Linnæus University
School of Computer Science, Physics and Mathematics

Degree Project

Online assignment submission

Baha'Aldeen Amayreh
2010-12-22
Subject: Software Technology
Level: Master
Course code: 4DV01E

Abstract

This project is aimed at downloading and uploading online assignments for students; with each assignment having information about the instructions, description, deadline, and submission details.

The main goal of this thesis is to design and implement online assignment submission and provide an interface use for uploading test program (Nant script file) by instructors, who would be able to evaluate assignments automatically.

The system provides an interface for testing assignments such that they can be plugged in by the teachers. This testing could invoke a compiler and make a test-run of the compiled code and check the result or test for plagiarism, existence of certain documents or simply check the file type (extension).

The most obvious advantage offered by online assignment submission is that it offers faster transmission of assignments than using traditional way by using online system. The interface use to invoke different testing program by teachers, So Save the time and cost for teachers by enabling them to put up a fast response for students as well as increasing the quality of the feedback provided to students.

Keywords: UML (Unified Modeling Language), Nant (Not Ant), OSS (online assignment Submission), Script file, DotNet2005.

Acknowledgements

Also, I would like to extend my warmest thanks and appreciation for the **Linnaeus** University for giving me the opportunity to complete my graduate studies. My thanks go particularly to Professor Welf Löwe as well as to the Program coordinator Dr.Jonas Lundberg.

Finally, I would like to thank Mr. Mathias Hedenborg, and every other individual who have facilitated this work.

Table of content

List of figures

List of tables

1. Introduction

Informational challenges play a significant role in societal development since they contribute immensely in the proliferation of knowledge. Easy and effective proliferation of knowledge should be the main raison d'être of modern information technology.

Colleges and universities are considered the main provider of know-how in various fields. At these institutions, various courses of studies are taught, covering several fields including applied sciences, math and computer. Normally, a given course at college consists of the theoretical as well as practical subject matter. To evaluate the degree of comprehension among students, assignments are given. However, evaluating and marking the assignments by instructors are a problematic and time-consuming process.

1.1 Background and purpose

At the University of **Linnaeus** there is a system of receiving student assignments called Blackboard. There is also the email system which is used to send assignments directly to the instructor's email address. However, this system is very much confined to receiving the assignments. In this case, the instructor is left with the formidable task of downloading the assignments, reading, evaluating and marking them, employing the conventional process.

So, this laborious process is still conventional despite the use of internet in sending and receiving student assignments. Indeed, the marking process carried out by the instructor or a Graduate Assistant takes a lot of time, which means more efforts and energy. It also means that that the marked assignments would have to be sent back to the students somewhat belatedly if not lately.

For the resolution of this problem, a special system ought to be devised in order to carry out the bulk of the marking process and verification of the assignments. Such a system would have to be capable of functioning through the internet. The same system would have also to be connected with available technological tools helping in the process of marking assignments.

The most obvious advantage offered by online assignment submission is that it offers faster transmission of assignments than using traditional way by using online system. The interface use to invoke different testing program by teachers, So Save the time and cost for teachers by enabling them to put up a fast response for students as well as increasing the quality of the feedback provided to students.

1.2 Goal of Study

This study is aimed at developing an electronic system (Generic Interface) which has the ability to receive, verify and mark assignments sent by the students to every respective course instructor.

The system should correct the submitted assignments (each assignment contains 3 visual basic files) within 3 minutes for twenty students and updates the results automatically.

1.3 Restriction

Hardware interfaces: This system needs a web server on which the system is being run and other computers via the Internet.
Software interfaces: Net platform 2005 and Nant technology is used in this project. Explanation of these technologies and motivations will be given in chapter 2.

1.4 Report structure

Here I will introduce the structure of the thesis paper.

In chapter 2, we shall be explaining the technologies and tools, programming languages and methodology used in this project.

Chapter 3 will explore the system requirement analysis (Functional and non - Functional requirements) and requirement specifications.

Chapter 4 describes the system design of OSS and the topics in this chapter are Use-case diagrams, class diagram, sequence diagrams and data base.

Chapter 5 System implementation discusses the process of coding and implementation of the system.

Chapter 6 System Testing discusses a practical example of electronic marking through the use of the Generic Interface System.

Chapter 7 Conclusion, Recommendations and future work.

2. Environment of implementation

In this chapter, we shall be explaining the technologies and tools, programming languages and methodology used in this project.

2.1 Technologies and tools

In this project I will use the ASP.Net 2005 and Nant technologies.

ASP.NET is a framework for web applications developed and marketed by the Microsoft Company for the purpose of enabling programmers to build dynamic websites whereby the web applications can be written in several languages, e.g. Visual basic, C#, etc.

Visual basic shall be used in building this special website which will receive student assignments.

The choice of language was open and the choice ended with Visual basic since my knowledge in VB was good.

Nant (Not Ant) "is free .NET build tool, in the theory its kind make without makes, in particle it is a lot like Ant".[1]

Explanation of this technology will be given in chapter 5.

- **Alternatives**

Java language is considered a very strong and powerful language and this language supports desktop and internet applications(JSP) and it is free of charge, in addition to this java language is used to develop ant technology,

ASP technology is used cause the researcher expert is good in this technology, in addition to this the limited period (2 months) for this research obligates me to use it, since it is fast in development and easy to use and installation also it supports the ant (ANAT) technology, JSP language is left since the researcher expert in it is limited.

2.2 Thesis/project codes

All codes are programmed in **Visual basic2005 Dot Net**.

ASP.NET allows you to use a far greater selection of full programming languages, create faster, more reliable dynamic web pages with any of the programming languages supported by the .NET Framework, it is a platform- and device-independent system that is designed to work over the Internet, With Microsoft .NET, developing Web applications is much easier, now a days ASP.Net support more than 20 languages (VB, C#, JSP).

2.3 Methodology

It is essential that any programmer must thoroughly know the language he or she uses when designing and analyzing. The programmer would have to analyze the program and then knows the problem he needs to solve. He would then perform the process of coding while applying the process of design which he presented previously. Finally, he or she would have to test the program in order to ascertain compatibility with customer requirement.

The process I have mentioned, including Analysis, Design, Coding and Testing, identify the project's Software Development Life Cycle as any project would have to go

through all these processes using the appropriate methodology. Otherwise, chaos would ensue.

We shall use the Interactive and incremental development methodology in order to develop a prototype system. This process is characterized with flexibility and revision whenever necessary in all phases. The process would begin with an initial plan and concluded with interaction among he various phases and components.

As to designing the processes used in describing and interactions in this project, we shall be using the UML (**Unified Modeling Language).**

2.4 Economic Feasibility

The technology used in this research will be freely available since the framework of ASP.Net2005 and Microsoft Visual Studio Express Edition IDE (integrated development environment) are free of charge. [2] The same can be said about the "Nant technology".

3. Requirement analysis for an OSS (online assignment submission)

This chapter will explore the system requirement analysis (Functional and non - Functional requirements) and requirement specifications.

3.1 Requirement analysis and Functional requirements

Next, the functional requirements will be detailed as following:

3.1.1 Student

- The student requests the OSS site from web server using internet browser.
- The student inserts his username and password at log in area in the login Page.
- If the student inserts a valid username and password he or she will see the main menu that contains the following items:
    1. Name information.
    2. Personal information: personal number, name, email and password.
    3. Course information: course name, number, point, semester and related assignments for each course.

Functional requirements:

- Student can update his personal data.
- Student can upload the solution of any of his assignments before the deadline (if strict).
- Student will receive a message from the system after uploading the solution (accept or reject).
- Student gets feedback from the teacher.
- Student can see the information about the assignment (description and instruction, start-time, end-time, motivation, how to download and upload the assignment and type of work: individually or as groups).

3.1.2 Teacher

- The teacher requests the OSS site from web server using internet browser.
- The teacher inserts his username and password at login area in the login Page.
- If the teacher inserts a valid username and password he will see the teacher main menu that contains the following items:
    - Name details.
    - Address information: personal number, name, and email.

Functional requirements:

- Teacher can administer (add, delete, and update) courses and oversee students using a simple XML (Extensible Markup Language) file.
- In this XML file the teacher can add
  - § Assignment information (description and instruction, start-time, end-time, motivation, how to download the assignment and upload it, type of work: individually or as groups).
  - § Login information (accounts for the students)
- Teacher can upload a test Ant/Nant script for testing the assignments for all students by using the generic interface.
- Teacher can see the assignments submitted by students. He can assess the students by browsing a report of the assignments and students.
- Teacher can manually send feedback, marks and notes to students.

3.2 Non Functional requirements

- Safety, the system must have the ability to prevent illegal or incorrect operations from teachers or students by using certain tools such as validation control.
- Understandability, which makes it easy for students to use and deal with, user friendly by developing good interface and data accessibility, must be easy.
- Secure and private.
- The system developed for the purpose of supporting integration between existing and future systems.
- Accessibility, the system is available via the Internet and can be accessed any time and any place by the internet.
- Performance, the system must be fast.

3.3 Requirement specifications

This section explores all the OSS functions in details.

1. The user requests the OSS home page from web server.

Function: User requests the page from web server.
Description: This function provides ability to browse the OSS.
Input: The OSS URL.
Source: User.
Output: OSS home page.
Destination: Web server.
Require: Insert correct site address.
Pre-condition: Availability of internet service.
Post-condition: Displaying the OSS home page.

2. Login.

Function: Login.
Description: Enabling the student, teacher and administration to access his account, see his data and use menu by using valid username and password.
Input: Username and password.
Source: Student, teacher.
Output: The student or teacher home page**.**
Destination: Web server.
Require: Valid login by correct username and password and having an account on the OSS.
Pre-condition: OSS Home Page.
Post-condition: Connect student or teacher to OSS home page.

3. Display Student profile.

Function: Displaying the student information.
Description: By clicking the profile link from the main menu a new page will be displayed containing the student information.
Input: Clicking my profile link from student main menu.
Source: The student and student menu web form.
Output: The student profile**.**
Destination: Web server.
Require: Valid log in and single click on my profile item.
Pre-condition: Home Page and no student displayed in main menu.
Post-condition: The student can see his profile.

4. Update student profile.

Function: Update student data
Description: The student can update his information (name, email and password).
Input: By clicking my profile link from student main menu and entering some specific information to the OSS and saving them in the web server.
Source: The student and student menu web form.
Output: Save student data in the OSS system.
Destination: Web server.
Require: Valid login and single click on update –my- profile item.
Pre-condition: Student Home Page + having an account on the OSS.
Post-condition: Save data in web server (Student XML file).

5. See the assignment.

Function: Student can see the information about the assignment (description and instruction, start-time, end-time, motivation, how to download the assignment and upload it, type of work individually or as groups).
Description: By clicking the courses link from the main menu a new page will be displayed, containing courses and related assignments.
Input: Clicking my courses link from student main menu.
Source: The student and courses link from menu web form.
Output: Courses and related assignments**.**
Destination: web server.
Require: Valid log in and single click on courses link.
Pre-condition: Student Home Page and registered student courses.
Post-condition: Courses assignments displayed.

6. Uploading the solution.

Function: Upload solution of the assignments.
Description: By clicking the courses link from the main menu a new page will be displayed, containing the courses and related assignment information, the student can upload the solution of his assignments before the deadline (if strict).
Input: Clicking on my courses link from student main menu and select the assignments from the grid view and upload the solution and saving them in the web server.
Source: The student and student menu web form.
Output: Save solution information in the OSS.
Destination: Web server.
Require: Valid login and single click on course link.
Pre-condition: Student Home Page and registered student courses and submit the assignments before the deadline (if strict).
Post-condition: Save data in web server (solution XML file) and received message from the system after uploading the solution (accept or reject) when the teacher make the correction process.

7. Registration courses for students.

Function: Registration of courses for students.
Description: This function Provides administrators with the ability to register courses for students that offered at the beginning of current semester.
Input: Click on registration link from administrator menu and choose the course number and student number and saving them in the web server.
Source: Administrator and registration of web form.
Output: Save student registration.
Destination: Web server.
Require: Valid login and single click on the registration link**.**
Pre-condition: Select course and student number.
Post-condition: Displaying student and registered courses then save it in registration xml file.

8. Teacher can add, delete and update courses.

Function: Add, delete and update courses.
Description: This function Provides administrator the ability to insert, delete and update courses.
Input: Click on insert course link from administrator main menu and fill course name, course number, course credit, course level and the course type text, or click on update courses link to update course or delete it from courses xml file.
Source: Administrator and insert new course web form or update course.
Output: Save the course and its information in courses xml file.
Destination: Web server.
Require: Valid login and single click on courses link (insert or update).
Pre-condition: Admin Home Page and fill all textboxes with valid data type.
Post-condition: Save courses in simple course xml files or update it.

9. Teacher can add, delete and update Students.

Function: Add, delete and update Students.
Description: This function Provides administrator the ability to insert, delete and update Students.
Input: Click on insert Students link from administrator main menu and fill Students name, ID, password and the email text, or click on update Student link to update Students or delete students from courses xml file.
Source: Administrator and insert new Students web form or update Students.
Output: Save the Student and his information in Students xml file.
Destination: Web server.
Require: Valid login and single click on Students link (insert or update).
Pre-condition: Administration Home Page and fill all textboxes with valid data type.
Post-condition: Save Student in simple student xml files or update it.

10. Teacher can add, delete and update assignments.

Function: Add, delete and update assignments.
Description: this function Provides administrator the ability to insert, delete and update assignments.
Input: Click on insert assignments link from administrator main menu and fill assignments number, Course number, description, instruction, start time , end time, motivation, type of work, is strict and the URL text, or click on update assignments link to update assignments or delete students from assignments xml file.
Source: Administrator and insert new assignments web form or update assignments.
Output: Save the assignments and its information in assignments xml file.
Destination: Web server.
Require: Valid login and single click on assignments link (insert or update).
Pre-condition: Admin Home Page and fill all textboxes with valid data type.
Post-condition: Save assignment in simple assignment xml files or update it.

11. Teacher can upload a test Ant/Nant script for testing the assignments.

Function: Add Ant/Nant script for testing the assignments.
Description: This function Provides administrator the ability to upload a test Ant/Nant script for testing the assignments for all students.
Input: Click on correction link from administrator main menu and select the Course name and the assignments to upload the script text.
Source: Administrator and upload Ant/Nant script.
Output: Save the Ant/Nant script and its information in correction xml file.
Destination: Web server.
Require: Valid login and single click on correction link.
Pre-condition: Admin Home Page and fill all textboxes with valid data type.
Post-condition: Save Ant/Nant script in correction xml file.

12. Teacher can manually send feedback, marks and notes to students.

Function: Send feedback and notes to students.
Description: Administrator can provide students with the feedback and notes.
Input: Click feedback link from administrator main menu.
Source: Administrator and feedback web form.
Output: Display feedback and notes for students.
Destination: Web server.
Require: Valid login and single click on feedback link.
Pre-condition: Admin Home Page and fill all textboxes with valid data type.
Post-condition: Save feedback and notes in correction xml file.

13. Insert Marks

Function: Insert marks.
Description: This function Provides administrator the ability to insert marks of courses offered at the end of current semester.
Input: Click on insert marks link from administrator menu and choose the course number and student number then insert marks.
Source: Administrator and insert marks web form.
Output: Save courses marks for each student registered in the previous specified course.
Destination: Web server.
Require: Valid login and single click on marks link**.**
Pre-condition: Select course number and student number.
Post-condition: Display students and insert marks saved in registration xml file.
.

14. Generate reports

Function: Generate reports
Description: Teacher can see the assignments submitted by students. He can assess the students by browsing a report of the assignments and students.
Input: Click on report link from administrator menu.
Source: administrator and report web form.
Output: Report on student data and evaluation of home assignments.
Destination: Web server.
Require: Valid login and single click on reports link**.**
Pre-condition: Admin Home Page.
Post-condition: Displaying students

4. System design

This chapter describes the system design of OSS and the topics in this chapter are Use-case diagrams, class diagram, sequence diagrams and data base design.

4.1 Use case diagram for OSS

"Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases" [3].

4.1.1 Teacher- Administrator

Teacher can add, delete or update students; courses, assignments, and uploading Nant script file to testing all assignments.

Description

This package contains all the functionalities that an administrator can do.

Use Cases

- Login to OSS
- Add, delete and update students.
- Add, delete and update courses.
- Add, delete and update assignments.
- Registration courses for students.
- Uploading Ant/Nant script files.
- Running the Ant/Nant script files for testing the assignments.
- Generate Report for students.

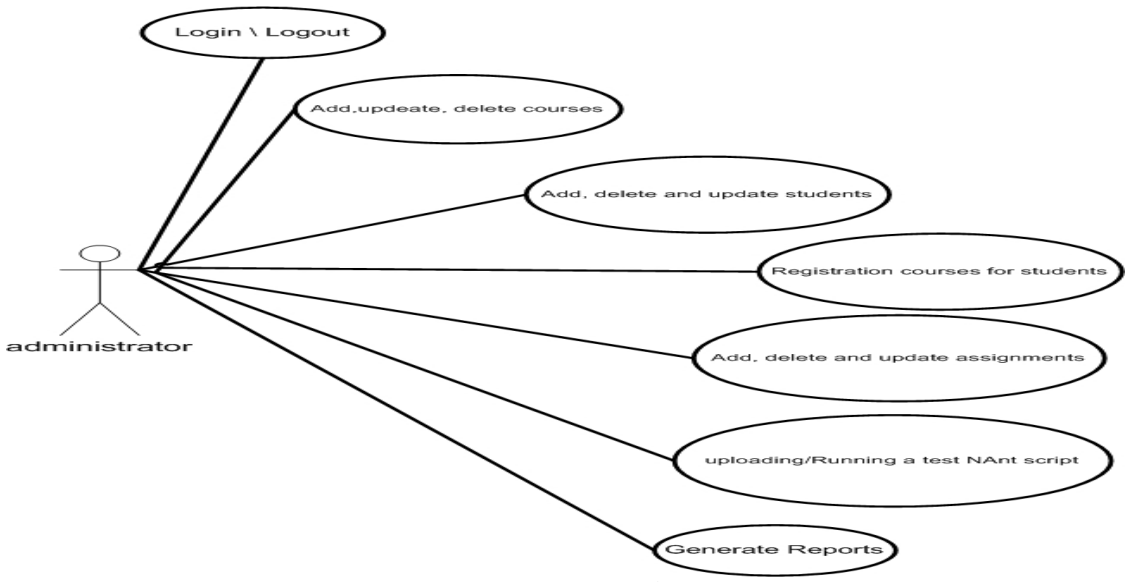


Figure 4.1: Use-case diagram for administrator

4.1.2 Student

Student can see the information about the assignment (description and instruction, start-time, end-time, motivation, how to download the assignment and upload it, type of work individually or as groups).

Description

This package contains all the functionalities that student can do.

Use Cases

- Login to OSS
- Update profile
- View the register courses
- View the assignments
- Uploading the solution of assignments.

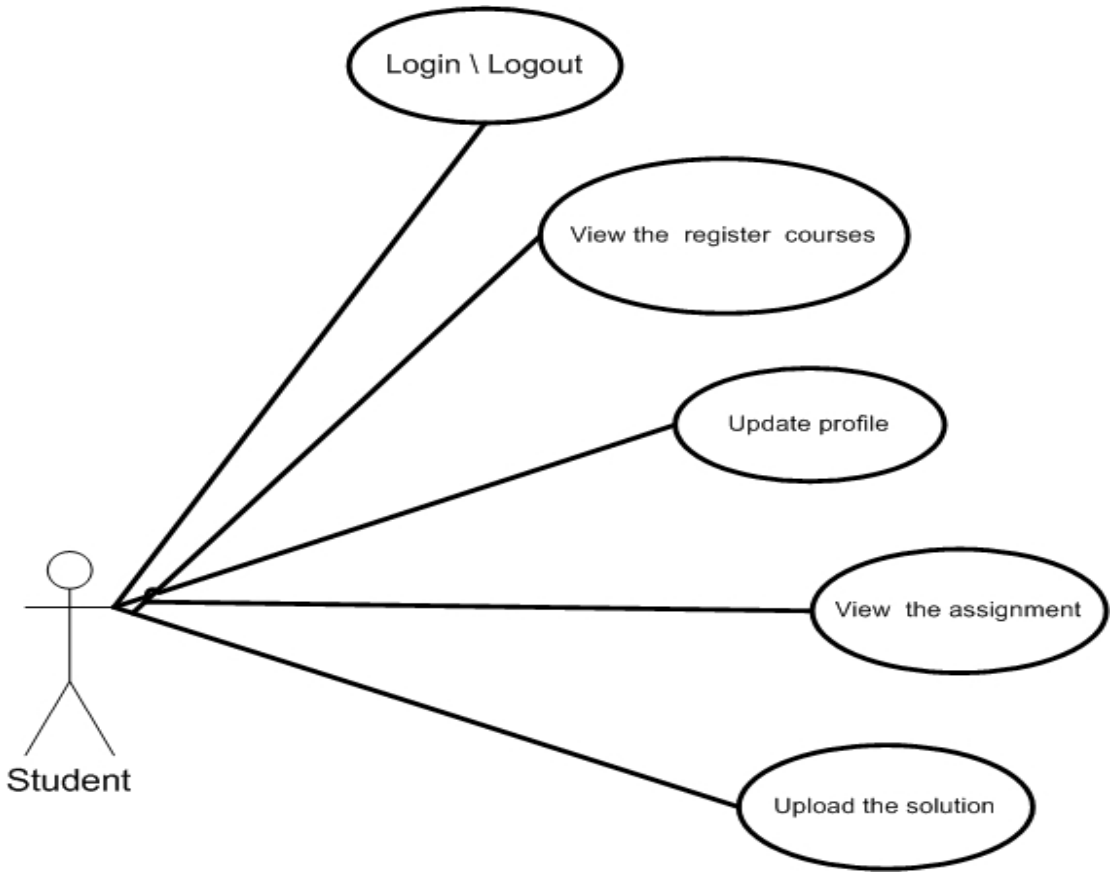


Figure 4.2: Use-case diagram for student

4.2 Class diagram for OSS

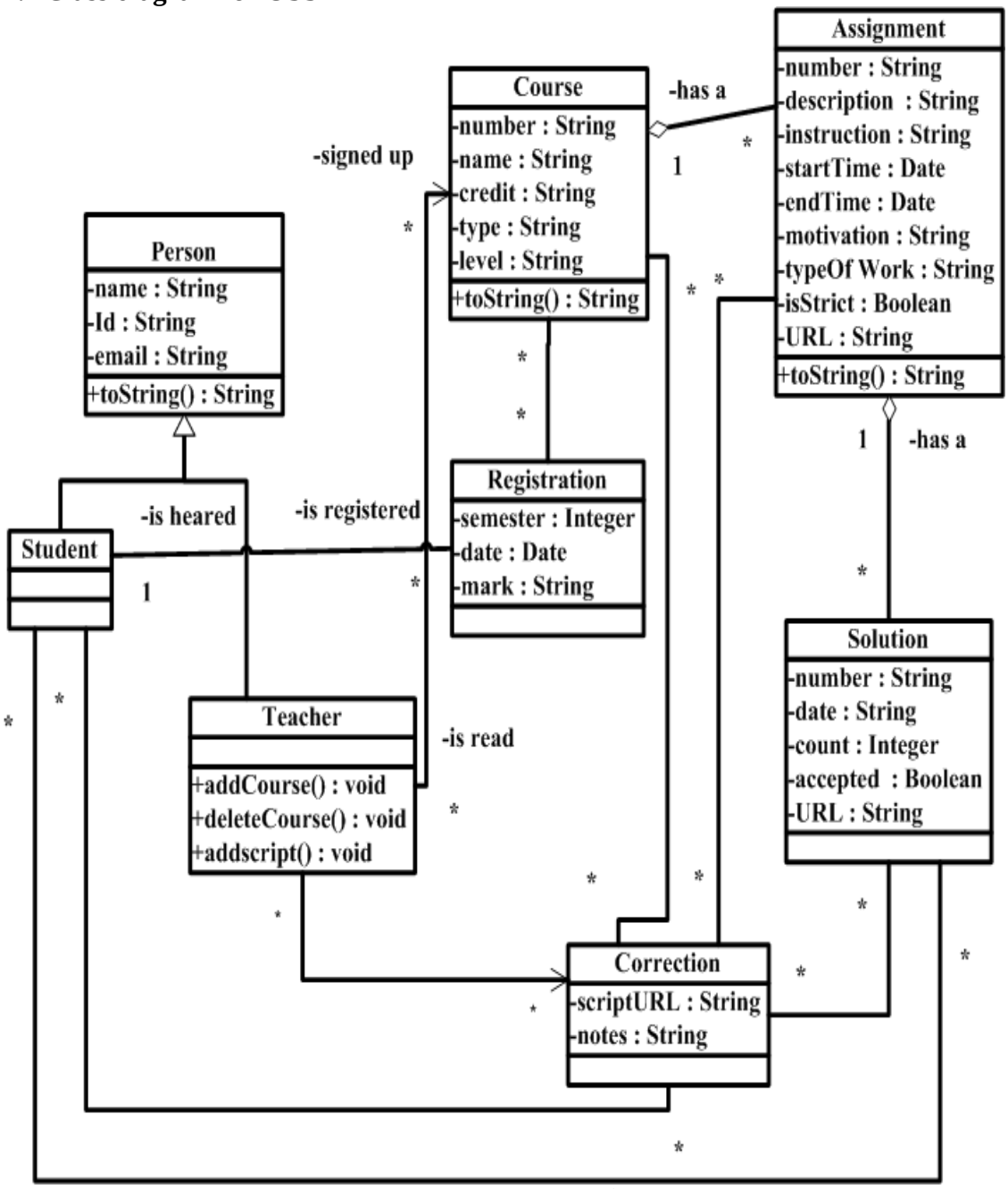


Figure 4.3: Class diagram for OSS

These classes are use to understand the problem domain of OSS, and to help designer in design data bas model, one of these classes is correction use to store data about Nant script file that is including script file name, the path of script file and the batch file use to run the script file.

