Project: ASP

This project will demonstrate that you know how to:

- Design a database with multiple tables containing relational fields
- Populate controls using data from a table
- Design a theme that utilizes master pages and CSS
- Utilize the SQL UPDATE statement
- Utilize the SQL DELETE statement
- Use data using JOIN
- Use parameters selected by user to extract specific records

This is a graded project. Do not share your code with other students. This is a reflection on how well **you** can do these programming tasks.

Project Specifications

Concept

Think of a collection or set of data that will involve several tables. For example: A music collection that contains several different artists, each artist possibly having several different albums and different genres of music.

Design the Database

Create a database named **dynamicSite.mdb**. It should have at least two tables that relate to each other. For example: a table containing a list of all musicians and another table of all albums. The two tables would be related by a common field named musicianID

The database should also contain multiple tables that will be used to populate drop-down list boxes so the user can select particular things to display on the page. For example: a list of genres for one table and a list of musicians in another table. Please set up your own tables. Do not use the data from the tutorials or the book.

This is the most important part of your project. If your tables are not well-thought out you will have difficulties in writing a good and useful UI (User Interface).

The tables only need to contain 3-4 records of data to demonstrate what you know and proof-of-concept. Don't burden your programming speed with long and cumbersome tables of data.

Tip: Be a smart DBA (DataBase Administrator) and use the naming conventions you know and love:

- Start with lower case
- useCamelCaseForReadability
- Don't use spaces
- No plurals
- Keep names succinct (short and sweet but descriptive).

This will simplify your SQL statements when you begin programming the application.

Documentation

The documentation for the site should be an html page named **documentation.html**.
It should include graphics and text explaining the following items:

- Drawing showing site design including repeating objects (for masterpage), font stack (for CSS) and color scheme (for CSS)

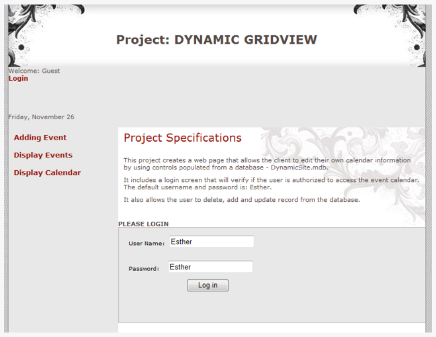
- Table structure and meta data showing relationships and key fields
- Drawings showing how the pages will look including names for each object, hex color codes, and font list.
- All pages must contain a comment block near the top of the page with:
    - Name of the file and its purpose
    - Your name and student ID and email address
    - Date the file was created
- Reflection - As you code you will run into problems. Reflect and write about at least two areas where you encountered a problem. Give a detailed description on how you solved the problem.
- Include a link to your video (see below) on this documentation page.

Site Design

Set up master pages using at least one external CSS file to make a pleasant, easy-to-read website using a specific color scheme.

- The App_Theme folder should contain the CSS file, the skin code, and graphics used by the CSS code.
- The CSS file should style at least four elements.
- The skin should style all the server-side controls on the form.
- There should be at least one graphic used by the CSS code.

Here are some shots showing how Esther Yen set up her pages. Note the professional color scheme, the "logout" option, as well as the repeating graphic elements. (Click on the images for a larger view.)



Project: DYNAMIC GRIDVIEW

Welcome: Guest
Login

Friday, November 26

Adding Event
Display Events
Display Calendar

Project Specifications

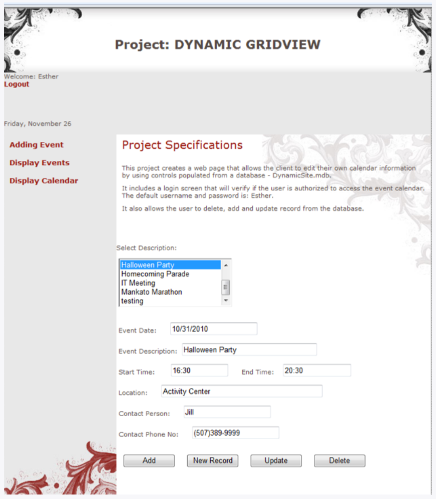
This project creates a web page that allows the client to edit their own calendar information by using controls populated from a database - DynamicSite.mdb.

