UNIVERSITY OF CINCINNATI

Date: November 10, 2005

I, Arunkumar Chandrasekaran
hereby submit this work as part of the requirements for the degree of:
Master of Science
in:
Industrial Engineering

It is entitled:
"Building and Implementing an Inventory Database Model for the Bearcat BRAT using Microsoft Access Database and C# Programming Language."

This work and its defense approved by:

Chair: Dr. Ernest L. Hall
       Dr. Richard L. Shell
       Dr. Ronald L. Huston
Building and Implementing an Inventory

Database Model using Microsoft Access

Database and C# Programming Language

A thesis submitted to the
Division of Graduate Studies and Research
Of the University of Cincinnati

In partial fulfillment of the
Requirements for the degree of
Master of Science
in the Department of Mechanical, Industrial and Nuclear Engineering
Of the College of Engineering

2005

by

Arunkumar Chandrasekaran

B.E (Mechanical Engineering)

Crescent Engineering College, Madras University, India, 2000.

Thesis Advisor and Committee Chair: Dr. Ernest L. Hall
Abstract

Data models are the basis of almost all operations that are performed throughout the world, be it a large manufacturing company or a small vendor who supplies parts to the manufacturing company. However, this data cannot be used in its native format and has to be formatted and processed into useful information to cater different interests and applications. With internet becoming increasingly popular and user-friendly these data models have to be made accessible throughout the world for various people to reap benefits from them. The advent of powerful programming languages such as C# (C\#), which is a part of the Microsoft Visual Studio .NET platform it has become easier to present applications which are highly user-friendly and highly portable across the world wide web (www).

The thesis details the use of the .NET framework technology, more specifically, the use of C# programming language in creating a user-friendly inventory management application. It is an effort to provide the UC Robotics Team with a universally accessible and easy to use data management tool. The purpose behind developing this tool is to provide easy access to information which would be useful when the team travels to different locations across the country participating in various IGVC competitions.

The Inventory Model is built using a simple Microsoft Access database and the accessing application is built using the C# programming language, with ASP.Net as the background framework. The reason behind using Microsoft Access database against other products in the same category is because it is easy to use and also very simple to maintain.
Acknowledgement:

I would like to thank my advisor, Dr. Ernest Hall, whose guidance and support has helped me complete this thesis successfully. His suggestions and feedback had greatly helped me in my thesis and I am grateful for his support and encouragement.

I would also like to thank Dr. Richard L. Shell and Dr. Ronald L. Huston for agreeing to chair the defense committee. I would also like them for readily helping me through the process.

I am very thankful for all the support from the UC Robotics team during my period of graduate studies at the University of Cincinnati. I would also like to thank all my friends who have constantly encouraged me all through the period of study.

Finally, I would like to express my gratitude to my parents whose relentless support and encouragement made this academic pursuit possible and fruitful.
# TABLE OF CONTENTS

## CHAPTER I
**Introduction** ........................................................................................................................................... 8

## CHAPTER II
**Bearcat BRAT - Overview** .................................................................................................................. 10

## CHAPTER III
**Database Fundamentals and Concepts** .................................................................................................. 13

- Data ......................................................................................................................................................... 13
- Database .................................................................................................................................................... 13
- Database Management System (DBMS) .................................................................................................... 14
- Data Definition .......................................................................................................................................... 14
- Data Maintenance .................................................................................................................................... 15
- Data Manipulation .................................................................................................................................... 15
- Data Display ............................................................................................................................................ 15
- Data Integrity ........................................................................................................................................... 15

## DATABASE MODELS
**Normalization** ........................................................................................................................................... 17

## Codd’s Rules
**The Information rule** ........................................................................................................................... 17
**The rule of guaranteed access** ................................................................................................................ 18
**The Systematic treatment of all null values** .......................................................................................... 18
**The database description rule** .............................................................................................................. 18
**Comprehensive data sub language** ...................................................................................................... 18
**The view updating rule** ........................................................................................................................ 19
**The insert and update rule** .................................................................................................................... 19
**The physical independence rule** ........................................................................................................... 19
**The logical data independence rule** ...................................................................................................... 19
**The Integrity independence rule** .......................................................................................................... 19
**The Distribution rule** ............................................................................................................................ 19
**The Nonsubversion rule** ......................................................................................................................... 20

## CHAPTER IV
**The .NET Framework – An Overview** ................................................................................................... 21
**CLR - Common Language Runtime** ...................................................................................................... 22
**MSIL – Microsoft Intermediate Language** ............................................................................................ 24
**The Truly Object-Oriented Nature of .NET Applications** ..................................................................... 24

## Evolution from ASP
**How well did ASP yield to web programming?** .................................................................................. 25
**How has .Net changed web programming?** ........................................................................................ 26
**Error Handling – Difference brought out in .NET against earlier ASP** .............................................. 26
**Server-Side Controls – ASP vs. .NET** .................................................................................................... 27
**State Management in the .NET environment** ....................................................................................... 28
**Event-Driven programming for web applications** .............................................................................. 29
Chapter I

Introduction

The internet has revolutionized the entire world in the way we communicate and share information. The present era demands collective information and learning from various sources of knowledge. However, it is imperative that there be a unified platform or environment through which the information is shared and used. The Microsoft Corporation’s .NET framework incorporates exactly this idea of integrating all technologies into a unified environment for data access and information sharing. The thesis is an attempt to provide the UC Robotics team with a unified environment to share information via the internet.

The Center for Robotics Research at the University of Cincinnati is a group of highly talented engineers, who have been instrumental in building intelligent autonomous vehicles. These autonomous vehicles are built and they are taken to be a part of the Annual Intelligent Ground Vehicles Competition. Therefore, constructing this data model becomes very useful to the current members and other prospective members to understand the structure of the robot and other supporting information.

Objective

Any project, would suffer if there happens to be insufficient information or lack of information at the appropriate time. Hence, for a project to be successful it is really important that the correct information is available to all members working in the project.
at the appropriate time. The speed with which information is retrieved and shared also plays a very important role in the success of a project. To keep abreast with latest trends and technologies it is wise to use the internet as the means of communication and information sharing, as it is a fast and very reliable source of information exchange.

The UC Robotics team has been an active participant in the Intelligent Ground Vehicles Competition (IGVC), held annually across the country and will continue to be in the future. Apart from the IGVC, there are also other competitions that the team actively participates in every year. It would greatly help the team if there is a centralized data store and simple application to keep track of the various technical and other aspects of the several autonomous guide vehicles that the team builds every year. Therefore, the key objective of this project is to provide the team with such an environment for centralized access to information pertaining to building the robot, not only for the current year, but also extensible easily for years to come.
Chapter II

Bearcat BRAT - Overview

The Bearcat Brat robot is an Intelligent Guided Vehicle (IGV) designed to operate in unstructured environments as part of the Intelligent Ground Vehicle Competition held annually, in various parts of the country. The vehicle was designed to perform all the tasks as laid down by the rules of the contest.

Components of the Bearcat BRAT

This chapter is an overview of the various functional component of the Bearcat Brat robot. The Automated Guided Vehicle (AGV) is an integration of various components
and operational subsystems which are controlled by a high level computer that enables the vehicle in its entirety to operate and function as an intelligent machine. The components and their subsystems were each designed to perform and meet with specific requirements outlined by the contest. The challenges that the Bearcat Brat undergoes at the contest are as follows:

- Line Following
- Obstacle Avoidance
- Pothole Detection
- Waypoint Navigation

For each of the above challenges there is a specific component in the Brat which plays a major part in accomplishing the required task. The **Line Following** task requires use of a vision system which is incorporated by using the ISCAN image tracking device and two CCD cameras. **Obstacle Avoidance** and **Pothole Detection** also make use of the cameras and the scanner in addition to a SONAR system which helps in the obstacle detection and avoidance operation. The **Waypoint Navigation** challenge utilizes a Global Positioning System (GPS) for co-ordinate processing from start to end point of the track and also for continual co-ordinate monitoring along the track.

All these components are mounted on a frame consisting mainly of aluminum bars and Plexiglas sheets. The use of aluminum against any other metal is to obviously reduce the weight of the entire assembly and make a compact and easily navigable vehicle. The central computer system is mounted on this aluminum frame along with the other components like the CCD cameras, SONAR scanner, and the image tracking device, the
Global Positioning System (GPS) and the batteries which power the entire system. The mechanical drive system is equipped with gears and wheels to move the robot in the desired direction with the inputs received from the cameras, scanners and the GPS.

The Brat also houses a manual stop unit in case of an emergency or any other undesirable operating conditions. This unit consists of a red manual push button located on the rear surface of the vehicle in a way it is easily and readily accessible. There is a remote controlled stop system incorporated into the robot as per contest regulations.
Chapter III

Database fundamentals and concepts

This chapter dwells briefly on some core database concepts and explains some database terminology.

Data

The term data refers to facts and figures that about any event or other situations that can be gathered, recorded and stored for the purpose of analysis. The data that has been stored has to be processed and analyzed to turn it into useful information.

Database

A database is defined as a collection of data which is organized in a particular format and which is logically related. The size and complexity of a database may vary as the situation warrants.

The raw data are the building blocks of information. A well formed database is a very good source of information and hence it requires a planned and systematic approach to designing a proper database. For all practical applications, a good database is one which can provide information in a properly organized manner which will increase the effectiveness of communication and knowledge transfer and information exchange.
**Database Management System (DBMS)**

A Database Management System is essentially a collection of interrelated data and a set of programs to access this data. This collection of data, as previously mentioned is called a database. The primary objective of a database management system is to provide a convenient environment to retrieve and store database information. Database systems support single user and multiple user scenarios. While these systems allow only one person to access the database at a given time, the relational database system, allows many users to simultaneously access the database.

A database system consists of two major parts namely the Database Management System and the Database Application. The Database Management System is the program that organizes the data and maintains the information. The Database Application is the program that lets the user view, retrieve and update the information stored in the DBMS.

The database management system has to protect a database against unintentional changes that could be caused by users and application errors. In case of multi-user system, it must be capable of notifying the users of any changes in the database structure or in the data contained.

A good DBMS should offer the following valuable services:

**Data Definition** – This is a method of data definition and storage.
**Data Maintenance** – The DBMS checks whether each record has fields containing all information about one particular item. For example, if an employee table is considered it is seen that all information about the employees like name, address, designation, salary, department name are recorded.

**Data Manipulation** – Allows data in the database to be inserted, updated, deleted and sorted.

**Data Display** – This helps in actually viewing the data which has been recorded in the database.

**Data Integrity** – This ensures that all recorded data which has been stored previously or has been added recently, is accurate and meets the Data Definition.

**Database Models**

Database models are broadly categorized into two major categories: They are as follows-

- Object based logical models
- Record bases logical models

The object based logical model can be defined as a collection of conceptual tools for describing data, data relationships and data constraints.
The Record based logical model describes the data structure and access techniques of a database management system. There are three types of record based logical database models. They are as follows:

- Hierarchical model
- Network model
- Relational model

The Hierarchical database model is exactly as the name suggests. The data is organized in a parent-child relationship structure. The database can be thought of as a logical tree where the origin of data is the root. The data located at different levels along a particular branch from the root is called the node. The last node in the series is called the leaf. This model supports only a “one to many” relationship. The main disadvantage in this system is that a new level in data cannot be inserted easily and requires a change in the tree structure. Also there is a tendency to have multiple copies of data in different levels thereby causing data redundancy.

The Network model brings about the “many-to-many” relationship in the data. The relationship between many data items is called sets. This is a slightly advanced system and uses data pointers to locate specific records. However, when the size and volume of data stored in a network model increases it becomes increasingly difficult to locate data as all individual models apply pointers and it becomes very complex with so many pointers.
To avoid these inherent disadvantages and complexities, the relational database model was devised by Dr. E.F.Codd. This model allows all data to be represented in a simple row-column format. Each datafield is considered as a column and each record a row of the table. Different relationships can be achieved between various tables by utilizing mathematical set functions namely JOIN and UNION.

In today’s world, most database management systems are modeled on the Relational Database model.

**Normalization**

Normalization theory is built around the concept of normal forms. Normalization reduces data redundancy. Redundancy is unnecessary repetition of data, which can cause problems with storage and retrieval of data. During the process of normalization, dependencies can be identified, dependencies which would cause potential problems during deletion and updating operations on the database. Normalization theory is based on the fundamental concept of Functional Dependency. Normalization helps in simplifying the structure of tables.

**Codd’s rules**

Any database management system should obey Codd’s twelve rules for it to be relational. The twelve rules are briefly discussed below:

**The Information rule**

All information is explicitly and logically represented in tables as data values.
The rule of guaranteed access

Every item of data must be logically addressable with the help of table name, primary key value and column name. Form this it is clear that any individual record can be retrieved with the use of a table name primary value of the row and the column name where it is to be found.

The Systematic treatment of all null values

The DBMS must be able to support null values to represent missing or inapplicable information. They must be distinct from zeros and spaces. Null values for all data types must be the same. One of the most important aspects that must be noted here is that there is a vast difference between a null value and zero and a space.

The database description rule

A description of database is maintained using the same logical structures with which data has been defined by the DBMS. These are accessible to users with appropriate authority and are stored in the data dictionary.

Comprehensive data sub language

According to this rule, the system must support the following

- Data definition
- View definition
- Data manipulation
- Integrity constraints
• Authorization

• Transaction management operations

The view updating rule

All views that are theoretically updateable must also be updateable by the system.

The insert and update rule

A single operand must hold good for all retrieval, update, delete and insert activities. This rule implies that all the data manipulation commands must be operational on sets of rows in relation rather than on a single row.

The physical independence rule

Application programs must remain unimpaired when any changes are made in storage representation or access methods.

The logical data independence rule

The changes that are made should not affect the user’s ability to work with the data. The change can be splitting the table into many more tables.

The Integrity independence rule

The integrity constraints should be stored in the system catalog or in the database as a table.

The Distribution rule

The system must be able to access or manipulate the data that is distributed in other systems.
The Nonsubversion rule

The nonsubversion rule states that different levels of the language cannot subvert or bypass the integrity rules and constraints. Simply put, if an RDBMS supports a lower level language then it should not bypass any integrity constraints defined in the higher level.
Chapter IV

The .Net Framework – an Overview

The .NET Framework is a development and execution environment that allows different programming languages & libraries to work together seamlessly to create Windows-based applications that are easier to build, manage, deploy, and integrate with other networked systems. It is the base that supports the runtime needs of all .Net applications.

The .NET Framework consists of two major components:

- **The Common Language Runtime (CLR)**

  The CLR is a language-neutral development & execution environment that provides services to help "manage" application execution. The CLR provides core services such as memory and thread management. It manages code written for the .Net environment and enforces type-safety.

- **The Framework Class Libraries (FCL)**

  The FCL is a consistent, object-oriented library of prepackaged functionality. This library of functions is provided by the .Net framework for developing programs that run under the CLR. The function library consists of object-oriented classes that simplify programming tasks and speed up development time. The library of classes supports development programs that run as command-line programs and as Windows based applications.
Code management is at the core of the CLR and the CLR itself is at the core of the .Net framework environment. Any programming language that targets the CLR is subject to code management. The security and programming model used by the .NET framework means that many applications that previously had to be installed on a computer locally, can now be deployed over the internet. A program may run components from different vendors and from various internet websites, and each component would have its own security and permissions on the local computer. That is it no longer necessary to wait endless months for a vendor to provide a service pack or program upgrade to fix or improve existing software applications. This is because, for obvious reasons, components on internet sites could be updated more quickly and deployment would be greatly simplified.

