Spring 2015

E-Exam Engine

Obaid Ahmed  
*Governors State University*

Muthyalamma Gangula  
*Governors State University*

Rakesh Margam  
*Governors State University*

Follow this and additional works at: [http://opus.govst.edu/capstones](http://opus.govst.edu/capstones)

Part of the [Databases and Information Systems Commons](http://opus.govst.edu/capstones/collections/29)

**Recommended Citation**

[http://opus.govst.edu/capstones/140](http://opus.govst.edu/capstones/140)

For more information about the academic degree, extended learning, and certificate programs of Governors State University, go to [http://www.govst.edu/Academics/Degree_Programs_and_Certifications/](http://www.govst.edu/Academics/Degree_Programs_and_Certifications/)

Visit the [Governors State Computer Science Department](http://www.govst.edu/Academics/ComputerScience/)

This Project Summary is brought to you for free and open access by the Student Capstone Projects at OPUS Open Portal to University Scholarship. It has been accepted for inclusion in All Capstone Projects by an authorized administrator of OPUS Open Portal to University Scholarship. For more information, please contact opus@govst.edu.
ABSTRACT

E-Exam Engine is a web application by which is designed for Educational Institutes like Grad Schools, Colleges, and Private Institutes to conduct logic tests of their students on a regular basis. Design to facilitate administrator and user friendly interface complete and secure information is provided to user scope.

The E-Exam Engine is the process of conducting exam online. This project provide accurate and flexible manner of conducting exam online. This Project provides more accurate and efficient way to take exam. It also provides flexibility to the user as one can give the exam at home. This is a one of the good project for Grad Students.

Development process of the system starts with System analysis. System analysis involves creating a formal model of the problem to be solved by understanding requirements.
First and foremost deeply thankful to Professor Dr. Soon-Ok Park, for her wonderful guidance during this project work in field of Computer Science, at Governors State University.

I am also thankful for her continuous feedback and encouragement throughout this project work. Her broad knowledge and hardworking attitude has left me with very deep impressions and they will greatly benefit me throughout my life.

I would like to thank my entire project Readers for their support throughout this project work.

I would like to thank all of my family members and for their affection, guidance and encouragement throughout the study period.

Finally, I would like to extend thanks to my close friends for their continuous support, discussions and suggestions in solving all issues during the project work.
## CONTENTS

ABSTRACT

ACKNOWLEDGEMENTS

LIST OF FIGURES & PROGRAM CODE

1.0 INTRODUCTION 1

2.0 OVERALL DESCRIPTION 2

2.1 SCOPE OF THE SOFTWARE 2

2.2 SOFTWARE SYSTEM DESCRIPTION 2

3.0 FUNCTIONAL REQUIREMENTS 4

3.1 SOFTWARE CONTEXT 4

3.1.1 ADMIN PROCESS 4

4.0 S/W and H/W REQUIREMENTS 23

4.0.1 ENVIRONMENT 23

4.0.2 REQUIREMENTS 23

5.0 DATA MANAGEMENT 26

5.1 DATA DESCRIPTION 26

5.2 DATA OBJECTS 26

5.3 E-R DIAGRAM 27

6.0 NON FUNCTIONAL / OPERATIONAL REQUIREMENTS 28

6.1 SECURITY, RELIABILITY 29

6.2 MAINTAINABILITY 29

7.0 CONCLUSION 31

8.0 REFERENCES 32
## List of Figures & Program Code:

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIGURE 1</td>
<td>Admin Login Page</td>
<td>4</td>
</tr>
<tr>
<td>FIGURE 2</td>
<td>Admin Logic Page Source Code</td>
<td>6</td>
</tr>
<tr>
<td>FIGURE 3</td>
<td>Admin Home Page</td>
<td>7</td>
</tr>
<tr>
<td>FIGURE 4</td>
<td>Registration page</td>
<td>8</td>
</tr>
<tr>
<td>FIGURE 5</td>
<td>Student Login Page</td>
<td>9</td>
</tr>
<tr>
<td>FIGURE 6</td>
<td>Student Details</td>
<td>12</td>
</tr>
<tr>
<td>FIGURE 3</td>
<td>Exam</td>
<td>13</td>
</tr>
</tbody>
</table>
1.0 Introduction

E-Exam Engine deals with accurate and flexible manner of conducting exams online.