It includes a login screen that will verify if the user is authorized to access the event calendar. The default username and password is: Esther.

It also allows the user to delete, add and update record from the database.

PLEASE LOGIN

User Name:	Esther
Password:	Esther



Project: DYNAMIC GRIDVIEW

Welcome: Esther
Logout

Friday, November 26

Adding Event
Display Events
Display Calendar

Project Specifications

This project creates a web page that allows the client to edit their own calendar information by using controls populated from a database - DynamicSite.mdb.

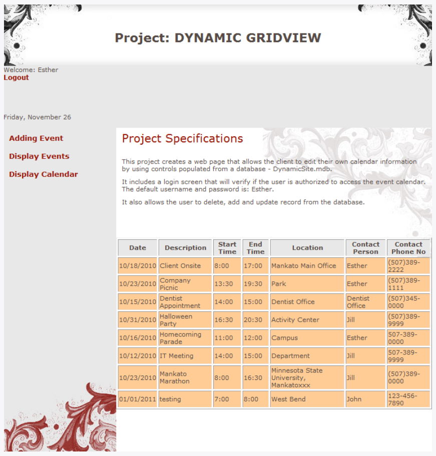
It includes a login screen that will verify if the user is authorized to access the event calendar. The default username and password is: Esther.

It also allows the user to delete, add and update record from the database.

Select Description:

Halloween Party
Homecoming Parade
IT Meeting
Mankato Marathon
testing

Event Date:	10/31/2010
Event Description:	Halloween Party
Start Time:	16:30
End Time:	20:30
Location:	Activity Center
Contact Person:	Jill
Contact Phone No:	(507)389-9999



Project: DYNAMIC GRIDVIEW

Welcome: Esther
Logout

Friday, November 26

Adding Event
Display Events
Display Calendar

Project Specifications

This project creates a web page that allows the client to edit their own calendar information by using controls populated from a database - DynamicSite.mdb.

It includes a login screen that will verify if the user is authorized to access the event calendar. The default username and password is: Esther.

It also allows the user to delete, add and update record from the database.

Date	Description	Start Time	End Time	Location	Contact Person	Contact Phone No
10/18/2010	Client Onsite	8:00	17:00	Mankato Main Office	Esther	(507)389-2222
10/23/2010	Company Picnic	13:30	19:30	Park	Esther	(507)389-1111
10/15/2010	Dentist Appointment	14:00	15:00	Dentist Office	Dentist Office	(507)345-0000
10/31/2010	Halloween Party	16:30	20:30	Activity Center	Jill	(507)389-9999
10/16/2010	Homecoming Parade	11:00	12:00	Campus	Esther	507-389-0000
10/12/2010	IT Meeting	14:00	15:00	Department	Jill	507-389-9999
10/23/2010	Mankato Marathon	8:00	16:30	Minnesota State University, Mankatoxxx	Jill	(507)389-0000
01/01/2011	testing	7:00	8:00	West Bend	John	123-456-7890

The Login Page

Create an ASP.NET Web Site that shows the contents of the database only after the user successfully logs in. The main page or login page must be named: **index.aspx**.

The login page should use at least one validation control.

The password field must be a textbox.

Include default settings so you don't have to waste time typing while you develop the program (and so I can quickly run through the program without having to look up user names and passwords.)

Using SQL statements, extract the username/password from the database. If there is no match (the userID/password is not valid), display a validation message on the login page using the validation control, setting it via code).

If there is a match, redirect to the DataView page (that uses the master page).