The .Net environment supports development of a number of programs such as console applications. Such programs have an advantage of being utility command-line programs that have access to Windows graphical user interface (GUI). The environment provides a set of reusable GUI objects in the Windows form classes. It also provides similar set of objects which are included in the Windows Web forms classes, for developing web applications.

**CLR - Common Language Runtime**

Programmers in the past have linked their code with various libraries and made use of external components for their applications. In addition to the runtime library, a number of specialized code libraries are available. Dynamic Link Libraries (DLLs) have provided
another source of code at runtime. For example, the code for the Windows Application Programming Interface (API), resides in several DLLs that a program can call as needed.

However, for the C# programming language, the .Net framework defines more than 1000 classes that a programmer may access directly or use as a template for deriving new classes to extend functionality that these classes provide.

The CLR is the environment used by the .Net environment. It manages code at runtime and provides services that make program development easier. The code that is being written for the CLR is called “managed” code.

According to Microsoft, the process of running code under the CLR is like running with a “contract of cooperation” with the runtime environment. Managed code undergoes a process of “verification” before the CLR will let it run on the computer. The CLR provides a type-safe environment where-in an application does not cause a memory fault, which is an attempt to access memory that it does not own. This is a common problem for C++ programmers who might sometimes forget to initialize a pointer before attempting to access memory.

C# also uses pointers in a way, but hides them from the programmer in the form of reference-type variables. The program will simply not allow the usage of a pointer variable or any regular variable without a value being assigned to it. However, the C# coding process is not perfect and it could still cause some errors to come up at runtime.

The CLR watches the code and catches these instances. When such a situation arises, the program throws an exception through the CLR. The programmer may choose to include
customized code in the program to watch for these “exceptions” and catch them in the program itself. If not properly handled or if not handled at all, the CLR will cause the program to end.

Programmers will find the CLR much more flexible in terms of programming language code interoperability. This is another very good feature of the .Net framework and the C# programming language.

**MSIL – Microsoft Intermediate Language**

The C# compiler converts source code files to an IL (intermediary language), which theoretically, may be used to run the same program on different operating systems and different computer platforms. When a program is executed, the runtime sends this information which is the IL or in this case the MSIL, to a just-in-time compiler, which in turn converts the code into native code for the computer on which the program is running. Microsoft uses just-in-time to indicate that an action is taken only when necessary.

**The Truly Object-Oriented nature of .Net applications**

A major change from earlier ASP applications is that .Net offers a completely object oriented application design and programming environment. All .Net applications are fully object oriented.

All the code resides inside some class. Everything is an object, even simple data types. There is full support for static and instance properties, static and instance methods, events, virtual functions, abstract classes, polymorphism, data hiding and inheritance.
Using these features allows us to create far more robust applications than were possible with Active Server Pages (ASP).

**Evolution from ASP – Some insight into how .NET has changed web programming**

**How well did ASP yield to web programming?**

By moving to an object oriented design in our web applications, we also get away from the tight binding between the user interface and the business logic that was inevitable in ASP. ASP applications consisted of the ASP files that contained both the user interface design code, which is in the form of HTML tags, and the business logic was bound together by other special tags marking the beginning and end of the dynamically executed code. To write object oriented code programmers either had to use very limited and mostly non-intuitive objects that can be created within the ASP scripting languages or write run-time callable COM objects that would be called by the scripting code. This allowed for black-box objects to be used in ASP applications, but the only truly object oriented code is confined to living inside the COM object itself.

One nice feature of COM is that objects can be written using various languages and tools but not ASP scripting languages. Therefore, creating object oriented Web applications with ASP required programming in at least two different tools and languages. Doing this also subjected the programmer to the complexities of calling COM objects by their dispatch interface, as well as the inherent dangers of calling code at runtime with no early binding type safety. Also, in these cases an ASP programmer who did not possess the
skills to create COM objects would have to rely on another programmer to provide this portion of the application.

**How has .Net changed web programming?**

As already mentioned .Net applications are entirely object oriented inherently. In all .Net applications, COM is no longer the foundation for discovering, loading, and using binary black-box objects at runtime. .Net has retooled the way this works and has its own native way to handle this using meta data that is compiled into each and every object built on the common language runtime. Legacy COM objects are still accessible from any .NET application including ASP.NET. However, to really leverage the power that .NET provides, programmers can write objects of virtually unlimited complexity directly in the ASP.NET code.

**Error Handling – Difference brought out in .NET against earlier ASP**

ASP applications traditionally have had very poor support for error handling because of the limited features supported by the scripting languages. In VBScript, the support for *On Error* leaves a lot to be desired and is not at all extensible. In ASP.NET, the application is running within the .NET runtime, which supports a very robust exception handling mechanism using a try-catch-finally-and-throw syntax. In fact, the exception handling in .NET has been honed over many years of Microsoft’s providing programming languages that support exceptions. So, due to the full object support in .NET, programmers can implement powerful, custom error handling routines.
Server-Side Controls – ASP vs. .NET

To provide user interfaces, ASP programmers have had to include standard HTML controls in their ASP files and access them via their IDs within the Document Object Model (DOM) or create the HTML to render the control directly using script code. To use a control in an object oriented way, you would have to call a COM object that exposes an interface. In this case, however, the object is wrapping up the details of writing HTML control code onto the stream headed back to the browser.

.NET adds a host of feature rich controls called Server Controls. These Controls are implemented and accessible as objects in the .NET framework. To lay these controls out in the user interface, special tags are added to the HTML user interface files. These controls are also flagged in code such that they will be run on the server. When the page is loaded and processed, the ASP.NET engine creates the correct HTML code to render the controls and sends it back to the browser for the user. Throughout the code for the page the controls can be accessed very much the same way as a windows control, by referring to them by their variable names and calling methods and accessing properties on them.

Additionally, ASP.NET will automatically detect the browser devices’ capabilities and then render the control appropriately. This frees programming effort in terms of managing code for detecting different versions of HTML and DHTML. There are controls for almost every type of UI item that would be needed for web applications, and some complex controls as well. For example, to save the programmer time in creating tables, divs and spans to render data dynamically, grid controls inherently have these functionalities.
State Management in the .NET environment

In ASP, each time a page is loaded, the code is executed in a top-down fashion, and the response, including the dynamically generated HTML code, is sent to the browser. There is no state held between a browser and the server because HTTP is a connectionless protocol. The problem with this is that when the user reloads the page for any reason, including action taken on some UI item like a form submit button that causes a trip to the server, the page has to be completely reloaded. The programmer is responsible for making sure the elements in the UI are in the same state each subsequent time the server resends the page.

In ASP.NET this task is handled automatically and quite nicely. The ASP.NET page framework makes the stateless HTTP protocol appear to have a state. In terms of programming, this means that there is no longer the need to write code to keep the UI current with what the user has done to it between multiple server round trips on the same page.

In ASP, programmers often jump to a new page when a user performs an action that cause a round trip to the server. In ASP.NET, because the state of the UI is taken care of automatically, programmers will find themselves adding more functionality to a single page even if it requires multiple trips to the server. In addition to the UI state as the user sees it, state is maintained in the UI objects that are in the code behind the page as well.

In ASP form variables or even a query string must be consulted to see what value was in a UI control when the page was submitted. In ASP.NET, the values are in the most
natural place, the control object variable itself. This adds a whole new level of UI programmability to web applications.

**Event-Driven programming for web applications**

ASP provided nothing in the way of event-driven programming. In order to handle events programmers had to rely on handling form submissions or handle the event on the client using VBScript, JavaScript or JScript. In ASP.NET, when a user causes an event by pressing a button or selecting an item in a list on the browser, an event is fired in the code on the server. This occurs via some .NET-generated JavaScript and a form submittal, but that happens behind the scenes. Finally web programmers can handle real events all in one place- the server. All events in .NET are handled by what are called delegates, which are analogous to function pointers.

Actually these delegates are just like the event handlers in visual basic applications, and for most standard events, they can even be added via point and click in the Visual Studio .NET IDE. But unlike VB event handlers, .NET event handlers can be added both at design time and at runtime, so much more robust operations can performed, like creating customized events. This is totally supported with ASP.NET applications also.
Chapter V

Inventory Model for the Bearcat BRAT

The Bearcat Brat is a system comprising of various components such as motors, wheels, a central computer system, CCD cameras, GPS and a few others parts that have been integrated and assembled on a single frame to make it an autonomous guided vehicle. The data model for this robot has been derived by making a hierarchical classification of the components by creating a super system and having sub-systems under it for ease of classification. The BRAT has five main super systems which have their own child components. Below is the Schematic representation of the main classification of the different systems.

![Figure 2 – Bearcat Brat data model – Schematic Representation](image-url)
**Master Table**

A master table was created based on the above nomenclature to have only details of system names and assigning an individual character ID for each system. The database table “master” is shown below in design view.

![Figure 3 – Database view of the Master table](image)

**The Sub-system classification**

![Figure 4 – Mechanical System](image)
Electrical System

Drive

Drive Accessories

Figure 5 – Electrical System

Power system

Power Source

Accessories

Figure 6 – Power System

Central Controller System

Computer

Controller

Figure 7 – Central Controller System
The subsystem database in the database has been designed to incorporate the structure the above schematic diagrams have depicted.
Components table

The component table is structured in a way as to hold the component details as follows:

- Component ID
- Component Name
- Number of Units required of each component
- Name of the manufacturer for each component
- Linking all the above information to the corresponding subsystem to which the each component belongs.

![Figure 10 – The Database view of the Component table](image-url)
Manufacturer Details table

The manufacturer details table consists of brief details about the manufacturer as follows:

- Manufacturer ID
- Manufacturer Name
- Other Info – This field is a memo field which can take more than a single of text. This field can be used to put in information like address, or quick contact number which would help at the time of placing orders for a single or many components.

Figure 11 – The Database view of the Manufacturer Details table
Sponsor Details Table

The information of all the sponsors for the UC Robotics team is stored and updated. The Sponsor details have been structured as the following fields:

- Sponsor ID
- First Name
- Last Name
- Mailing Address
- City
- State
- Zip code
- Phone number
- Email

Figure 12 – The Database view of the Sponsors table
This table can be used to view and modify any of the above listed information corresponding to the sponsors. New sponsors can also be added and their information can be updated very easily. The main objective of building this table into the project was to help the team keep track of their regular sponsors and also provide space to add new sponsors and their details. Additional fields can be added with minimal effort and maintained very easily.

**Member Information Table**

This table holds information about the members in the UC Robotics team. All information can be viewed and modified very easily. The fields in this table are as follows:

- Member ID
- First Name
- Last Name
- Member address
- City
- State
- Zip code
- Phone number
- Email
Figure 13 – The Database view of the Members table

User Information

There is a separate table which has been created to maintain users’ login information. This table is exclusively for security administration within the data environment.

Figure 14 - The Database view of the Users table
Chapter VI

Implementing the Inventory Model for the Bearcat BRAT

The inventory database model has been implemented using a C# web application. The reason behind using C# as the programming language is because it is powerful and robust and provides very programmer friendly debugging capabilities.

The User interface is divided into two different sections based on whether a user wants to just read the information available in the database or a power user has to add or update information into the database. The user is presented with a different user interface depending upon whether he/she is a read-only user or read/update user.

The basic GUI (Graphical User Interface) has been developed and rendered using ASP.NET web forms and the code-behind which supports the functioning of the application is handled entirely by C#. The application has been built on a MS-Access Database for the simple reason of ready availability and portability. Access databases are also very easy to create and maintain.

The Web-Server Controls

There are various GUI controls provided by the .NET framework which simplifies the programming and also provides a very wide scope for various methods in database-programming and working with data-bound controls. To create a user-friendly interface
and for better data presentation and manipulation some special web-server controls have been used in this project. The controls are as follows:

**Data Controls** –

- OleDbDataAdapter
- OleDbConnection
- Dataset

**Web Server Controls** –

- Data Grid

The OleDbDataAdapter and the OleDbConnection controls can be dragged and dropped onto the user interface and can be configured to connect to specific tables in the database. After configuring the Data Adapter, the control by itself creates a connection control and using these controls a Data Set can be created. The Data Set is the interface between the user and the database and is used to store the changed data before the change is even reflected on to the database. The Data Grid control can then be mapped to the Data Set and thereby will display the contents of the database in a neatly formatted grid, which is very easy to read for the user.
Figure 15 – Data Controls

Figure 16 – Data Grid web-server control
The *DataGrid* can be formatted using its own property windows to create a simple grid which is bound to a table on the database. The *DataSource* property is used to bind it typically to a Data Set which has been generated using the specific table on the database. *DataMember* property is activated when the DataSource is selected and gives the table name which is being used currently. The *DataKey* Field is the *primary key* field on the table and is used to index the dataset and thereby the grid control itself. Once the configuration has been completed the data grid is ready to use with just a couple of lines of additional code which has to be put in the *Page_Load()* event of the web page which holds the data grid. The code-behind the actual working of the data grid has been attached in the Appendix section.

![DataGrid1 Properties](image)

*Figure 17 – Data grid configuration window*
The User Interface

The opening page or the home page of the application gives the users an option to go and view the data from several tables or to update data by logging in. The user log-on panel is a very standard panel, and has been integrated into the page to provide data security, to avoid accidental data access where not warranted.