On-line Exam System is very useful for Educational Institute to prepare an exam, safe the time that will take to check the paper and prepare mark sheets. It will help the Institute to testing of students and develop their skills. But the disadvantages for this system, it takes a lot of times when you prepare the exam at the first time for usage. And we are needs number of computers with the same number of students.

The effective use of "On-line Exam System", any Educational Institutes can be use it to develop their strategy for putting the exams, and for getting better results in less time.

The E-Engine system automates the process by streamlining the handling of the exams in class thereby reducing the manual intervention.

There will be different types of users that use this system – faculty, admin, and Student. 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 students they give papers according to their convenience and time and there is no need of using extra thing like paper, pen etc.

This report is followed with Methodology part which explains Scope of the project, software system description, functional requirements along with user case screenshots, Data management covers the database relationships, Non-functional requirements and conclusion are outlined.
2.0 Overall Description

2.1 Scope of the Software

The main function of the software will be to provide quick and easy access for a user for doing Exams online. The system will be therefore designed to allow a user and admin to perform the following functions:

- Log on either as an administrator, or user.
- Register as an administrator or user.
- View both administrator and user accounts.
- Admin can create task, edit task information and close task
- All types of user must log in by user ID and password. According to their type there will be different privilege.
- Administrator view allows adding exams, quizzes and editing existing exams and quizzes
- Administrator can manage all things in this system.

2.2 Software System Description

- E-Exam system is designed for Educational Institutes (like schools, universities, training centers).
- The system handles all the operations, and generates reports as soon as the test is finish, that includes name, mark, time spent to solve the exam.
- Allow students to see or display his answers after the exam is finish.
- The type of questions is only multiple choice or true and false.

All types of user must log in by user ID and password. According to their type there will be different privilege.

3 Layer architecture

In this project I use three layer architecture. UI layer collaborate with BLL layer, BLL layer collaborate with DAL layer. I use three layer architecture because it gives flexibility. After some time if any change is needed in particular layer I would only
change in that layer. That time I will not bother for other layer. Suppose I want to change some business logic I will only concentrate on the business logic layer (BLL).
3.0 Functional Requirements

3.1 Software context

The system is designed for Educational Institutes like Schools, Colleges, and Private Institutes to conduct logic tests of their students on a regular basis. Design to facilitate administrator and user friendly interface complete and secure information is provided to user scope.

3.1.1 Admin Login Page:

Admin have the username and password. Only admin have all access like conducting exams, verify the student details and etc

Admin Login page Screen shot:
Code behind Admin login page:

```html

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title></title>
<link href="../css/bootstrap.min.css" rel="stylesheet" type="text/css" />
<link href="css/login.css" rel="stylesheet" type="text/css" />
</head>
<body>
<form id="form1" runat="server">
<div>
<div>
<div class="adminlogo" align="center"><img src="/images/Untitled-1.png" /></div>
<div class="adminlogo" align="center"><img src="/images/logo.png" /></div>
<div class="login_content">
<form class="form-horizontal" action="" method="post">
<h3 class="form-title">
Admin Login
</h3>
<div class="control-group">
<label>
Username
</label>
<asp:TextBox ID="txtusername" runat="server" placeholder="Username" class="form-control"></asp:TextBox>
</div>
<div class="control-group">
<label class="control-label visible-ie8 visible-ie9">
Password
</label>
<asp:TextBox ID="txtpassword" type="password" runat="server" placeholder="Password" class="form-control"></asp:TextBox>
</div>
<div class="form-actions">
<label class="checkbox">
</label>
<asp:Button ID="txtlogin" runat="server" Text="Login" class="btn btn-primary pull-right" onclick="txtlogin_Click" />
</div>
</form>
</div>
</div>
</div>
</form>
</body>
</html>
```
After log in Admin user and admin will see different type of home page according to their user type.

