'>< span>Contribute< /span>< /a>< /li>
   < li>< a href='#'>< span>Contact us< /span>< /a>< /li>
< /ul>
< /div>
```

Checking whether the user is logged in or not

```
< c: if test='${not empty sessionScope.user}'>

< div style="position:absolute;top:70px;left:1100px">
Logged as < a href="#" class="button username">${sessionScope.user}< /a>
< /div>

< div style="position:absolute;top:70px;left:1300px">
< a href='${pageContext.request.contextPath}/logout'>Logout< /a>
< /div>
< /c:if>
```

Showing the quiz images on home page

```
< div style="position:absolute;left:120px;top:60px">
< table cellpadding="0" cellspacing="50">

< tr>
< td>< a href="takeExam?test=java">< img height="200" width="200"
src="${pageContext.request.contextPath}/images/java.png"/>< /a>< /td>
< td>< a href="takeExam?test=javascript">< img height="200" width="200"
src="${pageContext.request.contextPath}/images/javascript.png"/>< /a>< /td>
< td>< a href="takeExam?test=sql">< img height="200" width="200"
src="${pageContext.request.contextPath}/images/sql-logo.png"/>< /a>< /td>
< td>< a href="takeExam?test=python">< img height="200" width="200"
src="${pageContext.request.contextPath}/images/python.jpg"/>< /a>< /td>
< /tr>

< tr>
< td>< a href="takeExam?test=css">< img height="200" width="200"
src="${pageContext.request.contextPath}/images/css.jpg"/>< /a>< /td>
< td>< a href="takeExam?test=php">< img height="200" width="200"
src="${pageContext.request.contextPath}/images/php-logo.jpg"/>< /a>< /td>
< td>< a href="takeExam?test=linux">< img height="200" width="200"
src="${pageContext.request.contextPath}/images/logo-linux.png"/>< /a>< /td>
< td>< a href="takeExam?test=mongodb">< img height="200" width="200"
src="${pageContext.request.contextPath}/images/mongodb_logo.png"/>< /a>< /td>
```

```
< /tr>


< /table>
< /div>
```

Creating the User Registration Page

There is nothing weird on the signup page; just an HTML form waiting for the user to provide their name, email and password. Once we have received that, we transfer this to the Registration Control register to create an account.

Note: We do not do any password verification that should contain at least 8 characters with at least one character, one number and a special character. We'll do that in a future post, as we expand this program.

Registration Code

```
protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
                String username=request.getParameter("username");
                String email=request.getParameter("email");
                String password=request.getParameter("password");

                Connection
con=DatabaseConnectionFactory.createConnection();

                try
                {
                 Statement st=con.createStatement();
                 String sql = "INSERT INTO users values
('"+username+"','"+password+"','"+email+"')";
                                System.out.println(sql);
```

```
                         st.executeUpdate(sql);
                        }catch(SQLException sqe){System.out.println("Error : While
Inserting record in database");}
                        try
                        {
                         con.close();
                        }catch(SQLException se){System.out.println("Error : While
Closing Connection");}
        request.setAttribute("newUser",username);
                        RequestDispatcher
dispatcher=request.getRequestDispatcher("/WEB-INF/jsps/regSuccess.jsp");
                        dispatcher.forward(request, response);
             }
```

Getting Database Connection

In this application we used the MySQL database to store user information. To access a database connection, we described a static method to create a connection to the Database Connection Factory class, where a database is stored.

Users' table

```
create table users(username varchar(50),email varchar(50),password
varchar(50))
```

If you are working with some other database like Oracle you have to change the properties of the DatabaseConnectionFactory class accordingly.

DatabaseConnectionFactory.java

```
public class DatabaseConnectionFactory {
        private static String dbURL="jdbc:mysql://localhost/quiz";
        private static String dbUser="root";
        private static String dbPassword="";

        public static Connection createConnection()
        {
                 Connection con=null;
                try{
                        try {
                                  Class.forName("com.mysql.jdbc.Driver");
                                }
                                catch(ClassNotFoundException ex) {
                                  System.out.println("Error: unable to load
driver class!");
                                  System.exit(1);
                                }
                   con =
DriverManager.getConnection(dbURL,dbUser,dbPassword);
                  }
                 catch(SQLException sqe){ System.out.println("Error: While
Creating connection to database");sqe.printStackTrace();}
                return con;
        }

}
```

Creating the Login Page

Login page is very much similar to registration page where we are providing two input fields asking user to provide a username and password. Once we get the username and password entered by the user, we pass it to Login Controller to authenticate user.

Login Validation Code

```
protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {

            String username=request.getParameter("username");
            String password=request.getParameter("password");