The Home Page – Login / Read only data access
### The Master Table [Read-Only]

![Master Table](image)

<table>
<thead>
<tr>
<th>machine_id</th>
<th>system_name</th>
</tr>
</thead>
<tbody>
<tr>
<td>central</td>
<td>central control system</td>
</tr>
<tr>
<td>elec</td>
<td>electrical system</td>
</tr>
<tr>
<td>mech</td>
<td>mechanical</td>
</tr>
</tbody>
</table>

[Home Page](#)

### Sub System Table [Read-Only]

![Sub System Table](image)

<table>
<thead>
<tr>
<th>sub_id</th>
<th>sub_name</th>
<th>system_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>central_sub1</td>
<td>computer</td>
<td>central</td>
</tr>
<tr>
<td>central_sub2</td>
<td>centralizer</td>
<td>central</td>
</tr>
<tr>
<td>elec_sub1</td>
<td>drive</td>
<td>elec</td>
</tr>
<tr>
<td>elec_sub2</td>
<td>driveacc</td>
<td>elec</td>
</tr>
<tr>
<td>mech_sub1</td>
<td>mech</td>
<td>frames</td>
</tr>
<tr>
<td>mech_sub2</td>
<td>transmission</td>
<td>mech</td>
</tr>
</tbody>
</table>

[Home Page](#)
Component Table [Read-Only]

Manufacturer Details [Read-Only]
Member Details [Read-Only]

<table>
<thead>
<tr>
<th>Member</th>
<th>Last Name</th>
<th>First Name</th>
<th>Address</th>
<th>City</th>
<th>State</th>
<th>Zip</th>
<th>Phone</th>
<th>Email</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Ernest</td>
<td>Hall</td>
<td></td>
<td>Cincinnati</td>
<td>OH</td>
<td>45219</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Anandkumar</td>
<td>Chandra</td>
<td></td>
<td>Charlotte</td>
<td>NC</td>
<td>28262</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Dinesh</td>
<td>Dharamsray</td>
<td></td>
<td>Cincinnati</td>
<td>OH</td>
<td>45220</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Vidyasagar</td>
<td>Maruthy</td>
<td></td>
<td>MI</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Sponsor Details [Read-Only]

<table>
<thead>
<tr>
<th>Sponsor</th>
<th>Sponsered By</th>
<th>Address</th>
<th>City</th>
<th>State</th>
<th>Zip</th>
<th>Phone</th>
<th>Email</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FANUC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Doitold Inc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Ernest</td>
<td>Hall</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>34535th</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Home Page
The Central update console

Master Table [Update]
Sub-System Table [Update]

Component Details [Update]
Member Details [Update]

User Administration screen
The data grid control has one very excellent feature which is worth mentioning. Typically database bound applications are very difficult to program especially when addition and updation of data is involved. Usually, programmers have to resort the use of multiple button controls to accomplish selecting a specific row and updating information for that row of data. Multiple buttons means clutter in the user interface and hence does not make it very presentable in some cases when the database table has a lot of columns to be updated at once. To eliminate this difficulty, the data grid has something called “button columns” and these columns provide the use of “hyperlink” type buttons or regular solid buttons within the grid for data manipulation. In the above update-able grids, there are two buttons within the grid control – Select and Delete. The Delete button has been named “Remove” to make it more intuitive for the end user. Each table in the database has been mapped to individual edit/update screens for ease of use and programmer friendliness in case any interface has to be updated in the future to keep up with changes in the database, which can be accomplished with very minimal effort.
Chapter VII

Conclusions

Inventory data models are an essential part of any continuing project or regularly functioning organization or establishment. Since the UC Robotics team participates annually in several competitions, it would be a very handy tool to store and retrieve information out of the inventory application. Although the MS – Access database that is used is not very robust, it has been used to demonstrate how an easy to use and regularly maintainable inventory can be modeled and implemented. Inventory databases, can also be used to store pictures and the application can also be extended to store important documents and files. Such heavy-duty full-scale databases can be built using SQL Server or Oracle, when the volume of data which is being handled is enormous. Once again, just to maintain simplicity in the project, the MS-Access database has been used, since the volume of data is not phenomenal as in the case of corporate databases.

In order to get the maximum benefit from such database applications, it is necessary to keep them updated at regular intervals. Therefore it is very essential to avoid repetitive data input which will cause data redundancy. Data redundancy needs to be avoided to make the application more effective and also to avoid database corruption down the road, since such databases are typically designed to be used by multiple users.

In a multi user environment extreme care should be taken to avoid indiscriminate data manipulation, and thereby user level security has to be established and robust validation
techniques have to be incorporated for the aforementioned security purposes. This application has been integrated with the above mentioned data security to avoid such erroneous operations and thereby provide a safe ground for storing and managing valuable information.
References

Text References:


Web References:

1. http://www.igvc.org/deploy/design/design_years.asp
2. http://libraries.uc.edu/
Appendix

The C# code behind

bratInv.aspx.cs

using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;

namespace BratInventory
{
    /// <summary>
    /// Summary description for bratInv.
    /// </summary>
    public class bratInv : System.Web.UI.Page
    {
        private OleDbDataAdapter adp;
        private DataSet ds;

        protected System.Web.UI.WebControls.Label Label1;
        protected System.Web.UI.WebControls.Label Label2;
        protected System.Web.UI.WebControls.TextBox txtPWD;
        protected System.Web.UI.WebControls.TextBox txtUID;
        protected System.Web.UI.WebControls.Button btnLogIn;
        protected System.Web.UI.WebControls.Label Label3;
        protected System.Web.UI.WebControls.Label Label4;
        protected System.Web.UI.WebControls.Button btnMasterRead;
        protected System.Web.UI.WebControls.Button btnSubRead;
        protected System.Web.UI.WebControls.Button btnCompRead;
        protected System.Web.UI.WebControls.Button btnMemRead;
        protected System.Web.UI.WebControls.Button btnSponsorRead;
        protected System.Web.UI.WebControls.Label Label5;
        protected System.Web.UI.WebControls.Label Label6;
        protected System.Web.UI.WebControls.Button btnMfgRead;

        private void Page_Load(object sender, System.EventArgs e)
        {
            // Put user code to initialize the page here
        }

        #region Web Form Designer generated code
        override protected void OnInit(EventArgs e)
        {
            // CODEGEN: This call is required by the ASP.NET Web Form Designer.
        }
    }
}
//
InitializeComponent();
base.OnInit(e);
}

/// <summary>
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
    this.btnLogIn.Click += new System.EventHandler(this.btnLogIn_Click);
    this.btnSponsorRead.Click += new System.EventHandler(this.btnSponsorRead_Click);
    this.btnMemRead.Click += new System.EventHandler(this.btnMemRead_Click);
    this.btnCompRead.Click += new System.EventHandler(this.btnCompRead_Click);
    this.btnSubRead.Click += new System.EventHandler(this.btnSubRead_Click);
    this.btnMasterRead.Click += new System.EventHandler(this.btnMasterRead_Click);
    this.btnMfgRead.Click += new System.EventHandler(this.btnMfgRead_Click);
    this.Load += new System.EventHandler(this.Page_Load);
}
#endregion

private void btnMasterRead_Click(object sender, System.EventArgs e)
{
    Response.Redirect("RbratMaster.aspx");
}

private void btnSubRead_Click(object sender, System.EventArgs e)
{
    Response.Redirect("RbratSubsys.aspx");
}

private void btnCompRead_Click(object sender, System.EventArgs e)
{
    Response.Redirect("RbratComponents.aspx");
}

private void btnMfgRead_Click(object sender, System.EventArgs e)
{
    Response.Redirect("RbratMfgDetails.aspx");
}

private void btnMemRead_Click(object sender, System.EventArgs e)
{
}
private void btnSponsorRead_Click(object sender, System.EventArgs e)
{
    Response.Redirect("RbratMembers.aspx");
}

private void btnLogIn_Click(object sender, System.EventArgs e)
{
    string userid = txtUID.Text;
    string password = txtPWD.Text;
    string sql, pass;

    if (IsPostBack)
    {
        try
        {
            OleDbConnection log_conn = new OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Password="";User ID=Admin;Data Source=E:\ak\bratTest\BratInventory.mdb;Mode=ReadWrite;Extended Properties="";Jet OLEDB:System database="";Jet OLEDB:Registry Path="";Jet OLEDB:Database Password="";Jet OLEDB:Engine Type=5;Jet OLEDB:Database Locking Mode=1;Jet OLEDB:Global Partial Bulk Ops=2;Jet OLEDB:Global Bulk Transactions=1;Jet OLEDB:New Database Password="";Jet OLEDB:Create System Database=False;Jet OLEDB:Encrypt Database=False;Jet OLEDB:Don't Copy Locale on Compact=False;Jet OLEDB:Compact Without Replica Repair=False;Jet OLEDB:SFP=False");
            adp = new OleDbDataAdapter();
            sql = "Select user_id, pwd FROM users WHERE user_id =" + userid + " and pwd =" + password + "";
            OleDbCommand selectCmd = new OleDbCommand(sql, log_conn);
            adp.SelectCommand = selectCmd;
            DataSet ds = new DataSet();
            adp.Fill(ds, "users");

            if (ds.Tables["users"].Rows.Count > 0)
            {
                Session["uid"] = userid;
                Response.Redirect("bratInvUpdate.aspx");
            }
            else
            {
                Response.Redirect("bratInv.aspx");
            }
        }
        catch (OleDbException oleEx)
        {
            oleEx.ToString();
        }
    }
}
catch (Exception ex)
{
    Response.Write("Exception: "+
    ex.ToString());
}

bratInvUpdate.aspx.cs
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;
namespace BratInventory
{
    /// <summary>
    /// Summary description for bratInvUpdate.
    /// </summary>
    public class bratInvUpdate : System.Web.UI.Page
    {
        protected System.Web.UI.WebControls.Label Label1;
        protected System.Web.UI.WebControls.Button btnMasterUpdate;
        protected System.Web.UI.WebControls.Button btnSubUpdate;
        protected System.Web.UI.WebControls.Button btnCompUpdate;
        protected System.Web.UI.WebControls.LinkButton LinkButton1;
        // Put user code to initialize the page here
    
    //region Web Form Designer generated code
    override protected void OnInit(EventArgs e)
    {
        // CODEGEN: This call is required by the ASP.NET Web Form Designer.
    }
}
///
/// InitializeComponent();
/// base.OnInit(e);
///
/// <summary>
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
    this.btnMasterUpdate.Click += new System.EventHandler(this.btnMasterUpdate_Click);
    this.btnUserUpdate.Click += new System.EventHandler(this.btnUserUpdate_Click);
    this.btnMembersUpdate.Click += new System.EventHandler(this.btnMembersUpdate_Click);
    this.btnSubUpdate.Click += new System.EventHandler(this.btnSubUpdate_Click);
    this.btnCompUpdate.Click += new System.EventHandler(this.btnCompUpdate_Click);
    this.btnMfgUpdate.Click += new System.EventHandler(this.btnMfgUpdate_Click);
    this.btnSponsorUpdate.Click += new System.EventHandler(this.btnSponsorUpdate_Click);
    this.LinkButton1.Click += new System.EventHandler(this.LinkButton1_Click);
    this.Load += new System.EventHandler(this.Page_Load);
}

#region

private void btnMasterUpdate_Click(object sender, System.EventArgs e)
{
    Response.Redirect("UbratMaster.aspx");
}

private void btnSubUpdate_Click(object sender, System.EventArgs e)
{
    Response.Redirect("UbratSubsys.aspx");
}

private void btnCompUpdate_Click(object sender, System.EventArgs e)
{
    Response.Redirect("UbratComponents.aspx");
}

private void btnMfgUpdate_Click(object sender, System.EventArgs e)
{
    Response.Redirect("UbratMfgDetails.aspx");
}

#endregion
private void btnSponsorUpdate_Click(object sender, System.EventArgs e)
{
    Response.Redirect("UbratSponsors.aspx");
}

private void btnMembersUpdate_Click(object sender, System.EventArgs e)
{
    Response.Redirect("UbratMembers.aspx");
}

private void btnUserUpdate_Click(object sender, System.EventArgs e)
{
    Response.Redirect("UbratUsers.aspx");
}

private void LinkButton1_Click(object sender, System.EventArgs e)
{
    Session["uid"] = "";
    Response.Redirect("bratInv.aspx");
}

RbratMaster.aspx.cs

using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;

namespace BratInventory
{
    /// <summary>
    /// Summary description for bratmaster.
    /// </summary>
    public class bratmaster : System.Web.UI.Page
    {
        protected System.Data.OleDb.OleDbDataAdapter masterAdapter;
        protected BratInventory.dsMaster dsMaster1;
        protected System.Web.UI.WebControls.DataGrid masterGrid;
        protected System.Web.UI.WebControls.Label Label1;
        protected System.Data.OleDb.OleDbConnection masterConn;
        protected System.Web.UI.WebControls.LinkButton LinkButton1;
        protected System.Data.OleDb.OleDbCommand oleDbSelectCommand1;
    }
}
private void Page_Load(object sender, System.EventArgs e) {
    masterAdapter.Fill(dsMaster1);
    if (!IsPostBack)
    {
        masterGrid.DataBind();
    }
    // Put user code to initialize the page here
}

#region Web Form Designer generated code
override protected void OnInit(EventArgs e)
{
    //
    // CODEGEN: This call is required by the ASP.NET Web
    // Form Designer.
    //
    InitializeComponent();
    base.OnInit(e);
}

///<summary>
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
///</summary>
private void InitializeComponent()
{
    this.masterAdapter = new System.Data.OleDb.OleDbDataAdapter();
    this.oleDbSelectCommand1 = new System.Data.OleDb.OleDbCommand();
    this.masterConn = new System.Data.OleDb.OleDbConnection();
    this.dsMaster1 = new BratInventory.dsMaster();
    ((System.ComponentModel.ISupportInitialize)(this.dsMaster1)).BeginInit();
    this.LinkButton1.Click += new System.EventHandler(this.LinkButton1_Click);

    this.masterAdapter.SelectCommand = this.oleDbSelectCommand1;
    this.masterAdapter.TableMappings.AddRange(new System.Data.Common.DataTableMapping[] {
        } }
new System.Data.Common.DataColumnMapping("system_name", "system_name"));

// oleDbSelectCommand1
this.oleDbSelectCommand1.CommandText = "SELECT system_id, system_name FROM master order by system_id";
this.oleDbSelectCommand1.Connection = this.masterConn;

// masterConn
this.masterConn.ConnectionString = @"Provider=Microsoft.Jet.OLEDB.4.0;Password=";User ID=Admin;Data Source=E:\ak\bratTest\BratInventory.mdb;Mode=ReadWrite|Share Deny None;Extended Properties=";Jet OLEDB:System database=";Jet OLEDB:Registry Path=";Jet OLEDB:Database Password=";Jet OLEDB:Engine Type=5;Jet OLEDB:Database Locking Mode=1;Jet OLEDB:Global Partial Bulk Ops=2;Jet OLEDB:Global Bulk Transactions=1;Jet OLEDB:New Database Password=";Jet OLEDB:Create System Database=False;Jet OLEDB:Encrypt Database=False;Jet OLEDB:Don't Copy Locale on Compact=False;Jet OLEDB:Compact Without Replica Repair=False;Jet OLEDB:SFP=False;"

// dsMaster1
this.dsMaster1.DataSetName = "dsMaster";
this.dsMaster1.Namespace = "http://www.tempuri.org/dsMaster.xsd";
this.Load += new System.EventHandler(this.Page_Load);

((System.ComponentModel.ISupportInitialize)(this.dsMaster1)).EndInit();

#endregion

deserialization and deserialization

private void LinkButton1_Click(object sender, System.EventArgs e)
{
    Response.Redirect("bratInv.aspx");
}
}
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;

namespace BratInventory
{
    /// <summary>
    /// Summary description for bratsubsys.
    /// </summary>
    public class bratsubsys : System.Web.UI.Page
    {
        protected System.Web.UI.WebControls.Label Label1;
        protected System.Data.OleDb.OleDbDataAdapter subAdapter;
        protected System.Data.OleDb.OleDbCommand oleDbSelectCommand1;
        protected System.Data.OleDb.OleDbConnection subConn;
        protected BratInventory.dsSub dsSub1;
        protected System.Web.UI.WebControls.LinkButton LinkButton1;
        protected System.Web.UI.WebControls.DataGrid subGrid;

        private void Page_Load(object sender, System.EventArgs e)
        {
            subAdapter.Fill(dsSub1);
            if (!IsPostBack)
            {
                subGrid.DataBind();
            }
            // Put user code to initialize the page here
        }

        #region Web Form Designer generated code
        override protected void OnInit(EventArgs e)
        {
            // CODEGEN: This call is required by the ASP.NET Web Form Designer.
            // InitializeComponent();
            base.OnInit(e);
        }
        /// <summary>
        /// Required method for Designer support - do not modify the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
        }
    }
}
this.subAdapter = new System.Data.OleDb.OleDbDataAdapter();
this.oleDbSelectCommand1 = new System.Data.OleDb.OleDbCommand();
this.subConn = new System.Data.OleDb.OleDbConnection();
this.dsSub1 = new BratInventory.dsSub();