4.3 Sequence diagrams

A sequence diagram is a graphical description of objects participating in a use case or scenario, sequence diagrams are derived from the use cases module and The structure of the sequence diagram help me to determine how decentralized the system is.

4.3.1 The main scenario by Teacher- Administrator

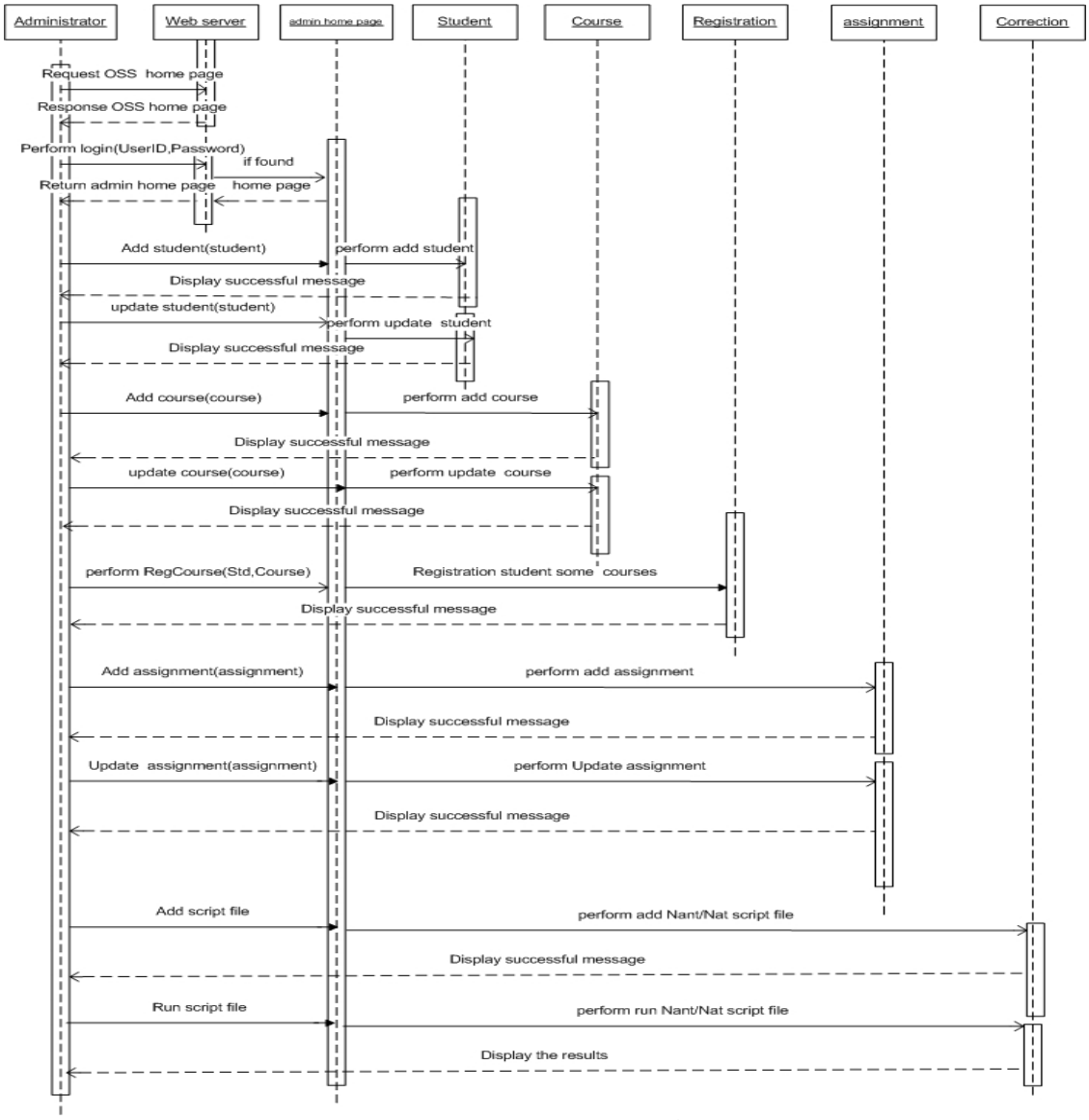


Figure 4.4: Sequence diagrams for teacher

Main scenario of teacher by clicking on correction link from administrator main menu and select the Course name and the assignments to upload the script text (Nant script XML file).

4.3.2 The main scenario by Student

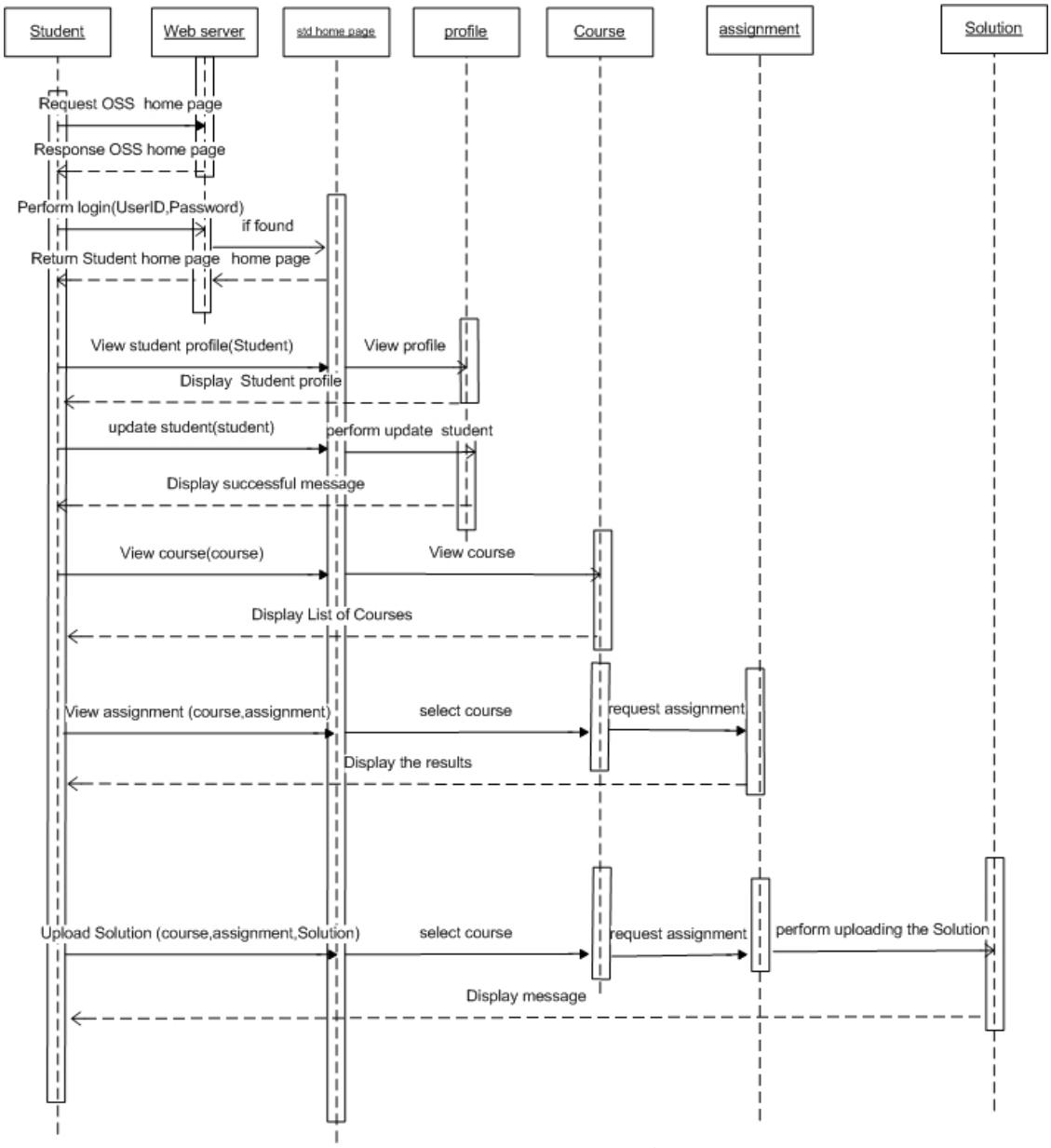


Figure 4.5: Sequence diagrams for student

Student can be interactive with many objects like course, assignment, solution and student profile, the main scenario of student by clicking the courses link from the main Menu a new page will be displayed, containing the courses and related assignment information; the student can upload the solution of his assignments before the deadline (if strict).

4.4 Data Base

Since this project is in its initial stages, and is still being developed and updated, we shall be using XML (Extensible Markup Language) files in order to store various data. This technology is useful for several reasons, including the possibility of speedily and easily amending database, XML standards so that structured data will be uniform and independent of applications, the flexibility encountered in the use of the system as well as the ease in recalling and reading the data and ensuring applicability through the internet [4].

The project includes seven basic tables which will be used for storing data pertaining to students and instructors as well as course and college assignments and other materials, show in below.

Field Name	Data Type	Null	Key	References	Length	Description
name	String	No			50	Teacher name
ID	String	No	PK		50	Teacher number
password	String	No			50	Teacher password
Email	String	No			50	Teacher email

Table 4.1: teacher table

Field Name	Data Type	Null	Key	References	Length	Description
name	String	No			50	Student name
ID	String	No	PK		50	Student number
password	String	No			50	Student password
Email	String	No			50	Student email

Table 4.2: student table

Field Name	Data Type	Null	Key	References	Length	Description
name	String	No			50	Course name
number	String	No	PK		50	Course number
credit	Integer	No			4	Course credit
type	String	No			50	Course type
level	String	No			50	Course level

Table 4.3: course table

Field Name	Data Type	Null	Key	References	Length	Description
Number	String	No	PK		50	Assignment number
CourseNumber	String	No	PK	Course (number)	50	Course number
Description	String	No			50	Description of the assignment
Instruction	String	No			50	How to upload and download the assignment
StartTime	Datetime	No	PK		8	Start time of the assignment
EndTime	Datetime	No	PK		8	End time of the assignment
Motivation	String	Yes			50	The benefits of this assignment
TypeOfWork	Integer	Yes			4	the allowed number of students in this assignment
IsStrict	Boolean	No			2	If it is possible to submit the assignment after the deadline
URL	String	Yes			50	Uniform resource locator of the assignment

Table 4.4: assignments table

Field Name	Data Type	Null	Key	References	Length	Description
StdID	String	No	PK	Student (ID)	50	Student number
StdName	String	No		Student (name)	50	Student name
CourseID	String	No		Course (number)	50	Course number
CourseName	String	No		Course (name)	50	Course Name
Semester	Integer	No	PK		4	Course semester
Date	Datetime	No	PK		8	Date of registration
Mark	Integer	Yes			4	Student Mark

Table 4.5: registration table

Field Name	Data Type	Null	Key	References	Length	Description
StudentNumber	String	No	PK	Student (ID)	50	Student number
CourseNumber	String	No		Course (number)	50	Course number
AssignmentNumber	String	No		Assignment (number)	50	Assignment number
StartTime	Datetime	No	PK		8	Start time of the assignment
EndTime	Datetime	No	PK		8	End time of the assignment
SubmitDate	Datetime	Yes			8	Submit date
count	Integer	Yes			4	The number of times of uploading the assignment
accepted	Boolean	Yes			2	Accepted the assignment or not
URL	String	Yes			50	Uniform resource locator of the solution

Table 4.6: solution table

Field Name	Data Type	Null	Key	References	Length	Description
CourseName	String	No		Course (name)	50	Course name
CourseNumber	String	No	PK	Course (number)	50	Course number
AssignmentNumber	String	No		Assignment (number)	50	Assignment number
StartTime	Datetime	No	PK		8	Start time of the assignment
EndTime	Datetime	No	PK		8	End time of the assignment
ScriptURL	String	No			50	Uniform resource locator of the script file(NANT)
BatURL	String	No			50	Uniform resource locator of the batch file
notes	String	Yes			100	feedback

Table 4.7: correction table

Notices:

NULL: is not a value, it is something unknown.
References: is a referential constraint between two tables (foreign key).
Foreign key: "identifies a column or a set of columns in one (referencing) table that refers to a set of columns in another (referenced) table" [5].

(For implementation see Appendix A: Data Base implementation)

5. System implementation

In this chapter, I will discuss the process of coding and implementation of the system.

5.1 Introducton

The development of my software system to work properly to meet its predetermined requirements on the internet architecture needs a set of certain software and hardware products found in a platform configured to be suitable for the deployment process.

This system is an internet application that depends on a number of technologies that need to be installed, maintained, and updated, continuously.

There exist a large number of software development packages that belong to different companies such as Microsoft.

This system is built basically on a group of Microsoft technologies such as ASP.NET 2005 and Microsoft windows XP, also included Nant technology that serve to automate the entire build process.

5.2 Coding

The principle improvements in programming are the reusability and code auto generation, this improvement and others reduce the time of programmer to write code.

For development OSS, I used visual Studio.net 2005; which simplifies the development of powerful and reliable software application by providing familiar and shared development environment.

It contain pre-built component and programming wizard, as well the ability to use component built using various language.

In visual studio.net there is a single integrated development environment (IDE), which provides a sense of what you see is what you get (the visual programming environment) [6].

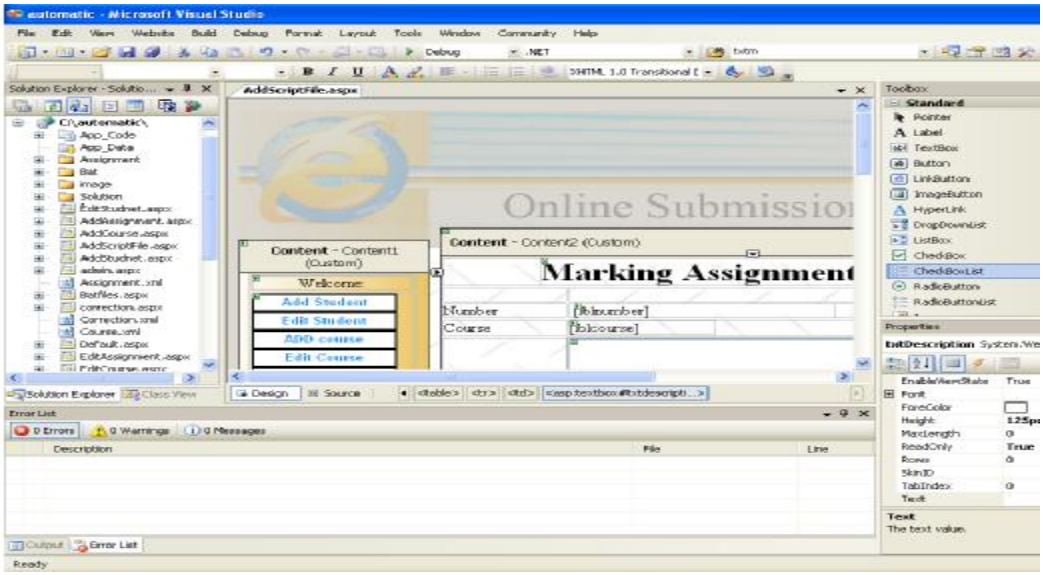


Figure 5.1: Microsoft Visual Studio Express Edition2005

The usage of this tool for the purpose of programming and coding reduce the time and efforts and thereby increasing the performance.

When using Visual Studio.Net as a programming environment we gain the benefits of the separation between writing the logical code (the program functionality) from one side and the design of the appearance and graphical user interface (GUI) from the other side.

5.2.1 Default.aspx.vb

This class enable students or teacher to login to OSS.

```
Imports System.Web.Security
Imports System.Xml
Imports System.Xml.XPath
Imports System.Data
Partial Class _Default
   Inherits System.Web.UI.Page
   Protected Sub btnsubmit_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles btnsubmit.Click
      Dim student As New DataSet
      Dim teacher As New DataSet
      Dim j As Integer = 0
      Dim found As Boolean = False
      Dim x As String
      Dim name As String = ""
      teacher.ReadXml(MapPath("Teacher.xml"))
      While (j < teacher.Tables(0).Rows.Count And found = False)
         If (teacher.Tables(0).Rows(j).Item("ID") = Me.txtuser.Text And
teacher.Tables(0).Rows(j).Item("Password") =
FormsAuthentication.HashPasswordForStoringInConfigFile(Me.txtpass.Text, "Md5"))
Then
            found = True
            x = teacher.Tables(0).Rows(j).Item("ID")
            Session("x") = x
            Session("name") = teacher.Tables(0).Rows(j).Item("Name")
            Me.Response.Redirect("admin.aspx")
            Me.lblerror.Visible = False
         End If
         j = j + 1
      End While
      If found = False Then
         student.ReadXml(MapPath("Student.xml"))
         j = 0
         While (j < student.Tables(0).Rows.Count And found = False)
            If (student.Tables(0).Rows(j).Item("ID") = Me.txtuser.Text And
```

```
student.Tables(0).Rows(j).Item("Password") =
FormsAuthentication.HashPasswordForStoringInConfigFile(Me.txtpass.Text, "Md5"))
Then
                found = True
                x = student.Tables(0).Rows(j).Item("ID")
                Session("Password") = student.Tables(0).Rows(j).Item("Password")
                Session("Email") = student.Tables(0).Rows(j).Item("Email")
                Session("x") = x
                Session("name") = student.Tables(0).Rows(j).Item("Name")
                Session("counter") = j
                Me.Response.Redirect("student.aspx")
                Me.lblerror.Visible = False
            End If
            j = j + 1
        End While
    End If
    If found = False Then
        Me.lblerror.Visible = True
    End If
  End Sub
End Class
```

5.2.2 Studnet.aspx.vb

This class Provides administrator the ability to insert new student.

```
Imports System.Web.Security
Imports System.Xml
Imports System.Xml.XPath
Imports System.Data
Partial Class AddStudnet
  Inherits System.Web.UI.Page
  Protected Sub Button1_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Button1.Click
    Dim ds As New DataSet
    Dim Dr As DataRow
    Dim j As Integer = 0
    Dim found As Boolean = False
    ds.ReadXml(MapPath("Student.xml"))
    While (j < ds.Tables(0).Rows.Count And found = False)
        If (ds.Tables(0).Rows(j).Item(1) = Me.txtId.Text And
ds.Tables(0).Rows(j).Item(0) <> "") Then
            found = True
        End If
```

```
            j = j   1
        End While
        If found = False Then
            Dr = ds.Tables(0).NewRow
            Dr("Name") = Me.txtName.Text
            Dr("ID") = Me.txtId.Text
            Dr("Password") =
FormsAuthentication.HashPasswordForStoringInConfigFile(Me.txtPassword.Text,
"Md5")
            Dr("Email") = Me.txtEmail.Text
            ds.Tables(0).Rows.Add(Dr)
            ds.WriteXml(Request.PhysicalApplicationPath   "Student.xml")
            Me.Label1.Text = "This student added in  XMl student file."
            Me.txtName.Text = ""
            Me.txtId.Text = ""
            Me.txtPassword.Text = ""
            Me.txtEmail.Text = ""
        Else
            Me.Label1.Text = "This student ID already in XMl file, you can not add two
student with the same ID."
        End If
    End Sub

    Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load
        Dim x As String
        Dim name As String
        x = Session("x")
        name = Session("Name")
        If Session("x") Is Nothing Then
            Me.Response.Redirect("default.aspx")
        Else
            If Me.IsPostBack = False Then
                Me.lblwelcome.Text  = name
            End If
        End If
    End Sub
End Class
```

