**CONTENTS**

|  |  |  |
| --- | --- | --- |
| NO | DESCRIPTION | PAGE NO. |
| 1 | Synopsis | 2 |
| 2 | Acknowledgment. | 5 |
| 3 | Project Specification. | 6 |
| 4 | Software Study Asp.Net. | 7 |
| 5 | Intro & Overview of the project. | 10 |
| 6 | User Interface. | 13 |
| 7 | Software Architecture | 14 |
| 8 | Tools & Platform. | 15 |
| 9 | Process Flowchart:  .  ~ D.F.D Diagram.  ~ Use Case Diagram. | 16  17  18 |
| 10 | Data Dictionary (Tables of database). | 19 |
| 11  12 | Screen Shots of Website.  Source Code | 24  33 |
| 13  14 | Validations in forms.  Conclusion | 46  46 |
| 15 | Bibliography. | 47 |

**SYNOPSIS**

**Project Title:** RNH Recruitment Process

Team member: **Nisha raiyani**

**Payal basiya**

**Introduction:-**

*Online Exams* is being launched because a need for a destination that is beneficial for both

Company and Candidate. With this site, company can recruit candidate easily through online exams.company will conduct exam and candidate can give exams and view their results. This site is an attempt to remove the existing flaws in the manual system of conducting exams.

The purpose of on-line test simulator is to take online test in an efficient manner and no time wasting for checking the paper. The main objective of on-line test simulator is to efficiently evaluate the candidate thoroughly through a fully automated system that not only saves lot of time but also gives fast results.

For candidate they give papers according to their convenience and time and there is no need of using extra thing like paper, pen etc.

**Purpose:-**

***Online Exams System*** fulfills the requirements of the company to conduct the exams online.

They do not have to go to any physical destination being able to

conduct exams online. They just have to login on the site and enter the exam details and the

lists of the candidate which can appear in the exam.

candidate can give exam without the need of going to any physical destination. They can view

the result at the same time.

Thus the purpose of the site is to provide a system that saves the efforts and time of both the

company and the candidate.

**Objective:**

This project is based on the RDBMS technology; the main objective of this project is

to computerize the manual system & reduce the time consumption.

In other words we can say that our project has the following objectives:

•Make all the system computerize

•Reduce time consumption

•Reduce error scope

•Easy operate all the work

•No paper work requirement

**Project Category:**

This project as title “RNH Recruitment Process” is comes under the RelationalDatabase Management System (RDBMS).This application is developed with the help of “Visual studio 2010 and SQL Server 2008. This application can also be run on thenetwork environment so it can be said as network application.

**Tools/Plateform:**

This project is developed using the tools, which are most suited for development of the

Application Package. These tools are as follows: -

1.Visual studio 2010 (For front end)

2.SQL Server 2008 (For Database Storage as Back end)

**Hardware & Software Requirement:**

**Software requirement:**

* **Operating System:-** Windows Xp,Windows8
* **Front end tool:-** Asp.net with C#
* **Back end tool:-** Sql Server 2008
* **Any other tool:-** Internet Explorer and Any Browser

**Hardware requirement:**

* Main Processor …………………. Pentium IV
* Visual display unit ……………….Colourmonitor
* RAM ……………………………..1 GB
* Hard disk ………………………..120 GB

**A Bibliography:**

* Comdex .Net Programming-Vikas Gupta.
* ASP.NET WROX Profrssion.
* ASP.NET Complete Reference-Matthew MacDownald.
* ASP.NET with C# Complete Reference.

**Future scope of the project:**

Nothing is perfect in this world. So, we are also no exception. Although, we have tried our best to present the information effectively, yet, there can be further enhancement in the Application. We have taken care of all the critical aspects, which need to take care of during the development of the Project. Like the things this project also has some limitations and can further be enhances by some one, because there are certain drawbacks that do not permit the system to be 100% accurate.

**ACKNOWLEDGEMENT**

Now that I have successfully completed **“ENTERPRISE COMPUTING MINI PROJECT”**, I would take this opportunity to thank one and all who have in some form or other been responsible for the completion of this Project.

We are indebted to our management and faculty of **Sambhram Academy of Management Studies** , for giving a good opportunity and necessary support extended to this course.

I take this opportunity to express my deep sense of gratitude to our principal **Dr. K.C. John** for the valuable guidance, keen interest and helpful criticism, during the course of studies.

I once again convey my sincere gratitude to **Dr. Antony Joseph** (Director of MCA Dept.) and **Mrs. Nirmala** (H.O.D of MCA Dept.) for their kind patronage and all the facilities provided.

I am deeply indebted to our Subject in-charge **Mr.Nandish**  for his constant encouragement and sustained motivation during the various phases of the lab work.

We express our sincere gratitude to our respected head of the department, for providing us with facilities to complete the project in due course of time.

Our heartfelt thanks to all the lecturers for their support, guidance. Encouragement, inspiration and co-operation during this project period.

Last but not the least, we would like to thank the entire computer science department, friends and our families for their help and co-operation without which this project wouldn’t have been a success.

#### PROJECT SPECIFICATIONS

#### Project Title : RNH Recruitment Process

**Front End Tool :** ASP.NET 2010

**Back End Tool :** Sql Server 2008

**Documentation Tool :** Microsoft Word 2007

**Operating System :** Microsoft Windows XP,Windows 8

**Web Server :** ASP.NET

**Web Browser :** Internet Explorer 6.0 or Opera or Mozila firefox

SOFTWARE STUDY

#### About ASP.NET

#### ASP.NET is the .NET framework layer that handles Web requests for specific types of files, among those are .aspx and .ascx, and .asmx extensions as well as several others (you can get the full list by checking the Application Configuration properties for the Web site in the Internet Services Manager application). The ASP.NET engine provides a robust object model for creating dynamic content and is loosely integrated into the .NET framework. This integration makes it easy to change the implementation when the .NET framework migrates to platforms other than Windows.

#### ASP.NET is implemented in an assembly that exposes classes and objects that perform predetermined specific tasks. If you are familiar with "classic" ASP (the versions of ASP that preceded .NET), you'll find that your approach to programming in ASP.NET is somewhat different, but the concepts behind building a Web application are much the same. If you're not familiar with classic ASP, so much the better— you won't have as much information to forget!

#### ASP.NET programs are centralized applications hosted on one or more Web servers that respond dynamically to client requests. The responses are dynamic because ASP.NET intercepts requests for pages with specific extensions, for example .aspx or .ascx, and hands off the responsibility for answering those requests to just-in-time (JIT) compiled code files that can build a response on-the-fly.

#### The server doesn't "serve" any of these file types— it returns the appropriate content type to the client. The configuration file types contain initialization and settings for a specific application or portion of an application. Another configuration file, called machine.web, contains machine-level initialization and settings. The server ignores requests for Web files, because serving them might constitute a security breach. Server to load, parses, and executes code to return a dynamic response.

#### For Web Forms, the response usually consists of HTML or WML. For Web Services, the server typically creates a Simple Object Access Protocol (SOAP) response. While SOAP requests are inherently stateless and can thus execute immediately, Web Forms are stateful by default. Web Forms maintain state by round-tripping user interface and other persistent values between the client and the server automatically for each request. In

#### Form can use View State, Session State, or Application State to maintain values between requests. It is possible (but not the default) to take advantage of ASP.NET's state maintenance architecture from a Web Service, but for performance reasons, you should generally avoid doing so. Both Web Forms and Web Services requests can take advantage of ASP.NET's integrated security and data access through ADO.NET and can run code that uses system services to construct the response.

#### The major difference between a static request and a dynamic request is that a typical Web request references a static file. The server reads the file and responds with the contents of the requested file.

#### With ASP.NET there's no such limitation. You don't have to respond with an existing file— you can respond to a request with anything you like, including dynamically created HTML, Extensible Markup Language (XML), graphics, raw text, or binary data— anything. Capability, by itself, is nothing new— you've been able to create CGI programs, Java Server Pages, classic ASP pages, ColdFusion, and Net- Objects Fusion pages for quite some time.

#### All these technologies give you the capability to respond to an HTTP request dynamically. So what are the differences? Unlike classic ASP, ASP.NE uses .NET languages. Therefore, you have access to the full power of any .NET assembly or class in exactly the same way as you do from any other Windows application written in C#. In this sense, ASP.NET is similar to early compiled CGI programs, but with CGI, a separate copy of the program had to be loaded and executed for each request. ASP.NET code exists in multithreaded JIT-compiled DLL assemblies, which can be loaded on demand. Once loaded, the ASP.NET DLLs can service multiple requests from a single in-memory copy.

#### An ASP.NET supports all the .NET languages (currently C#, C++, VB.NET, and JScript, but there are well over 20 different languages in various stages of development or deployment for .NET), so you will eventually be able to write Web applications in your choice of almost any modern programming language.

#### In addition, there are open source groups working with Intel and Hewlett-Packard to support .NET on various flavors of Unix and Linux. Java Server Pages support only Java, but because Java now has a wide support base, that's not much of a limitation. Java Servlets are more like ASP.NET but offer little support for state maintenance, Web Services, or XML. In addition, no Java design environment competes in features and quality to Visual Studio.

#### A Classic ASP supported several scripting languages, although in practice, VBScript and JScript were by far the most prevalent. Although the scripting languages you could use with classic ASP were untyped, interpreted, and not particularly powerful, you could extend ASP's basic functionality by writing DLLs in any COM-compliant language. Another ASP.NET competitor, ColdFusion, uses

#### ColdFusion Markup Language (CFML) tags, which have a powerful but limited set of capabilities; however, you can extend CFML with custom programming in C++ or Java.

#### Microsoft was able to draw on millions of hours of developer experience with classic ASP, so in addition to huge increases in speed and power, ASP.NET provides substantial development improvements, such as seamless server-to-client debugging, automatic validation of form data, and a programming model very similar to that of a Windows application.

Introduction and Overview of the Project:

* This Web Application provides facility to conduct online

examination world wide.

* It saves time as it allows number of students to give the exam at a

time and displays the results as the test gets over, so no need to

wait for the result. It is automatically generated by the server.

**Purpose:**

* This Web Application provides facility to conduct online

examination world wide.

* It saves time as it allows number of students to give the exam at a

time and displays the results as the test gets over, so no need to

wait for the result. It is automatically generated by the server.

* Administrator has a privilege to create, modify and delete the test

papers and its particular questions.

* User can register, login and give the test with his specific id, and
* can see the results as well.