((System.ComponentModel.ISupportInitialize)(this.dsSub1)).BeginInit();

// subAdapter
//
this.subAdapter.SelectCommand = this.oleDbSelectCommand1;
this.subAdapter.TableMappings.AddRange(new System.Data.Common.DataTableMapping[] {


new System.Data.Common.DataColumnMapping("sub_name", "sub_name"),


//
// oleDbSelectCommand1
//
this.oleDbSelectCommand1.CommandText = "SELECT sub_id, sub_name, system_id FROM subsystem order by sub_id";
this.oleDbSelectCommand1.Connection = this.subConn;

//
// subConn
//
this.subConn.ConnectionString = @"Provider=Microsoft.Jet.OLEDB.4.0;Password="";User ID=Admin;Data Source=E:\ak\bratTest\BratInventory.mdb;Mode=ReadWrite|Share Deny None;Extended Properties="";Jet OLEDB:System database="";Jet OLEDB:Registry Path="";Jet OLEDB:Database Password="";Jet OLEDB:Engine Type=5;Jet OLEDB:Database Locking Mode=1;Jet OLEDB:Global Partial Bulk Ops=2;Jet OLEDB:Global Bulk Transactions=1;Jet OLEDB:New Database Password="";Jet OLEDB:Create System Database=False;Jet OLEDB:Encrypt Database=False;Jet OLEDB:Don't Copy Locale on
namespace BratInventory
{
    public class bratcomponents : System.Web.UI.Page
    {
        protected System.Data.OleDb.OleDbDataAdapter componentAdapter;
        protected System.Web.UI.WebControls.Label Label1;
        protected System.Web.UI.WebControls.LinkButton LinkButton1;
        protected System.Data.OleDb.OleDbCommand oleDbSelectCommand1;
    }
}
protected System.Web.UI.WebControls.DataGrid componentGrid;

private void Page_Load(object sender, System.EventArgs e)
{
    componentAdapter.Fill(dsComponents1);
    if (!IsPostBack)
    {
        componentGrid.DataBind();
    }
    // Put user code to initialize the page here
}

#region Web Form Designer generated code
override protected void OnInit(EventArgs e)
{
    // CODEGEN: This call is required by the ASP.NET Web Form Designer.
    InitializeComponent();
    base.OnInit(e);
}

/// <summary>
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
    this.componentAdapter = new System.Data.OleDb.OleDbDataAdapter();
    this.oleDbSelectCommand1 = new System.Data.OleDb.OleDbCommand();
    this.mfgConn = new System.Data.OleDb.OleDbConnection();
    this.dsComponents1 = new BratInventory.dsComponents();
    ((System.ComponentModel.ISupportInitialize)( this.dsComponents1)).BeginInit();
    this.LinkButton1.Click += new System.EventHandler(this.LinkButton1_Click);
    // componentAdapter
    this.componentAdapter.SelectCommand = this.oleDbSelectCommand1;
    this.componentAdapter.TableMappings.AddRange( new System.Data.Common.DataTableMapping[]{
        new System.Data.Common.DataTableMapping("Table", "components",
            // more mappings
        }
    };
    this.componentAdapter.SelectCommand = this.oleDbSelectCommand1;
    this.componentAdapter.TableMappings.AddRange(new
    System.Data.Common.DataTableMapping[] { new
    System.Data.Common.DataTableMapping("Table", "components",
    new System.Data.Common DataColumnMapping[] { new
    System.Data.Common DataColumnMapping("Id", "int") })
    
}


new System.Data.Common.DataColumnMapping("num_units", "num_units"),


//
// oleDbSelectCommand1
//
this.oleDbSelectCommand1.CommandText = "SELECT component_id, comp_name, num_units, mfgr_id, sub_id FROM components order by component_id";
this.oleDbSelectCommand1.Connection = this.mfgConn;

//
// mfgConn
//
this.mfgConn.ConnectionString = @"Provider=Microsoft.Jet.OLEDB.4.0;Password=";User ID=Admin;Data Source=E:\ak\bratTest\BratInventory.mdb;Mode=ReadWrite\Share Deny None;Extended Properties=";Jet OLEDB:System database=";Jet OLEDB:Registry Path=";Jet OLEDB:Database Password=";Jet OLEDB:Engine Type=5;Jet OLEDB:Database Locking Mode=1;Jet OLEDB:Global Partial Bulk Ops=2;Jet OLEDB:Global Bulk Transactions=1;Jet OLEDB:New Database Password=";Jet OLEDB:Create System Database=False;Jet OLEDB:Encrypt Database=False;Jet OLEDB:Don't Copy Locale on Compact=False;Jet OLEDB:Compact Without Replica Repair=False;Jet OLEDB:SFP=False";

//
// dsComponents1
//
this.dsComponents1.DataSetName = "dsComponents";
this.dsComponents1.Namespace = "http://www.tempuri.org/dsComponents.xsd";
this.Load += new System.EventHandler(this.Page_Load);
((System.ComponentModel.Model.ISupportInitialize)(this.dsComponents1)).EndInit();

#endregion

private void LinkButton1_Click(object sender, System.EventArgs e)
{
    Response.Redirect("bratInv.aspx");
}

namespace BratInventory
{
    /// <summary>
    /// Summary description for bratMfgDetails.
    /// </summary>
    public class bratMfgDetails : System.Web.UI.Page
    {
        protected System.WebControls.Label Label1;
        protected System.Data.OleDb.OleDbDataAdapter mfgAdapter;
        protected System.Data.OleDb.OleDbCommand oleDbSelectCommand1;
        protected System.Data.OleDb.OleDbConnection mfgConn;
        protected BratInventory.dsManufacturer dsManufacturer1;
        protected System.WebControls.LinkButton LinkButton1;
        protected System.WebControls.DataGrid mfgGrid;

        private void Page_Load(object sender, System.EventArgs e)
        {
            mfgAdapter.Fill(dsManufacturer1);
            if (!IsPostBack)
            {
                mfgGrid.DataBind();
            }
            // Put user code to initialize the page here
        }
    }
}
#region Web Form Designer generated code

```csharp
override protected void OnInit(EventArgs e)
{
    //
    // CODEGEN: This call is required by the ASP.NET Web
    Form Designer.
    //
    InitializeComponent();
    base.OnInit(e);
}
```

/// <summary>
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
    this.mfgAdapter = new System.Data.OleDb.OleDbDataAdapter();
    this.oleDbSelectCommand1 = new System.Data.OleDb.OleDbCommand();
    this.mfgConn = new System.Data.OleDb.OleDbConnection();
    this.dsManufacturer1 = new BratInventory.dsManufacturer();
    ((System.ComponentModel.ISupportInitialize)(this.dsManufacturer1)).BeginInit();

    this.LinkButton1.Click += new System.EventHandler(this.LinkButton1_Click);
    // mfgAdapter
    //
    this.mfgAdapter.SelectCommand = this.oleDbSelectCommand1;
    this.mfgAdapter.TableMappings.AddRange(new System.Data.Common.DataTableMapping[] {


            new System.Data.Common.DataColumnMapping("manuf_name", "manuf_name"),

        }));