5.2.3 AddScriptFile.aspx.vb

This class enable teacher to Add Ant/Nant script for testing and evolution the assignments automatically .

```
Imports System.Xml
Imports System.Xml.XPath
Imports System.Data
Imports System.IO
Imports System.IO.StreamReader
Imports System.IO.FileStream

Partial Class AddScriptFile
   Inherits System.Web.UI.Page
   Dim assignment As New course
   Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load
      Dim x As String
      Dim name As String
      Dim visible As Boolean
      x = Session("x")
      name = Session("Name")
      visible = Session("yes")
      If Session("x") Is Nothing Or Session("assignment") Is Nothing Then
         Me.Response.Redirect("Correction.aspx")
      End If
      If Me.IsPostBack = False Then
         Me.lblwelcome.Text += name
         Dim ds As New DataSet
         Dim j As Integer = 0
         Dim found As Boolean = False
         assignment = Session("assignment")
         ds.ReadXml(MapPath("Assignment.xml"))
         Me.lblnumber.Text = assignment.assignment_number
         Me.lblcourse.Text = assignment.course_name
         Me.txtcornumber.Text = assignment.course_number
         While (j < ds.Tables(0).Rows.Count And found = False)
            If (ds.Tables(0).Rows(j).Item("Number") = assignment.assignment_number
               And ds.Tables(0).Rows(j).Item("CourseNumber") =
               assignment.course_number And ds.Tables(0).Rows(j).Item("StartTime") =
               assignment.assignment_Sdate And ds.Tables(0).Rows(j).Item("EndTime") =
               assignment.assignment_Edate) Then
              found = True
              Me.txtDescription.Text = ds.Tables(0).Rows(j).Item("Description")
              Me.lblinstruction.Text = ds.Tables(0).Rows(j).Item("Instruction")
```

```
            Me.lblStime.Text = ds.Tables(0).Rows(j).Item("StartTime")
            Me.lblEtime.Text = ds.Tables(0).Rows(j).Item("EndTime")
            Me.txtMotivation.Text = ds.Tables(0).Rows(j).Item("Motivation")
            Me.lblTypeofWork.Text = ds.Tables(0).Rows(j).Item("TypeOfWork")
            Me.lblstrict.Text = ds.Tables(0).Rows(j).Item("IsStrict")
            If (ds.Tables(0).Rows(j).Item("URL") <> "") Then
              Me.URL.Visible = True
              Me.URL.NavigateUrl="Assignment\"+ds.Tables(0).Rows(j).Item("URL")
            End If

          End If
          j = j + 1
        End While
      End If
      If visible = True Then
        Me.Button1.Visible = True
      Else
        Me.Button1.Visible = False
      End If
    End Sub

    Protected Sub btnupload_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles btnupload.Click
      Me.txtstdnames.Text = ""
      lblupload.Text = ""
      Me.txtbat.Text = ""
      If FileUploadURL.HasFile Then
        Dim exc As String =
FileUploadURL.FileName.Substring(FileUploadURL.FileName.LastIndexOf("."))
        If exc = ".build" Then
          Dim strname As String
          Dim correction As New DataSet
          Dim Dr As DataRow
          Dim j As Integer = 0
          Dim bat As String = ""
          Dim ds As New DataSet
          Dim i As Integer = 0
          Dim found As Boolean = False
          ds.ReadXml(MapPath("Solution.xml"))
          Dim str As String = ""
          Dim startTime(Me.lblStime.Text.Length) As Char
          Dim endTime(Me.lblEtime.Text.Length) As Char
          startTime = Me.lblStime.Text.ToCharArray
          endTime = Me.lblEtime.Text.ToCharArray
```

```
            For i = 0 To ds.Tables(0).Rows.Count - 1
               If (ds.Tables(0).Rows(i).Item("CourseNumber") = Me.txtcornumber.Text
                And ds.Tables(0).Rows(i).Item("AssignmentNumber") =
                Me.lblnumber.Text And ds.Tables(0).Rows(i).Item("StartTime") =
                Me.lblStime.Text And ds.Tables(0).Rows(i).Item("EndTime") =
                Me.lblEtime.Text) Then
                  strname = ds.Tables(0).Rows(i).Item("URL")
                  Me.txtstdnames.Text += strname.Substring(0, strname.LastIndexOf("."))
                  + Environment.NewLine
               End If
            Next

File.WriteAllText("c:\automatic\solution\projectname.txt",Me.txtstdnames.Text)
            For i = 0 To Me.lblStime.Text.Length - 1
               If (startTime(i) <> "/") Then
                  str += startTime(i).ToString
               End If
            Next
            For i = 0 To Me.lblEtime.Text.Length - 1
               If (endTime(i) <> "/") Then
                  str += endTime(i).ToString
               End If
            Next
            txtupload.Text = Me.lblnumber.Text + Me.txtcornumber.Text + str
            bat = txtupload.Text + ".bat"
            txtupload.Text = txtupload.Text + exc
            Try
               FileUploadURL.SaveAs(Server.MapPath("solution/" & txtupload.Text))
               lblupload.Text = "upload file "
               Me.txtbat.Text = "@echo" + Environment.NewLine
               Me.txtbat.Text += "cd\" + Environment.NewLine
               Me.txtbat.Text += "cls" + Environment.NewLine
               Me.txtbat.Text += "cd c:\automatic\solution" + Environment.NewLine
               Me.txtbat.Text += "nant.exe -buildfile:" + txtupload.Text
                + Environment.NewLine
               Me.txtbat.Text += "@pause" + Environment.NewLine
               File.WriteAllText(Server.MapPath("bat\") + bat, Me.txtbat.Text)
               System.Diagnostics.Process.Start(Server.MapPath("bat\") + bat)
               correction.ReadXml(MapPath("Correction.xml"))
               While (j < correction.Tables(0).Rows.Count And found = False)
                  If (correction.Tables(0).Rows(j).Item("CourseNumber") =
                  Me.txtcornumber.Text And
                  correction.Tables(0).Rows(j).Item("AssignmentNumber") =
                  Me.lblnumber.Text And correction.Tables(0).Rows(j).Item("StartTime") =
```

```
                Me.lblStime.Text And correction.Tables(0).Rows(j).Item("EndTime") =
                Me.lblEtime.Text) Then
                    found = True
                End If
                j = j + 1
            End While
            ' save the data to xml file correction
            If Not found Then
                Dr = correction.Tables(0).NewRow
                Dr("CourseNumber") = Me.txtcornumber.Text
                Dr("CourseName") = Me.lblcourse.Text
                Dr("AssignmentNumber") = Me.lblnumber.Text
                Dr("StartTime") = Me.lblStime.Text
                Dr("EndTime") = Me.lblEtime.Text
                Dr("ScriptURL") = "solution/" & txtupload.Text
                Dr("BatURL") = "bat\" + bat
                Dr("notes") = ""
                correction.Tables(0).Rows.Add(Dr)
                correction.WriteXml(Request.PhysicalApplicationPath+"Correction.xml")
            End If
            Session("yes") = True
            Me.Button1.Visible = True
        Catch ex As Exception
            lblupload.Text = "ERROR: " & ex.Message
        End Try

    Else
        lblupload.Text = "this file is not Ant\Nant"
    End If
  End If
 End Sub
End Class
```

(For more classes see Appendix B: System Coding)

6. System Testing

In this chapter, I will discuss a practical example of electronic marking through the use of the Generic Interface System.

6.1 Introduction

The most important implication of this system is the student ability to access the system and see the courses and all pertinent assignments with respect to each course as well as the ability of the students to submit the assignments before deadline.

As to the instructor, the test of his function is realized by accessing the system and adding the student accounts as well as courses and students' home assignments pertaining to the courses. This is in addition to adding examination files (NANT Script Files) in order to test these courses and apply electronic marking. At the end the instructors would activate these files and see the results. To carry out this process, the instructor needs to employ the NANT technology which would facilitate the automatic marking process.

The NANT technology is text file written in the XML language and consisting of Tags which contain certain tasks. The NANT technology also contains specific properties.

For these reasons, we shall use a practical example whereby the OSS system will be interacted with the NANT. We will also use the VB.net compiler to translate student home assignment, keeping in mind that the OSS system works through the internet.

An example: If there are 30 students registered in a VB.net course, and the instructor requested an assignment on calculating the area of a circle using the VB.net and submitting the assignment before a given deadline. Then the teacher would upload the NANT file to treat the sent assignments, which would translate all the assignments, after which the results would be available as follows: Some of the files or all of them will be functioning perfectly which means that there will be no mistakes made in the application of the VB. net. Some other files would not function due to certain mistakes in applying the programming language.

In this stage, the courses will be corrected and marked automatically by admitting or not admitting these assignments as this will be processed by using the NANT files and storing the results.

As to the programming codes pertaining to the NANT language, we do the following:

First, the data are read from a text file located in a zip folder containing home assignments for all students.

Second, the translation process is carried out for each student separately. The process would go on even if there were mistakes in some student's files. After the process is over, the results are screened on a DOS screen.

Below the results of running Nant script file for testing VB.Net assignments (the outcome after running Nant technology).

1. **C:\>cd c:\automatic\solution**
2. **C:\automatic\Solution>nant.exe -buildfile:oneDV112782010492010.build**
3. **NAnt 0.91 (Build 0.91.3801.0; alpha1; 29/05/2010)**

4. **Copyright (C) 2001-2010 Gerry Shaw**
5. **http://nant.sourceforge.net**

6. **Buildfile: file:///C:/automatic/Solution/oneDV112782010492010.build**
7. **Target framework: Microsoft .NET Framework 2.0**
8. **Target(s) specified: exection**

9. **build**

[echo] --------start compilation visual basic net files -----
[unzip] Unzipping 'C:\automatic\Solution\1970VB.Netone2782010492010.zip' to 'C:\automatic\Solution\1970VB.Netone2782010492010'.
[echo] C:\automatic\Solution\1970VB.Netone2782010492010\1\square.vb
[vbc] Compiling 1 files to 'C:\automatic\Solution\1970VB.Netone2782010492010\1\square.vb.exe'.

[echo] C:\automatic\Solution\1970VB.Netone2782010492010\2\rectangle.vb
[vbc] Compiling 1 files to 'C:\automatic\Solution\1970VB.Netone2782010492010\2\rectangle.vb.ex e'.
[echo] C:\automatic\Solution\1970VB.Netone2782010492010\3\circle.vb
[vbc] Compiling 1 files to 'C:\automatic\Solution\1970VB.Netone2782010492010\3\circle.vb.exe'.

[echo] ----------

10. **[unzip] Unzipping 'C:\automatic\Solution\1980VB.Netone2782010492010.zip' to 'C:\automatic\Soluti on\1980VB.Netone2782010492010'.**
[echo] C:\automatic\Solution\1980VB.Netone2782010492010\hom work\circle.vb
[vbc] Compiling 1 files to 'C:\automatic\Solution\1980VB.Netone2782010492010\hom work\circle.v b.exe'.
[echo] C:\automatic\Solution\1980VB.Netone2782010492010\hom work\rectangle.vb
[vbc] Compiling 1 files to 'C:\automatic\Solution\1980VB.Netone2782010492010\hom work\rectangl e.vb.exe'.
[echo] C:\automatic\Solution\1980VB.Netone2782010492010\hom work\square.vb
[vbc] Compiling 1 files to 'C:\automatic\Solution\1980VB.Netone2782010492010\hom work\square.v b.exe'.
[echo] ----------

11. **[unzip] Unzipping 'C:\automatic\Solution\1972VB.Netone2782010492010.zip' to 'C:\automatic\Soluti**

```
on\1972VB.Netone2782010492010'.
    [echo] C:\automatic\Solution\1972VB.Netone2782010492010\1\square.vb
     [vbc] Compiling 1 files to
'C:\automatic\Solution\1972VB.Netone2782010492010\1\square.vb.exe'.

    [echo] C:\automatic\Solution\1972VB.Netone2782010492010\2\rectangle.vb
     [vbc] Compiling 1 files to
'C:\automatic\Solution\1972VB.Netone2782010492010\2\rectangle.vb.ex
e'.
    [echo] C:\automatic\Solution\1972VB.Netone2782010492010\3\circle.vb
     [vbc] Compiling 1 files to
'C:\automatic\Solution\1972VB.Netone2782010492010\3\circle.vb.exe'.

 12.  [echo] ----------------------------------------------------
   [unzip] Unzipping 'C:\automatic\Solution\123VB.Netone2782010492010.zip' to
'C:\automatic\Solutio
n\123VB.Netone2782010492010'.
    [echo] C:\automatic\Solution\123VB.Netone2782010492010\1\square.vb
     [vbc] Compiling 1 files to
'C:\automatic\Solution\123VB.Netone2782010492010\1\square.vb.exe'.
    [echo] C:\automatic\Solution\123VB.Netone2782010492010\2\rectangle.vb
     [vbc] Compiling 1 files to
'C:\automatic\Solution\123VB.Netone2782010492010\2\rectangle.vb.exe
'.
    [echo] C:\automatic\Solution\123VB.Netone2782010492010\3\circle.vb
     [vbc] Compiling 1 files to
'C:\automatic\Solution\123VB.Netone2782010492010\3\circle.vb.exe'.
    [echo] ----------------------------------------------------
    [echo] --------finish compilation visual basic net files -----

13.  Execution:

    [echo] --------start execution visual basic net files -----
    [echo] C:\automatic\Solution\1970VB.Netone2782010492010\1\square.vb.exe
    [exec] The area of square is =100
    [echo] C:\automatic\Solution\1970VB.Netone2782010492010\2\rectangle.vb.exe
    [exec] The area of rectangle is  =150
    [echo] C:\automatic\Solution\1970VB.Netone2782010492010\3\circle.vb.exe
    [exec] The area of circle is  =314
    [echo] ----------------------------------------------------
    [echo] C:\automatic\Solution\1980VB.Netone2782010492010\hom
work\circle.vb.exe
    [exec] The area of circle is  =314
```

[echo] C:\automatic\Solution\1980VB.Netone2782010492010\hom work\rectangle.vb.exe
[exec] The area of rectangle is =150
[echo] C:\automatic\Solution\1980VB.Netone2782010492010\hom work\square.vb.exe
[exec] The area of square is =100
[echo] ----------
[echo] C:\automatic\Solution\1972VB.Netone2782010492010\1\square.vb.exe
[exec] The area of square is =100
[echo] C:\automatic\Solution\1972VB.Netone2782010492010\2\rectangle.vb.exe
[exec] The area of rectangle is =150
[echo] C:\automatic\Solution\1972VB.Netone2782010492010\3\circle.vb.exe
[exec] The area of circle is =314
[echo] ----------
[echo] C:\automatic\Solution\123VB.Netone2782010492010\1\square.vb.exe
[exec] The area of square is =100
[echo] C:\automatic\Solution\123VB.Netone2782010492010\2\rectangle.vb.exe
[exec] The area of rectangle is =150
[echo] C:\automatic\Solution\123VB.Netone2782010492010\3\circle.vb.exe
[exec] The area of circle is =314
[echo] ----------
[echo] --------finish execution visual basic net files -----

BUILD SUCCEEDED
Total time: 21.1 seconds.
Press any key to continue . . .

Here we want to explain the results of program testing, as to the line no 1 the command line change the directory to the solution folder, and this folder contains the submitted assignments from students.

As to the second line, we run the Nant technology, which executes the oneDV112782010492010 file, this file contains instructions for unzipping the assignment for all students and executes visual basic file for all students by invoking the visual basic compiler of the program and showing the results for all students.

Lines 3, 4 and 5: are concerning with people who develop this technology and the website for it.

Lines 6 and 7: Nant technology selects the file that will be run by it, and also it selects the Target framework: Microsoft .NET Framework 2.0.

As to lines 9, 10, 11 and 12: they unzip the files and invoke the visual basic compiler to make the exe files.

Finally, line 13: it shows the results of translating these files.

6.2 NANT Code

NANT file to treat the sent assignments, which would translate all the assignments and make an exe files.

```
<?xml version="1.0"?>
<project name="compilation visual basic" default="exection">
  <target name="build">
    <echo message="---start compilation visual basic net files ----"/>
    <foreach item="Line" in="projectname.txt" property="projectname">
      <property name="name" value="${projectname}.zip"/>
      <property name="file" value="${file::exists(name)}"/>
        <unzip zipfile="${projectname}.zip" todir="${projectname}"/>
      <foreach item="Folder" in="${projectname}"  property="folder">
      <foreach item="File" in="${folder}"  property="filename">
        <if test="${string::ends-with(filename,'vb')}">
          <echo message="${filename}"/>
          <vbc target="exe" output="${filename}.exe"  failonerror="false"  rebuild="true"  >
            <sources>
              <include name="${filename}"/>
            </sources>
          </vbc>
        </if>
      </foreach>
      </foreach>
      <echo message="-----------------------------------------"/>
    </foreach>
    <echo message="--finish compilation visual basic net files ---"/>
  </target>
  <target name="exection" depends="build" >
    <echo message="----start execution visual basic net files -----"/>
    <foreach item="Line" in="projectname.txt" property="projectname">
      <foreach item="Folder" in="${projectname}"  property="folder">
        <foreach item="File" in="${folder}"  property="filename">
          <if test="${string::ends-with(filename,'exe')}">
            <echo message="${filename}"/>
            <exec program="${filename}" failonerror="false"/>
          </if>
       </foreach>
    </foreach>
    </foreach>
    <echo message="----finish execution visual basic net files -----"/>
  </target>
</project>
```

7. Conclusion, Recommendations and future work

This chapter contains brief description of what I have done and what can be added in the future.

7.1 Introduction

Employing the conventional method of marking college assignments is no longer adequate in this age, given the technological, informational, and electronic advancement in various fields of life. Hence, it is necessary to move from the erstwhile old systems to new electronic systems that are more effective. In this case, the possibility of marking assignments electronically becomes real and available.

The electronic marking of student assignments will save time, efforts and energy as well as expenses. It will also mean more accuracy and reliability for both the students and their instructors.

None the less, there are a number of considerations that ought to be taken into account, when dealing with this task:

- This system had limited resources and time for development. This is why the researcher decided to summarize the conclusion and recommendations.

7.2 Results

After running the system in real world and make some testing, the researcher has reached the following observations:

1. The employment of the system achieves more accuracy in work performance.

   The usage of computer in routine works will lead to accuracy in work; humans can do mistakes whereas previously tested software programs in an accurate way will never do mistakes.

   For example ,if we have a hundred students who submitted their homework, the possibility to have mistakes in the correction process by the teacher is present, whereas the computer will never be affected by the number of the received homework, and eventually will not do mistakes, and this will lead to accuracy in work.
2. It reduces efforts needed in carrying out various administrative tasks.

   The usage of computer completely in the automation process will require less time and effort, and eventually gives more efficacy in work.

   For example to correct 40 homework in the traditional way by the teacher requires at least one hour, at the same time this will require not more few minutes if we use the computer.
3. It helps utilize time effectively.

   By the usage of computer, the correction process will require few minutes, as we notice in page 50, the correction process is done in 22 seconds for four students, whereas the usage of traditional ways in the correction will require more time, where more homework means more time is needed by the teacher to complete the correction.

4. It helps reduce the number of employees in administrative jobs.

   Here in Linnaeus University, usually there are two teachers one for the course and the other for the lab and homework in order to distribute work, the usage of automated correction programs will reduce work and eventually reduces the number of workers in this field, and enables the teacher of the course to complete the work alone.

7.3 Recommendations and future work

The researcher recommends the following:

- Students and instructors should use the system for an experimental period, while having their feedback.
- Implementation the system for other platforms like Java, C# and using data base like SqlServer, MySql, Oracle.
- Implementation of the system in a way that can handle the plagiarism between students.
- Furthering research into this subject, given its great potentials.

References

Literature

Grady Booch, Robert A. Maksimchuk, Michael W. Engel, Bobbi J. Young, Jim Conallen, Kelli A. Houston, Object-Oriented Analysis and Design with Applications, 2007 3rd Edition, Addison-Wesley.

Web links

[1] http://nant.sourceforge.net/ last visited time on 2010-8-31
[2] http://en.wikipedia.org/wiki/Microsoft_Visual_Studio last visited time on 2010-9-11
[3] http://en.wikipedia.org/wiki/Use_case_diagram last visited time on 2010-9-8
[4] http://www.w3schools.com/xml/default.asp last visited time on 2010-8-6
[5] http://en.wikipedia.org/wiki/Foreign_key last visited time on 2010-12-18
[6] http://msdn.microsoft.com/en-US/library/fx6bk1f4%28v=VS.80%29.aspx last visited time on 2010-9-11

Appendices

Appendix A: Data Base implementation

Data base implementation using XML technology for OSS.

- **Teacher information**

```
<?xml version="1.0" standalone="yes"?>
< Teacher>
  <data>
   <Name />
   <ID />
   <Password />
   <Email />
   </data>
 </Teacher>
```

- **Student information**

```
<?xml version="1.0" standalone="yes"?>
<Student>
  <data>
    <Name />
    <ID />
    <Password />
    <Email />
  </data>
</Student>
```

- **Course information**

```
<?xml version="1.0" standalone="yes"?>
<NewDataSet>
<Courses>
   <Name />
   <Number />
   <Credit />
   <Type />
   <Level />
 </Courses>
</NewDataSet>
```

- **Assignment information**

```
<?xml version="1.0" standalone="yes"?>
<Assignments>
  <Assignment>
    <Number />
    <CourseNumber />
    <Description />
```

```
  <Instruction />
  <StartTime />
  <EndTime />
  <Motivation />
  <TypeOfWork />
  <IsStrict />
  <URL />
</Assignment>
</Assignments >
```

- **Registration information**

```
<?xml version="1.0" standalone="yes"?>
<NewDataSet>
  <Registration>
    <StdID />
    <StdName />
    <CourseID />
    <CourseName />
    <Semester />
    <Date />
    <Mark />
  </Registration>
  </NewDataSet>
```

- **solution information**

```
<?xml version="1.0" standalone="yes"?>
<solution>
  <Assignment>
    <StudentNumber />
    <CourseNumber />
    <AssignmentNumber />
    <StartTime />
    <EndTime />
    <SubmitDate />
    <count />
    <accepted />
    <URL />
  </Assignment>
  < /solution >
```

- **Courrection information**

```
<?xml version="1.0" standalone="yes"?>
<Correction>
  <script>
```

```
    <CourseNumber />
    <CourseName />
    <AssignmentNumber />
    <StartTime />
    <EndTime />
    <ScriptURL />
    <BatURL />
    <notes />
  </script>
</Correction>
```

Appendix B: System Coding

These are the codes used to build the online assignment submission.