**Admin home page:**

Admin home page has a menu which provides administrative privilege.

Admin HomePage screen Shot:

---

**Student Registration page:**

Every Student must register with his/her details and then submit they got hall ticket for attending the exam.

Student registration screen shot:
After Student Registered:
Student login page:
Student can login with his hall ticket
### All Questions

**Search By:** Select Subject

<table>
<thead>
<tr>
<th>ID</th>
<th>Question_Paper_Name</th>
<th>Question_Type</th>
<th>Question_Name</th>
<th>Answer_A</th>
<th>Answer_B</th>
<th>Answer_C</th>
<th>Answer_D</th>
<th>Correct_Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Mathematics</td>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>D</td>
</tr>
<tr>
<td>2</td>
<td>Mathematics</td>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>A</td>
</tr>
<tr>
<td>3</td>
<td>PHYSICS</td>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>4</td>
<td>CHEMISTRY</td>
<td>SELECT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
</tbody>
</table>
### Exam Appeared Students List

<table>
<thead>
<tr>
<th>Roll Number</th>
<th>Student Name</th>
<th>Father Name</th>
<th>Sub</th>
<th>Mobile</th>
<th>EMAIL</th>
<th>Age</th>
<th>City</th>
<th>REACH</th>
<th>Country/Zone</th>
<th>Country/Region</th>
</tr>
</thead>
<tbody>
<tr>
<td>200001</td>
<td>NIK</td>
<td>J.K</td>
<td>123</td>
<td>1234567890</td>
<td>abcdf</td>
<td>123</td>
<td>Nizki</td>
<td>123</td>
<td>Andhra Pradesh</td>
<td>India</td>
</tr>
<tr>
<td>200007</td>
<td>niphi</td>
<td>j.k</td>
<td>123</td>
<td>1234567890</td>
<td>abcdf</td>
<td>123</td>
<td>Nizki</td>
<td>123</td>
<td>Andhra Pradesh</td>
<td>India</td>
</tr>
<tr>
<td>200013</td>
<td>1</td>
<td>1</td>
<td>123</td>
<td>1234567890</td>
<td>abcdf</td>
<td>123</td>
<td>Pat Bali</td>
<td>123</td>
<td>Andhra Pradesh</td>
<td>India</td>
</tr>
<tr>
<td>200019</td>
<td>1</td>
<td>1</td>
<td>123</td>
<td>1234567890</td>
<td>abcdf</td>
<td>123</td>
<td>Kadauna</td>
<td>123</td>
<td>Andhra Pradesh</td>
<td>India</td>
</tr>
</tbody>
</table>
Code Behind Business Logic:
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.SqlClient;
using System.Data;
using Datalayer;
using RepositryLayer;

namespace Businesslayer
{
    public class Businesslogic
    {
        datalogic ob = new datalogic();

        public DataTable reg(SqlParameter[] p, string proc)
        {
            DataTable dt = ob.getdetailswithparameters(p, proc);
            return dt;
        }
    }
}
public DataTable adminlogin(SqlParameter[] p, string proc)
{
    DataTable dt = ob.getdetailswithparameters(p, proc);
    return dt;
}

public DataSet gethallticket(SqlParameter[] p, string proc)
{
    DataSet dt = ob.details(p, proc);
    return dt;
}

public DataSet searchcandidates(SqlParameter[] p, string proc)
{
    DataSet dt = ob.details(p, proc);
    return dt;
}

public DataTable getcandidates(string proc)
{
    DataTable dt = ob.getdetails(proc);
    return dt;
}

public DataTable getcandidatewithid(SqlParameter[] p, string proc)
{
    DataTable dt = ob.getdetailswithparameters(p, proc);
    return dt;
}