**Specific Requirements:**

User interface Requirements:

Visual studio 2010 should be the front-end tool with a good I/P and O/P design and SQL Server 2008 should be the back-end tool.

**Functional Requirements:**

* The security of the system should be well provided i.e. there should be appropriate privileges depending on the type of the user.
* The performance of the system should be good enough to handle large data.
* The system should have customer registration possible through online registration and also through windows if in any case the user is not willing to register online.
* The new system should be able to reduce the manual work as much as possible and necessary reports should be generated from the computer.
* Most of the records must be saved in the database itself thereby reducing the manual work of saving useful information.
* Most of the errors should be efficiently caught thereby generating an error-free system. It should be easy to implement, modify and maintain by any of the users who are even without any kind of programming knowledge.

**Requirements of the User:**

For operating the system, a user is required to have some basic understanding of Windows operating system and some basic knowledge of computers. Other than that, some basic knowledge of the Graphical User Interface is desired.

**An overview of the project**

A system of this kind should be analyzed thoroughly first and then a step-by-step approach should be carried out.

A system study is step-by-step process used to identify and develop specific improvements in any organizations information system.

Following are the steps for system study.

1. System Analysis :

Following were the steps for the analysis of the system.

* Collection of Data
* ERD Diagrams
* General flow of the system and
* The end results

1. System Design :

In this process, the shortcomings of the existing system are taken into consideration first and an alternative solution to the pre-existing scenario is provided.

1. Programming Perspective:

Programming is all about disintegrating the design specifications into specific inputs, outputs, logic/comparison and storage retrieval operations.

1. Programming steps:

The required operation as determined by the flowcharts should be converted into a program using the C# Programming language. Programming steps can be summarized as follows:

1) Programming in the chosen language.

2) Editing and valuation check.

1. Implementation and Maintenance

After the completion of the code, exception handling also needs to be done and software testing also requires to be done. No software can ever be said complete as the requirements of the clients are ever-changing and every developer thinks in a different manner.

**User Interface**

Generally we see that a system Analyst interviews users, collects data and returns it back to the information service department so that a new system can be created. Most of the time, such type of approach is followed. But it is a fact that user participation plays a vital role for successful creation of a system. But most of the times, it is seen that the participation of the user is ignored to a certain extent and the system analyst is most of the times in charge of the project.

Instead of relying heavily on the analyst, the user’s idea should also be taken into consideration. This means that the user will decide the format of his design. Ultimately, the software has to be used by the user and thus the requirements of the user are to be taken into consideration. User participation in the designing of software reduces the amount of power transferred from the users to the information services department and thus avoids conflict with the Information and Services department.

By performing few of the Information department tasks, the user department reduces the task differences between their department and the information department. User participation also makes the users more familiar with the system which leads to a better understanding of the system consequently making the information services department and the user department mutually adaptable with each other.

A system design is of no use till the user is not able to understand that system. A successful completion of the system software with a proper output response is of no use till the user is not able to understand that system. So, user’s response is of equal importance as the only a user can fully be aware of the problems that he is facing. He can then interact with the Information services department and thus he can actively participate in the system design as per his own requirements.

**Software Architecture**

**Three Tier Architecture:**

This entire project of RNH Recruitment Process has been based on the Three-Tier Architecture. The system works on three layers namely:

1) The Presentation Layer

2) Data Access Layer (DAL) and

3) Business Logic Layer (BAL)

The three tier architecture is used when an effective distributed client/server design is needed that provides (when compared to the two tier) increased performance, flexibility, maintainability, reusability and scalability while hiding the complexity of the distributed processing from the user.

The user only understands the Presentation Layer. The user has no direct access to the DAL. The BAL is the core layer. The Presentation layer fires queries on the BAL which in turn passes them onto the DAL. DAL can only communicate with the BAL. Thus the entire project is divided into different layers. Thus the entire project of system development becomes smoother thereby reducing the complexities of the system development.

**Tools & Platform:**

TOOLS:-

ASP.NET:- ASP.NET 2010 is again used as a Front-End Tool with some Dynamic and Static pages. Its used to develop a Web – Based Application which is very interactive with the user. Using ASP.NET enables us to design a web-page more effectively and colorfully.

SQL-SERVER 2008:- SQL-SERVER 2008 is used as Back-End Tool

Microsoft Word:- Word is used for documenting our project files which can be used as a reference in the future for further enhancements in the software.

PLATFORM:- Windows xp or Higher is more convenient and easier to use. All the tools mentioned above works better under Windows compared to other platform.

**DFD Diagram**

Log Out

Username

Password

**Online Exam**

Login Confirmation

User

Attend Exam

**Use Case Diagram**

After attending exam user can not able to give exam second time or he will try to login automatic redirect to home.

**User**

This is about us page .

All candidate touch with us.

Candidate can login here.

Candidate Register with this page.

**Admin**

The Administrator can view the feedbacks if the users in this section.

The Administrator can add category of exam ex-maths,English etc..

The Administrator can add sub-category of exam

The Administrator can manage paper according to category.

The Administrator can give the grant to candidate which paper he should attend.

The Administrator can change password here.

The Administrator can activate deactivate admin user here.

Admin manage all the event from here.

**DATABASE TABLE:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr. No. | Field Name | Data Type | Size | Allow Nulls |
|  | AID | Int |  |  |
|  | CatgID | Int |  | Yes |
|  | SubCatgID | Int |  | Yes |
|  | QuestionID | Int |  | Yes |
|  | UserID | Int |  | Yes |
|  | CreatedBy | Varchar | 10 | Yes |
|  | CreatedOn | Varchar | 50 | Yes |
|  | ModifiedOn | Varchar | 50 | Yes |
|  | Isdeleted | Int |  | Yes |
|  | SetTiming | Varchar | 50 | Yes |
|  | Temp2 | Varchar | 50 | Yes |

Table Name : Assign table

|  |
| --- |
| Table Name : Evaluation |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr. No. | Field Name | Data Type | Size | Allow Nulls |
|  | EID | Int |  |  |
|  | CatID | Int |  | Yes |
|  | SubCatID | Int |  | Yes |
|  | UserID | Int |  | Yes |
|  | QID | Int |  | Yes |
|  | Answer | Varchar | 50 | Yes |
|  | Status | Int |  | Yes |
|  | CreatedBy | Varchar | 50 | Yes |
|  | CreatedOn | Varchar | 50 | Yes |
| 10. | ModifiedOn | Varchar | 50 | Yes |
| 11. | Temp | Varchar | 50 | Yes |
|  | Temp1 | Varchar | 50 | Yes |
|  | Temp2 | Varchar | 50 | yes |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr. No. | Field Name | Data Type | Size | Allow Nulls |
|  | TId | Int |  |  |
|  | TeamMembers | Varchar | 50 | Yes |
|  | Adhoc | Varchar | 50 | Yes |
|  | AtRisk | Varchar | 50 | Yes |
|  | GenMeetings | Varchar | 50 | Yes |
|  | IdleTime | Varchar | 50 | Yes |
|  | Leave | Varchar | 50 | Yes |
|  | OneonOne | Varchar | 50 | Yes |
|  | OperationMeeting | Varchar | 50 | Yes |
|  | PoInquiries | Varchar | 50 | Yes |
|  | PoInqtimetaken | Varchar | 50 | Yes |
|  | PoTracking | Varchar | 50 | Yes |
|  | PTG | Varchar | 50 | Yes |
|  | QualityAudit | Varchar | 50 | Yes |
|  | GrandTotal | Varchar | 50 | Yes |
|  | Temp | Varchar | 50 | Yes |
|  | Temp1 | Varchar | 50 | Yes |
|  | Temp2 | Varchar | 50 | Yes |

|  |
| --- |
| Table Name : Excel |

|  |
| --- |
| Table Name : Category |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr. No. | Field Name | Data Type | Size | Allow Nulls |
|  | CatgID | Int |  |  |
|  | CatgName | Varchar | 150 | Yes |
|  | Status | Varchar | 10 | Yes |
|  | CreatedBy | Varchar | 10 | Yes |
|  | CreatedOn | Varchar | 50 | Yes |
|  | ModifiedOn | Varchar | 50 | Yes |

|  |
| --- |
| Table Name : Events |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr. No. | Field Name | Data Type | Size | Allow Nulls |
|  | EventId | Int |  |  |
|  | Events | varchar | 4000 | Yes |
|  | Projects | varchar | 4000 | Yes |
|  | ParameterValue | varchar | 20 | Yes |
|  | CreatedBy | varchar | 10 | Yes |
|  | CreatedOn | varchar | 50 | Yes |
|  | ModifiedOn | varchar | 50 | Yes |

|  |
| --- |
| ble Name : QuestionPaper |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr. No. | Field Name | Data Type | Size | Allow Nulls |
|  | Question | Varchar | 3000 | Yes |
|  | OptionA | Varchar | 500 | Yes |
|  | OptionB | Varchar | 500 | Yes |
|  | OptionC | Varchar | 500 | Yes |
|  | OptionD | Varchar | 500 | Yes |
|  | Answer | Varchar | 500 | Yes |
|  | CreatedBy | Varchar | 10 | Yes |
|  | CreatedOn | Varchar | 50 | Yes |
|  | ModifiedOn | Varchar | 50 | Yes |
|  | Isdeleted | Int |  | Yes |
|  | ImageQuestion | Varchar | 1000 | Yes |
|  | Temp | Varchar | 50 | Yes |

|  |
| --- |
| Table Name : Result |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr. No. | Field Name | Data Type | Size | Allow Nulls |
|  | RID | Int |  |  |
|  | TotalQuest | Int |  | Yes |
|  | TotalAnswer | Int |  | Yes |
|  | CorrectAnswer | Int |  | Yes |
|  | UserID | Int |  | Yes |
|  | Result | Int |  | Yes |
|  | Percentage | Float |  | Yes |
|  | Grade | Varchar | 10 | Yes |