- **Admin.aspx.vb**

```
Partial Class admin
    Inherits System.Web.UI.Page
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
        System.EventArgs) Handles Me.Load
        Dim x As String
        Dim name As String
        x = Session("x")
        name = Session("Name")
        If Session("x") Is Nothing Then
            Me.Response.Redirect("default.aspx")
        Else
            If Me.IsPostBack = False Then
                Me.lblwelcome.Text += name
            End If
        End If
    End Sub
End Class
```

- **AddStudnet.aspx.vb**

```
Imports System.Web.Security
Imports System.Xml
Imports System.Xml.XPath
Imports System.Data
Partial Class AddStudnet
    Inherits System.Web.UI.Page
    Protected Sub Button1_Click(ByVal sender As Object, ByVal e As
            System.EventArgs) Handles Button1.Click
        Dim ds As New DataSet
        Dim Dr As DataRow
        Dim j As Integer = 0
        Dim found As Boolean = False
        ds.ReadXml(MapPath("Student.xml"))
        While (j < ds.Tables(0).Rows.Count And found = False)
            If (ds.Tables(0).Rows(j).Item(1) = Me.txtId.Text And
            ds.Tables(0).Rows(j).Item(0) <> "") Then
                found = True
            End If
            j = j + 1
        End While
        If found = False Then
            Dr = ds.Tables(0).NewRow
            Dr("Name") = Me.txtName.Text
            Dr("ID") = Me.txtId.Text
            Dr("Password") =
            FormsAuthentication.HashPasswordForStoringInConfigFile(Me.t
            xtPassword.Text, "Md5")
```

```
            Dr("Email") = Me.txtEmail.Text
            ds.Tables(0).Rows.Add(Dr)
            ds.WriteXml(Request.PhysicalApplicationPath +
            "Student.xml")
            Me.Label1.Text = "This student is added in  XMl student
            file."
            Me.txtName.Text = ""
            Me.txtId.Text = ""
            Me.txtPassword.Text = ""
            Me.txtEmail.Text = ""
        Else
            Me.Label1.Text = "This student ID already in XMl file, you
            can not add two student with the same ID."
        End If
    End Sub
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
            System.EventArgs) Handles Me.Load
        Dim x As String
        Dim name As String
        x = Session("x")
        name = Session("Name")
        If Session("x") Is Nothing Then
            Me.Response.Redirect("default.aspx")
        Else
            If Me.IsPostBack = False Then
                Me.lblwelcome.Text += name
            End If
        End If
    End Sub
End Class
```

- **EditStudnet.aspx.vb**

```
Imports System.Web.Security
Imports System.Xml
Imports System.Xml.XPath
Imports System.Data
Partial Class _EditStudnet
    Inherits System.Web.UI.Page
    Protected Sub Button1_Click(ByVal sender As Object, ByVal e As
           System.EventArgs) Handles Button1.Click
        Dim ds As New DataSet
        Dim i As Integer = 0
        Dim found As Boolean = False
        ds.ReadXml(MapPath("Student.xml"))
        While (i < ds.Tables(0).Rows.Count And found = False)
               If (ds.Tables(0).Rows(i).Item(1) = Me.TextBox1.Text And
              ds.Tables(0).Rows(i).Item(0) <> "") Then
                  Me.txtName.Text=ds.Tables(0).Rows(i).Item(0).ToString
                   Me.txtId.Text =ds.Tables(0).Rows(i).Item(1).ToString
               Me.txtPassword.Text=ds.Tables(0).Rows(i).Item(2).ToString
                  Me.txtEmail.Text=ds.Tables(0).Rows(i).Item(3).ToString
                   found = True
                   Session("ds") = ds
```

```
            Session("i") = i
        End If
        i = i + 1
    End While
    If found = False Then
        If Me.TextBox1.Text = "" Then
            Me.Label1.Text = "Please enter the Student ID."
        Else
            Me.Label1.Text = "The student not found."
        End If

        Me.Label2.Visible = False
        Me.Label3.Visible = False
        Me.Label4.Visible = False
        Me.Label5.Visible = False
        Me.txtName.Visible = False
        Me.txtId.Visible = False
        Me.txtPassword.Visible = False
        Me.txtEmail.Visible = False
        Me.btnDelete.Visible = False
        Me.btnUpdate.Visible = False
        Me.Label6.Visible = False
    Else
        Me.Label1.Text = ""
        Me.Label2.Visible = True
        Me.Label3.Visible = True
        Me.Label4.Visible = True
        Me.Label5.Visible = True
        Me.txtName.Visible = True
        Me.txtId.Visible = True
        Me.txtPassword.Visible = True
        Me.txtEmail.Visible = True
        Me.btnDelete.Visible = True
        Me.btnUpdate.Visible = True
        Me.Label6.Visible = True
    End If
    Me.Label6.Text = ""
End Sub
Protected Sub btnUpdate_Click(ByVal sender As Object, ByVal e As
       System.EventArgs) Handles btnUpdate.Click
    Dim ds As New DataSet
    Dim i As Integer
    Dim j As Integer = 0
    Dim count As Integer = 0
    Dim found As Boolean = False
    ds = Session("ds")
    i = Val(Session("i"))
    While (j < ds.Tables(0).Rows.Count)
        If (ds.Tables(0).Rows(j).Item(1) = Me.txtId.Text And
       ds.Tables(0).Rows(j).Item(0) <> "") Then
            found = True
            count = j
        End If
        j = j + 1
```

```
            End While
            If found = False Or (count = i) Then
                ds.Tables(0).Rows(i).Item(0) = Me.txtName.Text
                ds.Tables(0).Rows(i).Item(1) = Me.txtId.Text
                If Me.txtPassword.Text.Length < 30 Then
                    ds.Tables(0).Rows(i).Item(2) =
                      FormsAuthentication.HashPasswordForStoringInConfig
                      File(Me.txtPassword.Text, "Md5")
                Else
                    ds.Tables(0).Rows(i).Item(2) = Me.txtPassword.Text
                End If
                ds.Tables(0).Rows(i).Item(3) = Me.txtEmail.Text
                ds.WriteXml(Request.PhysicalApplicationPath +
              "Student.xml")
                Me.Label6.Text = "Data Update in Student XML file"
            Else
                Me.Label6.Text = "This student ID already in XMl file,
              you can not add two student with the same ID"
            End If
        End Sub
    Protected Sub btnDelete_Click(ByVal sender As Object, ByVal e As
           System.EventArgs) Handles btnDelete.Click
        Dim ds As New DataSet
        Dim i As Integer
        ds = Session("ds")
        i = Val(Session("i"))
        ds.Tables(0).Rows(i).Delete()
        ds.WriteXml(Request.PhysicalApplicationPath + "Student.xml")
        Me.Label6.Text = "Data Delete from Student XML file"
        Me.txtName.Text = ""
        Me.txtId.Text = ""
        Me.txtPassword.Text = ""
        Me.txtEmail.Text = ""
    End Sub
   Protected Sub Page_Load(ByVal sender As Object, ByVal e As
      System.EventArgs) Handles Me.Load
        Dim x As String
        Dim name As String
        x = Session("x")
        name = Session("Name")
        If Session("x") Is Nothing Then
            Me.Response.Redirect("default.aspx")
        Else
            If Me.IsPostBack = False Then
                Me.lblwelcome.Text += name
            End If
        End If
    End Sub
End Class
```

- **AddAssignment.aspx.vb**

```
Imports System.Xml
```

```
Imports System.Xml.XPath
Imports System.Data
Partial Class AddAssignment
    Inherits System.Web.UI.Page
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
           System.EventArgs) Handles Me.Load
        Dim x As String
        Dim name As String
        x = Session("x")
        name = Session("Name")
        If Session("x") Is Nothing Then
            Me.Response.Redirect("default.aspx")
        Else
            If Me.IsPostBack = False Then
                Me.lblwelcome.Text += name
                Dim i As Integer
                Dim ds As New DataSet
                ds.ReadXml(MapPath("Course.xml"))
                For i = 0 To ds.Tables(0).Rows.Count - 1
                    If ds.Tables(0).Rows(i)(0) <> "" Then
                        Me.ddlCourse.Items.Add(New
                          ListItem(ds.Tables(0).Rows(i)(0),
                          ds.Tables(0).Rows(i)(1)))
                    End If
                Next
            End If
        End If
    End Sub
  Protected Sub Button1_Click(ByVal sender As Object, ByVal e As
           System.EventArgs) Handles Button1.Click
        Dim ds As New DataSet
        Dim Dr As DataRow
        Dim j As Integer = 0
        Dim found As Boolean = False
        ds.ReadXml(MapPath("Assignment.xml"))
        While (j < ds.Tables(0).Rows.Count And found = False)
            If (ds.Tables(0).Rows(j).Item("Number") =
                 Me.ddlNumber.SelectedItem.Text And
                 ds.Tables(0).Rows(j).Item("Number") <> "" And
                 ds.Tables(0).Rows(j).Item("CourseNumber") =
                 Me.ddlCourse.SelectedItem.Value And
                 ds.Tables(0).Rows(j).Item("StartTime") =
                 Me.txtStart.Text) Then
                found = True
            End If
            j = j + 1
        End While
        If found = False Then
            Dr = ds.Tables(0).NewRow
            Dr("Number") = Me.ddlNumber.SelectedItem.Value
            Dr("CourseNumber") = Me.ddlCourse.SelectedItem.Value
            Dr("Description") = Me.txtDescription.Text
            Dr("Instruction") = Me.txtinstruction.Text
            Dr("StartTime") = Me.txtStart.Text
```

```
        Dr("EndTime") = Me.txtEnd.Text
        Dr("Motivation") = Me.txtMotivation.Text
        Dr("TypeOfWork") = Me.rblTypeOfWork.SelectedItem.Text

        If (Me.cbIsStrict.Checked = True) Then
            Dr("IsStrict") = True
        Else
            Dr("IsStrict") = False
        End If
        Dr("URL") = Me.txtupload.Text
        ds.Tables(0).Rows.Add(Dr)
        ds.WriteXml(Request.PhysicalApplicationPath +
           "Assignment.xml")
        Me.Label1.Text = "This Assignment is added in  XMl
           Assignment file."
        Me.ddlNumber.SelectedIndex = 0
        Me.ddlCourse.SelectedIndex = 0
        Me.txtDescription.Text = ""
        Me.txtinstruction.Text = ""
        Me.txtStart.Text = ""
        Me.txtEnd.Text = ""
        Me.txtMotivation.Text = ""
        Me.rblTypeOfWork.SelectedIndex = -1
        Me.cbIsStrict.Checked = False
        txtupload.Text = ""
    Else
        Me.Label1.Text = "This Assignment ID already in XMl file,
           you can not add two Assignment with the same ID."
    End If
End Sub
Protected Sub btnupload_Click(ByVal sender As Object, ByVal e As
        System.EventArgs) Handles btnupload.Click
    Dim ds As New DataSet
    Dim j As Integer = 0
    Dim found As Boolean = False
    ds.ReadXml(MapPath("Assignment.xml"))
    While (j < ds.Tables(0).Rows.Count And found = False)
        If (ds.Tables(0).Rows(j).Item("Number") =
          Me.ddlNumber.SelectedItem.Text And
          ds.Tables(0).Rows(j).Item("Number") <> "" And
          ds.Tables(0).Rows(j).Item("CourseNumber") =
          Me.ddlCourse.SelectedItem.Value And
          ds.Tables(0).Rows(j).Item("StartTime") = Me.txtStart.Text)
          Then
            found = True
        End If
        j = j + 1
    End While
    If found = False Then
        Dim start(txtStart.Text.Length) As Char
        Dim str As String
        start = Me.txtStart.Text.ToCharArray
        Dim i As Integer
        For i = 0 To Me.txtStart.Text.Length - 1
```

```
                If (start(i) <> "/") Then
                    str += start(i).ToString
                End If
            Next
            txtupload.Text = Me.ddlNumber.SelectedItem.Value +
                    Me.ddlCourse.SelectedItem.Value + str
            If FileUploadURL.HasFile Then
                txtupload.Text = txtupload.Text +
                  FileUploadURL.FileName.Substring(FileUploadURL.FileNa
                  me.LastIndexOf("."))
                Try
                    FileUploadURL.SaveAs(Server.MapPath("Assignment/" &
                        txtupload.Text))
                    lblupload.Text = "upload file "
                Catch ex As Exception
                    lblupload.Text = "ERROR: " & ex.Message
                End Try
            End If
        Else
            lblupload.Text = "This Assignment ID already in XMl file,
                 you can not add two Assignment with the same ID."
        End If
    End Sub
End Class
```

- **EditAssignment.aspx.vb**

```
Imports System.Xml
Imports System.Xml.XPath
Imports System.Data
Partial Class EditAssignment
    Inherits System.Web.UI.Page

    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
              System.EventArgs) Handles Me.Load
        Dim x As String
        Dim name As String
        x = Session("x")
        name = Session("Name")
        If Session("x") Is Nothing Then
            Me.Response.Redirect("default.aspx")
        Else
            If Me.IsPostBack = False Then
                Me.lblwelcome.Text += name
                Dim i As Integer
                Dim ds As New DataSet
                ds.ReadXml(MapPath("Course.xml"))
                For i = 0 To ds.Tables(0).Rows.Count - 1
                    If ds.Tables(0).Rows(i)(0) <> "" Then
                        Me.ddlCourse.Items.Add(New
                           ListItem(ds.Tables(0).Rows(i)(0),
                           ds.Tables(0).Rows(i)(1)))
                    End If
                Next
```

```
        End If
    End If
End Sub

Protected Sub ddlCourse_SelectedIndexChanged(ByVal sender As
        Object, ByVal e As System.EventArgs) Handles
        ddlCourse.SelectedIndexChanged

    Me.lbldescription.Visible = False
    Me.lblInstruction.Visible = False
    Me.LBLEndTime.Visible = False
    Me.LBLMotivation.Visible = False
    Me.LBLTypeOfWork.Visible = False
    Me.LBLIsStrict.Visible = False
    Me.LBLURL.Visible = False
    Me.txtDescription.Visible = False
    Me.txtinstruction.Visible = False
    Me.txtEnd.Visible = False
    Me.txtMotivation.Visible = False
    Me.rblTypeOfWork.Visible = False
    Me.cbIsStrict.Visible = False
    Me.FileUploadURL.Visible = False
    Me.BTNupdate.Visible = False
    Me.btnupload.Visible = False
    Me.lblDownload.Visible = False
    Me.HyperLink1.Visible = False
    Me.btndelete.Visible = False
    Me.DDLStime.Items.Clear()
    Me.lblupload.Text = ""
    Me.Label1.Text = ""
    Dim ds As New DataSet
    Dim i As Integer = 0
    Dim found As Boolean = False
    ds.ReadXml(MapPath("Assignment.xml"))
    While (i < ds.Tables(0).Rows.Count)
        If (ds.Tables(0).Rows(i).Item("Number") =
           Me.ddlNumber.SelectedItem.Value And
           ds.Tables(0).Rows(i).Item("Number") <> "" And
           ds.Tables(0).Rows(i).Item("CourseNumber") =
           Me.ddlCourse.SelectedItem.Value And
           ds.Tables(0).Rows(i).Item("CourseNumber") <> "") Then
           Me.DDLStime.Items.Add(New
              ListItem(ds.Tables(0).Rows(i).Item("StartTime"),
              ds.Tables(0).Rows(i).Item("StartTime")))
            found = True
        End If
        i = i + 1
    End While
    If found = False Then
        Me.lblmsg.Visible = True
        Me.btnretrieve.Visible = False
    Else
        Me.lblmsg.Visible = False
        Me.btnretrieve.Visible = True
```

```
    End If
End Sub

Protected Sub Button2_Click(ByVal sender As Object, ByVal e As
       System.EventArgs) Handles btnretrieve.Click
    Dim ds As New DataSet
    Dim i As Integer = 0
    Dim found As Boolean = False
    ds.ReadXml(MapPath("Assignment.xml"))
    Me.lbldescription.Visible = True
    Me.lblInstruction.Visible = True
    Me.LBLEndTime.Visible = True
    Me.LBLMotivation.Visible = True
    Me.LBLTypeOfWork.Visible = True
    Me.LBLIsStrict.Visible = True
    Me.LBLURL.Visible = True
    Me.txtDescription.Visible = True
    Me.txtinstruction.Visible = True
    Me.txtEnd.Visible = True
    Me.txtMotivation.Visible = True
    Me.rblTypeOfWork.Visible = True
    Me.cbIsStrict.Visible = True
    Me.FileUploadURL.Visible = True
    Me.BTNupdate.Visible = True
    Me.btnupload.Visible = True
    Me.lblDownload.Visible = True
    Me.HyperLink1.Visible = True
    Me.btndelete.Visible = True
    While (i < ds.Tables(0).Rows.Count And found = False)
        If (ds.Tables(0).Rows(i).Item("Number") =
             Me.ddlNumber.SelectedItem.Value And
             ds.Tables(0).Rows(i).Item("Number") <> "" And
             ds.Tables(0).Rows(i).Item("CourseNumber") =
             Me.ddlCourse.SelectedItem.Value And
             ds.Tables(0).Rows(i).Item("CourseNumber") <> "" And
             ds.Tables(0).Rows(i).Item("StartTime") =
             Me.DDLStime.SelectedItem.Value And
             ds.Tables(0).Rows(i).Item("StartTime") <> "") Then
            Me.txtDescription.Text =
              ds.Tables(0).Rows(i).Item("Description").ToString
            Me.txtinstruction.Text =
              ds.Tables(0).Rows(i).Item("Instruction").ToString
            Me.txtEnd.Text =
              ds.Tables(0).Rows(i).Item("EndTime").ToString
            Me.txtMotivation.Text =
              ds.Tables(0).Rows(i).Item("Motivation").ToString
            If ds.Tables(0).Rows(i).Item("TypeOfWork").ToString =
              "One Student" Then
                Me.rblTypeOfWork.SelectedIndex = 0
            ElseIf ds.Tables(0).Rows(i).Item("TypeOfWork").ToString
              = "Two Student" Then
                Me.rblTypeOfWork.SelectedIndex = 1
            ElseIf ds.Tables(0).Rows(i).Item("TypeOfWork").ToString
              = "Three Student" Then
```

```
                Me.rblTypeOfWork.SelectedIndex = 2
            End If
            If ds.Tables(0).Rows(i).Item("IsStrict").ToString =
              "True" Then
                Me.cbIsStrict.Checked = True
            Else
                Me.cbIsStrict.Checked = False
            End If
            Me.txtupload.Text =
              ds.Tables(0).Rows(i).Item("URL").ToString
            If Me.txtupload.Text <> "" Then
                Dim path As String = Server.MapPath("") +
              "\Assignment\" + Me.txtupload.Text
                Me.HyperLink1.Visible = True
                Me.HyperLink1.NavigateUrl = path
            Else
                Me.HyperLink1.Visible = False
            End If
            found = True
            Session("ds") = ds
            Session("i") = i
        End If
        i = i + 1
    End While
End Sub
Protected Sub BTNupdate_Click(ByVal sender As Object, ByVal e As
      System.EventArgs) Handles BTNupdate.Click
    Dim ds As New DataSet
    Dim i As Integer
    ds = Session("ds")
    i = Val(Session("i"))
    ds.Tables(0).Rows(i).Item("Description") =
           Me.txtDescription.Text
    ds.Tables(0).Rows(i).Item("Instruction") =
           Me.txtinstruction.Text
    ds.Tables(0).Rows(i).Item("EndTime") = Me.txtEnd.Text
    ds.Tables(0).Rows(i).Item("Motivation") = Me.txtMotivation.Text
    ds.Tables(0).Rows(i).Item("TypeOfWork") =
           Me.rblTypeOfWork.SelectedItem.Text
    If (Me.cbIsStrict.Checked = True) Then
        ds.Tables(0).Rows(i).Item("IsStrict") = True
    Else
        ds.Tables(0).Rows(i).Item("IsStrict") = False
    End If
    ds.Tables(0).Rows(i).Item("URL") = Me.txtupload.Text
    If Me.txtupload.Text <> "" Then
        Dim path As String = Server.MapPath("") + "\Assignment\" +
           Me.txtupload.Text
        Me.HyperLink1.Visible = True
        Me.HyperLink1.NavigateUrl = path
    Else
        Me.HyperLink1.Visible = False
    End If
    ds.WriteXml(Request.PhysicalApplicationPath + "Assignment.xml")
```

```
    Me.Label1.Text = "Data Update in  XMl Assignment file."
End Sub
Protected Sub btnupload_Click(ByVal sender As Object, ByVal e As
           System.EventArgs) Handles btnupload.Click
    If Me.txtupload.Text <> "" Then
        Dim path As String = Server.MapPath("") + "\Assignment\" +
           Me.txtupload.Text
        Dim fileExists As Boolean
        fileExists = My.Computer.FileSystem.FileExists(path)
        If fileExists Then
            My.Computer.FileSystem.DeleteFile(path,
           FileIO.UIOption.OnlyErrorDialogs,
           FileIO.RecycleOption.DeletePermanently)
        End If
    End If
    Dim j As Integer = 0
    Dim start(Me.DDLStime.SelectedItem.Text.Length) As Char
    Dim str As String
    start = Me.DDLStime.SelectedItem.Text.ToCharArray
    Dim i As Integer
    For i = 0 To Me.DDLStime.SelectedItem.Text.Length - 1
        If (start(i) <> "/") Then
            str += start(i).ToString
        End If
    Next
    txtupload.Text = Me.ddlNumber.SelectedItem.Value +
           Me.ddlCourse.SelectedItem.Value + str
    If FileUploadURL.HasFile Then
        txtupload.Text = txtupload.Text +
           FileUploadURL.FileName.Substring(FileUploadURL.FileName.
           LastIndexOf("."))
        Try
            FileUploadURL.SaveAs(Server.MapPath("Assignment/" &
           txtupload.Text))
            lblupload.Text = "upload file "
        Catch ex As Exception
            lblupload.Text = "ERROR: " & ex.Message
        End Try
    End If
End Sub
Protected Sub btndelete_Click(ByVal sender As Object, ByVal e As
           System.EventArgs) Handles btndelete.Click
    Dim ds As New DataSet
    Dim i As Integer
    ds = Session("ds")
    i = Val(Session("i"))
    ds.Tables(0).Rows(i).Delete()
    ds.WriteXml(Request.PhysicalApplicationPath + "Assignment.xml")
    Me.Label1.Text = "Data Delete from Assignment XML file"
    Me.lbldescription.Visible = False
    Me.lblInstruction.Visible = False
    Me.LBLEndTime.Visible = False
    Me.LBLMotivation.Visible = False
    Me.LBLTypeOfWork.Visible = False
```

```
        Me.LBLIsStrict.Visible = False
        Me.LBLURL.Visible = False
        Me.txtDescription.Visible = False
        Me.txtinstruction.Visible = False
        Me.txtEnd.Visible = False
        Me.txtMotivation.Visible = False
        Me.rblTypeOfWork.Visible = False
        Me.cbIsStrict.Visible = False
        Me.FileUploadURL.Visible = False
        Me.BTNupdate.Visible = False
        Me.btnupload.Visible = False
        Me.lblDownload.Visible = False
        Me.HyperLink1.Visible = False
        Me.btndelete.Visible = False
    End Sub
End Class
```

- **AddCourse.aspx.vb**

```
Imports System.Xml
Imports System.Xml.XPath
Imports System.Data
Partial Class AddCourse
    Inherits System.Web.UI.Page
    Protected Sub Button1_Click(ByVal sender As Object, ByVal e As
               System.EventArgs) Handles Button1.Click
        Dim ds As New DataSet
        Dim Dr As DataRow
        Dim j As Integer = 0
        Dim found As Boolean = False
        ds.ReadXml(MapPath("Course.xml"))

        While (j < ds.Tables(0).Rows.Count And found = False)
            If (ds.Tables(0).Rows(j).Item(1) = Me.txtNumber.Text And
               ds.Tables(0).Rows(j).Item(0) <> "") Then
                found = True
            End If
            j = j + 1
        End While
        If found = False Then
            Dr = ds.Tables(0).NewRow
            Dr("Name") = Me.txtname.Text
            Dr("Number") = Me.txtNumber.Text
            Dr("Credit") = Me.txtCredit.Text
            Dr("Type") = Me.txtType.Text
            Dr("Level") = Me.txtLevel.Text
            ds.Tables(0).Rows.Add(Dr)
            ds.WriteXml(Request.PhysicalApplicationPath + "Course.xml")
            Me.Label1.Text = "This course is added in  XMl course
               file."
            Me.txtname.Text = ""
            Me.txtNumber.Text = ""
            Me.txtCredit.Text = ""
            Me.txtType.Text = ""
```

```
            Me.txtLevel.Text = ""
        Else
            Me.Label1.Text = "This course ID already in XMl file, you
               can not add two course with the same ID."
        End If
    End Sub
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
               System.EventArgs) Handles Me.Load
        Dim x As String
        Dim name As String
        x = Session("x")
        name = Session("Name")
        If Session("x") Is Nothing Then
            Me.Response.Redirect("default.aspx")
        Else
            If Me.IsPostBack = False Then
                Me.lblwelcome.Text += name
            End If
            End If
    End Sub
End Class
```