public DataTable Get_country()
{
    return ob.getdetails("sp_getcountry");
}

public DataTable Get_state(SqlParameter[] p, string proc)
{
    DataTable dt = ob.getdetailswithparameters(p, proc);
    return dt;
}
public DataTable Get_city(SqlParameter[] p, string proc)
{
    DataTable dt = ob.getdetailswithparameters(p, proc);
    return dt;
}

public DataTable selectallreg(string proc)
{
    DataTable dt = ob.getdetails(proc);
    return dt;
}

public DataSet insert_Question(AddQuestion obj)
{
    datalogic obj1 = new datalogic();
    return obj1.insert_Question(obj);
}

public DataSet insert_Eng_Passage(AddQuestion obj)
{
    datalogic obj1 = new datalogic();
    return obj1.insert_Eng_Passage(obj);
}

public DataSet insert_Eng_Question(AddQuestion obj)
{
    datalogic obj1 = new datalogic();
    return obj1.insert_Eng_Question(obj);
}

public DataSet get_all_questions()
{
    datalogic obj1 = new datalogic();
    return obj1.get_all_questions();
}

public DataSet get_eng_questions()
{
datalogic obj1 = new datalogic();
return obj1.get_eng_questions();
}

public DataSet get_question_with_param(AddQuestion obj)
{
    datalogic obj1 = new datalogic();
    return obj1.get_question_with_param(obj);
}

public DataSet get_eng_question_with_param(AddQuestion obj)
{
    datalogic obj1 = new datalogic();
    return obj1.get_eng_question_with_param(obj);
}

public DataSet select_question_details(AddQuestion obj)
{
    datalogic obj1 = new datalogic();
    return obj1.select_question_details(obj);
}

public DataSet update_question_details(AddQuestion obj)
{
    datalogic obj1 = new datalogic();
    return obj1.update_question_details(obj);
}

public DataSet select_Eng_question_details(AddQuestion obj)
{
    datalogic obj1 = new datalogic();
    return obj1.select_Eng_question_details(obj);
}

public DataSet update_Eng_question_details(AddQuestion obj)
{
    datalogic obj1 = new datalogic();
    return obj1.update_Eng_question_details(obj);
}
public DataSet update_eng_question_details1(AddQuestion obj) {
    datalogic obj1 = new datalogic();
    return obj1.update_Eng_Question1(obj);
}

public AddQuestion Verify_hallticket(int htno) {
    datalogic obj1 = new datalogic();
    return obj1.Verify_hallticket(htno);
}

public DataSet get_maths_questions() {
    datalogic obj1 = new datalogic();
    return obj1.get_maths_questions();
}

public DataSet get_physics_questions() {
    datalogic obj1 = new datalogic();
    return obj1.get_physics_questions();
}

public DataSet get_Chemistry_questions() {
    datalogic obj1 = new datalogic();
    return obj1.get_Chemistry_questions();
}

public DataSet get_English_questions1() {
    datalogic obj1 = new datalogic();
    return obj1.get_English_questions1();
}

public DataSet get_English_questions2() {
    datalogic obj1 = new datalogic();
    return obj1.get_English_questions2();
}
public DataSet get_English_questions3()
{
datalogic obj1 = new datalogic();
return obj1.get_English_questions3();
}

public DataSet get_English_questions4()
{
datalogic obj1 = new datalogic();
return obj1.get_English_questions4();
}

public DataSet get_English_questions5()
{
datalogic obj1 = new datalogic();
return obj1.get_English_questions5();
}

public DataSet get_English_questions6()
{
datalogic obj1 = new datalogic();
return obj1.get_English_questions6();
}

public DataSet Online_Exam_final(AddQuestion obj)
{
datalogic obj1 = new datalogic();
return obj1.Online_Exam_final(obj);
}

public DataSet Online_Exam_marks(AddQuestion obj)
{
datalogic obj1 = new datalogic();
return obj1.Online_Exam_marks(obj);
}

public DataSet get_all_students_list()
{
datalogic obj1 = new datalogic();
return obj1.get_all_students_list();
}

public DataSet get_students_with_param(AddQuestion obj)
{
    datalogic obj1 = new datalogic();
    return obj1.get_students_with_param(obj);
}

public DataSet get_students_statewise(AddQuestion obj)
{
    datalogic obj1 = new datalogic();
    return obj1.get_students_statewise(obj);
}