|  |
| --- |
| Table Name : SubCategory |

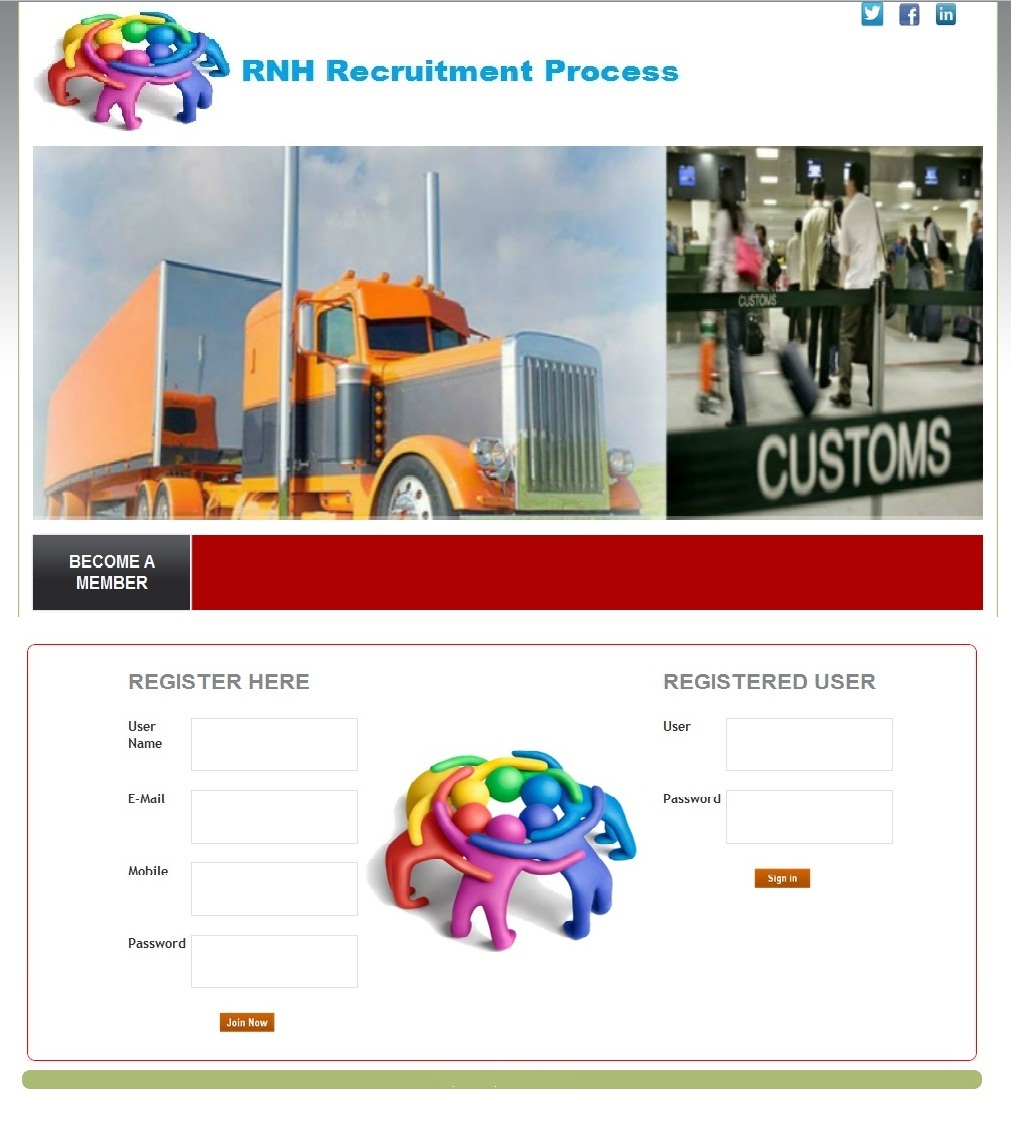
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr. No. | Field Name | Data Type | Size | Allow Nulls |
|  | SubCatgID | Int |  |  |
|  | CatgID | Int |  |  |
|  | SubCatgName | Varchar | 150 | Yes |
|  | Status | Varchar | 10 | Yes |
|  | CreatedBy | Varchar | 10 | Yes |
|  | CreatedOn | Varchar | 50 | Yes |
|  | ModifiedOn | Varchar | 50 | Yes |

|  |
| --- |
| Table Name : Users |

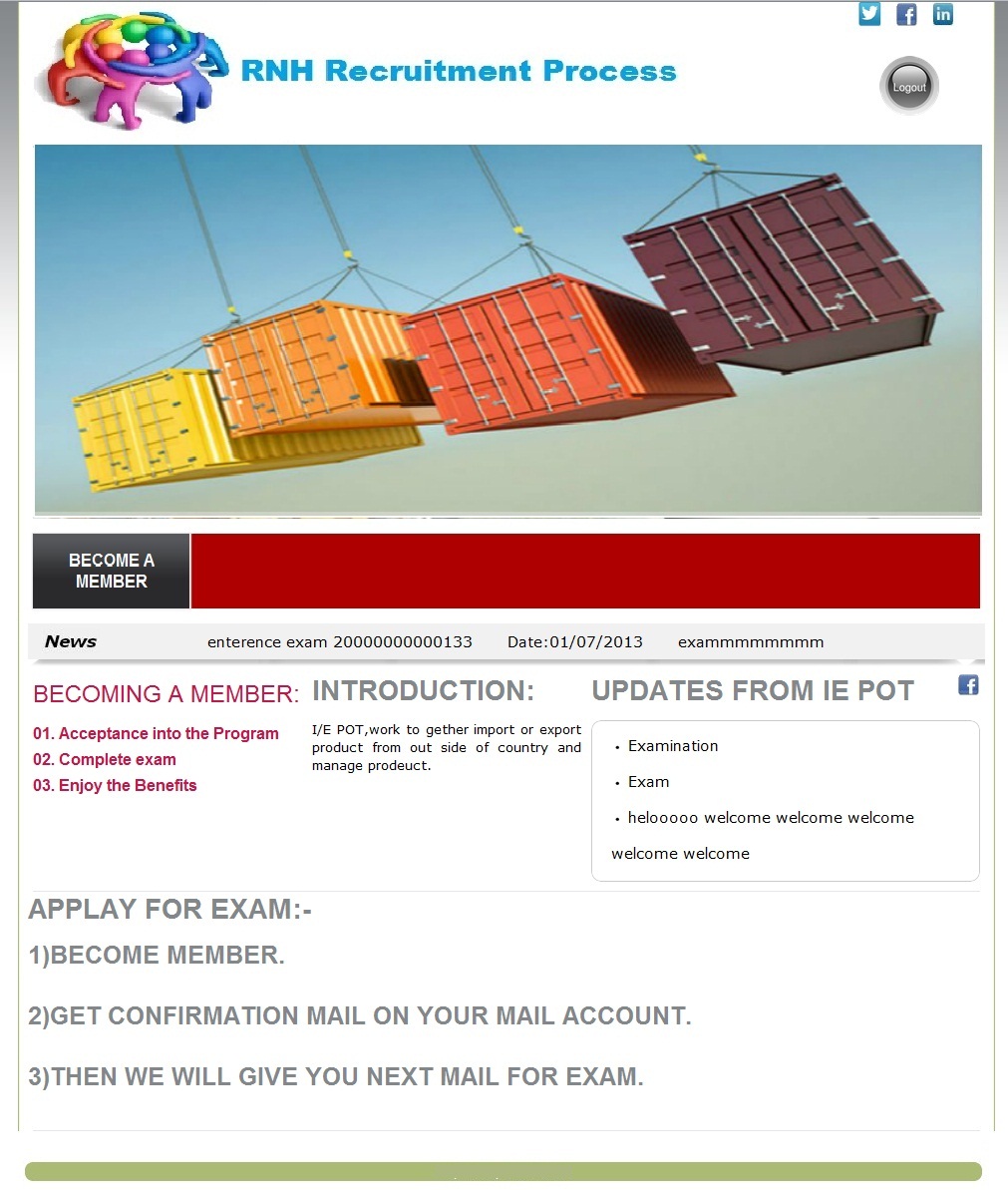
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr. No. | Field Name | Data Type | Size | Allow Nulls |
|  | Uid | Int |  |  |
|  | Username | Varchar | 50 | Yes |
|  | Password | Varchar | 30 | Yes |
|  | EmailId | Varchar | 50 | Yes |
|  | PhoneNo | Varchar | 20 | Yes |
|  | Status | Int |  | Yes |
|  | Usertype | Int |  | Yes |
|  | RegisteredOn | Varchar | 50 | Yes |
|  | ModifiedOn | Varchar | 50 |  |