- **EditCourse.aspx.vb**

```
Imports System.Xml
Imports System.Xml.XPath
Imports System.Data
Partial Class EditCourse
    Inherits System.Web.UI.Page

    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
           System.EventArgs) Handles Me.Load
        Dim x As String
        Dim name As String
        x = Session("x")
        name = Session("Name")
        If Session("x") Is Nothing Then
            Me.Response.Redirect("default.aspx")
        Else
            If Me.IsPostBack = False Then
                Me.lblwelcome.Text += name
                BindGrid()
            End If
        End If
    End Sub
    Private Sub BindGrid()
        Dim i As Integer = 0
        Dim found As Boolean = False
        Dim ds As New DataSet
        ds.ReadXml(MapPath("Course.xml"))
        While i < ds.Tables(0).Rows.Count And found <> True
            If (ds.Tables(0).Rows(i).Item(0) = "") Then
                found = True
                ds.Tables(0).Rows(i).Delete()
```

```
            End If
            i = i + 1
        End While
        Me.GridView1.DataSource = ds
        Me.GridView1.DataBind()
    End Sub
    Protected Sub GridView1_PageIndexChanging(ByVal sender As Object,
           ByVal e As System.Web.UI.WebControls.GridViewPageEventArgs)
           Handles GridView1.PageIndexChanging
        Me.Label1.Text = ""
        GridView1.PageIndex = e.NewPageIndex
        BindGrid()
    End Sub
    Protected Sub GridView1_RowCancelingEdit(ByVal sender As Object,
      ByVal e As System.Web.UI.WebControls.GridViewCancelEditEventArgs)
      Handles GridView1.RowCancelingEdit
        Me.Label1.Text = ""
        GridView1.EditIndex = -1
        BindGrid()
    End Sub

    Protected Sub GridView1_RowDeleting(ByVal sender As Object, ByVal e
      As System.Web.UI.WebControls.GridViewDeleteEventArgs) Handles
      GridView1.RowDeleting
        Me.Label1.Text = ""
        BindGrid()
        Dim ds As DataSet = GridView1.DataSource
        Dim Dr As DataRow
       Dr = ds.Tables(0).NewRow
ds.Tables(0).Rows(GridView1.Rows(e.RowIndex).DataItemIndex).Delete()
        Dr("Name") = ""
        Dr("Number") = ""
        Dr("Credit") = ""
        Dr("Type") = ""
        Dr("Level") = ""
        ds.Tables(0).Rows.Add(Dr)
        ds.WriteXml(Request.PhysicalApplicationPath + "Course.xml")
        BindGrid()
    End Sub
    Protected Sub GridView1_RowEditing(ByVal sender As Object, ByVal e
      As System.Web.UI.WebControls.GridViewEditEventArgs) Handles
      GridView1.RowEditing
        Me.Label1.Text = ""
        GridView1.EditIndex = e.NewEditIndex
        BindGrid()
    End Sub
    Protected Sub GridView1_RowUpdating(ByVal sender As Object, ByVal e
     As System.Web.UI.WebControls.GridViewUpdateEventArgs) Handles
     GridView1.RowUpdating
        Me.Label1.Text = ""
        Dim i As Integer = GridView1.Rows(e.RowIndex).DataItemIndex
        Dim name As String =
              CType(GridView1.Rows(e.RowIndex).Cells(2).Controls(0),
              TextBox).Text
```

```
        Dim number As String =
              CType(GridView1.Rows(e.RowIndex).Cells(3).Controls(0),
              TextBox).Text
        Dim credit As String =
              CType(GridView1.Rows(e.RowIndex).Cells(4).Controls(0),
              TextBox).Text
        Dim type As String =
              CType(GridView1.Rows(e.RowIndex).Cells(5).Controls(0),
              TextBox).Text
        Dim level As String =
              CType(GridView1.Rows(e.RowIndex).Cells(6).Controls(0),
              TextBox).Text
        Dim j As Integer = 0
        Dim count As Integer = 0
        Dim found As Boolean = False
        GridView1.EditIndex = -1
        BindGrid()
        Dim ds As DataSet = GridView1.DataSource
        While (j < ds.Tables(0).Rows.Count)
            If (ds.Tables(0).Rows(j).Item(1) = number And
              ds.Tables(0).Rows(j).Item(0) <> "") Then
                found = True
                count = j
            End If
            j = j + 1
        End While
        If found = False Or (count = i) Then
            Dim Dr As DataRow
            Dr = ds.Tables(0).NewRow
            ds.Tables(0).Rows(i).Item(0) = name
            ds.Tables(0).Rows(i).Item(1) = number
            ds.Tables(0).Rows(i).Item(2) = credit
            ds.Tables(0).Rows(i).Item(3) = type
            ds.Tables(0).Rows(i).Item(4) = level
            Dr("Name") = ""
            Dr("Number") = ""
            Dr("Credit") = ""
            Dr("Type") = ""
            Dr("Level") = ""
            ds.Tables(0).Rows.Add(Dr)
            ds.WriteXml(Request.PhysicalApplicationPath + "Course.xml")
            BindGrid()
        Else
            Me.Label1.Text = "This course ID already in XMl file, you
              can not add two course with the same ID,Updated canceled."
        End If

    End Sub
End Class
```

- **Registration.aspx.vb**

```
Imports System.Xml
Imports System.Xml.XPath
```

```
Imports System.Data
Partial Class Registration
    Inherits System.Web.UI.Page
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
        System.EventArgs) Handles Me.Load
        Dim x As String
        Dim name As String
        x = Session("x")
        name = Session("Name")
        If Session("x") Is Nothing Then
            Me.Response.Redirect("default.aspx")
        Else
            If Me.IsPostBack = False Then
                Me.lblwelcome.Text += name
                Dim i As Integer
                Dim ds As New DataSet
                Dim j As Integer
                Dim ds2 As New DataSet
                ds.ReadXml(MapPath("Student.xml"))
                ds2.ReadXml(MapPath("Course.xml"))
                For i = 0 To ds.Tables(0).Rows.Count - 1
                    If ds.Tables(0).Rows(i)("Name") <> "" Then
                        Me.DDLStudent.Items.Add(New
                          ListItem(ds.Tables(0).Rows(i)("Name"),
                          ds.Tables(0).Rows(i)("ID")))
                    End If
                Next
                For j = 0 To ds2.Tables(0).Rows.Count - 1
                    If ds2.Tables(0).Rows(j)("Name") <> "" Then
                        Me.DDLCourse.Items.Add(New
                          ListItem(ds2.Tables(0).Rows(j)("Name"),
                          ds2.Tables(0).Rows(j)("Number")))
                    End If
                Next
            End If
        End If
    End Sub
    Protected Sub Calendar1_SelectionChanged(ByVal sender As Object,
        ByVal e As System.EventArgs) Handles Calendar1.SelectionChanged
        Me.txtdate.Text = Me.Calendar1.SelectedDate.Date
    End Sub
    Protected Sub Button1_Click(ByVal sender As Object, ByVal e As
         System.EventArgs) Handles Button1.Click
        Dim ds As New DataSet
        Dim Dr As DataRow
        Dim j As Integer = 0
        Dim found As Boolean = False
        ds.ReadXml(MapPath("Registration.xml"))
        While (j < ds.Tables(0).Rows.Count And found = False)
            If (ds.Tables(0).Rows(j).Item("StdID") =
              Me.DDLStudent.SelectedItem.Value And
              ds.Tables(0).Rows(j).Item("CourseID") =
              Me.DDLCourse.SelectedItem.Value And
              ds.Tables(0).Rows(j).Item("Semester") =
```

```
            Me.DDLSems.SelectedItem.Value And
            ds.Tables(0).Rows(j).Item("Date") = Me.txtdate.Text And
            ds.Tables(0).Rows(j).Item(0) <> "") Then
              found = True
          End If
          j = j + 1
      End While
      If found = False Then
          Dr = ds.Tables(0).NewRow
          Dr("StdID") = Me.DDLStudent.SelectedItem.Value
          Dr("StdName") = Me.DDLStudent.SelectedItem.Text
          Dr("CourseID") = Me.DDLCourse.SelectedItem.Value
          Dr("CourseName") = Me.DDLCourse.SelectedItem.Text
          Dr("Semester") = Me.DDLSems.SelectedItem.Value
          Dr("Date") = Me.txtdate.Text
          Dr("Mark") = "Not Set"
          ds.Tables(0).Rows.Add(Dr)
          ds.WriteXml(Request.PhysicalApplicationPath +
              "Registration.xml")
          Me.Label1.Text = "This student is added in  XMl
              Registration file."
          Me.DDLStudent.SelectedIndex = -1
          Me.DDLCourse.SelectedIndex = -1
          Me.DDLSems.SelectedIndex = -1
      Else
          Me.Label1.Text = "This student ID  and course  already in
              XMl Registration file"
      End If
  End Sub
End Class
```

- **AddScriptFile.aspx.vb**

```
Imports System.Xml
Imports System.Xml.XPath
Imports System.Data
Imports System.IO
Imports System.IO.StreamReader
Imports System.IO.FileStream
Partial Class AddScriptFile
    Inherits System.Web.UI.Page
    Dim assignment As New course
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
           System.EventArgs) Handles Me.Load
        Dim x As String
        Dim name As String
        Dim visible As Boolean
        x = Session("x")
        name = Session("Name")
        visible = Session("yes")
        If Session("x") Is Nothing Or Session("assignment") Is Nothing
           Then
            Me.Response.Redirect("Correction.aspx")
        Else
```

```
        End If
        If Me.IsPostBack = False Then
            Me.lblwelcome.Text += name
            Dim ds As New DataSet
            Dim j As Integer = 0
            Dim found As Boolean = False
            assignment = Session("assignment")
            ds.ReadXml(MapPath("Assignment.xml"))
            Me.lblnumber.Text = assignment.assignment_number
            Me.lblcourse.Text = assignment.course_name
            Me.txtcornumber.Text = assignment.course_number
            While (j < ds.Tables(0).Rows.Count And found = False)
                If (ds.Tables(0).Rows(j).Item("Number") =
                  assignment.assignment_number And
                  ds.Tables(0).Rows(j).Item("CourseNumber") =
                  assignment.course_number And
                  ds.Tables(0).Rows(j).Item("StartTime") =
                  assignment.assignment_Sdate And
                  ds.Tables(0).Rows(j).Item("EndTime") =
                  assignment.assignment_Edate) Then
                    found = True
                    Me.txtDescription.Text =
                  ds.Tables(0).Rows(j).Item("Description")
                    Me.lblinstruction.Text =
                  ds.Tables(0).Rows(j).Item("Instruction")
                    Me.lblStime.Text =
                  ds.Tables(0).Rows(j).Item("StartTime")
                    Me.lblEtime.Text =
                  ds.Tables(0).Rows(j).Item("EndTime")
                    Me.txtMotivation.Text =
                  ds.Tables(0).Rows(j).Item("Motivation")
                    Me.lblTypeofWork.Text =
                  ds.Tables(0).Rows(j).Item("TypeOfWork")
                    Me.lblstrict.Text =
                  ds.Tables(0).Rows(j).Item("IsStrict")
                    If (ds.Tables(0).Rows(j).Item("URL") <> "") Then
                        Me.URL.Visible = True
                        Me.URL.NavigateUrl = "Assignment\" +
                  ds.Tables(0).Rows(j).Item("URL")
                    End If

                End If
                j = j + 1
            End While
        End If
        If visible = True Then
            Me.Button1.Visible = True
        Else
            Me.Button1.Visible = False
        End If
    End Sub
    Protected Sub btnupload_Click(ByVal sender As Object, ByVal e As
           System.EventArgs) Handles btnupload.Click
        Me.txtstdnames.Text = ""
```

```
lblupload.Text = ""
Me.txtbat.Text = ""
If FileUploadURL.HasFile Then
    Dim exc As String =
   FileUploadURL.FileName.Substring(FileUploadURL.FileName.LastI
   ndexOf("."))
    If exc = ".build" Then
        Dim strname As String
        Dim correction As New DataSet
        Dim Dr As DataRow
        Dim j As Integer = 0
        Dim bat As String = ""
        Dim ds As New DataSet
        Dim i As Integer = 0
        Dim found As Boolean = False
        ds.ReadXml(MapPath("Solution.xml"))
        Dim str As String = ""
        Dim startTime(Me.lblStime.Text.Length) As Char
        Dim endTime(Me.lblEtime.Text.Length) As Char
        startTime = Me.lblStime.Text.ToCharArray
        endTime = Me.lblEtime.Text.ToCharArray
        For i = 0 To ds.Tables(0).Rows.Count - 1
            If (ds.Tables(0).Rows(i).Item("CourseNumber") =
             Me.txtcornumber.Text And
             ds.Tables(0).Rows(i).Item("AssignmentNumber") =
             Me.lblnumber.Text And
             ds.Tables(0).Rows(i).Item("StartTime") =
             Me.lblStime.Text And
             ds.Tables(0).Rows(i).Item("EndTime") =
             Me.lblEtime.Text) Then
                strname = ds.Tables(0).Rows(i).Item("URL")
                Me.txtstdnames.Text += strname.Substring(0,
                strname.LastIndexOf(".")) + Environment.NewLine
            End If
        Next
      File.WriteAllText("c:\automatic\solution\projectname.txt",
                  Me.txtstdnames.Text)
        For i = 0 To Me.lblStime.Text.Length - 1
            If (startTime(i) <> "/") Then
                str += startTime(i).ToString
            End If
        Next
        For i = 0 To Me.lblEtime.Text.Length - 1
            If (endTime(i) <> "/") Then
                str += endTime(i).ToString
            End If
        Next
        txtupload.Text = Me.lblnumber.Text +
         Me.txtcornumber.Text + str
        bat = txtupload.Text + ".bat"
        txtupload.Text = txtupload.Text + exc
        Try
            FileUploadURL.SaveAs(Server.MapPath("solution/" &
                txtupload.Text))
```

```
            lblupload.Text = "upload file "
            Me.txtbat.Text = "@echo" + Environment.NewLine
            Me.txtbat.Text += "cd\" + Environment.NewLine
            Me.txtbat.Text += "cls" + Environment.NewLine
            Me.txtbat.Text += "cd c:\automatic\solution" +
                Environment.NewLine
            Me.txtbat.Text += "nant.exe -buildfile:" +
                txtupload.Text + Environment.NewLine
            Me.txtbat.Text += "@pause" + Environment.NewLine
            File.WriteAllText(Server.MapPath("bat\") + bat,
                Me.txtbat.Text)

               System.Diagnostics.Process.Start(Server.MapPath("
               bat\") + bat)
            correction.ReadXml(MapPath("Correction.xml"))
            While (j < correction.Tables(0).Rows.Count And
           found = False)

             If(correction.Tables(0).Rows(j).Item("CourseNumber
             ") = Me.txtcornumber.Text And
             correction.Tables(0).Rows(j).Item("AssignmentNumbe
             r") = Me.lblnumber.Text And
             correction.Tables(0).Rows(j).Item("StartTime") =
             Me.lblStime.Text And
             correction.Tables(0).Rows(j).Item("EndTime") =
             Me.lblEtime.Text) Then
                       found = True
              End If
                j = j + 1
            End While
            ' save the data to xml file correction
            If Not found Then
                Dr = correction.Tables(0).NewRow
                Dr("CourseNumber") = Me.txtcornumber.Text
                Dr("CourseName") = Me.lblcourse.Text
                Dr("AssignmentNumber") = Me.lblnumber.Text
                Dr("StartTime") = Me.lblStime.Text
                Dr("EndTime") = Me.lblEtime.Text
                Dr("ScriptURL") = "solution/" & txtupload.Text
                Dr("BatURL") = "bat\" + bat
                Dr("notes") = ""
                correction.Tables(0).Rows.Add(Dr)
            correction.WriteXml(Request.PhysicalApplicationPath
                  + "Correction.xml")
            End If
            Session("yes") = True
            Me.Button1.Visible = True
        Catch ex As Exception
            lblupload.Text = "ERROR: " & ex.Message
        End Try

    Else
        lblupload.Text = "this file is not Ant\Nant"
    End If
```

```
        End If
    End Sub
End Class
```

o **Batfiles.aspx.vb**

```
Imports System.Xml
Imports System.Xml.XPath
Imports System.Data
Imports System.IO
Imports System.IO.StreamReader
Imports System.IO.FileStream
Partial Class Batfiles
    Inherits System.Web.UI.Page
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
           System.EventArgs) Handles Me.Load
        Dim x As String
        Dim name As String
        x = Session("x")
        name = Session("Name")
        If Session("x") Is Nothing Then
            Me.Response.Redirect("default.aspx")
        Else
            If Me.IsPostBack = False Then
                Me.lblwelcome.Text += name
                Dim ds As New DataSet
                Dim i As Integer = 0
                ds.ReadXml(MapPath("Correction.xml"))
                Me.GridView1.DataSource = ds
                For i = ds.Tables(0).Rows.Count - 1 To 0 Step -1
                    If ds.Tables(0).Rows(i).Item("ScriptURL") = "" Then
                        ds.Tables(0).Rows(i).Delete()
                    End If
                Next
                Me.GridView1.DataBind()
            End If
        End If
    End Sub
    Protected Sub GridView1_SelectedIndexChanged(ByVal sender As
           Object, ByVal e As System.EventArgs) Handles
           GridView1.SelectedIndexChanged
        Dim txtcornumber As String =
               CType(Me.GridView1.SelectedRow.Cells(0).FindControl("Cnu
               mber"), Label).Text
        Dim lblnumber As String =
               CType(Me.GridView1.SelectedRow.Cells(0).FindControl("Snu
               mber"), Label).Text
        Dim lblStime As String =
           CType(Me.GridView1.SelectedRow.Cells(0).FindControl("sdate"),
           Label).Text
        Dim lblEtime As String =
           CType(Me.GridView1.SelectedRow.Cells(0).FindControl("edate"),
           Label).Text
        Dim i As Integer
```

```
        Dim ds As New DataSet
        Dim strname As String
        ds.ReadXml(MapPath("Solution.xml"))
        For i = 0 To ds.Tables(0).Rows.Count - 1
            If (ds.Tables(0).Rows(i).Item("CourseNumber") =
          txtcornumber And
          ds.Tables(0).Rows(i).Item("AssignmentNumber") = lblnumber And
          ds.Tables(0).Rows(i).Item("StartTime") = lblStime And
          ds.Tables(0).Rows(i).Item("EndTime") = lblEtime) Then
                strname = ds.Tables(0).Rows(i).Item("URL")
                Me.txtstdnames.Text += strname.Substring(0,
          strname.LastIndexOf(".")) + Environment.NewLine
            End If
        Next
        File.WriteAllText("c:\automatic\solution\projectname.txt",
          Me.txtstdnames.Text)
    End Sub
End Class
```

o **correction.aspx.vb**

```
Imports System.Xml
Imports System.Xml.XPath
Imports System.Data
Imports System.IO
Partial Class correction
    Inherits System.Web.UI.Page
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
          System.EventArgs) Handles Me.Load
        Dim x As String
        Dim name As String
        x = Session("x")
        name = Session("Name")
        If Session("x") Is Nothing Then
            Me.Response.Redirect("default.aspx")
        Else
            If Me.IsPostBack = False Then
                Me.lblwelcome.Text += name
                Dim ds As New DataSet
                Dim i As Integer = 0
                ds.ReadXml(MapPath("Course.xml"))
                For i = 0 To ds.Tables(0).Rows.Count - 1
                    If ds.Tables(0).Rows(i)("Number") <> "" Then
                        Me.DropDownList1.Items.Add(New
                        ListItem(ds.Tables(0).Rows(i)("Name"),
                        ds.Tables(0).Rows(i)("Number")))
                    End If
                Next
            End If
        End If
    End Sub
    Protected Sub DropDownList1_SelectedIndexChanged(ByVal sender As
          Object, ByVal e As System.EventArgs) Handles
          DropDownList1.SelectedIndexChanged
```

```
    Dim currentdate As Date = System.DateTimeOffset.Now.Date.Date
    Dim assdate As Date
    Dim ds As New DataSet
    Dim i As Integer = 0
    Dim j As Integer = 0
    Dim courses As New DataSet
    Dim courseNumber As String =
      Me.DropDownList1.SelectedItem.Value
    Dim result As Integer = -2
    Dim counter As Integer = 0
    ds.ReadXml(MapPath("Assignment.xml"))
    courses.Tables.Add("Assignment")
    courses.Tables("Assignment").Columns.Add("Start Time")
    courses.Tables("Assignment").Columns.Add("EndTime")
    courses.Tables("Assignment").Columns.Add("Type Of Work")
    courses.Tables("Assignment").Columns.Add("Number")
    courses.Tables("Assignment").Columns.Add("Is Strict")
    Me.Label1.Text = ""
    For i = 0 To ds.Tables(0).Rows.Count - 1
        If courseNumber = ds.Tables(0).Rows(i)("courseNumber") Then
            assdate = ds.Tables(0).Rows(i)("EndTime")
            result = Date.Compare(currentdate, assdate)
            If result = -1 Or result = 0 Or result = 1 Then
                courses.Tables("Assignment").Rows.Add()
                courses.Tables("Assignment").Rows(counter)(0) =
      ds.Tables(0).Rows(i)("StartTime")
                courses.Tables("Assignment").Rows(counter)(1) =
      ds.Tables(0).Rows(i)("EndTime")
                courses.Tables("Assignment").Rows(counter)(2) =
      ds.Tables(0).Rows(i)("TypeOfWork")
                courses.Tables("Assignment").Rows(counter)(3) =
      ds.Tables(0).Rows(i)("Number")
                courses.Tables("Assignment").Rows(counter)(4) =
      ds.Tables(0).Rows(i)("IsStrict")
                counter = counter + 1
            End If
        End If
    Next
    If courses.Tables("Assignment").Rows.Count = 0 And
      Me.DropDownList1.SelectedItem.Value <> "Current courses" Then
        Me.Label1.Text = "The are no assignments for this course"
    ElseIf Me.DropDownList1.SelectedItem.Value = "Current courses"
      Then
        Me.Label1.Text = "Please select course from Drop Down List
      "
    End If
    Me.GridView1.DataSource = courses
    Me.GridView1.DataBind()
End Sub
Protected Sub GridView1_SelectedIndexChanged(ByVal sender As
      Object, ByVal e As System.EventArgs) Handles
      GridView1.SelectedIndexChanged
    Dim assignment As New course
    assignment.course_name = Me.DropDownList1.SelectedItem.Text
```

```
        assignment.course_number = Me.DropDownList1.SelectedItem.Value
        assignment.assignment_Sdate =
          Me.GridView1.SelectedRow.Cells(1).Text
        assignment.assignment_Edate =
          Me.GridView1.SelectedRow.Cells(2).Text
        assignment.assignment_number =
          Me.GridView1.SelectedRow.Cells(4).Text
        assignment.assignment_IsStrict =
          Me.GridView1.SelectedRow.Cells(5).Text
        Session("assignment") = assignment
        Me.Response.Redirect("AddScriptFile.aspx")
    End Sub
End Class
```