    });
```
new System.Data.Common.DataColumnMapping("other_info", "other_info");

// OleDbSelectCommand1
// this.oleDbSelectCommand1.CommandText = "SELECT manuf_id, manuf_name, other_info FROM mfg_details order by manuf_id"
// this.oleDbSelectCommand1.Connection = this.mfgConn;
//
// mfgConn
//
this.mfgConn.ConnectionString =
"Provider=Microsoft.Jet.OLEDB.4.0;Password=";User ID=Admin;Data Source=E:\ak\bratTest\BratInventory.mdb;Mode=ReadWrite|Share Deny None;Extended Properties=";Jet OLEDB:System database=";Jet OLEDB:Registry Path=";Jet OLEDB:Database Password=";Jet OLEDB:Engine Type=5;Jet OLEDB:Database Locking Mode=1;Jet OLEDB:Global Partial Bulk Ops=2;Jet OLEDB:Global Bulk Transactions=1;Jet OLEDB:New Database Password=";Jet OLEDB:Create System Database=False;Jet OLEDB:Encrypt Database=False;Jet OLEDB:Don't Copy Locale on Compact=False;Jet OLEDB:Compact Without Replica Repair=False;Jet OLEDB:SFP=False";

// dsManufacturer1
// this.dsManufacturer1.DataSetName = "dsManufacturer";
this.dsManufacturer1.Locale = new System.Globalization.CultureInfo("en-US");
this.dsManufacturer1.Namespace = "http://www.tempuri.org/dsManufacturer.xsd";
this.Load += new System.EventHandler(this.Page_Load);

((System.ComponentModel.ISupportInitialize)(this.dsManufacturer1)).EndInit();

private void LinkButton1_Click(object sender, System.EventArgs e)
{
    Response.Redirect("bratInv.aspx");
}

RbratMembers.aspx.cs
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;

namespace BratInventory
{
    /// <summary>
    /// Summary description for RbratMembers.
    /// </summary>
    public class RbratMembers : System.Web.UI.Page
    {
        protected System.Data.OleDb.OleDbDataAdapter memberAdapter;
        protected System.Web.UI.WebControls.Label Label1;
        protected System.Web.UI.WebControls.LinkButton LinkButton1;
        protected System.Data.OleDb.OleDbCommand oleDbSelectCommand1;
        protected System.Data.OleDb.OleDbConnection memberConn;
        protected BratInventory.dsMembers dsMembers1;
        protected System.Web.UI.WebControls.DataGrid RgridMembers;

        private void Page_Load(object sender, System.EventArgs e)
        {
            memberAdapter.Fill(dsMembers1);
            if (!IsPostBack)
            {
                RgridMembers.DataBind();
                // Put user code to initialize the page here
            }
        }

        #region Web Form Designer generated code
        override protected void OnInit(EventArgs e)
        {
            // CODEGEN: This call is required by the ASP.NET Web Form Designer.
            InitializeComponent();
            base.OnInit(e);
        }

        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        ///
        private void InitializeComponent()
        {
            this.memberAdapter = new System.Data.OleDb.OleDbDataAdapter();
            this.oleDbSelectCommand1 = new System.Data.OleDb.OleDbCommand();
            this.memberConn = new System.Data.OleDb.OleDbConnection();
            this.dsMembers1 = new BratInventory.dsMembers();
            ((System.ComponentModel.ISupportInitialize)(this.dsMembers1)).BeginInit();
            this/memberAdapter.Fill(dsMembers1);
            if (!IsPostBack)
            {
                RgridMembers.DataBind();
                // Put user code to initialize the page here
            }
        }
    }
}
this.LinkButton1.Click += new System.EventHandler(this.LinkButton1_Click);

//
// memberAdapter
//
this.memberAdapter.SelectCommand =
this.oleDbSelectCommand1;
this.memberAdapter.TableMappings.AddRange(new System.Data.Common.DataTableMapping[] { 


        new System.Data.Common.DataColumnMapping("m_lname", "m_lname"),

        new System.Data.Common.DataColumnMapping("m_address", "m_address"),

        new System.Data.Common.DataColumnMapping("m_city", "m_city"),

        new System.Data.Common.DataColumnMapping("m_state", "m_state"),


    })});
new
System.Data.Common.DataColumnMapping("m_phone", "m_phone"),

new
System.Data.Common.DataColumnMapping("m_email", "m_email"));

//
// oleDbSelectCommand1
//
this.oleDbSelectCommand1.CommandText = "SELECT member_id, m_fname, m_lname, m_address, m_city, m_state, m_zip, m_phone, m_email FROM members";
this.oleDbSelectCommand1.Connection = this.memberConn;

//
// memberConn
//
this.memberConn.ConnectionString = @"Provider=Microsoft.Jet.OLEDB.4.0;Password="""";User ID=Admin;Data Source=E:\ak\bratTest\BratInventory.mdb;Mode=ReadWrite|Share Deny None;Extended Properties="""";Jet OLEDB:System database="""";Jet OLEDB:Registry Path="""";Jet OLEDB:Database Password="""";Jet OLEDB:Engine Type=5;Jet OLEDB:Database Locking Mode=1;Jet OLEDB:Global Partial Bulk Ops=2;Jet OLEDB:Global Bulk Transactions=1;Jet OLEDB:New Database Password="""";Jet OLEDB:Create System Database=False;Jet OLEDB:Encrypt Database=False;Jet OLEDB:Don't Copy Locale on Compact=False;Jet OLEDB:Compact Without Replica Repair=False;Jet OLEDB:SFP=False";"

//
// dsMembers1
//
this.dsMembers1.DataSetName = "dsMembers";
this.dsMembers1.Locale = new System.Globalization.CultureInfo("en-US");
this.dsMembers1.Namespace = "http://www.tempuri.org/dsMembers.xsd";
this.Load += new System.EventHandler(this.Page_Load);

{(System.ComponentModel.Model.ISupportInitialize)(this.dsMembers1)).EndInit();

}
@endregion

private void LinkButton1_Click(object sender,
System.EventArgs e)
{
    Response.Redirect("bratInv.aspx");
}
}
namespace BratInventory
{
    /// <summary>
    /// Summary description for bratsubsys.
    /// </summary>
    public class bratsubsys : System.Web.UI.Page
    {
        protected System.Web.UI.WebControls.Label Label1;
        protected System.Data.OleDb.OleDbDataAdapter subAdapter;
        protected System.Data.OleDb.OleDbCommand oleDbSelectCommand1;
        protected System.Data.OleDb.OleDbConnection subConn;
        protected BratInventory.dsSub dsSub1;
        protected System.Web.UI.WebControls.LinkButton LinkButton1;
        protected System.Web.UI.WebControls.DataGrid subGrid;

        private void Page_Load(object sender, System.EventArgs e)
        {
            subAdapter.Fill(dsSub1);
            if (!IsPostBack)
            {
                subGrid.DataBind();
            }
            // Put user code to initialize the page here
        }

        #region Web Form Designer generated code
        override protected void OnInit(EventArgs e)
        {
            // CODEGEN: This call is required by the ASP.NET Web Form Designer.
            InitializeComponent();
            base.OnInit(e);
        }
        #endregion

        private void InitializeComponent()
        {
            // Required method for Designer support - do not modify
            // the contents of this method with the code editor.
            // CODEGEN: This call is required by the ASP.NET Web Form Designer.
            InitializeComponent();
            base.OnInit(e);
        }
    }
}
this.subAdapter = new System.Data.OleDb.OleDbDataAdapter();
this.oleDbSelectCommand1 = new System.Data.OleDb.OleDbCommand();
this.subConn = new System.Data.OleDb.OleDbConnection();
this.dsSub1 = new BratInventory.dsSub();
(System.ComponentModel.ISupportInitialize)(this.dsSub1)).BeginInit();

// subAdapter
// this.subAdapter.SelectCommand =
this.oleDbSelectCommand1;
this.subAdapter.TableMappings.AddRange(new System.Data.Common.DataTableMapping[] {


new System.Data.Common DataColumnMapping("sub_id", "sub_id"),

new System.Data.Common DataColumnMapping("sub_name", "sub_name"),

new System.Data.Common DataColumnMapping("system_id", "system_id")});
// oleDbSelectCommand1
// this.oleDbSelectCommand1.CommandText = "SELECT sub_id, sub_name, system_id FROM subsystem order by sub_id";
this.oleDbSelectCommand1.Connection = this.subConn;
//
// subConn
// this.subConn.ConnectionString =
"Provider=Microsoft.Jet.OLEDB.4.0;Password=";User ID=Admin;Data Source=E:\ak\bratTest\BratInventory.mdb;Mode=ReadWrite|Share Deny None;Extended Properties=";Jet OLEDB:System database=";Jet OLEDB:Registry Path=";Jet OLEDB:Database Password=";Jet OLEDB:Engine Type=5;Jet OLEDB:Database Locking Mode=1;Jet OLEDB:Global Partial Bulk Ops=2;Jet OLEDB:Global Bulk Transactions=1;Jet OLEDB:New Database Password=";Jet OLEDB:Create System Database=False;Jet OLEDB:Encrypt Database=False;Jet OLEDB:Don't Copy Locale on
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;
namespace BratInventory
{
    /// <summary>
    /// Summary description for UbratMaster.
    /// </summary>
    public class UbratMaster : System.Web.UI.Page
    {
        private OleDbDataAdapter master_uadp;
        private DataSet master_uds;
        protected System.Web.UI.WebControls.DataGrid UgridMaster;
        protected System.Web.UI.WebControls.Button btnMasterUpdate;
        protected System.Web.UI.WebControls.TextBox txtMaster_nameAdd;
        //
        // dsSub1
        //
        this.dsSub1.DataSetName = "dsSub";
        this.dsSub1.Locale = new System.Globalization.CultureInfo("en-US");
        this.dsSub1.Namespace = "http://www.tempuri.org/dsSub.xsd";
        this.LinkButton1.Click += new System.EventHandler(this.LinkButton1_Click);
        this.Load += new System.EventHandler(this.Page_Load);
        }((System.ComponentModel.ISupportInitialize)(this.dsSub1)).EndInit();
} endregion

private void LinkButton1_Click(object sender, System.EventArgs e)
{
    Response.Redirect("bratInv.aspx");
}
}

UbratMaster.aspx.cs
protected System.Web.UI.WebControls.TextBox txtMaster_IdAdd;
protected System.Web.UI.WebControls.TextBox txtMaster_Update;
protected System.Web.UI.WebControls.Label Label1;
protected System.Web.UI.WebControls.Label Label2;
protected System.Web.UI.WebControls.Label Label3;
protected System.Web.UI.WebControls.Label Label19;
protected System.Web.UI.WebControls.Button btnMasterAdd;
protected System.Web.UI.WebControls.Label Label5;
protected System.Web.UI.WebControls.LinkButton LinkButton1;
protected System.Web.UI.WebControls.LinkButton LinkButton2;
protected System.Web.UI.WebControls.Label Label4;

public UbratMaster()
{
    Page.Init += new System.EventHandler(Page_Init);
}

private void Page_Load(object sender, System.EventArgs e)
{
    if ((Session["uid"].ToString()) != "")
    {
        if (!IsPostBack)
        {
            try
            {

                OleDbConnection conn = new OleDbConnection(@"Provider=Microsoft.JET.OLEDB.4.0;" + @"DataSource=E:\ak\bratTest\BratInventory.mdb;Jet OLEDB:Engine Type=5;" + @"Jet OLEDB:Database Password=abc");
                OleDbConnection master_uconn = new OleDbConnection(@"Provider=Microsoft.JET.OLEDB.4.0;Password="";UserID=Admin;Data Source=E:\ak\bratTest\BratInventory.mdb;Mode=ReadWrite;Extended Properties="";Jet OLEDB:System database="";Jet OLEDB:Registry Path="";Jet OLEDB:Database Password="";Jet OLEDB:Engine Type=5;Jet OLEDB:Database Locking Mode=1;Jet OLEDB:Global Partial Bulk Ops=2;Jet OLEDB:Global Bulk Transactions=1;Jet OLEDB:New Database Password="";Jet OLEDB:Create System Database=False;Jet OLEDB:Encrypt Database=False;Jet OLEDB:Don't Copy Locale on Compact=False;Jet OLEDB:Compact Without Replica Repair=False;Jet OLEDB:SFP=False");
                master_uadp = new OleDbDataAdapter();
                OleDbCommand selectCmd = new OleDbCommand("Select * FROM master", master_uconn);
                master_uadp.SelectCommand = selectCmd;
                OleDbCommand updateCmd = new OleDbCommand("UPDATE master SET system_name = @sysname " + " WHERE system_id = @sysid", master_uconn);
            }
        }
    }
}
updateCmd.Parameters.Add("@sysname",OleDbType.VarChar,10);
updateCmd.Parameters["@sysname"].SourceColumn = "system_name";

updateCmd.Parameters.Add("@sysid",OleDbType.VarChar,10);
updateCmd.Parameters["@sysid"].SourceColumn = "system_id";
master_uadp.UpdateCommand = updateCmd;

OleDbCommand deleteCmd = new OleDbCommand("DELETE FROM master WHERE " +
"system_id=@sysid",master_uconn);
deleteCmd.Parameters.Add("@sysid",OleDbType.VarChar);
deleteCmd.Parameters["@sysid"].SourceColumn = "system_id";
master_uadp.DeleteCommand = deleteCmd;

OleDbCommand insertCmd = new OleDbCommand("INSERT INTO master (system_id, system_name)" +
"VALUES (@sysid, @sysname)", master_uconn);
insertCmd.Parameters.Add("@sysid",OleDbType.VarChar, 10);
insertCmd.Parameters["@sysid"].SourceColumn = "system_id";
insertCmd.Parameters.Add("@sysname",OleDbType.VarChar, 10);
insertCmd.Parameters["@sysname"].SourceColumn = "system_name";
master_uadp.InsertCommand = insertCmd;

master_uds = new DataSet();
master_uadp.Fill(master_uds,"master");
UgridMaster.DataSource = master_uds.Tables["master"].DefaultView;
UgridMaster.DataBind();
Session.Add("master_uadp", master_uadp);
Session.Add("master_uds", master_uds);

} catch (OleDbException OleDbEx)
Response.Write("OleDbException: "+
OleDbEx.ToString());
}
catch (Exception ex)
{
    Response.Write("Exception:
"+ex.ToString());
}
else
{
    master_uds =
(DataSet)Session["master_uds"];
    master_uadp =
(OleDbDataAdapter)Session["master_uadp"];
}
else
{
    Response.Redirect("bratInv.aspx");
}

private void Page_Init(object sender, EventArgs e)
{
    InitializeComponent();
}

#region Web Form Designer generated code
// override protected void OnInit(EventArgs e)
// {
//     // CODEGEN: This call is required by the ASP.NET Web
//     // Form Designer.
//     //
//     InitializeComponent();
//     //
//     base.OnInit(e);
// }

    // <summary>
    // Required method for Designer support - do not modify
    // the contents of this method with the code editor.
    // </summary>

    //endregion

    private void InitializeComponent()
    {
        this.UgridMaster.ItemCommand += new
System.Web.UI.WebControls.DataGridCommandEventHandler(this.OnItemClick);
        this.btnMasterUpdate.Click += new
System.EventHandler(this.btnMasterUpdate_Click);
        this.btnMasterAdd.Click += new
System.EventHandler(this.btnMasterAdd_Click);
    }
this.LinkButton1.Click += new System.EventHandler(this.LinkButton1_Click);
this.LinkButton2.Click += new System.EventHandler(this.LinkButton2_Click);
this.Load += new System.EventHandler(this.Page_Load);

protected void OnItemClicked(object source, DataGridCommandEventArgs e)
{
    int itemindex = (int)e.Item.ItemIndex;
    if (e.CommandName == "Select")
    {
        ViewState["SelectedItemIndex"] = itemindex;
        DataRow selectedRow = master_uds.Tables["master"].Rows[itemindex];
        txtMaster_Update.Text = selectedRow["system_name"].ToString();

        txtMaster_Update.Enabled = true;
        btnMasterUpdate.Enabled = true;
    }
    else
    {
        try
        {
            master_uds.Tables["master"].Rows[itemindex].Delete();
            master_uadp.Update(master_uds, "master");
            Session["master_uds"] = master_uds;
            UgridMaster.DataSource = master_uds.Tables["master"].DefaultView;
            UgridMaster.DataBind();
        }
        catch (OleDbException OleDbEx)
        {
            Response.Write("OleDbException: " + OleDbEx.ToString());
        }
        catch (Exception ex)
        {
            Response.Write("Exception: " + ex.ToString());
        }
    }
}

private void btnMasterAdd_Click(object sender, System.EventArgs e)
{
    try
    {

DataRow newRow =
master_uds.Tables["master"].NewRow();

newRow["system_id"] = txtMaster_IdAdd.Text;
newRow["system_name"] = txtMaster_nameAdd.Text;

master_uds.Tables["master"].Rows.Add(newRow);
master_uadp.Update(master_uds, "master");

master_uds.Clear();
master_uadp.Fill(master_uds, "master");

Session["master_uds"] = master_uds;

UgridMaster.DataSource =
master_uds.Tables["master"].DefaultView;
UgridMaster.DataBind();

txtMaster_IdAdd.Text = "";
txtMaster_nameAdd.Text = "";
}
catch (OleDbException OleDbEx)
{
    Response.Write("OleDbException: "+
OleDbEx.ToString());
}
catch (Exception ex)
{
    Response.Write("Exception: "+ ex.ToString());
}

private void btnMasterUpdate_Click(object sender,
System.EventArgs e)
{
    try
    {
        try
        {
            DataRow selectedRow =
master_uds.Tables["master"].Rows[(int)(ViewState["SelectedItemIndex"])];

            selectedRow["system_name"] =
txtMaster_Update.Text;

            master_uadp.Update(master_uds, "master");
            Session["master_uds"] = master_uds;

            UgridMaster.DataSource =
master_uds.Tables["master"].DefaultView;
            UgridMaster.DataBind();

            txtMaster_Update.Text = "";