**Code For Stored Procedure:**

1. Register
USE [E-Exam]
GO

******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER proc [dbo].[register]
(
    @name varchar(150),
    @fname varchar(150),
    @dob varchar(150),
    @add varchar(150),
    @state varchar(150),
    @pin varchar(150),
    @photo varchar(150),
@photo1 image,
@landline varchar(150),
@mobile varchar(150),
@email varchar(150),
@parentsprof varchar(150),
@income varchar(150),
@schoolname varchar(150),
@board varchar(150),
@preitcoach varchar(150),
@coachyears varchar(150),
@examcentre varchar(150),
@firstlang varchar(150),
@secondlang varchar(150),
@english varchar(150),
@maths varchar(150),
@science varchar(150),
@social varchar(150),
@total varchar(150),
@country varchar(150),
@city varchar(150),
@mark1 varchar(150),
@mark2 varchar(150)
)

as
begin
--if exists(select * from [dbo].[Registration] where ([Landline]=@landline))
--begin
--select 'landline' as message
--end

--else
if exists(select * from [dbo].[Registration] where ([Mobile]=@mobile))
begin
select 'mobile' as message
end

else if exists(select * from [dbo].[Registration] where ([Email]=@email))
begin
select 'email' as message
end

else
begin
insert into [dbo].[Registration]([Name],[Fname],[Dob],[Address],[Country],[State],[City],[Pincode], [Photo],[Photo1],[Landline], [Mobile],[Email],[ParentsProf],[Income],[SchoolName],[Board],[Boardother],[Preiitcoach],[Coachyears],[Firstlang], [Secondlang],[English],[Maths],[Science],[Social],[Total],[IdentificationMark1],[IdentificationMark2])
values(@name,@fname,@dob,@add,@country,@state,@city,@pin,@photo,@photo1,@landline,@mobile,@email,@parentsprof,@income,@schoolname,@board,@examcentre, @preiitcoach,@coachyears,@firstlang,@secondlang,@english,@maths,@science,@social,@total,@mark1,@mark2)
select 'success' as message, @@identity as ID
end
end

2. Get Hall Ticket
USE [E-Exam]
GO
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER proc [dbo].[gethallticket] (
 @id varchar(150)
) as
begin
if exists(select * from [dbo].[Registration] where ([Landline]=@id))
begin

select * from [dbo].[Registration] where ([Landline]=@id)
select '1' as message
end

else if exists(select * from [dbo].[Registration] where ([Mobile]=@id))
begin
select * from [dbo].[Registration] where ([Mobile]=@id)
select '1' as message
end

else if exists(select * from [dbo].[Registration] where ([Email]=@id))
begin
select * from [dbo].[Registration] where ([Email]=@id)
select '1' as message
end

else
begin
select '0' as message
end

end

3.Get All Students List
USE [E-Exam]
GO
/****** Object:  StoredProcedure [dbo].[sp_get_all_students_list]    Script Date: 5/6/2015 12:58:00 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER proc [dbo].[sp_get_all_students_list]
as
begin
select distinct HT_Number, StudentName, FName as FatherName, Dob, Mobile, Email, Address, e.City, d.Region, c.CountryName from tbl_Online_Exam_Student_list as a
inner join Registration as b on a.HT_Number = b.Sno
inner join tblCountry as c on b.Country = c.CountryID
inner join Tbl_State as d on b.State = d.RegionId
inner join tblCities as e on b.City = e.CityId
end

4.0 S/w and H/w requirements

4.0.1 Environment:

- **Servers:**
  - **Operating System Server:** - Microsoft Windows 7 or Higher
  - **Data Base Server:** Microsoft SQL Server 2012