- **Report.aspx.vb**

```
Imports System.Xml
Imports System.Xml.XPath
Imports System.Data
Imports System.IO
Partial Class report
    Inherits System.Web.UI.Page
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
          System.EventArgs) Handles Me.Load
        Dim x As String
        Dim name As String
        x = Session("x")
        name = Session("Name")
        If Session("x") Is Nothing Then
            Me.Response.Redirect("default.aspx")
        Else
            If Me.IsPostBack = False Then
                Me.lblwelcome.Text += name
                Dim ds As New DataSet
                Dim i As Integer = 0
                ds.ReadXml(MapPath("Course.xml"))
                For i = 0 To ds.Tables(0).Rows.Count - 1
                    If ds.Tables(0).Rows(i)("Number") <> "" Then
                        Me.DropDownList1.Items.Add(New
                        ListItem(ds.Tables(0).Rows(i)("Name"),
                        ds.Tables(0).Rows(i)("Number")))
                    End If
                Next
            End If
        End If
    End Sub

    Protected Sub DropDownList1_SelectedIndexChanged(ByVal sender As
          Object, ByVal e As System.EventArgs) Handles
          DropDownList1.SelectedIndexChanged
        Dim currentdate As Date = System.DateTimeOffset.Now.Date.Date
        Dim assdate As Date
        Dim ds As New DataSet
        Dim i As Integer = 0
```

```
        Dim j As Integer = 0
        Dim courses As New DataSet
        Dim courseNumber As String =
          Me.DropDownList1.SelectedItem.Value
        Dim result As Integer = -2
        Dim counter As Integer = 0
        ds.ReadXml(MapPath("Assignment.xml"))
        courses.Tables.Add("Assignment")
        courses.Tables("Assignment").Columns.Add("Start Time")
        courses.Tables("Assignment").Columns.Add("EndTime")
        courses.Tables("Assignment").Columns.Add("Type Of Work")
        courses.Tables("Assignment").Columns.Add("Number")
        courses.Tables("Assignment").Columns.Add("Is Strict")
        Me.Label1.Text = ""
        For i = 0 To ds.Tables(0).Rows.Count - 1
            If courseNumber = ds.Tables(0).Rows(i)("courseNumber") Then
                assdate = ds.Tables(0).Rows(i)("EndTime")
                result = Date.Compare(currentdate, assdate)
                If result = -1 Or result = 0 Or result = 1 Then
                    courses.Tables("Assignment").Rows.Add()
                    courses.Tables("Assignment").Rows(counter)(0) =
          ds.Tables(0).Rows(i)("StartTime")
                    courses.Tables("Assignment").Rows(counter)(1) =
          ds.Tables(0).Rows(i)("EndTime")
                    courses.Tables("Assignment").Rows(counter)(2) =
          ds.Tables(0).Rows(i)("TypeOfWork")
                    courses.Tables("Assignment").Rows(counter)(3) =
          ds.Tables(0).Rows(i)("Number")
                    courses.Tables("Assignment").Rows(counter)(4) =
          ds.Tables(0).Rows(i)("IsStrict")
                    counter = counter + 1
                End If
            End If
        Next
        If courses.Tables("Assignment").Rows.Count = 0 And
          Me.DropDownList1.SelectedItem.Value <> "Current courses" Then
            Me.Label1.Text = "The are no assignments for this course"
        ElseIf Me.DropDownList1.SelectedItem.Value = "Current courses"
          Then
            Me.Label1.Text = "Please select course from Drop Down List
          "
        End If
        Me.GridView1.DataSource = courses
        Me.GridView1.DataBind()
    End Sub

Protected Sub GridView1_SelectedIndexChanged(ByVal sender As Object,
     ByVal e As System.EventArgs) Handles GridView1.SelectedIndexChanged
        Dim courses As New DataSet
        Dim ds As New DataSet
        Dim i As Integer
        Dim counter As Integer
        ds.ReadXml(MapPath("solution.xml"))
        courses.Tables.Add("Report")
```

```
        courses.Tables("Report").Columns.Add("StudentNumber")
        courses.Tables("Report").Columns.Add("SubmitDate")
        courses.Tables("Report").Columns.Add("count")
        courses.Tables("Report").Columns.Add("accepted")
        courses.Tables("Report").Columns.Add("URL")
        For i = 0 To ds.Tables(0).Rows.Count - 1
            If Me.DropDownList1.SelectedItem.Value =
              ds.Tables(0).Rows(i)("courseNumber") And
              ds.Tables(0).Rows(i)("AssignmentNumber") =
              Me.GridView1.SelectedRow.Cells(4).Text And
              ds.Tables(0).Rows(i)("StartTime") =
              Me.GridView1.SelectedRow.Cells(1).Text And
              ds.Tables(0).Rows(i)("EndTime") =
              Me.GridView1.SelectedRow.Cells(2).Text Then
                 courses.Tables("Report").Rows.Add()
                 courses.Tables("Report").Rows(counter)(0) =
              ds.Tables(0).Rows(i)("StudentNumber")
                 courses.Tables("Report").Rows(counter)(1) =
              ds.Tables(0).Rows(i)("SubmitDate")
                 courses.Tables("Report").Rows(counter)(2) =
              ds.Tables(0).Rows(i)("count")
                 courses.Tables("Report").Rows(counter)(3) =
              ds.Tables(0).Rows(i)("accepted")
                 counter = counter + 1
            End If
        Next
        Me.GridView2.DataSource = courses
        Me.GridView2.DataBind()
    End Sub
End Class
```

- **Student.aspx.vb**

```
Partial Class student
    Inherits System.Web.UI.Page
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
        System.EventArgs) Handles Me.Load
        Dim x As String
        Dim name As String
        x = Session("x")
        name = Session("Name")
        If Session("x") Is Nothing Then
            Me.Response.Redirect("default.aspx")
        Else
            If Me.IsPostBack = False Then
                Me.lblwelcome.Text += name
            End If
        End If
    End Sub
End Class
```

- **Profile.aspx.vb**

```
Imports System.Web.Security
Imports System.Xml
```

```
Imports System.Xml.XPath
Imports System.Data
Partial Class profile
    Inherits System.Web.UI.Page
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
         System.EventArgs) Handles Me.Load
        Dim x As String
        Dim name As String
        x = Session("x")
        name = Session("Name")
        If Session("x") Is Nothing Then
            Me.Response.Redirect("default.aspx")
        Else
            If Me.IsPostBack = False Then
                Me.lblwelcome.Text += name
                Me.txtName.Text = name
                Me.txtId.Text = Session("x")
                Me.txtPassword.Text = Session("Password")
                Me.txtEmail.Text = Session("Email")
            End If
        End If
    End Sub
    Protected Sub btnUpdate_Click(ByVal sender As Object, ByVal e As
         System.EventArgs) Handles btnUpdate.Click
        Dim ds As New DataSet
        Dim i As Integer
        Dim j As Integer = 0
        Dim count As Integer = 0
        Dim found As Boolean = False
        ds.ReadXml(MapPath("Student.xml"))
        i = Val(Session("counter"))
        While (j < ds.Tables(0).Rows.Count)
            If (ds.Tables(0).Rows(j).Item("ID") = Me.txtId.Text And
              ds.Tables(0).Rows(j).Item(0) <> "") Then
                found = True
                count = j
            End If
            j = j + 1
        End While
        If found = False Or (count = i) Then
            ds.Tables(0).Rows(i).Item(0) = Me.txtName.Text
            ds.Tables(0).Rows(i).Item(1) = Me.txtId.Text
            If Me.txtPassword.Text.Length < 30 Then
                ds.Tables(0).Rows(i).Item(2) =
                 FormsAuthentication.HashPasswordForStoringInConfigFile(
                 Me.txtPassword.Text, "Md5")
            Else
                ds.Tables(0).Rows(i).Item(2) = Me.txtPassword.Text
            End If
            ds.Tables(0).Rows(i).Item(3) = Me.txtEmail.Text
            ds.WriteXml(Request.PhysicalApplicationPath +
               "Student.xml")
            Me.Label6.Text = "Data Update in Student XML file"
            Session("Password") = ds.Tables(0).Rows(i).Item("Password")
```

```
            Session("Email") = ds.Tables(0).Rows(i).Item("Email")
            Session("x") = ds.Tables(0).Rows(i).Item("ID")
            Session("name") = ds.Tables(0).Rows(i).Item("Name")
        Else
            Me.Label6.Text = "This student ID already in XMl file, you
            can not add two student with the same ID"
        End If
    End Sub
End Class
```

- **Viewcourses.aspx.vb**

```
Imports System.Xml
Imports System.Xml.XPath
Imports System.Data
Partial Class Viewcourses
    Inherits System.Web.UI.Page
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
       System.EventArgs) Handles Me.Load
        Dim x As String
        Dim name As String
        x = Session("x")
        name = Session("Name")
        Dim id As String
        id = x
        If Session("x") Is Nothing Then
            Me.Response.Redirect("default.aspx")
        Else
            If Me.IsPostBack = False Then
                Me.lblwelcome.Text += name
                Dim ds As New DataSet
                Dim i As Integer = 0
                Dim j As Integer = 0
                Dim found As Boolean = False
                ds.ReadXml(MapPath("Registration.xml"))
                For i = 0 To ds.Tables(0).Rows.Count - 1
                    found = False
                    j = 0
                    If ds.Tables(0).Rows(i)("StdID") = id And
                   ds.Tables(0).Rows(i)("StdID") <> "" Then
                        While j < Me.DropDownList1.Items.Count And
                          found <> True
                            If ds.Tables(0).Rows(i)("CourseID") =
                             Me.DropDownList1.Items(j).Value Then
                                found = True
                            End If
                            j = j + 1
                        End While
                        If found <> True Then
                            Me.DropDownList1.Items.Add(New
                           ListItem(ds.Tables(0).Rows(i)("CourseName"),
                           ds.Tables(0).Rows(i)("CourseID")))
                        End If
```

```
                End If
            Next
        End If
    End If
End Sub
Protected Sub DropDownList1_SelectedIndexChanged(ByVal sender As
        Object, ByVal e As System.EventArgs) Handles
        DropDownList1.SelectedIndexChanged
    Dim currentdate As Date = System.DateTimeOffset.Now.Date.Date
    Dim assdate As Date
    Dim ds As New DataSet
    Dim i As Integer = 0
    Dim j As Integer = 0
    Dim courses As New DataSet
    Dim courseNumber As String =Me.DropDownList1.SelectedItem.Value
    Dim result As Integer = -2
    Dim counter As Integer = 0
    ds.ReadXml(MapPath("Assignment.xml"))
    courses.Tables.Add("Assignment")
    courses.Tables("Assignment").Columns.Add("Start Time")
    courses.Tables("Assignment").Columns.Add("EndTime")
    courses.Tables("Assignment").Columns.Add("Type Of Work")
    courses.Tables("Assignment").Columns.Add("Number")
    courses.Tables("Assignment").Columns.Add("Is Strict")
    Me.Label1.Text = ""
    For i = 0 To ds.Tables(0).Rows.Count - 1
        If courseNumber = ds.Tables(0).Rows(i)("courseNumber") Then
            assdate = ds.Tables(0).Rows(i)("EndTime")
            result = Date.Compare(currentdate, assdate)
            If result = -1 Or result = 0 Or result = 1 Then
                courses.Tables("Assignment").Rows.Add()
                courses.Tables("Assignment").Rows(counter)(0) =
                    ds.Tables(0).Rows(i)("StartTime")
                courses.Tables("Assignment").Rows(counter)(1) =
                    ds.Tables(0).Rows(i)("EndTime")
                courses.Tables("Assignment").Rows(counter)(2) =
                    ds.Tables(0).Rows(i)("TypeOfWork")
                courses.Tables("Assignment").Rows(counter)(3) =
                    ds.Tables(0).Rows(i)("Number")
                courses.Tables("Assignment").Rows(counter)(4) =
                    ds.Tables(0).Rows(i)("IsStrict")
                counter = counter + 1
            End If
        End If
    Next
    If courses.Tables("Assignment").Rows.Count = 0 And
       Me.DropDownList1.SelectedItem.Value <> "Current courses" Then
        Me.Label1.Text = "The are no assignments for this course"
    ElseIf Me.DropDownList1.SelectedItem.Value = "Current courses"
Then
        Me.Label1.Text ="Please select course from Drop Down List "
    End If
    Me.GridView1.DataSource = courses
    Me.GridView1.DataBind()
```

```
    End Sub
    Protected Sub GridView1_SelectedIndexChanged(ByVal sender As
        Object, ByVal e As System.EventArgs) Handles
        GridView1.SelectedIndexChanged
         Dim id As String
         id = Session("ID")
         Dim assignment As New course
         assignment.student_number = id
         assignment.course_name = Me.DropDownList1.SelectedItem.Text
         assignment.course_number = Me.DropDownList1.SelectedItem.Value
         assignment.assignment_Sdate =
          Me.GridView1.SelectedRow.Cells(1).Text
          assignment.assignment_Edate =
          Me.GridView1.SelectedRow.Cells(2).Text
          assignment.assignment_number=Me.GridView1.SelectedRow.Cells(4)
          .Text
          assignment.assignment_IsStrict=Me.GridView1.SelectedRow.Cells(
          5).Text
         Session("assignment") = assignment
         Me.Response.Redirect("ViewAssignment.aspx")
    End Sub
End Class
```

- **ViewAssignment.aspx.vb**

```
Imports System.Xml
Imports System.Xml.XPath
Imports System.Data
Partial Class ViewAssignment
    Inherits System.Web.UI.Page
    Dim assignment As New course
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
                System.EventArgs) Handles Me.Load
         Dim x As String
         Dim name As String
         x = Session("x")
         name = Session("Name")
         If Session("x") Is Nothing Or Session("assignment") Is Nothing
                Then
             Me.Response.Redirect("default.aspx")
         Else
             If Me.IsPostBack = False Then
                 Me.lblwelcome.Text += name
                 Dim ds As New DataSet
                 Dim j As Integer = 0
                 Dim found As Boolean = False
                 assignment = Session("assignment")
                 ds.ReadXml(MapPath("Assignment.xml"))
                 Me.lblnumber.Text = assignment.assignment_number
                 Me.lblcourse.Text = assignment.course_name
                 Me.txtstdnumber.Text = x
                 Me.txtcornumber.Text = assignment.course_number
                 Me.txtstrict.Text = assignment.assignment_IsStrict
                 While (j < ds.Tables(0).Rows.Count And found = False)
```

```
                If (ds.Tables(0).Rows(j).Item("Number") =
            assignment.assignment_number And
            ds.Tables(0).Rows(j).Item("CourseNumber") =
            assignment.course_number And
            ds.Tables(0).Rows(j).Item("StartTime") =
            assignment.assignment_Sdate And
            ds.Tables(0).Rows(j).Item("EndTime") =
            assignment.assignment_Edate) Then
                     found = True
                        Me.txtDescription.Text =
                          ds.Tables(0).Rows(j).Item("Description")
                        Me.lblinstruction.Text =
                          ds.Tables(0).Rows(j).Item("Instruction")
                        Me.lblStime.Text =
                          ds.Tables(0).Rows(j).Item("StartTime")
                        Me.lblEtime.Text =
                          ds.Tables(0).Rows(j).Item("EndTime")
                        Me.txtMotivation.Text =
                          ds.Tables(0).Rows(j).Item("Motivation")
                        Me.lblTypeofWork.Text =
                          ds.Tables(0).Rows(j).Item("TypeOfWork")
                        Me.lblstrict.Text =
                          ds.Tables(0).Rows(j).Item("IsStrict")
                     If (ds.Tables(0).Rows(j).Item("URL") <> "")
            Then
                         Me.URL.Visible = True
                         Me.URL.NavigateUrl = "Assignment\" +
                          ds.Tables(0).Rows(j).Item("URL")
                     End If
                End If
                j = j + 1
             End While
        End If
        Dim t1 As Date = Me.lblEtime.Text
        Dim t2 As Date = System.DateTimeOffset.Now.Date.Date
        If Date.Compare(t1, t2) >= 0 Or Me.lblstrict.Text = False
           Then
            Me.FileUploadURL.Visible = True
            Me.btnupload.Visible = True
        Else
            Me.FileUploadURL.Visible = False
            Me.btnupload.Visible = False
            lblupload.Text = "you can not send the assignment after
           the deadline"
        End If
    End If
End Sub
Protected Sub btnupload_Click(ByVal sender As Object, ByVal e As
           System.EventArgs) Handles btnupload.Click
    Dim str As String
    Dim i As Integer
    Dim startTime(Me.lblStime.Text.Length) As Char
    Dim endTime(Me.lblEtime.Text.Length) As Char
    startTime = Me.lblStime.Text.ToCharArray
```

```
endTime = Me.lblEtime.Text.ToCharArray
str = Me.txtstdnumber.Text + Me.lblcourse.Text +
       Me.lblnumber.Text
For i = 0 To Me.lblStime.Text.Length - 1
    If (startTime(i) <> "/") Then
        str += startTime(i).ToString
    End If
Next
For i = 0 To Me.lblEtime.Text.Length - 1
    If (endTime(i) <> "/") Then
        str += endTime(i).ToString
    End If
Next
If FileUploadURL.HasFile Then
    str +=
       FileUploadURL.FileName.Substring(FileUploadURL.FileName.
       LastIndexOf("."))
    Try
        FileUploadURL.SaveAs(Server.MapPath("Solution/" & str))
        Dim ds As New DataSet
        Dim j As Integer = 0
        Dim found As Boolean = False
        Dim count As Integer = 0
        ds.ReadXml(MapPath("solution.xml"))
        While (j < ds.Tables(0).Rows.Count And found = False)
            If (ds.Tables(0).Rows(j).Item("StudentNumber") =
       Me.txtstdnumber.Text And
       ds.Tables(0).Rows(j).Item("CourseNumber") =
       Me.txtcornumber.Text And
       ds.Tables(0).Rows(j).Item("AssignmentNumber") =
       Me.lblnumber.Text And
       ds.Tables(0).Rows(j).Item("EndTime") = Me.lblEtime.Text
       And ds.Tables(0).Rows(j).Item("StartTime") =
       Me.lblStime.Text) Then
                found = True
            Else
                count = count + 1
            End If
            j = j + 1
        End While

        If found = False Then
            Dim Dr As DataRow
            Dr = ds.Tables(0).NewRow
            Dr("StudentNumber") = Me.txtstdnumber.Text
            Dr("CourseNumber") = Me.txtcornumber.Text
            Dr("AssignmentNumber") = Me.lblnumber.Text
            Dr("StartTime") = Me.lblStime.Text
            Dr("EndTime") = Me.lblEtime.Text
            Dr("SubmitDate") =
       System.DateTimeOffset.Now.Date.Date
            Dr("count") = 1
            Dr("accepted") = "not yet"
            Dr("URL") = str
```

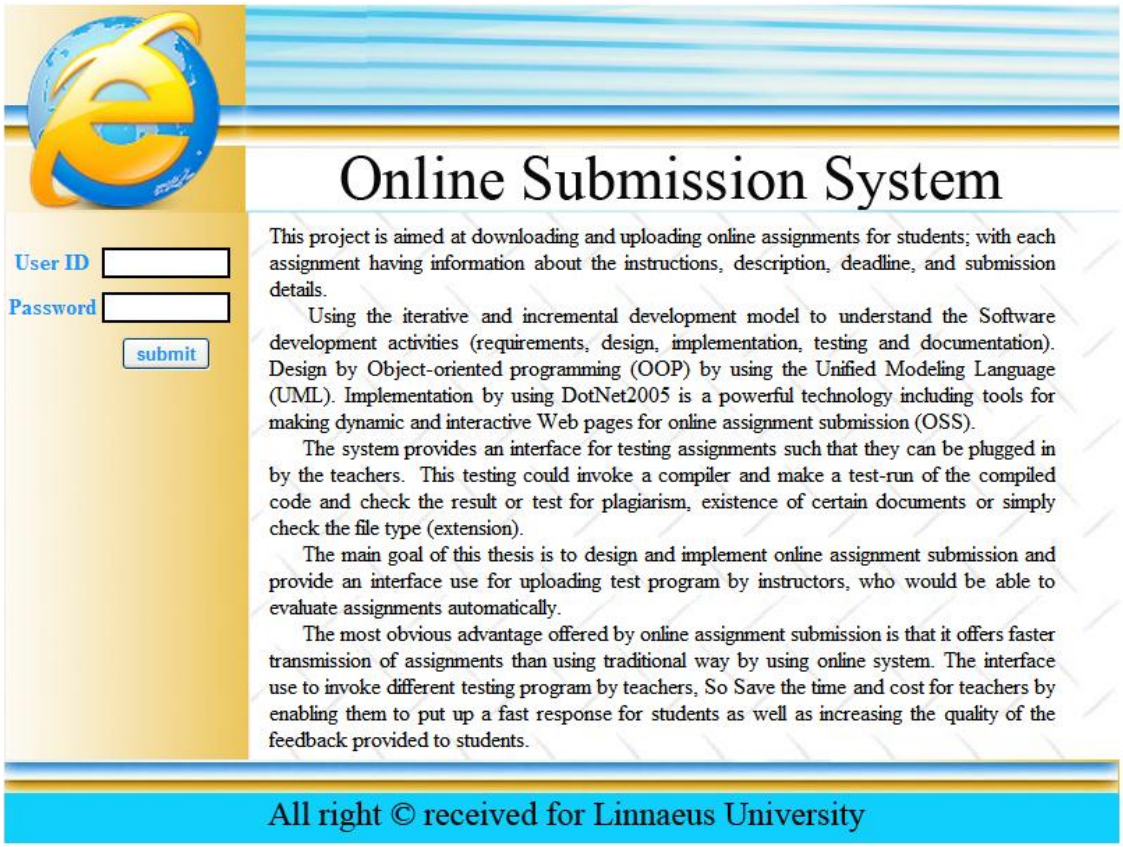
```
                ds.Tables(0).Rows.Add(Dr)
                ds.WriteXml(Request.PhysicalApplicationPath +
            "solution.xml")
                lblupload.Text = "upload the assignment successful
            "
             Else
                ds.Tables(0).Rows(count).Item("StudentNumber") =
            Me.txtstdnumber.Text
                ds.Tables(0).Rows(count).Item("CourseNumber") =
            Me.txtcornumber.Text
                ds.Tables(0).Rows(count).Item("AssignmentNumber") =
            Me.lblnumber.Text
                ds.Tables(0).Rows(count).Item("StartTime") =
            Me.lblStime.Text
                ds.Tables(0).Rows(count).Item("EndTime") =
            Me.lblEtime.Text
                ds.Tables(0).Rows(count).Item("SubmitDate") =
            System.DateTimeOffset.Now.Date.Date.Date
                ds.Tables(0).Rows(count).Item("count") =
            Val(ds.Tables(0).Rows(count).Item("count")) + 1
                ds.Tables(0).Rows(count).Item("accepted") = "not
            yet"
                ds.Tables(0).Rows(count).Item("URL") = str
                ds.WriteXml(Request.PhysicalApplicationPath +
            "solution.xml")
                lblupload.Text = "update the assignment successful
            "
             End If
        Catch ex As Exception
            lblupload.Text = "ERROR: " & ex.Message
        End Try
    End If
End Sub
End Class
```

Appendix C: User guide for OSS

In online assignment submission number of forms considering user friendly and interface consistency is designed, by using the standard web methods and tools in the ASP.Net such as master page and web user control and cascading style sheet.