            txtMaster_Update.Enabled = false;
            btnMasterUpdate.Enabled = false;
        }
    }
catch (OleDbException OleDbEx)
public class UbratSubsys : System.Web.UI.Page
{
    private OleDbDataAdapter sub_uadp;
    private DataSet sub_uds;
    protected System.Web.UI.WebControls.TextBox txtMasterIdAdd;
    protected System.Web.UI.WebControls.TextBox txtSubSysIdAdd;
    protected System.Web.UI.WebControls.TextBox txtSubSysNameAdd;
    protected System.Web.UI.WebControls.TextBox txtUpdate_SubSysName;
    protected System.Web.UI.WebControls.Button btnSubSys_Add;
    // ...
}
protected System.Web.UI.WebControls.Label Label1;
protected System.Web.UI.WebControls.Label Label2;
protected System.Web.UI.WebControls.Label Label4;
protected System.Web.UI.WebControls.Label Label5;
protected System.Web.UI.WebControls.Label Label6;
protected System.Web.UI.WebControls.Label Label7;
protected System.Web.UI.WebControls.TextBox txtUpdate_sysId;
protected System.Web.UI.WebControls.Label Label8;
protected System.Web.UI.WebControls.LinkButton LinkButton1;
protected System.Web.UI.WebControls.LinkButton LinkButton2;
protected System.Web.UI.WebControls.DataGrid UgridSubsys;

public UbratSubsys()
{
    Page.Init += new System.EventHandler(Page_Init);
}

private void Page_Load(object sender, System.EventArgs e)
{
    if ((Session["uid"]).ToString()) !="")
    {
        if (!IsPostBack)
        {
            OleDbConnection sub_conn = new OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Password="";User ID=Admin;Data Source=E:\ak\bratTest\BratInventory.mdb;Mode=ReadWrite;Extended Properties="";Jet OLEDB:System database="";Jet OLEDB:Registry Path="";Jet OLEDB:Database Password="";Jet OLEDB:Engine Type=5;Jet OLEDB:Database Locking Mode=1;Jet OLEDB:Global Partial Bulk Ops=2;Jet OLEDB:Global Bulk Transactions=1;Jet OLEDB:New Database Password="";Jet OLEDB:Create System Database=False;Jet OLEDB:Encrypt Database=False;Jet OLEDB:Don't Copy Locale on Compact=False;Jet OLEDB:Compact Without Replica Repair=False;Jet OLEDB:SFP=False");
            OleDbDataAdapter sub_uadp = new OleDbDataAdapter();
            OleDbCommand selectCmd = new OleDbCommand("Select * FROM subsystem", sub_conn);
            sub_uadp.SelectCommand = selectCmd;
            OleDbCommand updateCmd = new OleDbCommand("UPDATE subsystem SET system_id = @sysid, " + "sub_name = @subname WHERE sub_id = @subid", sub_conn);
            OleDbCommand cmd = new OleDbCommand("Update subsystem SET system_id = @sysid, " + "sub_name = @subname WHERE sub_id = @subid", sub_conn);
            OleDbCommand.Parameters.Add("@sysid", OleDbType.VarChar, 20);
            OleDbCommand.Parameters["@subname"].SourceColumn = "sub_name";
        }
    }
}
updateCmd.Parameters.Add("@sysid", OleDbType.VarChar, 10);
updateCmd.Parameters["@sysid"].SourceColumn = "system_id";

updateCmd.Parameters.Add("@subid", OleDbType.VarChar, 10);
updateCmd.Parameters["@subid"].SourceColumn = "sub_id";
sub_uadp.UpdateCommand = updateCmd;

OleDbCommand deleteCmd = new OleDbCommand("DELETE FROM subsystem WHERE " +
"sub_id=@subid", sub_conn);

deleteCmd.Parameters.Add("@subid", OleDbType.VarChar);
deleteCmd.Parameters["@subid"].SourceColumn = "sub_id";
sub_uadp.DeleteCommand = deleteCmd;

OleDbCommand insertCmd = new OleDbCommand("INSERT INTO subsystem (sub_id, sub_name, system_id)" +
"VALUES (@subid, @subname, @sysid)", sub_conn);

insertCmd.Parameters.Add("@subid", OleDbType.VarChar, 10);
insertCmd.Parameters["@subid"].SourceColumn = "sub_id";
insertCmd.Parameters.Add("@subname", OleDbType.VarChar, 10);
insertCmd.Parameters["@subname"].SourceColumn = "sub_name";
insertCmd.Parameters.Add("@sysid", OleDbType.VarChar, 10);
insertCmd.Parameters["@sysid"].SourceColumn = "system_id";
sub_uadp.InsertCommand = insertCmd;

sub_uds = new DataSet();
sub_uadp.Fill(sub_uds, "subsystem");

UgridSubsys.DataSource = sub_uds.Tables["subsystem"].DefaultView;
UgridSubsys.DataBind();

Session.Add("sub_uadp", sub_uadp);
Session.Add("sub_uds", sub_uds);
}

catch (OleDbException OleDbEx)
Response.Write("OleDbException: "+
OleDbEx.ToString());
    }
    catch (Exception ex)
    {
        Response.Write("Exception:
"+ex.ToString());
    }
else
{
    sub_uds = (DataSet)Session["sub_uds"]; 
    sub_uadp = (OleDbDataAdapter)Session["sub_uadp"]; 
}
else
{
    Response.Redirect("bratInv.aspx");
}
// Put user code to initialize the page here
private void Page_Init(object sender, EventArgs e)
{
    InitializeComponent();
}
#region Web Form Designer generated code
/// <summary>
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
    this.UgridSubsys.ItemCommand += new System.Web.UI.WebControls.DataGridCommandEventHandler(this.OnItemClick);
    this.btnSubSys_Add.Click += new System.EventHandler(this.btnSubSys_Add_Click);
    this.btnSubsys_Update.Click += new System.EventHandler(this.btnSubsys_Update_Click);
    this.LinkButton1.Click += new System.EventHandler(this.LinkButton1_Click);
    this.LinkButton2.Click += new System.EventHandler(this.LinkButton2_Click);
    this.Load += new System.EventHandler(this.Page_Load);
}
protected void OnItemClicked(object source, DataGridCommandEventArgs e)
{
    int itemindex = (int)e.Item.ItemIndex;

    if (e.CommandName == "Select")
    {
        ViewState["SelectedItemIndex"] = itemindex;
        DataRow selectedRow = sub_uds.Tables["subsystem"].Rows[itemindex];
        txtUpdate_sysId.Text = selectedRow["system_id"].ToString();
        txtUpdate_SubSysName.Text = selectedRow["sub_name"].ToString();

        txtUpdate_sysId.Enabled = true;
        txtUpdate_SubSysName.Enabled = true;
        btnSubsys_Update.Enabled = true;
    }
    else
    {
        try
        {
            sub_uds.Tables["subsystem"].Rows[itemindex].Delete();

            sub_uadp.Update(sub_uds, "subsystem");
            Session["sub_uds"] = sub_uds;
            UgridSubsys.DataSource = sub_uds.Tables["subsystem"].DefaultView;
            UgridSubsys.DataBind();
        }
        catch (OleDbException OleDbEx)
        {
            Response.Write("OleDbException: " + OleDbEx.ToString());
        }
        catch (Exception ex)
        {
            Response.Write("Exception: " + ex.ToString());
        }
    }
}

private void btnSubSys_Add_Click(object sender, System.EventArgs e)
{
    try
    {
DataRow newRow = sub_uds.Tables["subsystem"].NewRow();

newRow["sub_id"] = txtSubSysIdAdd.Text;
newRow["sub_name"] = txtSubSysNameAdd.Text;
newRow["system_id"] = txtMasterIdAdd.Text;

sub_uds.Tables["subsystem"].Rows.Add(newRow);
sub_uadp.Update(sub_uds, "subsystem");
sub_uds.Clear();
sub_uadp.Fill(sub_uds, "subsystem");

Session["sub_uds"] = sub_uds;
UgridSubsys.DataSource = sub_uds.Tables["subsystem"].DefaultView;
UgridSubsys.DataBind();
txtSubSysIdAdd.Text = ";
txtSubSysNameAdd.Text = ";
txtMasterIdAdd.Text = "
}

} catch(OleDbException OleDbEx)
{
    Response.Write("OleDbException: "+ OleDbEx.ToString());
}

} catch(Exception ex)
{
    Response.Write("Exception: "+ ex.ToString());
}

private void btnSubsys_Update_Click(object sender, System.EventArgs e)
{
    try
    {
        DataRow selectedRow = sub_uds.Tables["subsystem"].Rows[(int)(ViewState["SelectedItemIndex"])];
        selectedRow["sub_name"] = txtUpdate_SubSysName.Text;
        selectedRow["system_id"] = txtUpdate_sysId.Text;
        sub_uadp.Update(sub_uds, "subsystem");
        Session["sub_uds"] = sub_uds;
        UgridSubsys.DataSource = sub_uds.Tables["subsystem"].DefaultView;
        UgridSubsys.DataBind();
        txtUpdate_SubSysName.Text = "";
txtUpdate_SubSysName.Enabled = false;
txtUpdate_sysId.Text = "";
txtUpdate_sysId.Enabled = false;
btnSubsys_Update.Enabled = false;
}
catch (OleDbException OleDbEx)
{
    Response.Write("OleDbException: "+
OleDbEx.ToString());
}
catch (Exception ex)
{
    Response.Write("Exception: "+ ex.ToString());
}

private void LinkButton1_Click(object sender,
System.EventArgs e)
{
    Session["uid"] = "";
    Response.Redirect("bratInv.aspx");
}

private void LinkButton2_Click(object sender,
System.EventArgs e)
{
    Response.Redirect("bratInvUpdate.aspx");

}

UbratComponents.aspx.cs

using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;

namespace BratInventory
{
    /// <summary>
    /// Summary description for UbratComponents.
    /// </summary>
    public class UbratComponents : System.Web.UI.Page
    {
        private DataSet comp_uds;
        private OleDbDataAdapter comp_uadp;
        protected System.Web.UI.WebControls.DataGrid
        UgridComponents;
protected System.Web.UI.WebControls.Label Label1;
protected System.Web.UI.WebControls.Label Label2;
protected System.Web.UI.WebControls.Label Label3;
protected System.Web.UI.WebControls.Label Label4;
protected System.Web.UI.WebControls.Label Label5;
protected System.Web.UI.WebControls.Label Label6;
protected System.Web.UI.WebControls.TextBox txtComp_NameAdd;
protected System.Web.UI.WebControls.TextBox txtComp_unitsAdd;
protected System.Web.UI.WebControls.Label Label7;
protected System.Web.UI.WebControls.Button btnCompAdd;
protected System.Web.UI.WebControls.TextBox txtComp_subsysIdAdd;
protected System.Web.UI.WebControls.TextBox txtComp_mfAdd;
protected System.Web.UI.WebControls.TextBox txtComp_nameUpdate;
protected System.Web.UI.WebControls.TextBox txtComp_mfUpdate;
protected System.Web.UI.WebControls.Label Label8;
protected System.Web.UI.WebControls.Label Label9;
protected System.Web.UI.WebControls.TextBox txtComp_subIdUpdate;
protected System.Web.UI.WebControls.TextBox txtComp_unitUpdate;
protected System.Web.UI.WebControls.Label Label10;
protected System.Web.UI.WebControls.Label Label11;
protected System.Web.UI.WebControls.Button btnCompUpdate;
protected System.Web.UI.WebControls.Label Label19;
protected System.Web.UI.WebControls.LinkButton LinkButton1;
protected System.Web.UI.WebControls.LinkButton LinkButton2;

public UbratComponents()
{
    Page.Init += new System.EventHandler(Page_Init);
}

private void Page_Load(object sender, System.EventArgs e)
{
    if ((Session["uid"]).ToString()) != "")
    {
        if (!IsPostBack)
        {
            try
            {
                //
                OleDbConnection conn = new OleDbConnection("Provider=Microsoft.JET.OLEDB.4.0;"+"DataSource=\ak\bratTest\BratInventory.mdb;Jet OLEDB:Engine Type=5;"+"Jet OLEDB:Database Password=abc");
                OleDbConnection ucomp_conn = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Password=";User ID=admin;Data Source=\ak\bratTest\BratInventory.mdb;Mode=ReadWrite;Extended Properties=";Jet OLEDB:System database=";Jet OLEDB:Registry Path=";Jet OLEDB:Database Password=";Jet OLEDB:Engine Type=5;Jet
OLEDB:Database Locking Mode=1;Jet OLEDB:Global Partial Bulk Ops=2;Jet OLEDB:Global Bulk Transactions=1;Jet OLEDB:New Database Password="";Jet OLEDB:Create System Database=False;Jet OLEDB:Encrypt Database=False;Jet OLEDB:Don't Copy Locale on Compact=False;Jet OLEDB:Compact Without Replica Repair=False;Jet OLEDB:SFP=False");

comp_uadp = new OleDbDataAdapter();

OleDbCommand selectCmd = new OleDbCommand("Select * FROM components order by component_id", ucomp_conn);

comp_uadp.SelectCommand = selectCmd;

OleDbCommand updateCmd = new OleDbCommand("UPDATE components SET comp_name = @compname, num_units = @units, mfgr_id = @mfgid," + " sub_id = @subid WHERE component_id = @compid", ucomp_conn);

updateCmd.Parameters.Add("@compname", OleDbDbType.VarChar,10);
updateCmd.Parameters["@compname"].SourceColumn = "comp_name";

updateCmd.Parameters.Add("@units", OleDbDbType.Numeric,3);
updateCmd.Parameters["@units"].SourceColumn = "num_units";

updateCmd.Parameters.Add("@mfgid", OleDbDbType.VarChar,10);
updateCmd.Parameters["@mfgid"].SourceColumn = "mfgr_id";

updateCmd.Parameters.Add("@subid", OleDbDbType.VarChar,10);
updateCmd.Parameters["@subid"].SourceColumn = "sub_id";

updateCmd.Parameters.Add("@compid", OleDbDbType.Numeric,10);
updateCmd.Parameters["@compid"].SourceColumn = "component_id";

comp_uadp.UpdateCommand = updateCmd;

OleDbCommand deleteCmd = new OleDbCommand("DELETE FROM components WHERE " + "component_id=@compid", ucomp_conn);
deleteCmd.Parameters.Add("@compid",OleDbType.VarChar);
deleteCmd.Parameters["@compid"].SourceColumn = "component_id";
comp_uadp.DeleteCommand =
deleteCmd;
OleDbCommand insertCmd = new OleDbCommand("INSERT INTO components (component_id, comp_name, num_units, mfgr_id, sub_id)" + "VALUES (@compid, @compname, @units, @mfgid, @subid)", ucomp_conn);
insertCmd Parameters.Add("@compid",OleDbType.VarChar, 10);
insertCmd.Parameters["@compid"].SourceColumn = "component_id";
insertCmd.Parameters.Add("@compname",OleDbType.VarChar, 10);
insertCmd.Parameters["@compname"].SourceColumn = "comp_name";
insertCmd.Parameters.Add("@units",OleDbType.Numeric, 3);
insertCmd.Parameters["@units"].SourceColumn = "num_units";
insertCmd.Parameters.Add("@mfgid",OleDbType.Numeric, 20);
insertCmd.Parameters["@mfgid"].SourceColumn = "mfgr_id";
insertCmd.Parameters.Add("@subid",OleDbType.VarChar, 10);
insertCmd.Parameters["@subid"].SourceColumn = "sub_id";
comp_uadp.InsertCommand =
insertCmd;
comp_uds = new DataSet();
comp_uadp.Fill(comp_uds, "components");
UgridComponents.DataSource = comp_uds.Tables["components"].DefaultView;
UgridComponents.DataBind();
Session.Add("comp_uadp",
comp_uadp);
Session.Add("comp_uds", comp_uds);
}
catch (OleDbException OleDbEx)
{
    Response.Write("OleDbException: " + OleDbEx.ToString());
}
catch (Exception ex)
Response.Write("Exception:");
        + ex.ToString();
    }
    else
    {
        comp_uds = (DataSet)Session["comp_uds"];  
        comp_uadp = (OleDbDataAdapter)Session["comp_uadp"];  
    }
    else
    {
        Response.Redirect("bratInv.aspx");
    }

    // Put user code to initialize the page here
}
private void Page_Init(object sender, EventArgs e)
{
    InitializeComponent();
}

#region  Web Form Designer generated code
//  override protected void OnInit(EventArgs e)
//  {
//    //
//    //  CODEGEN: This call is required by the ASP.NET Web
//    //  Form Designer.
//    //
//    InitializeComponent();
//    base.OnInit(e);
//  }
/// <summary>
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>
#endregion
private void InitializeComponent()
{
    this.UgridComponents.ItemCommand += new System.Web.UI.WebControls.DataGridCommandEventHandler(this.OnItemClick);
    this.btnCompUpdate.Click += new System.EventHandler(this.btnCompUpdate_Click);
    this.btnCompAdd.Click += new System.EventHandler(this.btnCompAdd_Click);
    this.LinkButton1.Click += new System.EventHandler(this.LinkButton1_Click);
    this.LinkButton2.Click += new System.EventHandler(this.LinkButton2_Click);
    this.Load += new System.EventHandler(this.Page_Load);
}
protected void OnItemClicked(object source, DataGridCommandEventArgs e)
{
    int itemindex = (int)e.Item.ItemIndex;

    if (e.CommandName == "Select")
    {
        ViewState["SelectedItemIndex"] = itemindex;
        DataRow selectedRow =
            comp_uds.Tables["components"].Rows[itemindex];
        txtComp_nameUpdate.Text =
            selectedRow["comp_name"].ToString();
        txtComp_mfUpdate.Text =
            selectedRow["mfgr_id"].ToString();
        txtComp_subIdUpdate.Text =
            selectedRow["sub_id"].ToString();
        txtComp_unitUpdate.Text =
            selectedRow["num_units"].ToString();

        txtComp_nameUpdate.Enabled = true;
        txtComp_mfUpdate.Enabled = true;
        txtComp_subIdUpdate.Enabled = true;
        txtComp_unitUpdate.Enabled = true;
        btnCompUpdate.Enabled = true;
    }
    else
    {
        try
        {
            comp_uds.Tables["components"].Rows[itemindex].Delete();
            comp_uadp.Update(comp_uds, "components");
            Session["comp_uds"] = comp_uds;
            UgridComponents.DataSource =
                comp_uds.Tables["components"].DefaultView;
            UgridComponents.DataBind();
        }
        catch (OleDbException OleDbEx)
        {
            Response.Write("OleDbException: " +
                OleDbEx.ToString());
        }
        catch (Exception ex)
        {
            Response.Write("Exception: " +
                ex.ToString());
        }
    }
}
private void btnCompAdd_Click(object sender, System.EventArgs e)
{
    try
    {
        DataRow newRow = comp_uds.Tables["components"].NewRow();
        newRow["component_id"] = txtComp_IdAdd.Text;