**Screen Shots of Project**

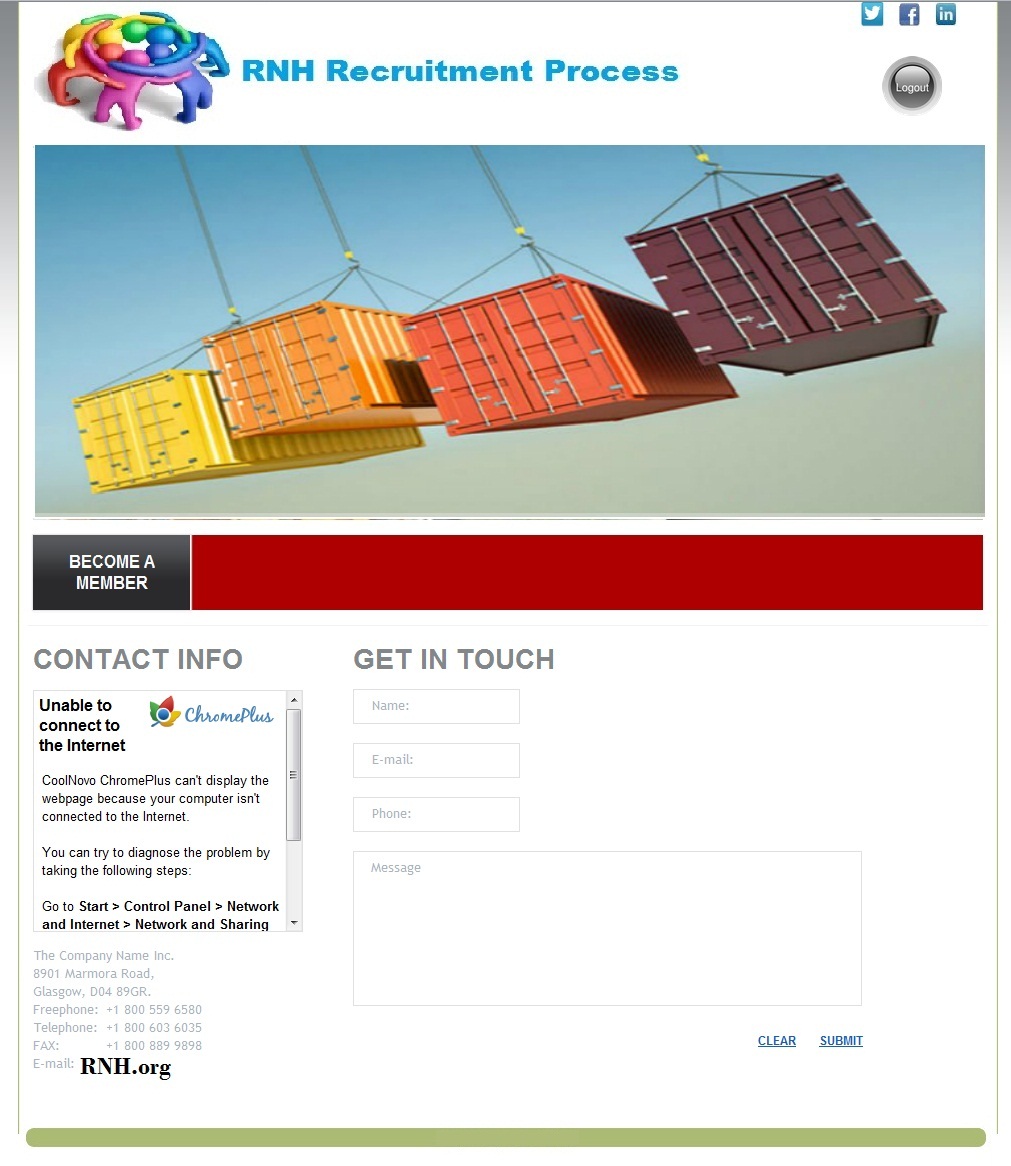
**Index:**

****

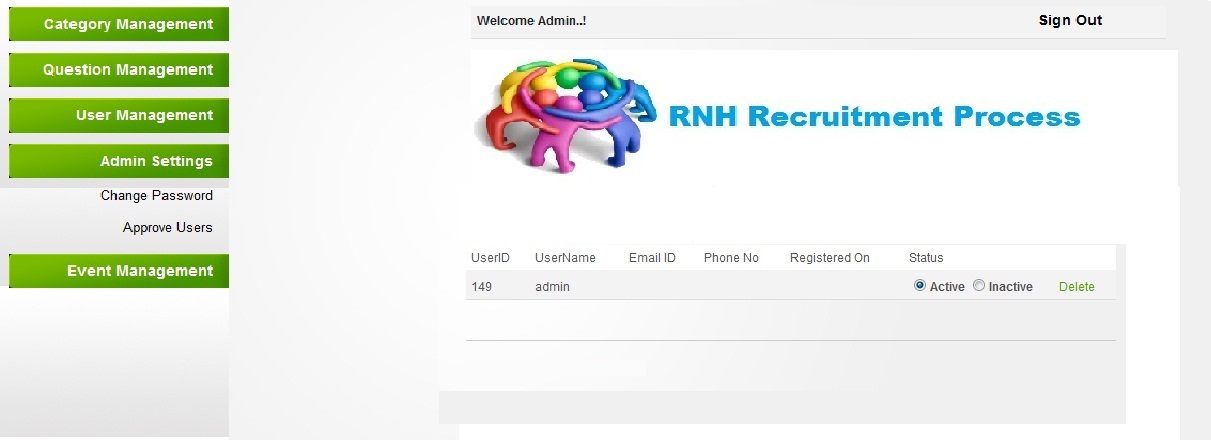
**Home:**

****

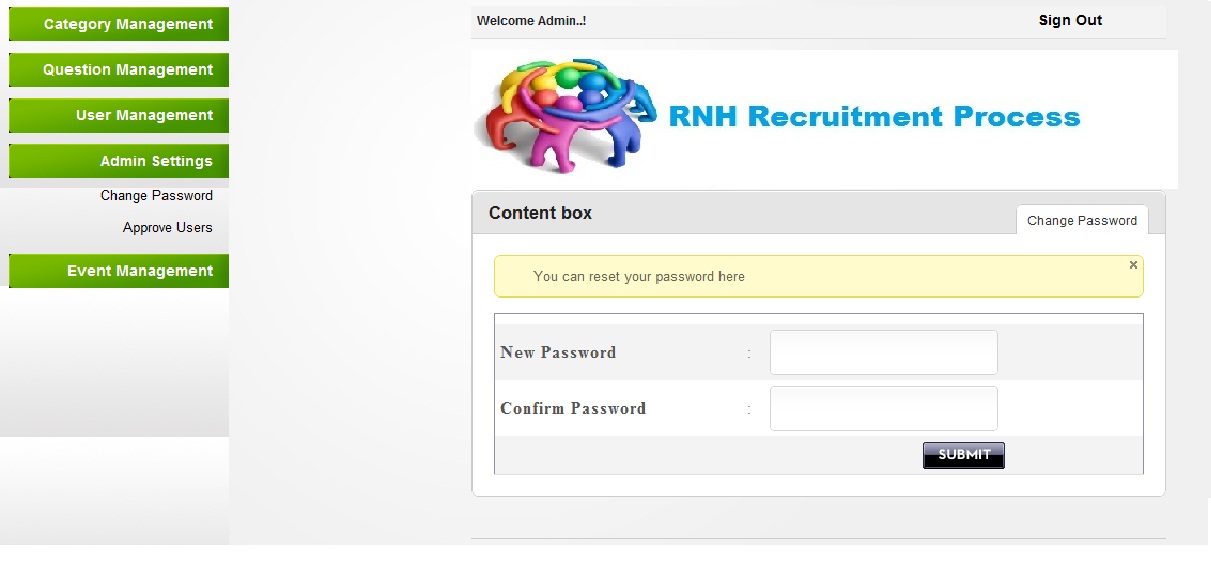
**Contact Us:**

****

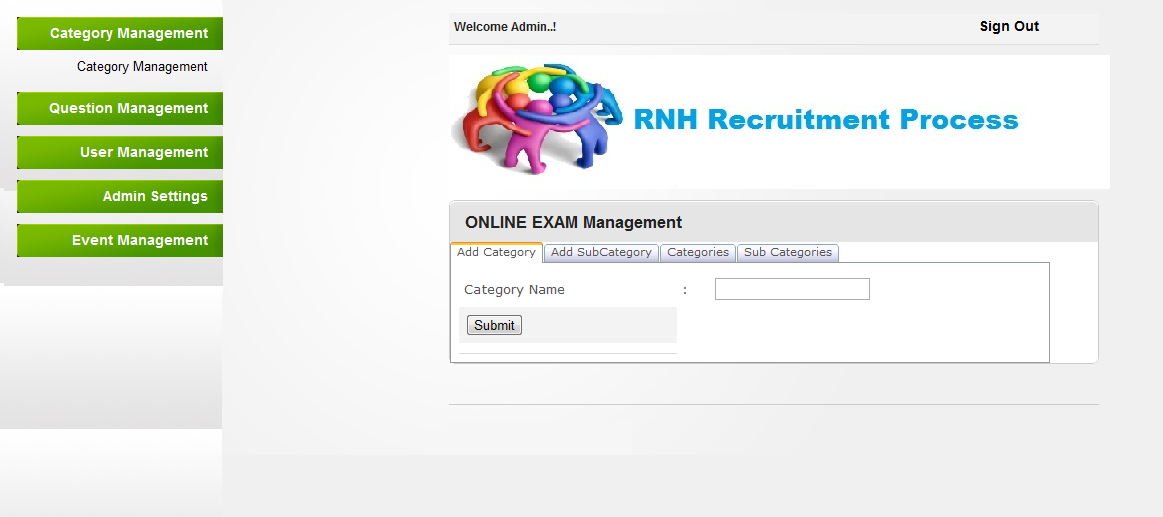
**Adminsetings(approveuser):**

****

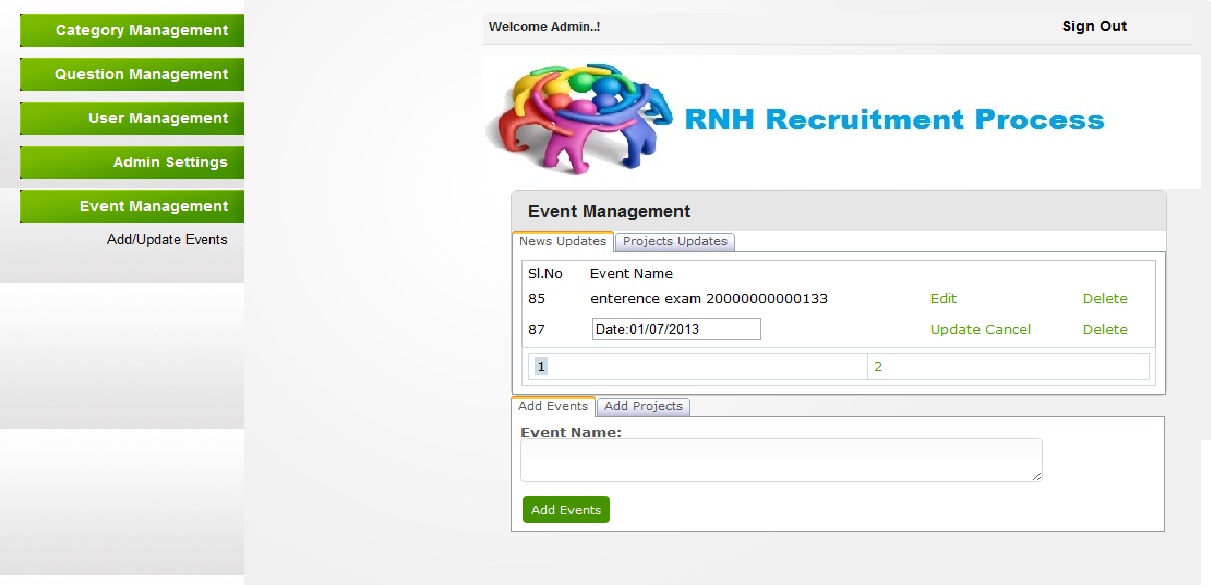
**Adminsetings(changepass):**

****

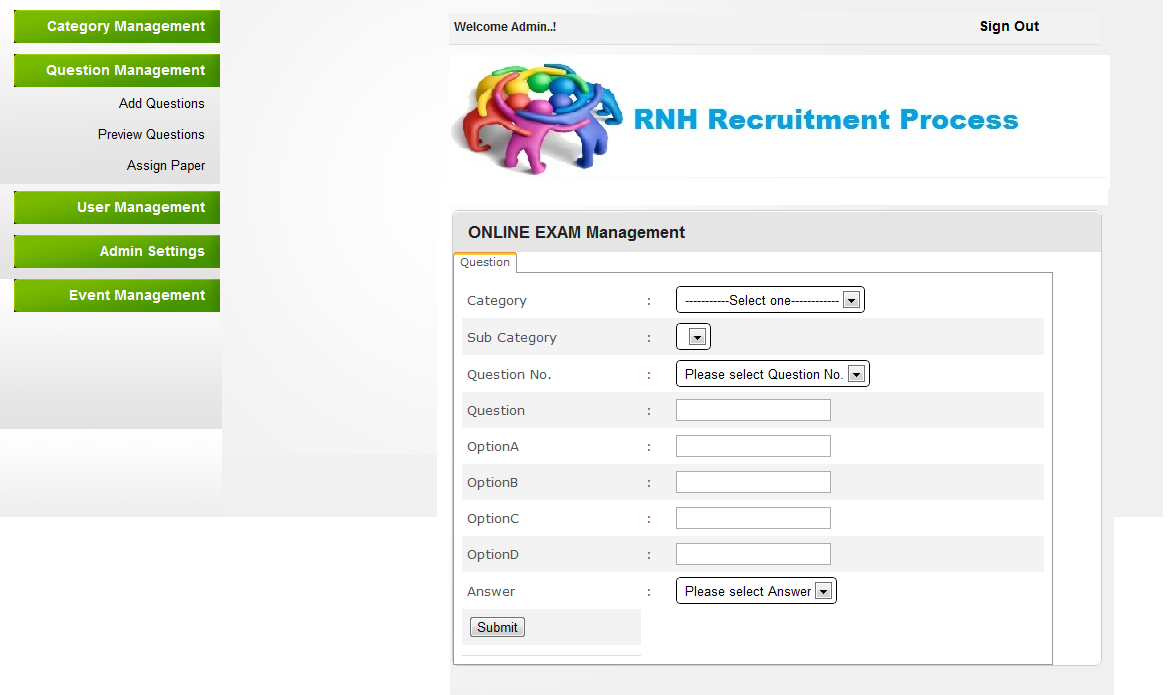
**Category management:**

****

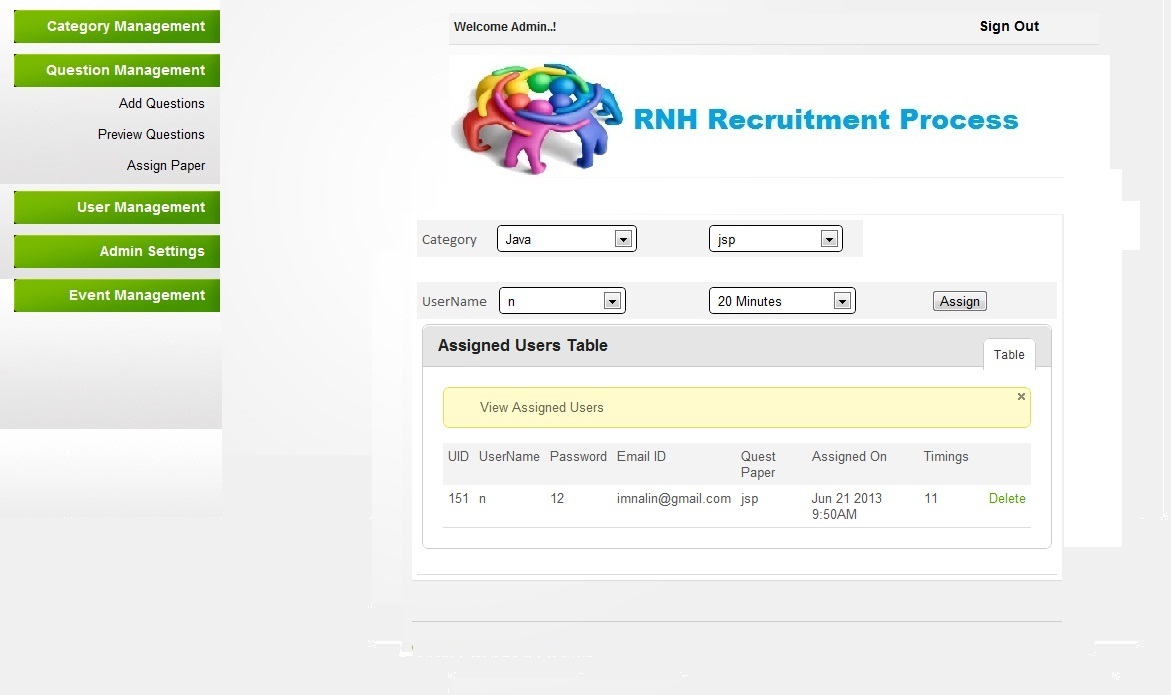
**Eventmanage(addupdateevnt):**

****

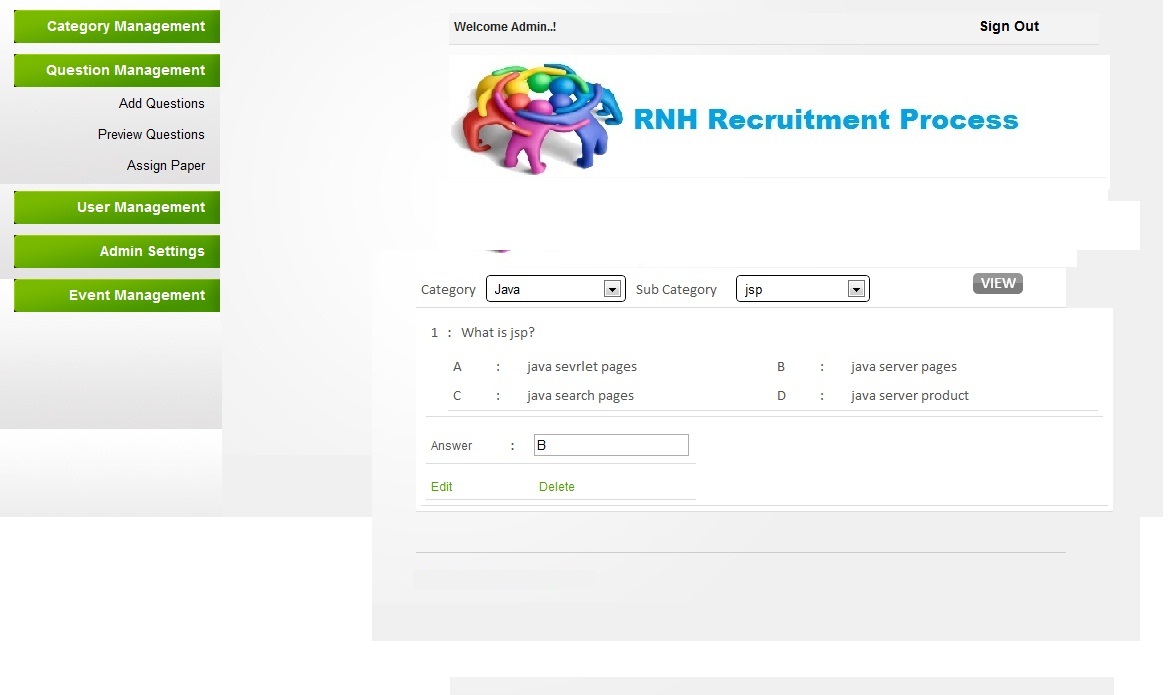
**Quetionmanage(addquetion):**

****

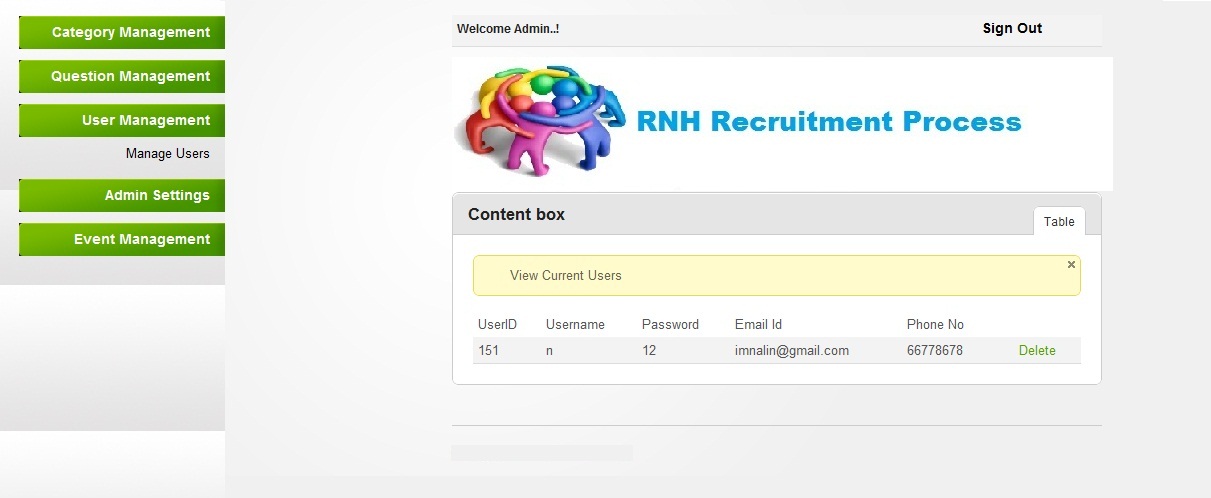
**Quetionmanage(assign paper):**

****

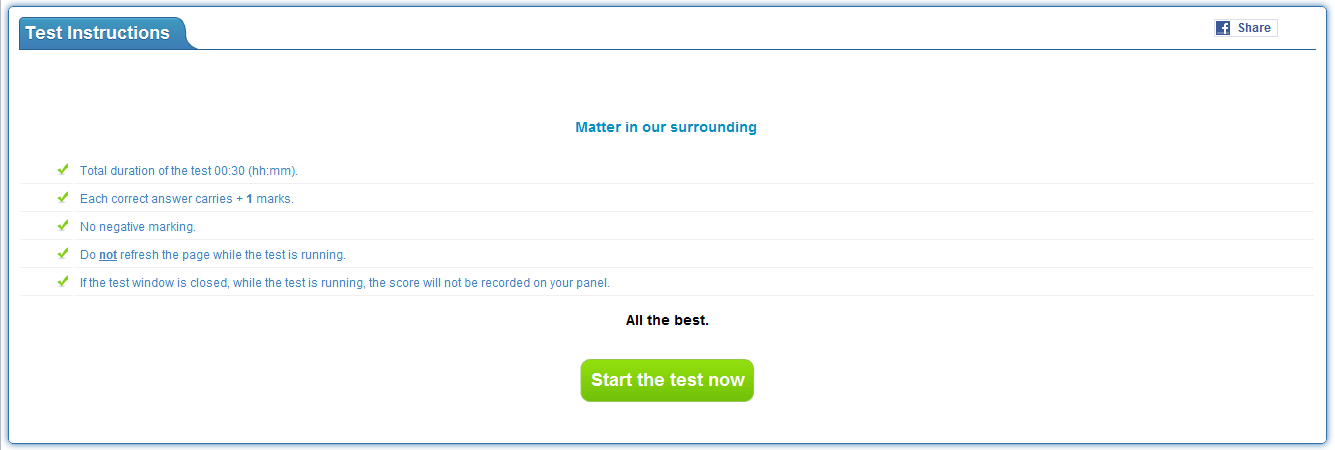
**Quetionmanage(prevquetion):**

****

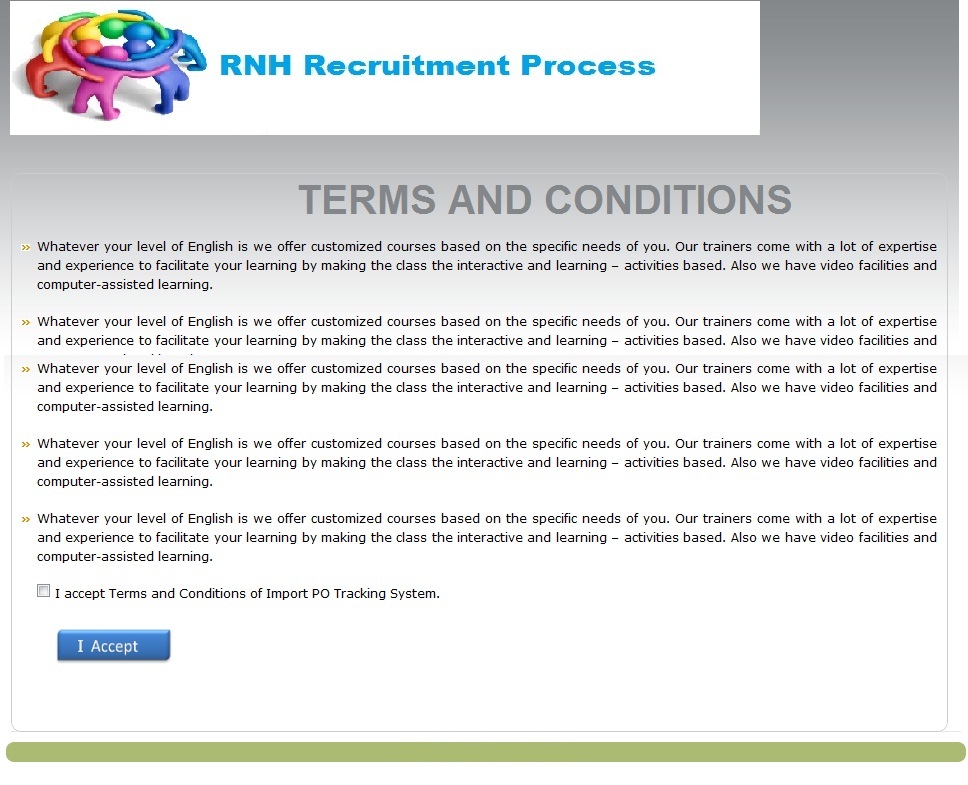
**Usermanage(manageuser):**

****

**Exam(start):**

****

**(Exam)policy:**

****

**(Exam)userexam:**

****

BAL(Business Acess Layer).

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data;

using System.Data.SqlClient;

using dal;

namespace bal

{

public class Class1

{

public int checklogin(int Uid, string Username, string Password, string Emailid, string Phoneno, int Status, int Usertype, string Registeredon, string Modifiedon)

{

clsDataAccess.clsReturnStatus objRetnStatus = new clsDataAccess.clsReturnStatus();

clsDataAccess.SPParams[] objParam = new clsDataAccess.SPParams[9];

//clsDataAccess.AssignValues(ref objParam[0], "@flag", "SqlDbType.Int", "Input", flag);

clsDataAccess.AssignValues(ref objParam[0], "@Uid", "SqlDbType.Int", "Input", Uid);

clsDataAccess.AssignValues(ref objParam[1], "@username", "SqlDbType.VarChar", "Input", Username, 50);

clsDataAccess.AssignValues(ref objParam[2], "@password", "SqlDbType.VarChar", "Input", Password, 50);

clsDataAccess.AssignValues(ref objParam[3], "@Emailid", "SqlDbType.VarChar", "Input", Emailid, 50);

clsDataAccess.AssignValues(ref objParam[4], "@Phoneno", "SqlDbType.VarChar", "Input", Phoneno, 50);

clsDataAccess.AssignValues(ref objParam[5], "@Status", "SqlDbType.Int", "Input", Status);

clsDataAccess.AssignValues(ref objParam[6], "@Usertype", "SqlDbType.Int", "Input", Usertype);