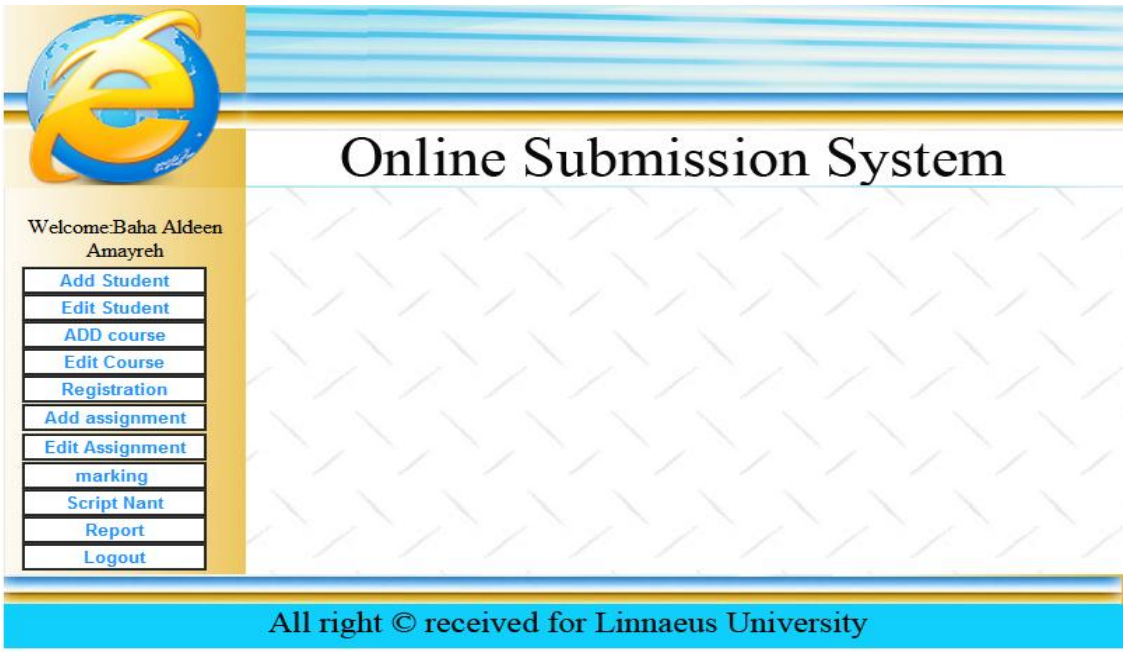
- **Login page**

This page is enabling the student, teacher and administration to access his account, see his data and use menu by using valid username and password.



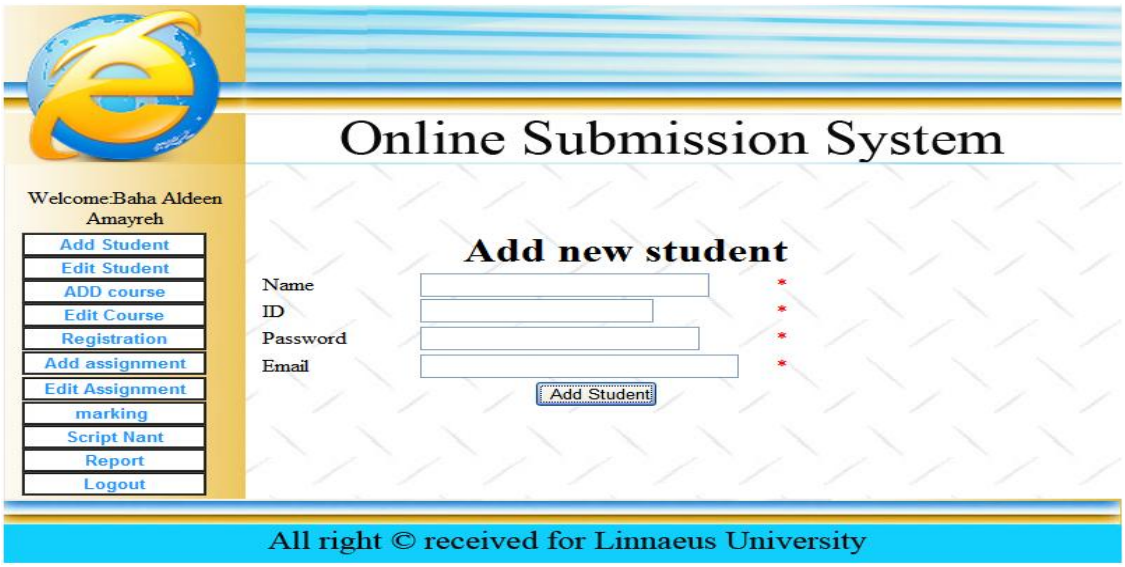
- **Teacher home page**

The main menu of teacher**.**



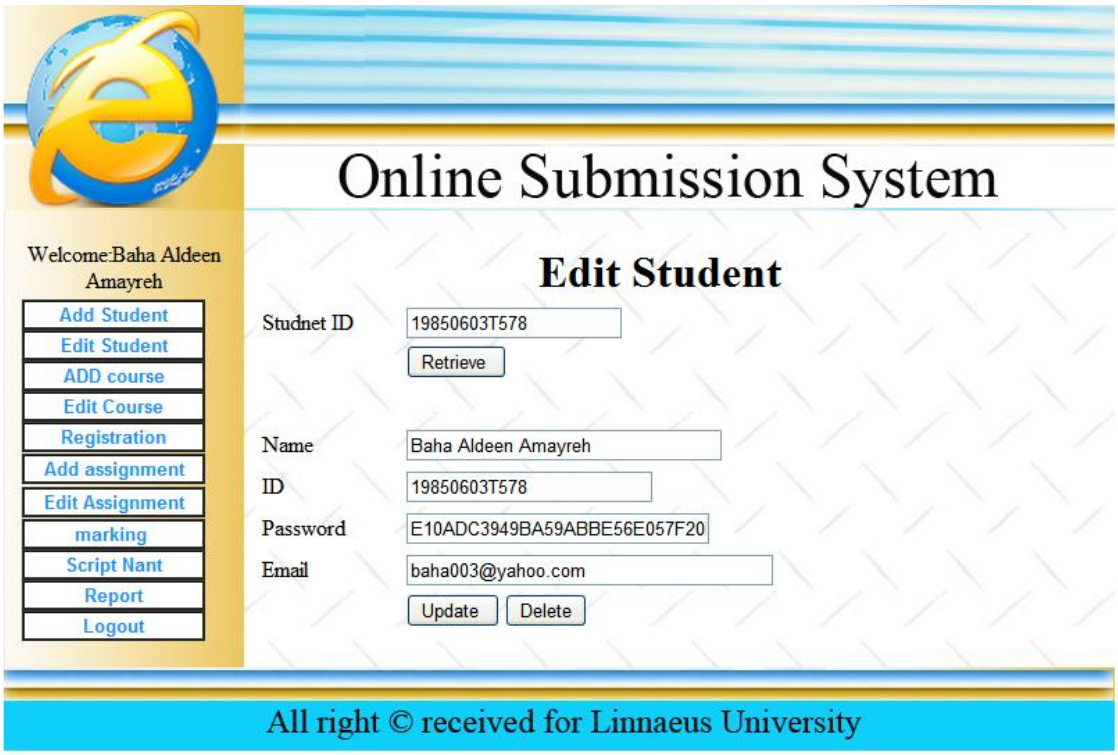
- **Add new student page**

This form enables the instructor to add new students to the database, keeping in mind that the student ID should not be repeated more than once.



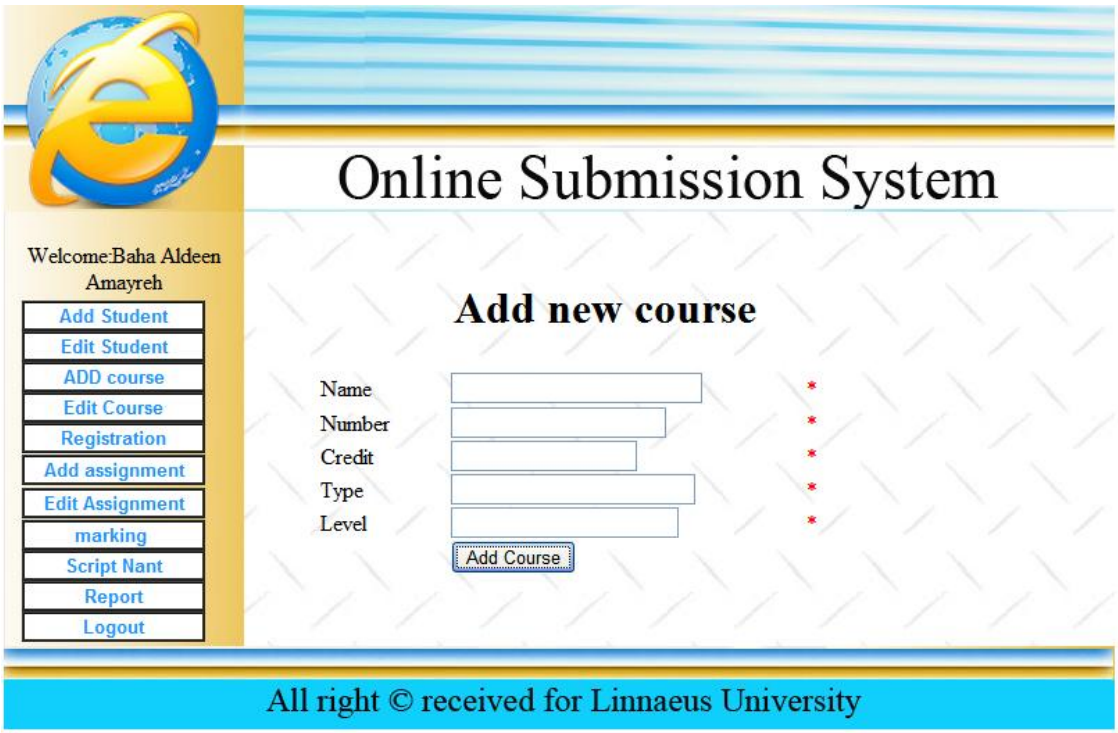
- **Edit student page**

The form also enables the instructor to revise student data and or delete students from the database.



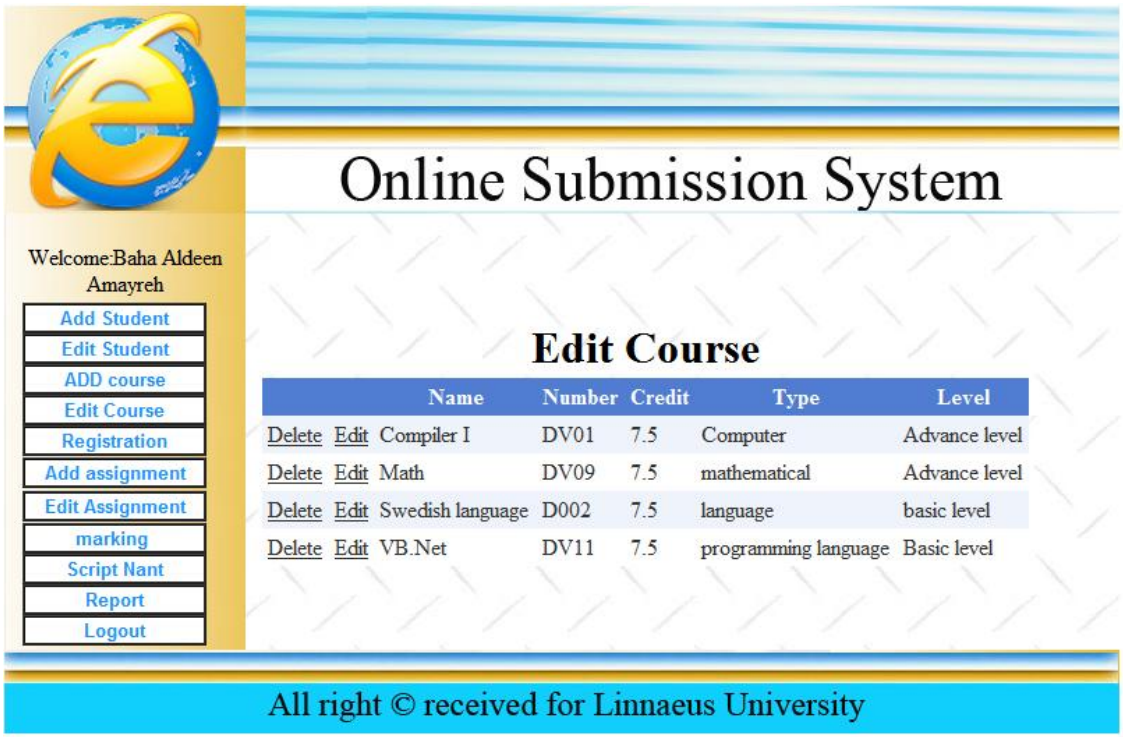
- **Add new course page**

The form enables the instructor to add courses to the database, keeping in mind that a course code or number should not be repeated more than once, since this code or number is a primary key.



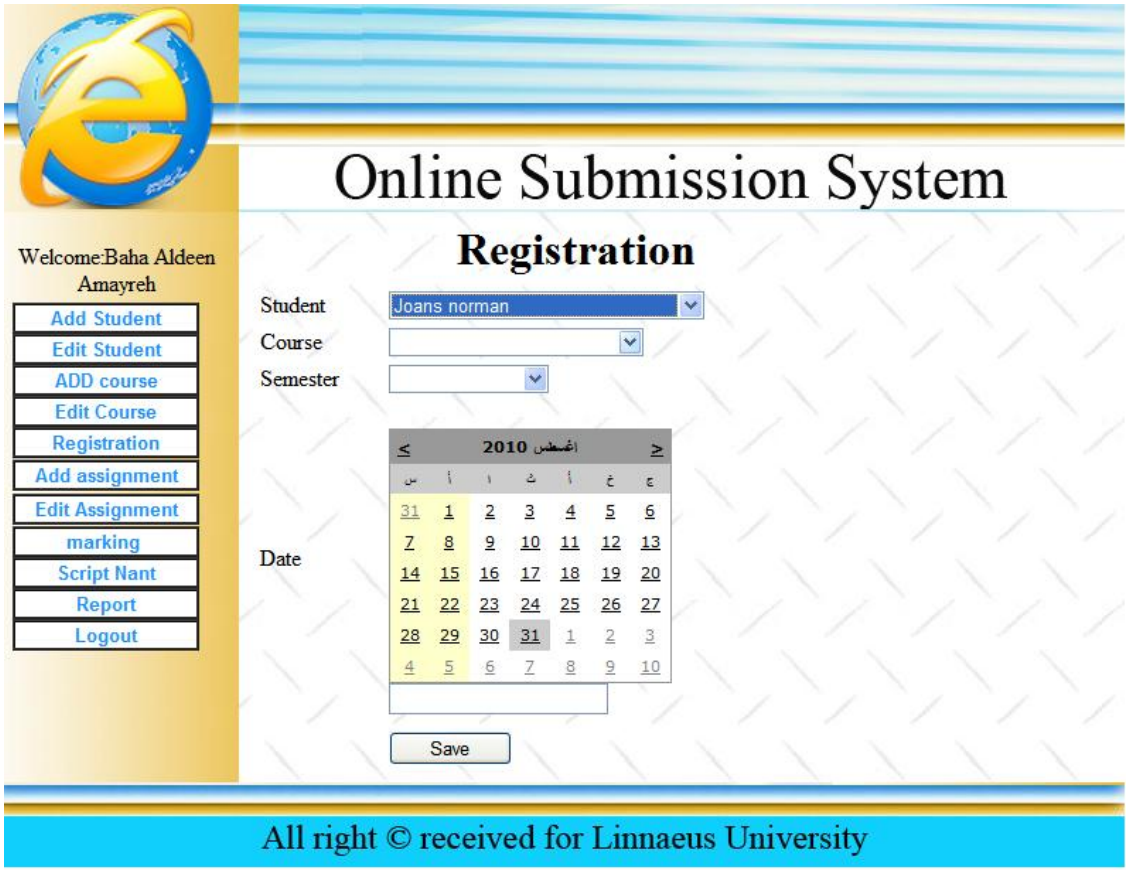
- **Edit course page**

The form also enables the instructor to revise the database of the same course or delete it entirely.



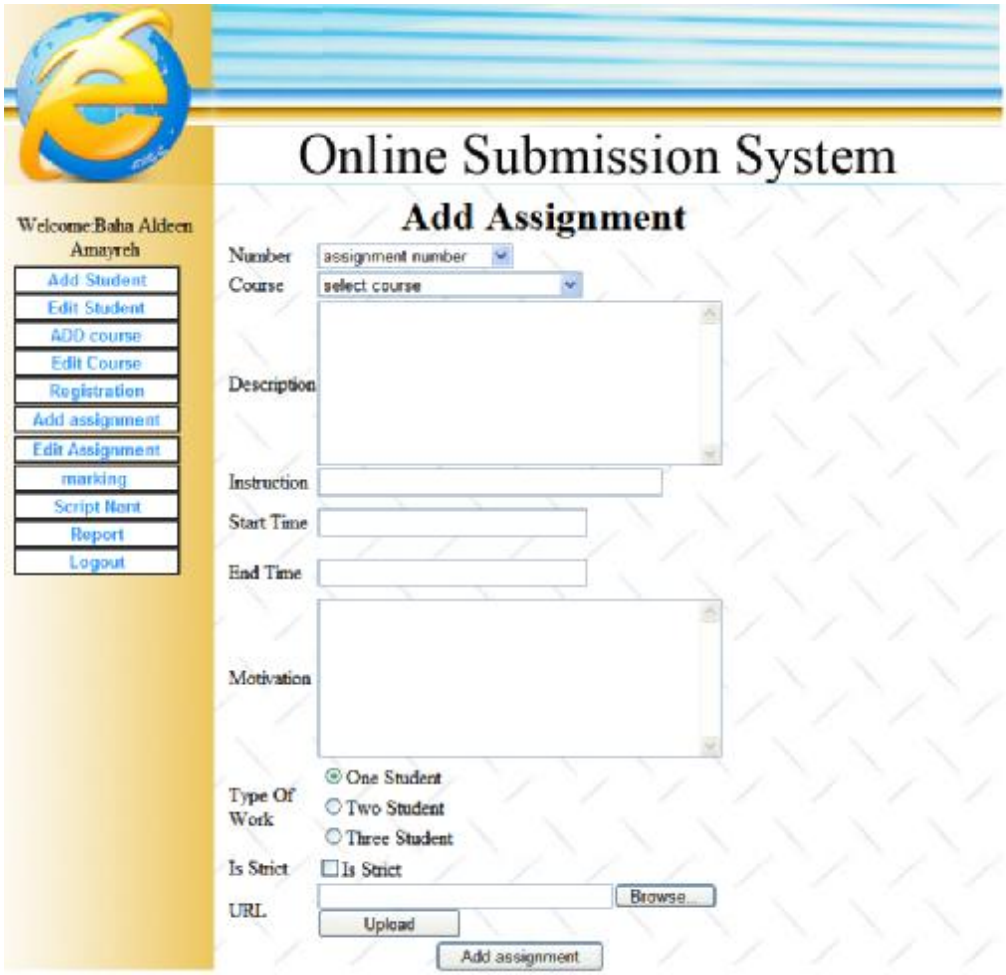
- **Registration page**

The form enables the instructor to register new students to available courses. In this case, both the student number, the number of the study course as well as the date of registration is considered primary keys.



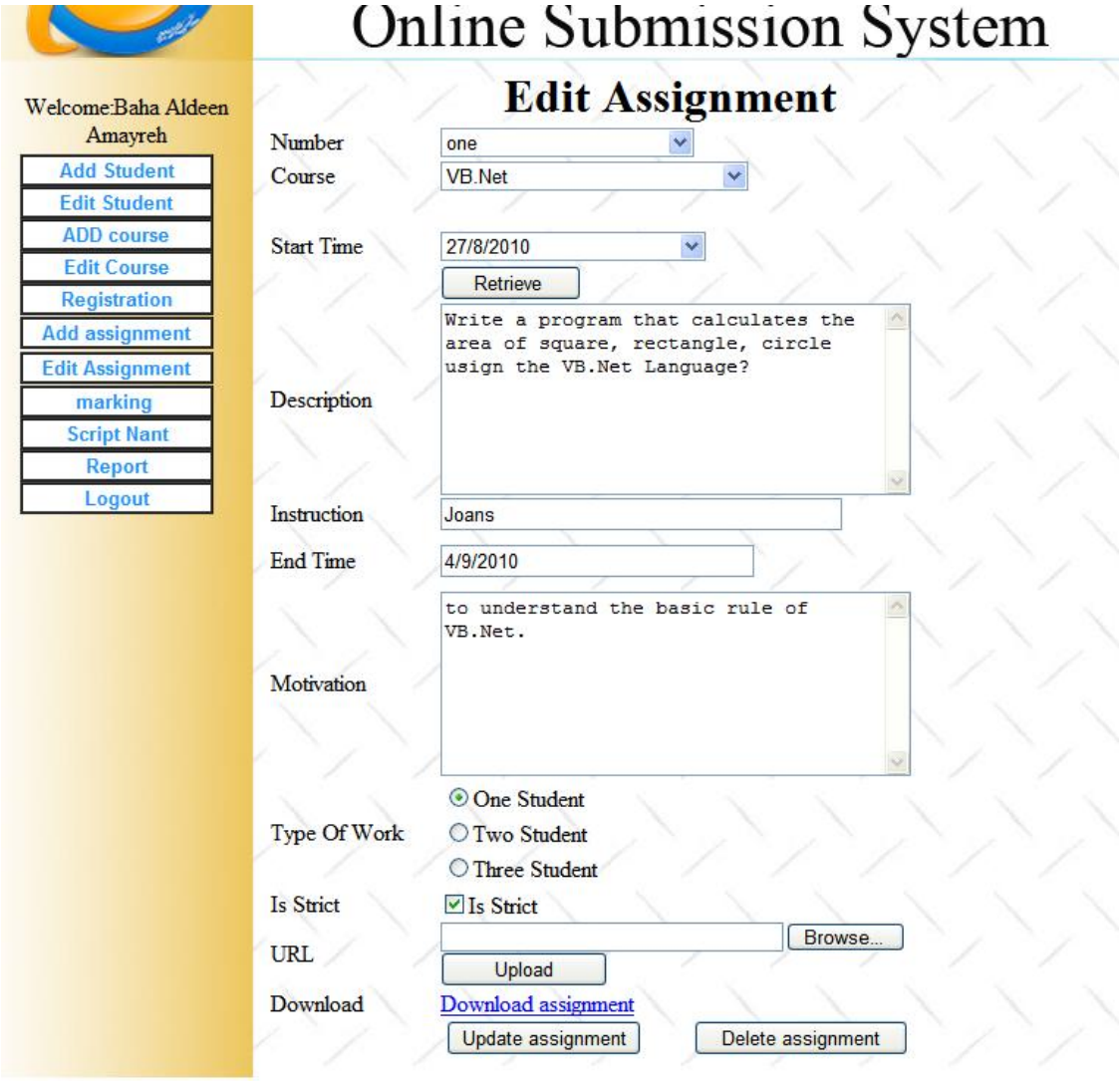
- **Add assignment page**

The form will enable the instructor to add fresh home assignments to the database, keeping in mind that the assignment code as well as the start-time and end-time should not be repeated more than once since these are primary keys.