        newRow["comp_name"] = txtComp_NameAdd.Text;
        newRow["num_units"] = txtComp_unitsAdd.Text;
        newRow["mfgr_id"] = txtComp_mfAdd.Text;
        newRow["sub_id"] = txtComp_subsysIdAdd.Text;

        comp_uds.Tables["components"].Rows.Add(newRow);
        comp_uadp.Update(comp_uds, "components");
        comp_uds.Clear();
        comp_uadp.Fill(comp_uds, "components");
        Session["comp_uds"] = comp_uds;
        UgridComponents.DataSource = comp_uds.Tables["components"].DefaultView;
        UgridComponents.DataBind();
        txtComp_IdAdd.Text = ";
        txtComp_NameAdd.Text = ";
        txtComp_unitsAdd.Text = ";
        txtComp_mfAdd.Text = ";
        txtComp_subsysIdAdd.Text = ";
    }
    catch (OleDbException OleDbEx)
    {
        Response.Write("OleDbException: " + OleDbEx.ToString());
    }
    catch (Exception ex)
    {
        Response.Write("Exception: " + ex.ToString());
    }
}

private void btnCompUpdate_Click(object sender, System.EventArgs e)
{
    try
    {
        DataRow selectedRow = comp_uds.Tables["components"].Rows[(int)(ViewState["SelectedItemIndex"])];

        selectedRow["comp_name"] = txtComp_nameUpdate.Text;
        selectedRow["num_units"] = txtComp_unitUpdate.Text;
        selectedRow["mfgr_id"] = txtComp_mfUpdate.Text;
        selectedRow["sub_id"] = txtComp_subsysIdUpdate.Text;
    }
    catch (OleDbException OleDbEx)
    {
        Response.Write("OleDbException: " + OleDbEx.ToString());
    }
    catch (Exception ex)
    {
        Response.Write("Exception: " + ex.ToString());
    }
}
selectedRow["mfgr_id"] = txtComp_mfUpdate.Text;
selectedRow["sub_id"] = txtComp_subIdUpdate.Text;

//comp_uadp.Update(comp_uds, "components");
comp_uadp.Update(comp_uds,"components");
Session["comp_uds"] = comp_uds;

UgridComponents.DataSource = comp_uds.Tables["components"].DefaultView;
UgridComponents.DataBind();

txtComp_nameUpdate.Text = "";
txtComp_nameUpdate.Enabled = false;
txtComp_unitUpdate.Text = "";
txtComp_unitUpdate.Enabled = false;
txtComp_mfUpdate.Text = "";
txtComp_mfUpdate.Enabled = false;
txtComp_subIdUpdate.Text = "";
txtComp_subIdUpdate.Enabled = false;
btnCompUpdate.Enabled = false;

} catch (OleDbException OleDbEx)
{
    Response.Write("OleDbException: "+ OleDbEx.ToString());
} catch (Exception ex)
{
    Response.Write("Exception: "+ ex.ToString());
}

private void LinkButton1_Click(object sender,
System.EventArgs e)
{
    Session["uid"] = "";
    Response.Redirect("bratInv.aspx");
}

private void LinkButton2_Click(object sender,
System.EventArgs e)
{
    Response.Redirect("bratInvUpdate.aspx");
}

UbratMfgDetails.aspx.cs
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;

namespace BratInventory
{
    /// <summary>
    /// Summary description for UbratMfgDetails.
    /// </summary>
    public class UbratMfgDetails : System.Web.UI.Page
    {
        private OleDbDataAdapter mfg_uadp;
        private DataSet mfg_uds;
        protected System.Web.UI.WebControls.TextBox txtMfg_IdAdd;
        protected System.Web.UI.WebControls.TextBox txtMfg_NameAdd;
        protected System.Web.UI.WebControls.TextBox txtMfg_otherAdd;
        protected System.Web.UI.WebControls.Button btnMfg_Add;
        protected System.Web.UI.WebControls.TextBox txtMfg_nameUpdate;
        protected System.Web.UI.WebControls.DataGrid UgridMfg;
        protected System.Web.UI.WebControls.Label Label19;
        protected System.Web.UI.WebControls.Label Label1;
        protected System.Web.UI.WebControls.Label Label4;
        protected System.Web.UI.WebControls.Label Label5;
        protected System.Web.UI.WebControls.Label Label6;
        protected System.Web.UI.WebControls.Label Label2;
        protected System.Web.UI.WebControls.Label Label3;
        protected System.Web.UI.WebControls.Label Label7;
        protected System.Web.UI.WebControls.LinkButton LinkButton1;
        protected System.Web.UI.WebControls.LinkButton LinkButton2;
        protected System.Web.UI.WebControls.TextBox txtMfg_otherUpdate;
        public UbratMfgDetails()
        {
            Page.Init += new System.EventHandler(Page_Init);
        }
        private void Page_Load(object sender, System.EventArgs e)
        {
            if ((Session["uid"].ToString()) != 
            {
                if (!IsPostBack)
                {
                    try
                    {
                        OleDbConnection mfgConn = new OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Password="";User ID=Admin;Data Source=E:\ak\bratTest\BratInventory.mdb;Mode=ReadWrite;Extended Properties="";Jet OLEDB:System database="";Jet OLEDB:Registry Path="";Jet OLEDB:Database Password="";Jet OLEDB:Engine Type=5;Jet OLEDB:Database Locking Mode=1;Jet OLEDB:Global Partial Bulk Ops=2;Jet

96
mfg_uadp = new OleDbDataAdapter();
OleDbCommand selectCmd = new OleDbCommand("Select * FROM mfg_details",mfgConn);
mfg_uadp.SelectCommand = selectCmd;
OleDbCommand updateCmd = new OleDbCommand("UPDATE mfg_details SET manuf_name = @mfgname, " + "other_info = @otherinfo WHERE manuf_id = @mfgid", mfgConn);

updateCmd.Parameters.Add("@mfgname",OleDbType.VarChar,30);
updateCmd.Parameters["@mfgname"].SourceColumn = "manuf_name";

updateCmd.Parameters.Add("@otherinfo",OleDbType.VarChar,10);
updateCmd.Parameters["@otherinfo"].SourceColumn = "other_info";

updateCmd.Parameters.Add("@mfgid",OleDbType.Numeric,3);
updateCmd.Parameters["@mfgid"].SourceColumn = "manuf_id";

mfg_uadp.UpdateCommand = updateCmd;
OleDbCommand deleteCmd = new OleDbCommand("DELETE FROM mfg_details WHERE " + "manuf_id=@mfgid",mfgConn);
deleteCmd.Parameters.Add("@mfgid",OleDbType.Numeric);
deleteCmd.Parameters["@mfgid"].SourceColumn = "manuf_id";

mfg_uadp.DeleteCommand = deleteCmd;
OleDbCommand insertCmd = new OleDbCommand("INSERT INTO mfg_details (manuf_id, manuf_name, other_info)" + "VALUES (@mfgid, @mfgname, @otherinfo)", mfgConn);
insertCmd.Parameters.Add("@mfgid",OleDbType.Numeric, 3);
insertCmd.Parameters["@mfgid"].SourceColumn = "manuf_id";
insertCmd.Parameters.Add("@mfgname",OleDbType.VarChar, 30);
insertCmd.Parameters["@mfgname"].SourceColumn = "manuf_name";
insertCmd.Parameters.Add("@otherinfo", OleDbType.VarChar, 30);
insertCmd.Parameters["@otherinfo"].SourceColumn = "other_info";
mfg_uadp.InsertCommand = insertCmd;
mfg_uds = new DataSet();
mfg_uadp.Fill(mfg_uds, "mfg_details");
UgridMfg.DataSource = mfg_uds.Tables["mfg_details"].DefaultView;
UgridMfg.DataBind();
Session.Add("mfg_uadp", mfg_uadp);
Session.Add("mfg_uds", mfg_uds);
}
} catch (OleDbException OleDbEx)
{
    Response.Write("OleDbException: "+ OleDbEx.ToString());
} catch (Exception ex)
{
    Response.Write("Exception: "+ex.ToString());
} else
{
    mfg_uds = (DataSet)Session["mfg_uds"]; 
mfg_uadp = (OleDbDataAdapter)Session["mfg_uadp"]; 
} 
else
{
    Response.Redirect("bratInv.aspx");
} // Put user code to initialize the page here
private void Page_Init(object sender, EventArgs e)
{
    InitializeComponent();
}
#region Web Form Designer generated code
// override protected void OnInit(EventArgs e)
// {
//     // CODEGEN: This call is required by the ASP.NET Web Form Designer.
//     //
//     InitializeComponent();
//     //
//     base.OnInit(e);
// }
private void InitializeComponent()
{
    this.UgridMfg.ItemCommand += new System.Web.UI.WebControls.DataGridCommandEventHandler(this.OnItemClicked);
    this.btnMfg_Add.Click += new System.EventHandler(this.btnMfg_Add_Click);
    this.btnMfg_Update.Click += new System.EventHandler(this.btnMfg_Update_Click);
    this.LinkButton1.Click += new System.EventHandler(this.LinkButton1_Click);
    this.LinkButton2.Click += new System.EventHandler(this.LinkButton2_Click);
    this.Load += new System.EventHandler(this.Page_Load);
}
#endregion

protected void OnItemClicked(object source, DataGridCommandEventArgs e)
{
    int itemindex = (int)e.Item.ItemIndex;
    if (e.CommandName == "Select")
    {
        ViewState["SelectedItemIndex"] = itemindex;
        DataRow selectedRow = mfg_uds.Tables["mfg_details"].Rows[itemindex];
        txtMfg_nameUpdate.Text = selectedRow["manuf_name"].ToString();
        txtMfg_otherUpdate.Text = selectedRow["other_info"].ToString();
        txtMfg_nameUpdate.Enabled = true;
        txtMfg_otherUpdate.Enabled = true;
        btnMfg_Update.Enabled = true;
    }
    else
    {
        try
        {
            mfg_uds.Tables["mfg_details"].Rows[itemindex].Delete();
            mfg_uadp.Update(mfg_uds, "mfg_details");
            Session["mfg_uds"] = mfg_uds;
            UgridMfg.DataSource = mfg_uds.Tables["mfg_details"].DefaultView;
            UgridMfg.DataBind();
        }
        catch (Exception ex)
        {
            // Handle exception
        }
    }
}
private void btnMfg_Add_Click(object sender, System.EventArgs e)
{
    try
    {
        DataRow newRow = mfg_uds.Tables["mfg_details"].NewRow();
        newRow["manuf_id"] = txtMfg_IdAdd.Text;
        newRow["manuf_name"] = txtMfg_NameAdd.Text;
        newRow["other_info"] = txtMfg_otherAdd.Text;
        mfg_uds.Tables["mfg_details"].Rows.Add(newRow);
        mfg_uadp.Update(mfg_uds, "mfg_details");
        mfg_uadp.Clear();
        mfg_uadp.Fill(mfg_uds, "mfg_details");
        Session["mfg_uds"] = mfg_uds;
        UgridMfg.DataSource = mfg_uds.Tables["mfg_details"].DefaultView;
        UgridMfg.DataBind();
        txtMfg_IdAdd.Text = ";
        txtMfg_NameAdd.Text = ";
        txtMfg_otherAdd.Text = ";
    }
    catch (OleDbException OleDbEx)
    {
        Response.Write("OleDbException: " + OleDbEx.ToString());
        return;
    }
    catch (Exception ex)
    {
        Response.Write("Exception: "+ ex.ToString());
    }
}

private void btnMfg_Update_Click(object sender, System.EventArgs e)
try {
    DataRow selectedRow = mfg_uads.Tables["mfg_details"].Rows[(int)(ViewState["SelectedItemIndex"])];
    selectedRow["manuf_name"] = txtMfg_nameUpdate.Text;
    selectedRow["other_info"] = txtMfg_otherUpdate.Text;
    mfg_uadp.Update(mfg_uads, "mfg_details");
    Session["mfg_uads"] = mfg_uads;
    UgridMfg.DataSource = mfg_uads.Tables["mfg_details"].DefaultView;
    UgridMfg.DataBind();
    txtMfg_nameUpdate.Text = ""
    txtMfg_nameUpdate.Enabled = false;
    txtMfg_otherUpdate.Text = ""
    txtMfg_otherUpdate.Enabled = false;
    btnMfg_Update.Enabled = false;
} catch (OleDbException OleDbEx) {
    Response.Write("OleDbException: "+ OleDbEx.ToString());
} catch (Exception ex) {
    Response.Write("Exception: "+ ex.ToString());
}

private void LinkButton1_Click(object sender, EventArgs e) {
    Session["uid"] = "";
    Response.Redirect("bratInv.aspx");
}

private void LinkButton2_Click(object sender, EventArgs e) {
    Response.Redirect("bratInvUpdate.aspx");
}

UbratSponsors.aspx.cs
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;
namespace BratInventory
{
    /// <summary>
    /// Summary description for UbratSponsors.
    /// </summary>
    public class UbratSponsors : System.Web.UI.Page
    {
        private OleDbDataAdapter sponsor_uadp;
        private DataSet sponsor.udS;
        protected System.Web.UI.WebControls.TextBox txtSponsor_AddressAdd;
        protected System.Web.UI.WebControls.TextBox txtSponsor_lnameAdd;
        protected System.Web.UI.WebControls.TextBox txtSponsor_FnameAdd;
        protected System.Web.UI.WebControls.TextBox txtSponsor_IdAdd;
        protected System.Web.UI.WebControls.TextBox txtSponsor_EmailAdd;
        protected System.Web.UI.WebControls.TextBox txtSponsor_PhAdd;
        protected System.Web.UI.WebControls.TextBox txtSponsor_ZipAdd;
        protected System.Web.UI.WebControls.TextBox txtSponsor_StateAdd;
        protected System.Web.UI.WebControls.TextBox txtSponsor_CityAdd;
        protected System.Web.UI.WebControls.TextBox txtSponsor_fnameUpdate;
        protected System.Web.UI.WebControls.TextBox txtSponsor_lnameUpdate;
        protected System.Web.UI.WebControls.TextBox txtSponsor_addressUpdate;
        protected System.Web.UI.WebControls.TextBox txtSponsor_cityUpdate;
        protected System.Web.UI.WebControls.TextBox txtSponsor_stateUpdate;
        protected System.Web.UI.WebControls.TextBox txtSponsor_zipUpdate;
        protected System.Web.UI.WebControls.TextBox txtSponsor_phUpdate;
        protected System.Web.UI.WebControls.DataGrid UgridSponsors;
        protected System.Web.UI.WebControls.Label Label1;
        protected System.Web.UI.WebControls.Label Label2;
        protected System.Web.UI.WebControls.Label Label3;
    }
}
protected System.Web.UI.WebControls.Label Label4;
protected System.Web.UI.WebControls.Label Label5;
protected System.Web.UI.WebControls.Label Label6;
protected System.Web.UI.WebControls.Label Label7;
protected System.Web.UI.WebControls.Label Label8;
protected System.Web.UI.WebControls.Label Label9;
protected System.Web.UI.WebControls.Label Label10;
protected System.Web.UI.WebControls.Label Label11;
protected System.Web.UI.WebControls.Label Label12;
protected System.Web.UI.WebControls.Label Label13;
protected System.Web.UI.WebControls.Label Label14;
protected System.Web.UI.WebControls.Label Label15;
protected System.Web.UI.WebControls.Label Label16;
protected System.Web.UI.WebControls.Label Label17;
protected System.Web.UI.WebControls.Label Label18;
protected System.Web.UI.WebControls.Label Label19;
protected System.Web.UI.WebControls.Label Label20;
protected System.Web.UI.WebControls.Label Label21;
protected System.Web.UI.WebControls.LinkButton LinkButton1;
protected System.Web.UI.WebControls.LinkButton LinkButton2;
protected System.Web.UI.WebControls.TextBox txtSponsor_emailUpdate;

public UbratSponsors()
{
    Page.Init += new System.EventHandler(Page_Init);
}

private void Page_Load(object sender, System.EventArgs e)
{
    if ((Session["uid"]).ToString()) != "")
    {
        if (!IsPostBack)
        {
            try
            {
                OleDbConnection sponsorConn = new
                OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Password=";User
                ID=Admin;Data
                Source=E:\ak\bratTest\BratInventory.mdb;Mode=ReadWrite;Extended
                Properties="";Jet OLEDB:System database="";Jet OLEDB:Registry
                Path="";Jet OLEDB:Database Password="";Jet OLEDB:Engine Type=5;Jet
                OLEDB:Database Locking Mode=1;Jet OLEDB:Global Partial Bulk Ops=2;Jet
                OLEDB:Global Bulk Transactions=1;Jet OLEDB:New Database
                Password="";Jet OLEDB:Create System Database=False;Jet OLEDB:Encrypt
                Database=False;Jet OLEDB:Don't Copy Locale on Compact=False;Jet
                OLEDB:Compact Without Replica Repair=False;Jet OLEDB:SFP=False");
               sponsor_uadp = new
                OleDbDataAdapter();

                OleDbCommand selectCmd = new
                OleDbCommand("Select * FROM sponsors order by sp_id", sponsorConn);
                sponsor_uadp.SelectCommand =
                selectCmd;
            }
        }
    }
}
OleDbCommand updateCmd = new OleDbCommand("UPDATE sponsors SET sp_fname = @firstname, sp_lname = @lastname, sp_address = @address, sp_city = @city, sp_state = @state, " + " sp_zip = @zip, sp_phone = @phone, sp_email = @email WHERE sp_id = @sponsorid", sponsorConn);

updateCmd.Parameters.Add("@firstname", OleDbType.VarChar, 10);
updateCmd.Parameters["@firstname"].SourceColumn = "sp_fname";

updateCmd.Parameters.Add("@lastname", OleDbType.VarChar, 10);
updateCmd.Parameters["@lastname"].SourceColumn = "sp_lname";

updateCmd.Parameters.Add("@address", OleDbType.VarChar, 50);
updateCmd.Parameters["@address"].SourceColumn = "sp_address";

updateCmd.Parameters.Add("@city", OleDbType.VarChar, 10);
updateCmd.Parameters["@city"].SourceColumn = "sp_city";

updateCmd.Parameters.Add("@state", OleDbType.VarChar, 2);
updateCmd.Parameters["@state"].SourceColumn = "sp_state";

updateCmd.Parameters.Add("@zip", OleDbType.Numeric, 5);
updateCmd.Parameters["@zip"].SourceColumn = "sp_zip";

updateCmd.Parameters.Add("@phone", OleDbType.VarChar, 15);
updateCmd.Parameters["@phone"].SourceColumn = "sp_phone";

updateCmd.Parameters.Add("@email", OleDbType.VarChar, 10);
updateCmd.Parameters["@email"].SourceColumn = "sp_email";

updateCmd.Parameters.Add("@sponsorid", OleDbType.Numeric, 3);
updateCmd.Parameters["@sponsorid"].SourceColumn = "sp_id";

sponsor_uadp.UpdateCommand = updateCmd;

OleDbCommand deleteCmd = new OleDbCommand("DELETE FROM sponsors WHERE " +