clsDataAccess.AssignValues(ref objParam[7], "@Registeredon", "SqlDbType.VarChar", "Input", Registeredon, 50);

clsDataAccess.AssignValues(ref objParam[8], "@Modifiedon", "SqlDbType.VarChar", "Input", Modifiedon, 50);

string strSpName = "usp\_login";

int intRtnStatus = 1;

string[] arrstrRet;

arrstrRet = objRetnStatus.ExecuteProcedure(strSpName, objParam);

intRtnStatus = Convert.ToInt32(arrstrRet[0]);

return intRtnStatus;

}

public DataSet checkloginnew(int Uid, string Username, string Password, string Emailid, string Phoneno, int Status, int Usertype, string Registeredon, string Modifiedon)

{

clsDataAccess.clsReturnDataSet objRetDataSet = new clsDataAccess.clsReturnDataSet();

clsDataAccess.SPParams[] objParam = new clsDataAccess.SPParams[9];

//clsDataAccess.AssignValues(ref objParam[0], "@flag", "SqlDbType.Int", "Input", flag);

clsDataAccess.AssignValues(ref objParam[0], "@Uid", "SqlDbType.Int", "Input", Uid);

clsDataAccess.AssignValues(ref objParam[1], "@username", "SqlDbType.VarChar", "Input", Username, 50);

clsDataAccess.AssignValues(ref objParam[2], "@password", "SqlDbType.VarChar", "Input", Password, 50);

clsDataAccess.AssignValues(ref objParam[3], "@Emailid", "SqlDbType.VarChar", "Input", Emailid, 50);

clsDataAccess.AssignValues(ref objParam[4], "@Phoneno", "SqlDbType.VarChar", "Input", Phoneno, 50);

clsDataAccess.AssignValues(ref objParam[5], "@Status", "SqlDbType.Int", "Input", Status);

clsDataAccess.AssignValues(ref objParam[6], "@Usertype", "SqlDbType.Int", "Input", Usertype);

clsDataAccess.AssignValues(ref objParam[7], "@Registeredon", "SqlDbType.VarChar", "Input", Registeredon, 50);

clsDataAccess.AssignValues(ref objParam[8], "@Modifiedon", "SqlDbType.VarChar", "Input", Modifiedon, 50);

DataSet ds = new DataSet();

string strSpName = "usp\_login";

ds = objRetDataSet.ExecuteProcedure(strSpName, objParam);

return ds;

}

public DataSet Insert\_to\_Registration(int flag, int Uid, string Username, string Password, string EmailId, string PhoneNo, int Status,

int Usertype, string RegisteredOn, string ModifiedOn)

{

clsDataAccess.clsReturnDataSet objRetDataSet = new clsDataAccess.clsReturnDataSet();