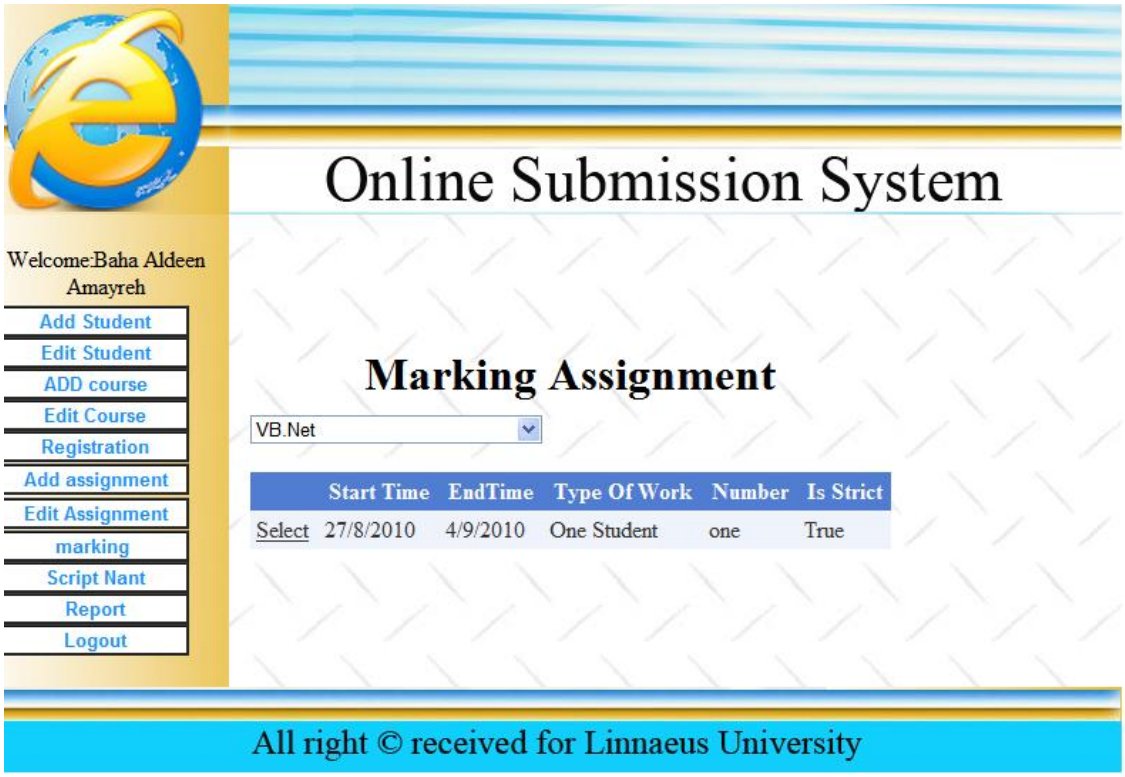
- **Edit assignment page**

The form will enable the instructor to revise data pertaining to home assignments or deleting the assignments from the database.



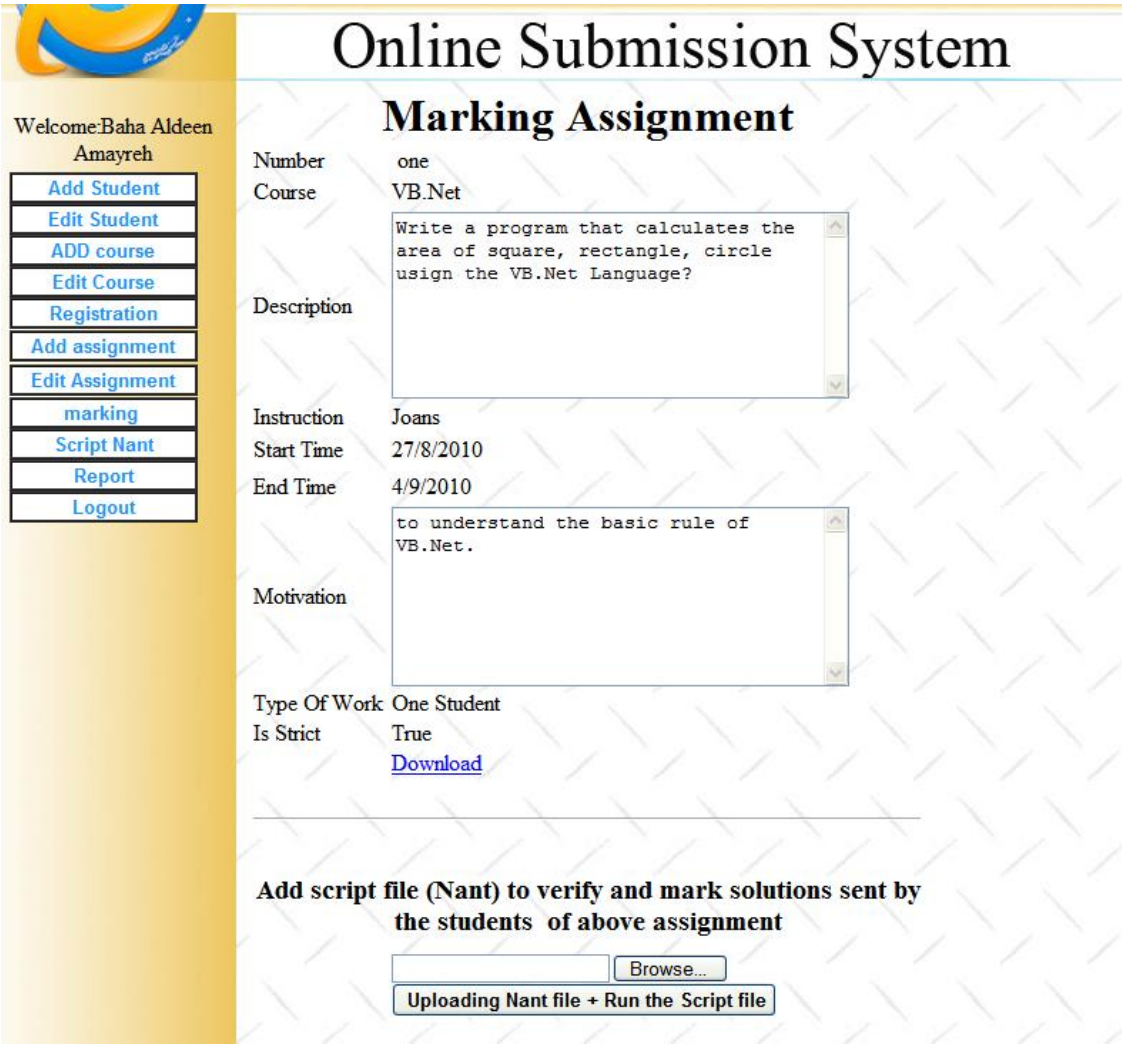
- **Select assignment to marking**

This form enables the instructor to select the course and home assignments in order to carry out the process of electronic marking. Through course testing, a list of home assignments for the current course appears. Then the instructor would upload a NANT file which helps him or her in the process of electronic marking by invoking the VB.net and translating all sent assignments by students registered in the VB.net.



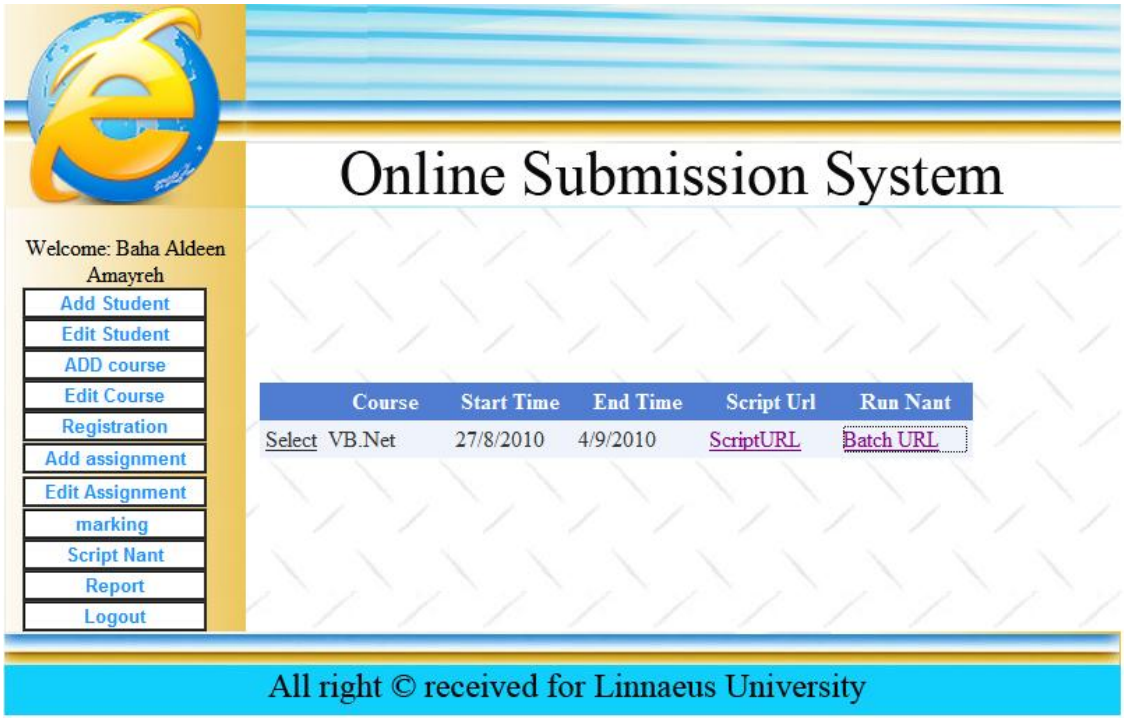
- **Marking assignment page**

After completing the translation process, the process of marking begins by ensuring that no linguistic mistakes exist in the student assignments. This would determine if the assignment will be accepted or not, as assignments containing mistakes won't be accepted.



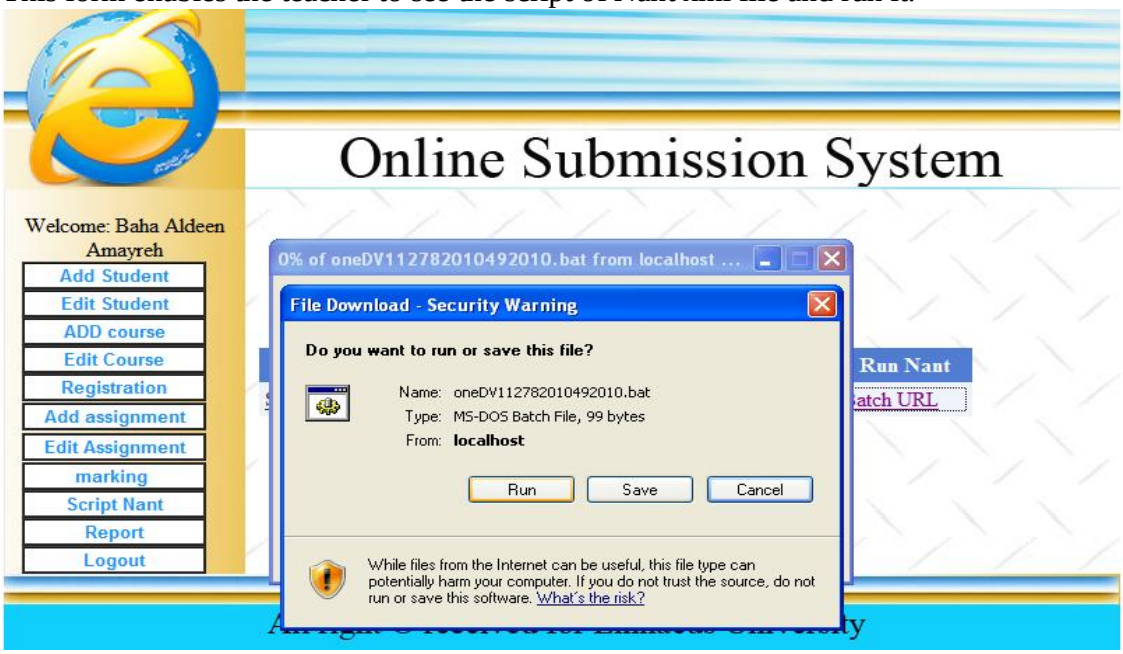
- **Selecting NANT batch file**

The form would enables the instructor to automatically recall NANT/ant files which had been uploaded, and getting them activated in order to perform automatic marking of home assignments determined previously.



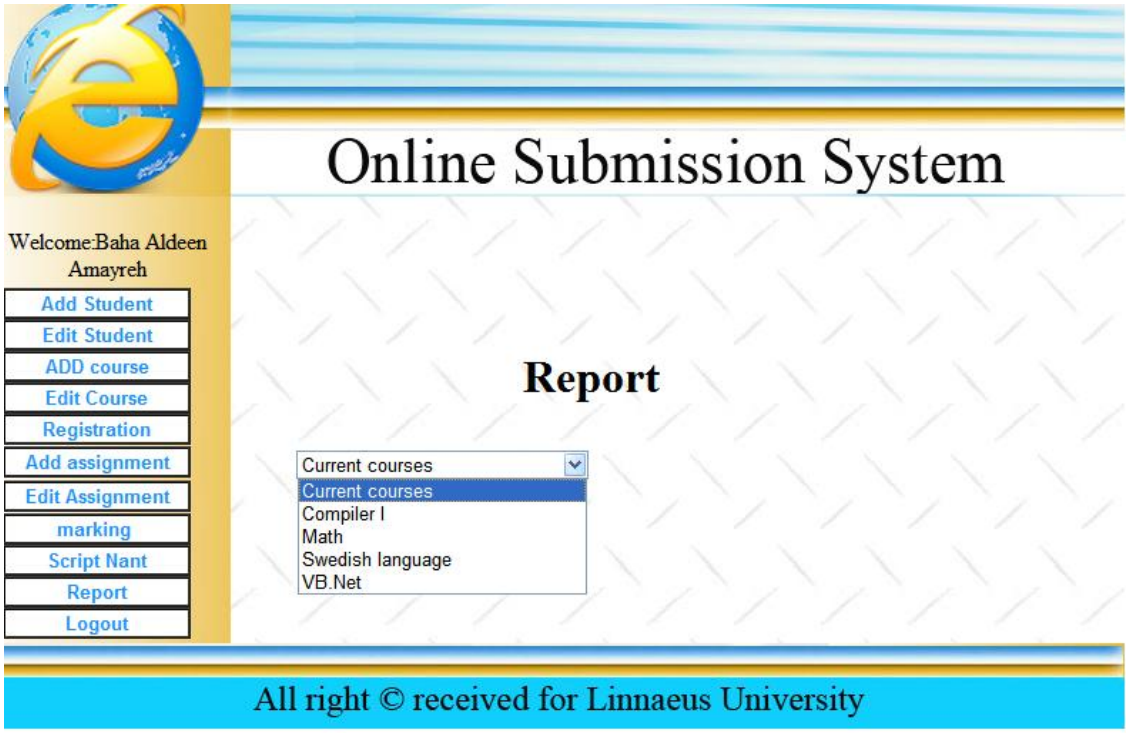
- **Run NANT batch file**

This form enables the teacher to see the script of Nant xml file and run it.



- **Report**

The form enables the instructor to compile student reports and know those students who have sent their home assignments and those who have not. It will also allow him to know how many times a given student has tried to upload his assignment before the expiry of the deadline.

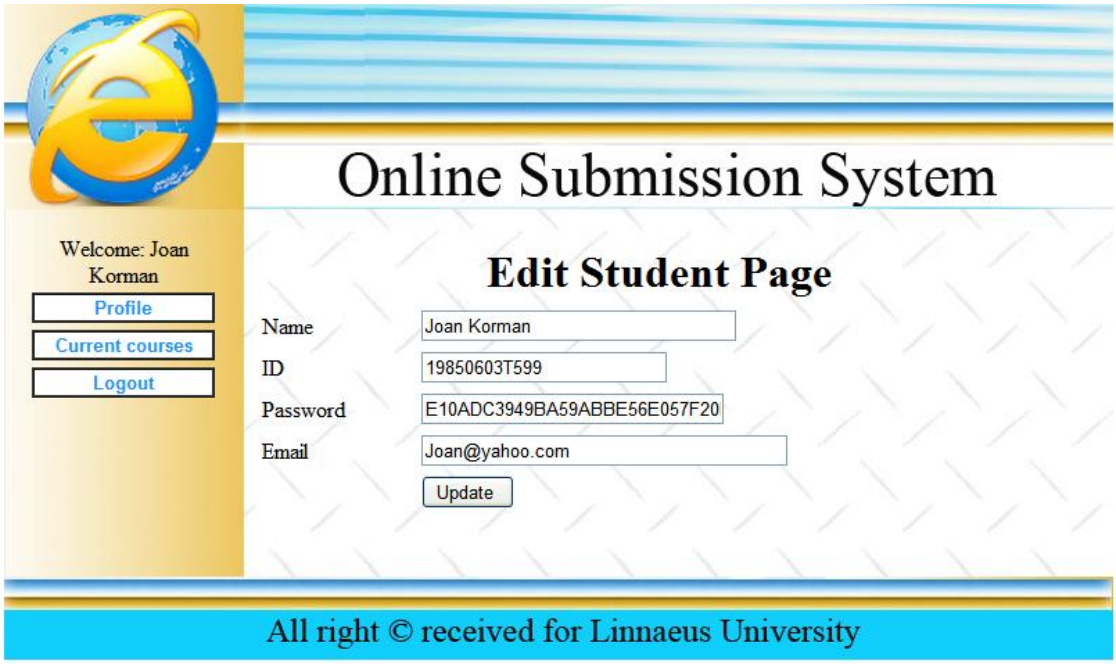


- **Student home page**

The main menu of teacher**.**

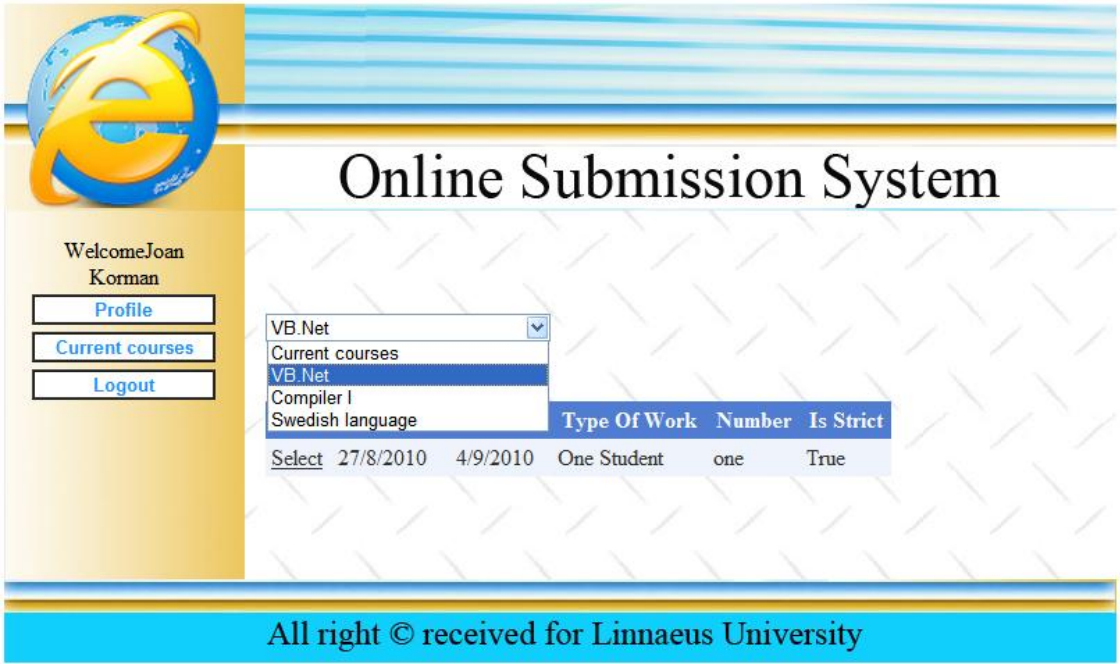
- **Student profile page**

This form enables the student to update his information (name, email and password).



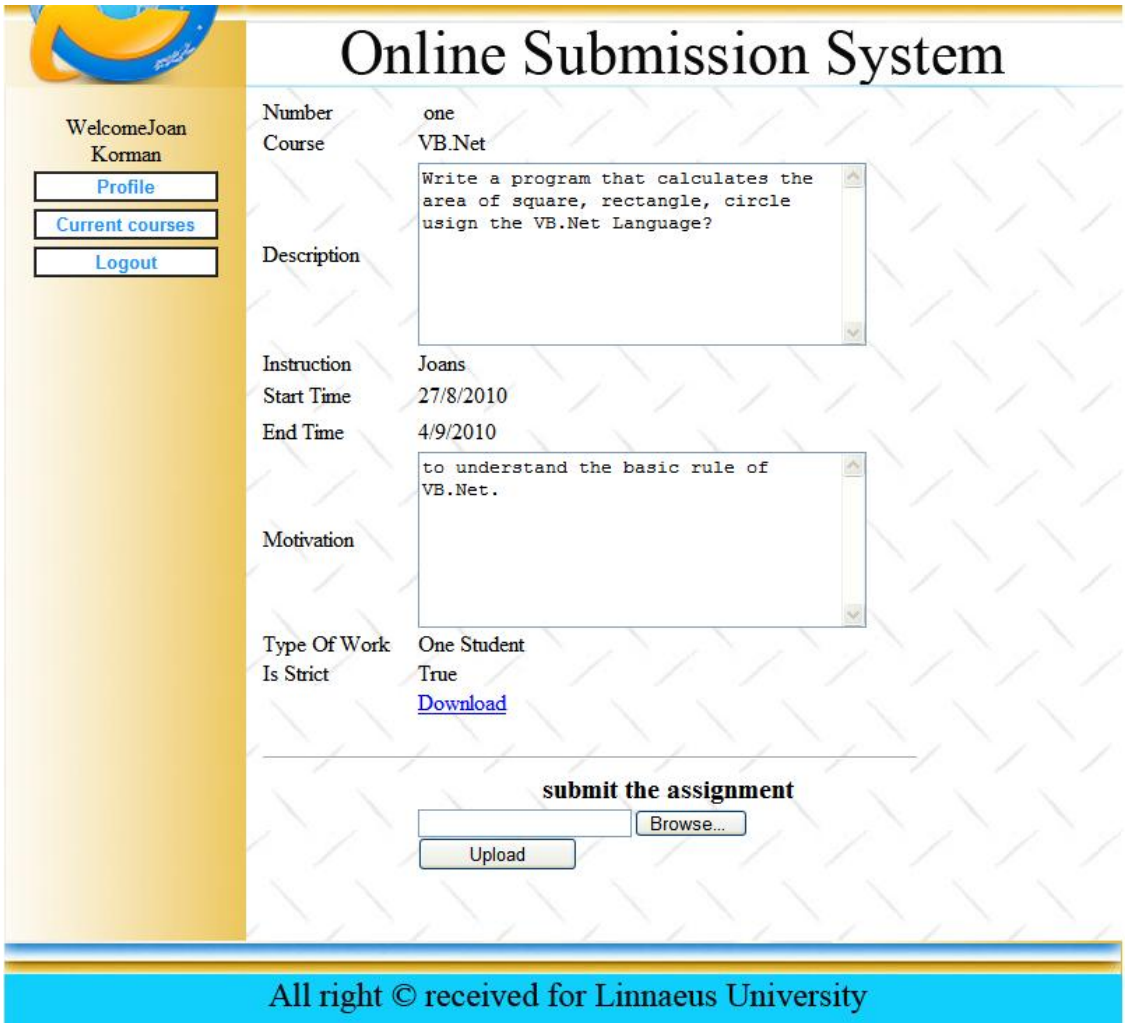
- **Student course page**

By clicking the courses link from the main menu a new page will be displayed, containing courses and related assignments.



- **Student assignment page**

This form enables the student to upload the solution of his assignments before the deadline (if strict).



Linnæus University
School of Computer Science, Physics and Mathematics

SE-351 95 Växjö / SE-391 82 Kalmar
Tel +46-772-28 80 00
dfm@lnu.se
Lnu.se