- **Clients:** : Microsoft Internet Explorer,
- **Tools:** : Microsoft Visual Studio .Net 2012
- **User Interface:** : ASP.NET
- **Code Behind**: : C# .NET
4.0.2 Requirements:

- **Hardware requirements:**

<table>
<thead>
<tr>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PC with 56 GB hard-disk and 4GB RAM</td>
</tr>
</tbody>
</table>

- **Software requirements:**

<table>
<thead>
<tr>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Windows 7/8.1/ or Higher with MS-office</td>
</tr>
<tr>
<td>2</td>
<td>SQL server2012</td>
</tr>
<tr>
<td>4</td>
<td>Ms-Internet Explorer</td>
</tr>
</tbody>
</table>

**C#.NET**

**ADO.NET OVERVIEW**

ADO.NET is an evolution of the ADO data access model that directly addresses user requirements for developing scalable applications. It was designed specifically for the web with scalability, statelessness, and XML in mind.

ADO.NET uses some ADO objects, such as the Connection and Command objects, and also introduces new objects. Key new ADO.NET objects include the Dataset, Data Reader, and Data Adapter.

The important distinction between this evolved stage of ADO.NET and previous data architectures is that there exists an object -- the DataSet -- that is separate and distinct from any data stores. Because of that, the DataSet functions as a standalone entity. You can think of the DataSet as an always disconnected recordset that knows nothing about the source or destination of the data it contains. Inside a DataSet, much like in a database, there are tables, columns, relationships, constraints, views, and so forth.

A DataAdapter is the object that connects to the database to fill the DataSet. Then, it connects back to the database to update the data there, based on operations performed
while the DataSet held the data. In the past, data processing has been primarily connection-based. Now, in an effort to make multi-tiered apps more efficient, data processing is turning to a message-based approach that revolves around chunks of information. At the center of this approach is the DataAdapter, which provides a bridge to retrieve and save data between a DataSet and its source data store. It accomplishes this by means of requests to the appropriate SQL commands made against the data store.

The XML-based DataSet object provides a consistent programming model that works with all models of data storage: flat, relational, and hierarchical. It does this by having no 'knowledge' of the source of its data, and by representing the data that it holds as collections and data types. No matter what the source of the data within the DataSet is, it is manipulated through the same set of standard APIs exposed through the DataSet and its subordinate objects.

While the DataSet has no knowledge of the source of its data, the managed provider has detailed and specific information. The role of the managed provider is to connect, fill, and persist the DataSet to and from data stores. The OLE DB and SQL Server .NET Data Providers (System.Data.OleDb and System.Data.SqlClient) that are part of the .Net Framework provide four basic objects: the Command, Connection, DataReader and DataAdapter. In the remaining sections of this document, we'll walk through each part of the DataSet and the OLE DB/SQL Server .NET Data Providers explaining what they are, and how to program against them.

The following sections will introduce you to some objects that have evolved, and some that are new. These objects are:

- Connections. For connection to and managing transactions against a database.
- Commands. For issuing SQL commands against a database.
- DataReaders. For reading a forward-only stream of data records from a SQL Server data source.
- DataSet. For storing, Remoting and programming against flat data, XML data and relational data.
- DataAdapters. For pushing data into a DataSet, and reconciling data against a database.

When dealing with connections to a database, there are two different options: SQL Server .NET Data Provider (System.Data.SqlClient) and OLE DB .NET Data Provider (System.Data.OleDb). In these samples we will use the SQL Server .NET Data Provider. These are written to talk directly to Microsoft SQL Server. The OLE DB .NET Data Provider is used to talk to any OLE DB provider (as it uses OLE DB underneath).
5.0 Data Management

5.1 Data Description

This database consists of

- **Student**: student personal information is added to database with Unique ID.
- **Project**: Admin can create Exams, edit Exam information, add / remove users to a exam and can close exam.