clsDataAccess.SPParams[] objParam = new clsDataAccess.SPParams[10];

clsDataAccess.AssignValues(ref objParam[0], "@flag ", "SqlDbType.Int", "Input", flag);

clsDataAccess.AssignValues(ref objParam[1], "@Uid ", "SqlDbType.Int", "Input", Uid);

clsDataAccess.AssignValues(ref objParam[2], "@Username ", "SqlDbType.VarChar", "Input", Username, 50);

clsDataAccess.AssignValues(ref objParam[3], "@EmailId", "SqlDbType.VarChar", "Input", EmailId, 50);

clsDataAccess.AssignValues(ref objParam[4], "@Password ", "SqlDbType.VarChar", "Input", Password, 30);

clsDataAccess.AssignValues(ref objParam[5], "@PhoneNo ", "SqlDbType.VarChar", "Input", PhoneNo, 20);

clsDataAccess.AssignValues(ref objParam[6], "@Status ", "SqlDbType.Int", "Input", Status);

clsDataAccess.AssignValues(ref objParam[7], "@Usertype ", "SqlDbType.Int", "Input", Usertype);

clsDataAccess.AssignValues(ref objParam[8], "@RegisteredOn ", "SqlDbType.VarChar", "Input", RegisteredOn, 50);

clsDataAccess.AssignValues(ref objParam[9], "@ModifiedOn", "SqlDbType.VarChar", "Input", ModifiedOn, 50);

DataSet ds = new DataSet();

string strSpName = "usp\_registration";

ds = objRetDataSet.ExecuteProcedure(strSpName, objParam);

return ds;

}

public DataSet Category(int flag, int @CatgID, string CatgName, string Status, string CreatedBy, string CreatedOn, string ModifiedOn)

{

clsDataAccess.clsReturnDataSet objRetDataSet = new clsDataAccess.clsReturnDataSet();

clsDataAccess.SPParams[] objParam = new clsDataAccess.SPParams[7];

clsDataAccess.AssignValues(ref objParam[0], "@flag ", "SqlDbType.Int", "Input", flag);

clsDataAccess.AssignValues(ref objParam[1], "@CatgID ", "SqlDbType.Int", "Input", CatgID);

clsDataAccess.AssignValues(ref objParam[2], "@CatgName", "SqlDbType.VarChar", "Input", CatgName, 150);

clsDataAccess.AssignValues(ref objParam[3], "@Status", "SqlDbType.VarChar", "Input", Status, 10);

clsDataAccess.AssignValues(ref objParam[4], "@CreatedBy", "SqlDbType.VarChar", "Input", CreatedBy, 10);

clsDataAccess.AssignValues(ref objParam[5], "@CreatedOn", "SqlDbType.VarChar", "Input", CreatedOn, 50);

clsDataAccess.AssignValues(ref objParam[6], "@ModifiedOn", "SqlDbType.VarChar", "Input", ModifiedOn, 50);