            Connection
con=DatabaseConnectionFactory.createConnection();
            ResultSet set=null;
            int i=0;
            try
            {
             Statement st=con.createStatement();
             String sql = "Select * from  users where
username='"+username+"' and password='"+password+"' ";
                        System.out.println(sql);
             set=st.executeQuery(sql);
             while(set.next())
             {
                   i=1;
             }
             if(i!=0)
             {   HttpSession session=request.getSession();
                session.setAttribute("user",username);
```

```
                                    RequestDispatcher
rd=request.getRequestDispatcher("/WEB-INF/jsps/home.jsp");
                                    rd.forward(request, response);


                        }
                        else
                        {   request.setAttribute("errorMessage","Invalid username or
password");
                                    RequestDispatcher
rd=request.getRequestDispatcher("/WEB-INF/jsps/login.jsp");
                                    rd.forward(request, response);
                        }
                        } catch(SQLException sqe) {System.out.println("Error : While
Fetching records from database");}
                        try
                        {
                         con.close();
                        }catch(SQLException se){System.out.println("Error : While
Closing Connection");}
            }
```

Main Controller for the Application

It is the Main Controller where we have written the code to redirect the user to appropriate page according to the incoming request URL.

```
@WebServlet(urlPatterns = { "/login", "/register", "/takeExam", "/logout" })
public class MainController extends HttpServlet {
            private static final long serialVersionUID = 1L;


            protected void doGet(HttpServletRequest request,
            HttpServletResponse response) throws
ServletException, IOException {

        String applicationContextPath = request.getContextPath();

        if (request.getRequestURI().equals(applicationContextPath +
"/")) {
            RequestDispatcher dispatcher = request
                    .getRequestDispatcher("/WEB-
INF/jsps/home.jsp");
            dispatcher.forward(request, response);
        } else if (request.getRequestURI(). equals(
                applicationContextPath + "/login")) {
            RequestDispatcher dispatcher = request
                    .getRequestDispatcher("/WEB-
INF/jsps/login.jsp");
            dispatcher.forward(request, response);
        } else if (request.getRequestURI().equals(
                applicationContextPath + "/register")) {
            RequestDispatcher dispatcher = request
                    .getRequestDispatcher("/WEB-
INF/jsps/register.jsp");
            dispatcher.forward(request, response);
        } else if (request.getRequestURI().equals(
                applicationContextPath + "/takeExam")) {
            request.getSession().setAttribute("currentExam", null);

            String exam = request.getParameter("test");
            request.getSession().setAttribute("exam", exam);
```

```
    System.out.println(request.getSession().getAttribute("user"));
            if (request.getSession().getAttribute("user") == null) {

    request.getRequestDispatcher("/login").forward(request,
                        response);

            } else {
                RequestDispatcher dispatcher = request
                        .getRequestDispatcher("/WEB-
INF/jsps/quizDetails.jsp");
                dispatcher.forward(request, response);
            }
        } else if (request.getRequestURI().equals(
                applicationContextPath + "/logout")) {
            request.getSession().invalidate();
            RequestDispatcher dispatcher = request
                    .getRequestDispatcher("/WEB-
INF/jsps/home.jsp");
            dispatcher.forward(request, response);
        }

    }

}
```

Implementing the Logout Functionality

Once the user clicks on logout, link session is invalidated and all the objects bind in the session are removed.

```
request.getSession().invalidate();
```

Storing the Quiz questions

Note that we have stored the questions in separate XML files, not in the database.

```
< quiz>
  < title>MongoDB Quiz (01/09/2015)< /title>
    < questions>
      < question>
        < quizquestion>MongoDB is a < /quizquestion>
        < answer>Relational Database< /answer>
        < answer>Object Relational Database< /answer>
        < answer>Graph Database< /answer>
        < answer>Document Database< /answer>
        < correct>3< /correct>
      < /question>

      < question>
        < quizquestion>What is the name of MongoDB server ?< /quizquestion>
        < answer>mongoserver< /answer>
        < answer>mongod< /answer>
        < answer>mongodb< /answer>
        < answer>mongo< /answer>
        < correct>1< /correct>
      < /question>

      < question>
        < quizquestion>What is the name of MongoDB client ?< /quizquestion>
        < answer>mongo< /answer>
    < answer>mongod< /answer>
    < answer>mongodb< /answer>
    < answer>mongo-client< /answer>
    < correct>0< /correct>
   < /question>
< /questions>
< /quiz>
```

How to Read the Questions Stored in XML File

To read the queries in an XML file we create a document that represents an XML file containing queries. Each time a user clicks on the next or previous button we call the set of Query (int I) method, we provide a query for the query we want to read and at the same time that query is stored in the Array Query List.

How to Represent a Question?

Quiz is a class that represents a single query; each query will have a number, query statement, options and one index for the correct option.

QuizQuestion.java