5.2 Data Objects

- **Admin**: Admin user name, admin password
- **Exam**: Exam_Id, Exam_Title.
- **Student**: Sno, SName, hall ticket no, etc.
5.3 ER-DIAGRAMS:
6.0 Non-Functional / Functional Requirements

Functional System Requirement:
This section gives a functional requirement that applicable to the On-Line Exam system. There are three sub modules in this phase.
  Candidate module.
  Examiner module.
  Administrator module.

The functionality of each module is as follows:
Candidate module: The candidate will logon to the software and take his examination. He can also check his previous examinations marks and his details. The candidate will get result immediately after the completion of the examination.

Examiner module: The database is prepared & loaded into the software. Selection for examination can be done language wise by the examiner. The results will be displayed immediately after completion of the examination.

Administrator module: The administrator collects all the results after successful completion of the examination and sends to the headquarters as and when required.

The features that are available to the Administrator are:
- The administrator has the full-fledged rights over the OES.
- Can create/delete an account.
- Can view the accounts.
- Can change the password.
- Can hide any kind of features from the both of users.
- Insert/delete/edit the information of available on OES.
- Can access all the accounts of the faculty members/students.

The features available to the Students are:
- Can view the different categories of Test available in their account.
- Can change password.
- Can view their marks.
- Can view the various reading material.
- Can view and modify its profile but can modify it to some limited range.

The features available to the Examiner are:
- Can view the different categories of Test conducted by users.
- Can change password.
• Can view their marks.
• Can view and modify Results.

Non-Functional System Requirements:

Performance Requirements
Some Performance requirements identified is listed below:
• The database shall be able to accommodate a minimum of 10,000 records of students.
• The software shall support use of multiple users at a time.
• There are no other specific performance requirements that will affect development.

6.1 Security and Reliability

6.1.1 Security

• Pages of the website must be access in the way they were intended to be accessed. Included files shall not be accessed outside of their parent file.
• Administrator can only perform administrative task on pages they are privileged to access. Customers will not be allowed to access the administrator pages.

6.1.2 Reliability

• The database should maintain data normalization by implementing a primary and foreign key system so that discrepancies do not occur within the data.

6.2 Maintainability

Maintenance the last phase in the software engineering process. As programs are developed.

A distributing trend has emerged the amount of effort and a resource expended on software maintenance is growing. In total project development maintenances takes 65% of effort. In software maintenance there are four. They are

• Adaptive Maintenance
• Corrective Maintenance
Adaptive Maintenance is applied when changes in the external environment precipitate modifications to software. I deal with adapting the software to new environments. Perfective Maintenance incorporates enhancements that are requested by user community. It deals with updating the software according to changes in user requirements. Corrective Maintenance acts to correct errors that are uncovered after the software is in use. It deals with fixing bugs in the code. Preventive Maintenance improves future maintainability and reliability and provides a basis for future enhancement. It deals with updating documentation and making the software more maintainable. Tasks performed during the software engineering process define maintainability and have an important impact in the success of any maintenance approach. Reverse Engineering and Reengineering are the tools and techniques used to maintain the project.

- Page loads should be returned and formatted in a timely fashion depending on the request being made.
7.0 Conclusion

It has been a great pleasure for me to work on this exciting and challenging project. This project proved good for me as it provided practical knowledge of not only programming in ASP.NET and C#.Net web based application and no some extent Windows Application and SQL Server, but also about all handling procedure related with “E-Engine system”. It also provides knowledge about the latest technology used in developing web enabled application and client server technology that will be great demand in future. This will provide better opportunities and guidance in future in developing projects independently.
8. REFERENCES:

- FOR .NET INSTALLATION
  www.support.microsoft.com
- FOR DEPLOYMENT AND PACKING ON SERVER
  www.developer.com
  www.15seconds.com
- FOR SQL
  www.msdn.microsoft.com
- FOR ASP.NET
  www.msdn.microsoft.com/net/quickstart/aspplus/default.com
  www.asp.net
  www.fmexpense.com/quickstart/aspplus/default.com
  www.asptoday.com
  www.aspfree.com
  www.4guysfromrolla.com/index.aspx