The DataView Page

Design a form that displays the data based on selections made by the client.
It should include:

- At least one GridView, one DropDown List, and either a RadioButton List or CheckBox List.
- All controls to be populated dynamically from the other controls on the page.
- Key field (id) should be either hidden from the user or set to read-only if displayed.
- Each time a control selection changes, update the data grid.
- Make certain each web page has its own unique <title>
- There should be editable fields. The user should be able to change data in the grid and update that record with the click of a button.
- The user should be able to delete a specific record with the click of a button.
- Demonstrate the ability to use these SQL commands: JOIN DELETE UPDATE INSERT

Include data entry fields (textboxes, radio buttons, checkboxes, buttons) that will allow the user to add a new record to one of the tables. Mark this clearly and separate from the grid so the user knows that these fields are specifically for new data being added to the database. (Don't forget to include default values to speed testing and grading.)

The Video

Create a video with narration showing the following:

- An improper login displaying the validation error.
- Successful login showing the redirect to the dataview page
- Stepping through a selection showing how SQL is used to redisplay specific data.
- Display the contents of the strSQL variable and set up a watch for that variable
- Include a link to the video as part of the HTML documentation.

Submission Guidelines

- Make certain the HTML documentation and graphics have been imported into your web site.
- Zip up the entire website folder and submit it to the D2L dropbox.
- Also please include the link in the Notes area of the D2L Dropbox. (UX - Make it an actual hyperlink, not just text.)

Assessment Checklist

75 points total possible.

Database - 5 points

Create an SQL Server database named **dynamicSite** with at least two tables that relate to each other.

The database must contain multiple tables for the main site as well as to populate drop-down list boxes.

Documentation - 10 points

An html page named **documentation.html** with:

- Drawing showing site design including repeating objects (for master page), font stack (for CSS) and color scheme (for CSS)
- Table structure and meta-data showing relationships and key fields
- Drawings showing how the pages will look including names for each object, hex color codes, and font list.
- Complete comment block near the top of the pages
- Reflections - As you code you will run into problems. Reflect and write about at least two areas where you encountered a problem. Give a detailed description on how you solved the problem.

Site Design - 10 points

Set up master page using at least one external CSS file

- The App_Theme folder should contain the CSS file, the skin code, and graphics used by the CSS code.
- The CSS file should style at least four elements.
- The skin should style all the server-side controls on the form.
- There should be at least one graphic used by the CSS code.

The Login Page - 15 points

index.aspx - An ASP.NET web site that shows the contents of the database only after the user successfully logs in.

- Uses the Master Page
- At least one validation control
- Include default values
- Password must be a textbox

- Use SQL to extract the username/password from the database
- If no match (the userID/password is not valid), display a validation message setting it via code
- If there is a match, redirect to a the DataView page (that uses the master page)

The DataView Page - 30 points

25 points - Design a form that displays the data based on selections made by the client.
It should include:

- At least one GridView, one DropDown List, and either a RadioButton List or CheckBox List.
- All controls to be populated dynamically from the other controls on the page.
- Key field (id) should be either hidden from the user or set to read-only if displayed.
- Each time a control selection changes, update the data grid.
- Make certain each web page has its own unique <title>
- There should be editable fields. The user should be able to change data in the grid and update that record with the click of a button.
- The user should be able to delete a specific record with the click of a button.
- Demonstrate the ability to use these SQL commands: JOIN DELETE UPDATE INSERT

5 points - Include data entry fields (textboxes, radio buttons, checkboxes, buttons) that will allow the user to add a new record to one of the tables. Mark this clearly and separate from the grid so the user knows that these fields are specifically for new data being added to the database. (Don't forget to include default values to speed testing and grading.)

The Video - 5 points

Create a video with narration following this script:

1. An improper login displaying the validation error.
2. Successful login showing the redirect to the dataview page
3. Stepping through a selection showing how SQL is used to redisplay specific data.
4. Display the contents of the strSQL variable and set up a watch for that variable

Revised: 04-13-15