```
public class QuizQuestion {


      int questionNumber;
      String question;
      String questionOptions[];
      int correctOptionIndex;
```

```
public String getQuestion()
{
	return question;
}

public int getQuestionNumber()
{
	return questionNumber;
}

public void setQuestionNumber(int i)
{
	questionNumber=i;
}

public int getCorrectOptionIndex()
{
	return correctOptionIndex;
}

public String[] getQuestionOptions()
{
	return questionOptions;
}

public void setQuestion(String s)
{
	question=s;
```

```
        }
        public void setCorrectOptionIndex(int i)
        {
                correctOptionIndex=i;
        }
        public void setQuestionOptions(String[]s)
        {
                questionOptions=s;
        }

}
```

Note that since this is a web app, many users will be taking tests at the same time. We must make sure that the tests of the other user do not fall into the other user's tests. For example, one user may have just started a Java test and another user is in question 5 for a SQL test; we have to treat them as two separate tests.

To do that we will maintain the status of each test using the session.

When the user clicks on the test button to start the tests, we will create a new test instance that exceeds the e.t test type. Java, PHP, CSS etc. Therefore, each user will have a different Exam lesson example (showing each exam).

Let's see what's in the test class

```
public class Exam {
        Document dom;
        public int currentQuestion=0;

        public Map selections=new LinkedHashMap();
        public ArrayList questionList = new ArrayList(10);
```

```
            public Exam(String test) throws
SAXException,ParserConfigurationException,IOException,
URISyntaxException{
                        dom=CreateDOM.getDOM(test);
            }


     // code
}
```

Note that to track the current question in the exam we have currentQuestion property in exam class.

Handling the Entire Exam

Exam control is the main controller from where we control the test. Here we save the choice of users (what the user answered with a question) on Map. The Exam Controller also allows the user to go through the queries by clicking the next and previous button, the back end is the Exam Controller that performs the function call calls to retrieve queries and save user responses.

Submitting the Exam and Evaluating Exam Result

When a user clicks the end button, the Exam Controller calls a method to calculate the Exam () result that passes the Exam object, calculates Result () compares the user's answers to the correct query option and returns how many answers the user has received.

```
public int calculateResult(Exam exam){
                        int totalCorrect=0;
                        Map<Integer,Integer> userSelectionsMap=exam.selections;

                        List userSelectionsList=new ArrayList(10);
```

```
        for (Map.Entry<Integer, Integer> entry
:userSelectionsMap.entrySet()) {
            userSelectionsList.add(entry.getValue());
        }
        List questionList=exam.questionList;
        List correctAnswersList=new ArrayList(10);
        for(QuizQuestion question: questionList){

    correctAnswersList.add(question.getCorrectOptionIndex());
        }

        for(int i=0;i<selections.size();i++){
            System.out.println(userSelectionsList.get(i)+" ---
"+correctAnswersList.get(i));
            if((userSelectionsList.get(i)-
1)==correctAnswersList.get(i)){
                totalCorrect++;
            }
        }

        System.out.println("You Got "+totalCorrect+" Correct");
        return totalCorrect;
    }
```

2. Advantages of the Project

• Students can appear in Questions from anywhere in the world.

• No interaction with pencil and paper.

- Questions are tactfully done.

- The system generates a new query depending on the results of the query that was last answered.

- The system generates different queries for different learners in the same item queries

- Teachers can use the program to assess students effectively, effectively and holistically

- It is an effective and effective way to test remote learners.

- There is provision for automatic marking, that is, teachers do not have to check the text of the answers as they do in written questions.

- Important teacher time is saved.

- The program will provide feedback for the learner while treating the soft side.

- The program uses distance education.

6. CONCLUSION

In this thesis, we focus on the automated system, which replaces the upcoming query system. But the interesting thing is that the thesis is not just an online questionnaire; it has its own intelligent power. This has actually been our intended feature of our thesis. Basically, we tried to introduce a way of assessing student performance and student feedback on what was done online. Here, questions arise based on the student's previous response. We have applied it to our concept and it is effective in our system. In this system, an educational institution can have their assessment process done automatically. It provides the authority to enrol teachers in a variety of subjects and also provides the facility for students to enrol in their exams using OIQS. Although this is an MCQ-based query system, it is an online MCQ complete system that provides many user centres, as the latest query system should have. This program is primarily for an educational institution but may be for other exam programs such as job interviews, a questionnaire and other terms.

7. REFERENCES

1. http://phplens.com/phpeverywhere/node/view/15
2. http://www.mysql.com/why-mysql/
3. http://student.bu.ac.bd/~shadid/
4. Object-Oriented Software Engineering book by Timothy C. Lethbridge and Robert Laganiere
5. www.google.com
6. www.scribd.com
7. www.wiki.com