DataSet ds = new DataSet();

string strSpName = "usp\_category";

ds = objRetDataSet.ExecuteProcedure(strSpName, objParam);

return ds;

}

// 3, catid, SubCatName, "", "", "", ""

public DataSet SubCategory(int flag, int SubCatgID, string SubCatgName, string Status, string CreatedBy, string CreatedOn, string ModifiedOn)

{

clsDataAccess.clsReturnDataSet objRetDataSet = new clsDataAccess.clsReturnDataSet();

clsDataAccess.SPParams[] objParam = new clsDataAccess.SPParams[7];

clsDataAccess.AssignValues(ref objParam[0], "@flag ", "SqlDbType.Int", "Input", flag);

clsDataAccess.AssignValues(ref objParam[1], "@SubCatgID ", "SqlDbType.Int", "Input", SubCatgID);

clsDataAccess.AssignValues(ref objParam[2], "@SubCatgName", "SqlDbType.VarChar", "Input", SubCatgName, 150);

clsDataAccess.AssignValues(ref objParam[3], "@Status", "SqlDbType.VarChar", "Input", Status, 10);

clsDataAccess.AssignValues(ref objParam[4], "@CreatedBy", "SqlDbType.VarChar", "Input", CreatedBy, 10);

clsDataAccess.AssignValues(ref objParam[5], "@CreatedOn", "SqlDbType.VarChar", "Input", CreatedOn, 50);

clsDataAccess.AssignValues(ref objParam[6], "@ModifiedOn", "SqlDbType.VarChar", "Input", ModifiedOn, 50);

DataSet ds = new DataSet();

string strSpName = "usp\_category";

ds = objRetDataSet.ExecuteProcedure(strSpName, objParam);

return ds;

}

public DataSet QuestionPaper(int flag, int QuestionID, int CatgID, int SubCatgID, int UserID, int QuestionNo, string Question, string OptionA, string OptionB, string OptionC, string OptionD, string Answer, string CreatedBy, string CreatedOn, string ModifiedOn, int Isdeleted)

{

clsDataAccess.clsReturnDataSet objRetDataSet = new clsDataAccess.clsReturnDataSet();

clsDataAccess.SPParams[] objParam = new clsDataAccess.SPParams[16];

clsDataAccess.AssignValues(ref objParam[0], "@flag ", "SqlDbType.Int", "Input", flag);

clsDataAccess.AssignValues(ref objParam[1], "@QuestionID ", "SqlDbType.Int", "Input", QuestionID);

clsDataAccess.AssignValues(ref objParam[2], "@CatgID ", "SqlDbType.Int", "Input", CatgID);

clsDataAccess.AssignValues(ref objParam[3], "@SubCatgID ", "SqlDbType.Int", "Input", SubCatgID);

clsDataAccess.AssignValues(ref objParam[4], "@UserID ", "SqlDbType.Int", "Input", UserID);

clsDataAccess.AssignValues(ref objParam[5], "@QuestionNo ", "SqlDbType.Int", "Input", QuestionNo);

clsDataAccess.AssignValues(ref objParam[6], "@Question", "SqlDbType.VarChar", "Input", Question, 3000);

clsDataAccess.AssignValues(ref objParam[7], "@OptionA", "SqlDbType.VarChar", "Input", OptionA, 500);

clsDataAccess.AssignValues(ref objParam[8], "@OptionB", "SqlDbType.VarChar", "Input", OptionB, 500);

clsDataAccess.AssignValues(ref objParam[9], "@OptionC", "SqlDbType.VarChar", "Input", OptionC, 500);

clsDataAccess.AssignValues(ref objParam[10], "@OptionD", "SqlDbType.VarChar", "Input", OptionD, 500);

clsDataAccess.AssignValues(ref objParam[11], "@Answer", "SqlDbType.VarChar", "Input", Answer, 500);

clsDataAccess.AssignValues(ref objParam[12], "@CreatedBy", "SqlDbType.VarChar", "Input", CreatedBy, 10);

clsDataAccess.AssignValues(ref objParam[13], "@CreatedOn", "SqlDbType.VarChar", "Input", CreatedOn, 50);

clsDataAccess.AssignValues(ref objParam[14], "@ModifiedOn", "SqlDbType.VarChar", "Input", ModifiedOn, 50);

clsDataAccess.AssignValues(ref objParam[15], "@Isdeleted ", "SqlDbType.Int", "Input", Isdeleted);

DataSet ds = new DataSet();

string strSpName = "usp\_QuestionPaper";

ds = objRetDataSet.ExecuteProcedure(strSpName, objParam);

return ds;

}

public DataSet Users(int flag, int Uid, string Username, string Password, string EmailId, string PhoneNo, int Status, int Usertype, string RegisteredOn, string ModifiedOn)

{

clsDataAccess.clsReturnDataSet objRetDataSet = new clsDataAccess.clsReturnDataSet();

clsDataAccess.SPParams[] objParam = new clsDataAccess.SPParams[10];

clsDataAccess.AssignValues(ref objParam[0], "@flag ", "SqlDbType.Int", "Input", flag);

clsDataAccess.AssignValues(ref objParam[1], "@Uid", "SqlDbType.Int", "Input", Uid);

clsDataAccess.AssignValues(ref objParam[2], "@Username", "SqlDbType.VarChar", "Input", Username, 50);

clsDataAccess.AssignValues(ref objParam[3], "@Password", "SqlDbType.VarChar", "Input", Password, 30);

clsDataAccess.AssignValues(ref objParam[4], "@EmailId", "SqlDbType.VarChar", "Input", EmailId, 50);

clsDataAccess.AssignValues(ref objParam[5], "@PhoneNo", "SqlDbType.VarChar", "Input", PhoneNo, 20);

clsDataAccess.AssignValues(ref objParam[6], "@Status", "SqlDbType.Int", "Input", Status);

clsDataAccess.AssignValues(ref objParam[7], "@Usertype", "SqlDbType.Int", "Input", Usertype);

clsDataAccess.AssignValues(ref objParam[8], "@RegisteredOn", "SqlDbType.VarChar", "Input", RegisteredOn, 50);

clsDataAccess.AssignValues(ref objParam[9], "@ModifiedOn", "SqlDbType.VarChar", "Input", ModifiedOn, 50);

DataSet ds = new DataSet();

string strSpName = "usp\_user";

ds = objRetDataSet.ExecuteProcedure(strSpName, objParam);

return ds;

}

public DataSet news(int flag, int EventId, string Events, string Projects, string ParameterValue, string CreatedBy, string CreatedOn, string ModifiedOn)

{

clsDataAccess.clsReturnDataSet objRetDataSet = new clsDataAccess.clsReturnDataSet();

clsDataAccess.SPParams[] objParam = new clsDataAccess.SPParams[8];

clsDataAccess.AssignValues(ref objParam[0], "@flag ", "SqlDbType.Int", "Input", flag);

clsDataAccess.AssignValues(ref objParam[1], "@EventId ", "SqlDbType.Int", "Input", EventId);

clsDataAccess.AssignValues(ref objParam[2], "@Events", "SqlDbType.VarChar", "Input", Events, 4000);

clsDataAccess.AssignValues(ref objParam[3], "@Projects", "SqlDbType.VarChar", "Input", Projects, 4000);

clsDataAccess.AssignValues(ref objParam[4], "@ParameterValue", "SqlDbType.VarChar", "Input", ParameterValue, 20);

clsDataAccess.AssignValues(ref objParam[5], "@CreatedBy", "SqlDbType.VarChar", "Input", CreatedBy, 10);

clsDataAccess.AssignValues(ref objParam[6], "@CreatedOn", "SqlDbType.VarChar", "Input", CreatedOn, 50);

clsDataAccess.AssignValues(ref objParam[7], "@ModifiedOn", "SqlDbType.VarChar", "Input", ModifiedOn, 50);

DataSet ds = new DataSet();

string strSpName = "usp\_events";

ds = objRetDataSet.ExecuteProcedure(strSpName, objParam);

return ds;

}

public DataSet Assign(int @flag, int @Aid, int @CatgID, int @SubCatgID, int @UserID, string @CreatedBy, string @CreatedOn, string @ModifiedOn, int @Isdeleted, string @SetTimings)

{

clsDataAccess.clsReturnDataSet objRetDataSet = new clsDataAccess.clsReturnDataSet();

clsDataAccess.SPParams[] objParam = new clsDataAccess.SPParams[10];

clsDataAccess.AssignValues(ref objParam[0], "@flag ", "SqlDbType.Int", "Input", @flag);

clsDataAccess.AssignValues(ref objParam[1], "@Aid ", "SqlDbType.Int", "Input", @Aid);

clsDataAccess.AssignValues(ref objParam[2], "@CatgID ", "SqlDbType.Int", "Input", @CatgID);

clsDataAccess.AssignValues(ref objParam[3], "@SubCatgID ", "SqlDbType.Int", "Input", @SubCatgID);

clsDataAccess.AssignValues(ref objParam[4], "@UserID ", "SqlDbType.Int", "Input", @UserID);

clsDataAccess.AssignValues(ref objParam[5], "@CreatedBy", "SqlDbType.VarChar", "Input", @CreatedBy, 10);

clsDataAccess.AssignValues(ref objParam[6], "@CreatedOn", "SqlDbType.VarChar", "Input", @CreatedOn, 50);

clsDataAccess.AssignValues(ref objParam[7], "@ModifiedOn", "SqlDbType.VarChar", "Input", @ModifiedOn, 50);

clsDataAccess.AssignValues(ref objParam[8], "@Isdeleted", "SqlDbType.Int", "Input", @Isdeleted);

clsDataAccess.AssignValues(ref objParam[9], "@SetTimings", "SqlDbType.VarChar", "Input", @SetTimings, 50);

DataSet ds = new DataSet();

string strSpName = "Usp\_Assign";

ds = objRetDataSet.ExecuteProcedure(strSpName, objParam);

return ds;

}

public DataSet Evaluation(int @flag, int EID, int CatID, int SubCatID, int UserID, int QID, string Answers, int Status, string CreatedBy, string CreatedOn, string ModifiedOn, string Temp, string Temp1, string Temp2)

{

clsDataAccess.clsReturnDataSet objRetDataSet = new clsDataAccess.clsReturnDataSet();

clsDataAccess.SPParams[] objParam = new clsDataAccess.SPParams[14];

clsDataAccess.AssignValues(ref objParam[0], "@flag ", "SqlDbType.Int", "Input", flag);

clsDataAccess.AssignValues(ref objParam[1], "@EID ", "SqlDbType.Int", "Input", EID);

clsDataAccess.AssignValues(ref objParam[2], "@CatID ", "SqlDbType.Int", "Input", CatID);

clsDataAccess.AssignValues(ref objParam[3], "@SubCatID ", "SqlDbType.Int", "Input", SubCatID);

clsDataAccess.AssignValues(ref objParam[4], "@UserID ", "SqlDbType.Int", "Input", UserID);

clsDataAccess.AssignValues(ref objParam[5], "@QID ", "SqlDbType.Int", "Input", QID);

clsDataAccess.AssignValues(ref objParam[6], "@Answers", "SqlDbType.VarChar", "Input", Answers, 50);

clsDataAccess.AssignValues(ref objParam[7], "@Status ", "SqlDbType.Int", "Input", Status);

clsDataAccess.AssignValues(ref objParam[8], "@CreatedBy", "SqlDbType.VarChar", "Input", CreatedBy, 50);

clsDataAccess.AssignValues(ref objParam[9], "@CreatedOn", "SqlDbType.VarChar", "Input", CreatedOn, 50);

clsDataAccess.AssignValues(ref objParam[10], "@ModifiedOn", "SqlDbType.VarChar", "Input", ModifiedOn, 50);

clsDataAccess.AssignValues(ref objParam[11], "@Temp", "SqlDbType.VarChar", "Input", Temp, 50);

clsDataAccess.AssignValues(ref objParam[12], "@Temp1", "SqlDbType.VarChar", "Input", Temp1, 50);

clsDataAccess.AssignValues(ref objParam[13], "@Temp2", "SqlDbType.VarChar", "Input", Temp2, 50);

DataSet ds = new DataSet();

string strSpName = "Usp\_Evaluation";

ds = objRetDataSet.ExecuteProcedure(strSpName, objParam);

return ds;

}

}

}

DAL(Data Acess Layer)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data;

using System.Data.SqlClient;

using System.Configuration;

using System.Web;

namespace dal

{

public class clsDataAccess

{

public clsDataAccess()

{ }

public struct SPParams

{

public int intSize;

public string strValue;

public string strDbType;

public string strFieldName;

public string strDirection;

}

private static string GetDSN()

{

// string strDSN = "Provider=SQLOLEDB;server=(local);uid=newots;pwd=ots;database=NewOts";

string strDSN =ConfigurationSettings.AppSettings["database"];

return strDSN;

}

public static void AssignValues(ref SPParams objParam, string strFieldName, string strDbType, string strDirection, int intValue)

{

objParam.strFieldName = strFieldName;

objParam.strDbType = strDbType;

objParam.strDirection = strDirection;

objParam.strValue = intValue.ToString();

objParam.intSize = 0;

}

public static void AssignValues(ref SPParams objParam, string strFieldName, string strDbType, string strDirection, string strValue, int intSize)

{

objParam.strFieldName = strFieldName;

objParam.strDbType = strDbType;

objParam.strDirection = strDirection;

objParam.strValue = strValue;

objParam.intSize = intSize;

}

#region Return Status and Output parameters

public class clsReturnStatus

{

public clsReturnStatus()

{ }

public string[] ExecuteProcedure(string strSPName, SPParams[] objParams)

{

string[] arrstrRet;

SqlConnection con = new SqlConnection();

try

{

int intOutputParamCount = 0;

con = new SqlConnection(clsDataAccess.GetDSN());

SqlCommand cmd = new SqlCommand();

cmd.CommandText = strSPName;

cmd.Connection = con;

cmd.CommandType = CommandType.StoredProcedure;

SqlParameter parReturn = new SqlParameter("@iReturn", SqlDbType.NVarChar);

parReturn.Direction = ParameterDirection.ReturnValue;

cmd.Parameters.Add(parReturn);

foreach (SPParams objSPParam in objParams)

{

SqlParameter parSP = new SqlParameter(objSPParam.strFieldName, objSPParam.strDbType);

parSP.Size = objSPParam.intSize;

if (objSPParam.strDirection == "Input")

{

parSP.Direction = ParameterDirection.Input;

parSP.Value = objSPParam.strValue;

}

else if (objSPParam.strDirection == "Output")

{

parSP.Direction = ParameterDirection.Output;

intOutputParamCount++;

}

cmd.Parameters.Add(parSP);

}

con.Open();

cmd.ExecuteNonQuery();

arrstrRet = new string[intOutputParamCount + 1];

int intIdx = 0;

arrstrRet[intIdx++] = parReturn.Value.ToString();

foreach (SqlParameter parSP in cmd.Parameters)

{

if (parSP.Direction == ParameterDirection.Output)

{

arrstrRet[intIdx++] = parSP.Value.ToString();

}

}

return arrstrRet;

}

catch (Exception ex)

{

throw new Exception(" DAL Error - " + strSPName + ":" + ex.Message);

}

finally

{

if (con.State == ConnectionState.Open)

{

con.Close();

con.Dispose();

}

}

}

public string[][] ExecuteProcedure(string[] strSPName, SPParams[][] objParams)

{

string[][] arrstrRet;

SqlConnection con = new SqlConnection();

try

{

int intOutputParamCount;

con = new SqlConnection(clsDataAccess.GetDSN());

arrstrRet = new string[strSPName.Length][];

for (int intIdx = 0; intIdx < strSPName.Length; intIdx++)

{

SqlCommand cmd = new SqlCommand();

cmd.CommandText = strSPName[intIdx];

cmd.Connection = con;

cmd.CommandType = CommandType.StoredProcedure;

SqlParameter parReturn = new SqlParameter("@iReturn", SqlDbType.NVarChar);

parReturn.Direction = ParameterDirection.ReturnValue;

cmd.Parameters.Add(parReturn);

intOutputParamCount = 0;

foreach (SPParams objSPParam in objParams[intIdx])

{

SqlParameter parSP = new SqlParameter(objSPParam.strFieldName, objSPParam.strDbType);

parSP.Size = objSPParam.intSize;

if (objSPParam.strDirection == "Input")

{

parSP.Direction = ParameterDirection.Input;

parSP.Value = objSPParam.strValue;

}

else if (objSPParam.strDirection == "Output")

{

parSP.Direction = ParameterDirection.Output;

intOutputParamCount++;

}

cmd.Parameters.Add(parSP);

}

con.Open();

cmd.ExecuteNonQuery();

arrstrRet[intIdx] = new string[intOutputParamCount + 1];

int intLoopVar = 2;

arrstrRet[intIdx][0] = parReturn.Value.ToString();

foreach (SqlParameter parSP in cmd.Parameters)

{

if (parSP.Direction == ParameterDirection.Output)

{

arrstrRet[intIdx][intLoopVar++] = parSP.Value.ToString();

}

}

cmd.Dispose();

}

return arrstrRet;

}

catch (Exception ex)

{

throw new Exception(" DAL Error - " + strSPName + ":" + ex.Message);

}

finally

{

if (con.State == ConnectionState.Open)

{

con.Close();

con.Dispose();

}

}

}

}

#endregion

#region Return DataSet

public class clsReturnDataSet

{

public clsReturnDataSet()

{ }

public DataSet ExecuteProcedure(string strSPName, SPParams[] objParams)

{

int intOutputParamCount = 0;

string strRet = "";

DataSet dsReturn = new DataSet("ResultSet");

SqlConnection con = new SqlConnection();

try

{

con = new SqlConnection(clsDataAccess.GetDSN());

SqlCommand cmd = new SqlCommand();

cmd.CommandText = strSPName;

cmd.Connection = con;

cmd.CommandType = CommandType.StoredProcedure;

SqlParameter parReturn = new SqlParameter("@iReturn", SqlDbType.NVarChar);

parReturn.Direction = ParameterDirection.ReturnValue;

cmd.Parameters.Add(parReturn);

foreach (SPParams objSPParam in objParams)

{

SqlParameter parSP = new SqlParameter(objSPParam.strFieldName, objSPParam.strDbType);

parSP.Size = objSPParam.intSize;

if (objSPParam.strDirection == "Input")

{

parSP.Direction = ParameterDirection.Input;

parSP.Value = objSPParam.strValue;

}

else if (objSPParam.strDirection == "Output")

{

parSP.Direction = ParameterDirection.Output;

intOutputParamCount++;

}

cmd.Parameters.Add(parSP);

}

con.Open();

SqlDataAdapter daReturn = new SqlDataAdapter(cmd);

daReturn.Fill(dsReturn, "DataTable");

if (intOutputParamCount > 0)

{

DataTable dtbOutput = new DataTable("OutputTable");

dtbOutput.Columns.Add("Value", Type.GetType("System.String"));

foreach (SqlParameter parSP in cmd.Parameters)

{

if (parSP.Direction == ParameterDirection.Output)

{

DataRow drow = dtbOutput.NewRow();

drow[0] = parSP.Value.ToString();

dtbOutput.Rows.Add(drow);

}

}

dsReturn.Tables.Add(dtbOutput);

}

strRet = parReturn.Value.ToString();

return dsReturn;

}

catch (Exception ex)

{

throw new Exception(" DAL Error - " + strSPName + ":" + ex.Message);

}

finally

{

if (strRet.ToString() != "0")

{

DataTable dtbError = new DataTable("ErrorTable");

dtbError.Columns.Add("ErrorCode", Type.GetType("System.String"));

DataRow drow = dtbError.NewRow();

drow[0] = strRet;

dtbError.Rows.Add(drow);

dsReturn.Tables.Add(dtbError);

}

if (con.State == ConnectionState.Open)

{

con.Close();

con.Dispose();

}

}

}

}

#endregion

}

}

**Validations:**

Validations are Provided in the Following Forms:-

* IN ASP.NET
  + Index.aspx
  + Register.aspx
  + All The Entry Form…..

**Conclusion**

This Web Application provides facility to conduct online examination world

wide. It saves time as it allows number of students to give the exam at a time

and displays the results as the test gets over, so no need to wait for the result.

It is automatically generated by the server.

Administrator has a privilege to create, modify and delete the test papers and

its particular questions. User can register, login and give the test with his

specific id, and can see the results as well.

**BIBLIOGRAPHY:**

* C# - WROX Professional
* ASP.NET WROX Professional
* ASP.NET – Black Book
* C# - WROX Beginning
* SQL-2005 – WROX Publications
* Project Management – Pearson Education
* UML Beginner’s Guide – Jason T. Roff
* Software Testing – Pearson Education
* Presentations provided by MKCL
* Website : [www.google.com](http://www.google.com)