```sql
DELETE FROM sponsors WHERE " +
```
"sp_id=@sponsorid", sponsorConn);

deleteCmd.Parameters.Add("@sponsorid", OleDbType.Numeric);
deleteCmd.Parameters["@sponsorid"].SourceColumn = "sp_id";
sponsor_uadp.DeleteCommand = deleteCmd;

OleDbCommand insertCmd = new OleDbCommand("INSERT INTO sponsors (sp_id, sp_fname, sp_lname, sp_address, sp_city, sp_state, sp_zip, sp_phone, sp_email) VALUES (@spid, @firstname, @lastname, @address, @city, @state, @zip, @phone, @email)", sponsorConn);

insertCmd.Parameters.Add("@spid", OleDbType.Numeric, 3);
insertCmd.Parameters["@spid"].SourceColumn = "sp_id";

insertCmd.Parameters.Add("@firstname", OleDbType.VarChar, 10);
insertCmd.Parameters["@firstname"].SourceColumn = "sp_fname";

insertCmd.Parameters.Add("@lastname", OleDbType.VarChar, 10);
insertCmd.Parameters["@lastname"].SourceColumn = "sp_lname";

insertCmd.Parameters.Add("@address", OleDbType.VarChar, 50);
insertCmd.Parameters["@address"].SourceColumn = "sp_address";

insertCmd.Parameters.Add("@city", OleDbType.VarChar, 10);
insertCmd.Parameters["@city"].SourceColumn = "sp_city";

insertCmd.Parameters.Add("@state", OleDbType.VarChar, 2);
insertCmd.Parameters["@state"].SourceColumn = "sp_state";

insertCmd.Parameters.Add("@zip", OleDbType.Numeric, 5);
insertCmd.Parameters["@zip"].SourceColumn = "sp_zip";

insertCmd.Parameters.Add("@phone", OleDbType.VarChar, 15);
insertCmd.Parameters["@phone"].SourceColumn = "sp_phone";
insertCmd.Parameters.Add("@email",OleDbType.VarChar, 10);
sponsor_uadp.InsertCommand = insertCmd;
sponsor_uds = new DataSet();
sponsor_uadp.Fill(sponsor_uds, "sponsors");
UgridSponsors.DataSource = sponsor_uds.Tables["sponsors"].DefaultView;
UgridSponsors.DataBind();
Session.Add("sponsor_uadp", sponsor_uadp);
Session.Add("sponsor_uds", sponsor_uds);

}  
catch (OleDbException OleDbEx)  
{  
    Response.Write("OleDbException: " + OleDbEx.ToString());  
}  
catch (Exception ex)  
{  
    Response.Write("Exception: "+ex.ToString());  
}  
else  
{  
    sponsor_uds = (DataSet)Session["sponsor_uds"];  
sponsor_uadp = (OleDbDataAdapter)Session["sponsor_uadp"];  
}  
else  
{  
    Response.Redirect("bratInv.aspx");  
}  
// Put user code to initialize the page here 

private void Page_Init(object sender, EventArgs e)  
{  
    InitializeComponent();  
}

#region Web Form Designer generated code
  // override protected void OnInit(EventArgs e)  
  //
/// // CODEGEN: This call is required by the ASP.NET Web Form Designer.
/// //
/// InitializeComponent();
/// //
/// base.OnInit(e);
///
/// <summary>
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
    this.UgridSponsors.ItemCommand += new System.Web.UI.WebControls.DataGridCommandEventHandler(this.OnItemClicked);
    this.btnSponsor_update.Click += new System.EventHandler(this.btnSponsor_update_Click);
    this.btnSponsor_add.Click += new System.EventHandler(this.btnSponsor_add_Click);
    this.LinkButton1.Click += new System.EventHandler(this.LinkButton1_Click);
    this.LinkButton2.Click += new System.EventHandler(this.LinkButton2_Click);
    this.Load += new System.EventHandler(this.Page_Load);
}
#endregion

protected void OnItemClicked(object source, DataGridCommandEventArgs e)
{
    int itemindex = (int)e.Item.ItemIndex;
    if (e.CommandName == "Select")
    {
        ViewState["SelectedItemIndex"] = itemindex;
        DataRow selectedRow = sponsor_uds.Tables["sponsors"].Rows[itemindex];
        txtSponsor_fnameUpdate.Text = selectedRow["sp_fname"].ToString();
        txtSponsor_lnameUpdate.Text = selectedRow["sp_lname"].ToString();
        txtSponsor_addressUpdate.Text = selectedRow["sp_address"].ToString();
        txtSponsor_cityUpdate.Text = selectedRow["sp_city"].ToString();
        txtSponsor_stateUpdate.Text = selectedRow["sp_state"].ToString();
        txtSponsor_zipUpdate.Text = selectedRow["sp_zip"].ToString();
        txtSponsor_phUpdate.Text = selectedRow["sp_phone"].ToString();
        txtSponsor_emailUpdate.Text = selectedRow["sp_email"].ToString();
    }
txtSponsor_fnameUpdate.Enabled = true;
txtSponsor_lnameUpdate.Enabled = true;
txtSponsor_addressUpdate.Enabled = true;
txtSponsor_cityUpdate.Enabled = true;
txtSponsor_stateUpdate.Enabled = true;
txtSponsor_zipUpdate.Enabled = true;
txtSponsor_phUpdate.Enabled = true;
txtSponsor_emailUpdate.Enabled = true;
btnSponsor_update.Enabled = true;
}
else
{
    try
    {
        sponsor_uds.Tables["sponsors"].Rows[itemindex].Delete();
        sponsor_uadp.Update(sponsor_uds, "sponsors");
        Session["sponsor_uds"] = sponsor_uds;
        UgridSponsors.DataSource = sponsor_uds.Tables["sponsors"].DefaultView;
        UgridSponsors.DataBind();
    }
    catch (OleDbException OleDbEx)
    {
        Response.Write("OleDbException: " + OleDbEx.ToString());
    }
    catch (Exception ex)
    {
        Response.Write("Exception: " + ex.ToString());
    }
}
}

private void btnSponsor_add_Click(object sender, System.EventArgs e)
{
    try
    {
        DataRow newRow = sponsor_uds.Tables["sponsors"].NewRow();
        newRow["sp_id"] = txtSponsor_IdAdd.Text;
        newRow["sp_fname"] = txtSponsor_FnameAdd.Text;
        newRow["sp_lname"] = txtSponsor_lnameAdd.Text;
        newRow["sp_address"] = txtSponsor_AddressAdd.Text;
        newRow["sp_city"] = txtSponsor_CityAdd.Text;
        newRow["sp_state"] = txtSponsor_StateAdd.Text;
        newRow["sp_zip"] = txtSponsor_ZipAdd.Text;
        newRow["sp_phone"] = txtSponsor_PhAdd.Text;
    }
newRow["sp_email"] = txtSponsor_EmailAdd.Text;

sponsor_uds.Tables["sponsors"].Rows.Add(newRow);

sponsor_uadp.Update(sponsor_uds, "sponsors");

sponsor_uds.Clear();
sponsor_uadp.Fill(sponsor_uds, "sponsors");

Session["sponsor_uds"] = sponsor_uds;

UgridSponsors.DataSource = sponsor_uds.Tables["sponsors"].DefaultView;
UgridSponsors.DataBind();

txtSponsor_IdAdd.Text = ";
txtSponsor_FnameAdd.Text = ";
txtSponsor_LnameAdd.Text = ";
txtSponsor_AddressAdd.Text = ";
txtSponsor_CityAdd.Text = ";
txtSponsor_StateAdd.Text = ";
txtSponsor_ZipAdd.Text = ";
txtSponsor_PhAdd.Text = ";

private void btnSponsor_update_Click(object sender, System.EventArgs e)
{
    try
    {
        DataRow selectedRow = sponsor_uds.Tables["sponsors"].Rows[(int)(ViewState["SelectedItemIndex"])];

        selectedRow["sp_fname"] = txtSponsor_fnameUpdate.Text;
        selectedRow["sp_lname"] = txtSponsor_lnameUpdate.Text;
        selectedRow["sp_address"] = txtSponsor_addressUpdate.Text;
        selectedRow["sp_city"] = txtSponsor_cityUpdate.Text;
        selectedRow["sp_state"] = txtSponsor_stateUpdate.Text;
    }
    catch (OleDbException OleDbEx)
    {
        Response.Write("OleDbException: "+ OleDbEx.ToString());
    }
    catch (Exception ex)
    {
        Response.Write("Exception: "+ ex.ToString());
    }
}
selectedRow["sp_zip"] =
txtSponsor_zipUpdate.Text;
selectedRow["sp_phone"] =
txtSponsor_phUpdate.Text;
selectedRow["sp_email"] =
txtSponsor_emailUpdate.Text;

sponsor_uadp.Update(sponsor_uds, "sponsors");
Session["sponsor_uads"] = sponsor_uads;

UgridSponsors.DataSource =
sponsor_uads.Tables["sponsors"].DefaultView;
UgridSponsors.DataBind();

txtSponsor_fnameUpdate.Text = "";
txtSponsor_fnameUpdate.Enabled = false;

txtSponsor lnameUpdate.Text = "";
txtSponsor lnameUpdate.Enabled = false;

txtSponsor_addressUpdate.Text = "";
txtSponsor_addressUpdate.Enabled = false;

txtSponsor_cityUpdate.Text = "";
txtSponsor_cityUpdate.Enabled = false;

txtSponsor_stateUpdate.Text = "";
txtSponsor_stateUpdate.Enabled = false;

txtSponsor_zipUpdate.Text = "";
txtSponsor_zipUpdate.Enabled = false;

txtSponsor_phUpdate.Text = "";
txtSponsor_phUpdate.Enabled = false;

btnSponsor_update.Enabled = false;
}
catch (OleDbException OleDbEx)
{
    Response.Write("OleDbException: "+
OleDbEx.ToString());
}
catch (Exception ex)
{
    Response.Write("Exception: "+ ex.ToString());
}

private void LinkButton1_Click(object sender,
System.EventArgs e)
{
    Session["uid"] = "";
    Response.Redirect("bratInv.aspx");
}
private void LinkButton2_Click(object sender, System.EventArgs e)
{
    Response.Redirect("bratInvUpdate.aspx");
}

UbratMembers.aspx.cs
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;
namespace BratInventory
{
/// <summary>
/// Summary description for UbratMembers.
/// </summary>
/// <summary>
public class UbratMembers : System.Web.UI.Page
{
    private OleDbDataAdapter member_uadp;
    private DataSet member_uds;
    protected System.Web.UI.WebControls.DataGrid UgridMembers;
    protected System.Web.UI.WebControls.Button btnMemberAdd;
    protected System.Web.UI.WebControls.TextBox txtMember_emailAdd;
    protected System.Web.UI.WebControls.TextBox txtMember_phAdd;
    protected System.Web.UI.WebControls.TextBox txtMember_zipAdd;
    protected System.Web.UI.WebControls.TextBox txtMember_stateAdd;
    protected System.Web.UI.WebControls.TextBox txtMember_cityAdd;
    protected System.Web.UI.WebControls.TextBox txtmember_lnameAdd;
    protected System.Web.UI.WebControls.TextBox txtMember_fnameAdd;
    protected System.Web.UI.WebControls.Button btnUpdateMember;
    protected System.Web.UI.WebControls.TextBox txtMember_addressUpdate;
    protected System.Web.UI.WebControls.TextBox txtMember_lnameUpdate;
}
public UbratMembers()
{
    Page.Init += new System.EventHandler(Page_Init);
}

private void Page_Load(object sender, System.EventArgs e)
{
    if ((Session["uid"].ToString()) != "")
    {
        if (!IsPostBack)
        {
            try
            {
                OleDbConnection member_conn = new OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Password="";User ID=Admin;Data Source=E:\ak\bratTest\BratInventory.mdb;Mode=ReadWrite;Extended Properties="";Jet OLEDB:System database="";Jet OLEDB:Registry Path="";Jet OLEDB:Database Password="";Jet OLEDB:Engine Type=5;Jet

member_uadp = new OleDbDataAdapter();

OleDbCommand selectCmd = new OleDbCommand("Select * FROM members", member_conn);

member_uadp.SelectCommand = selectCmd;

OleDbCommand updateCmd = new OleDbCommand("UPDATE members SET m_fname = @firstname, m_lname = @lastname, m_address = @address, m_city = @city, m_state = @state, m_zip = @zip, m_phone = @memphone, m_email = @email WHERE member_id = @memberid", member_conn);

updateCmd.Parameters.Add("@firstname", OleDbDbType.VarChar, 20);
updateCmd.Parameters["@firstname"].SourceColumn = "m_fname";

updateCmd.Parameters.Add("@lastname", OleDbDbType.VarChar, 20);
updateCmd.Parameters["@lastname"].SourceColumn = "m_lname";

updateCmd.Parameters.Add("@address", OleDbDbType.VarChar, 50);
updateCmd.Parameters["@address"].SourceColumn = "m_address";

updateCmd.Parameters.Add("@city", OleDbDbType.VarChar, 10);
updateCmd.Parameters["@city"].SourceColumn = "m_city";

updateCmd.Parameters.Add("@state", OleDbDbType.VarChar, 2);
updateCmd.Parameters["@state"].SourceColumn = "m_state";

updateCmd.Parameters.Add("@zip", OleDbDbType.Numeric, 5);
updateCmd.Parameters["@zip"].SourceColumn = "m_zip";

updateCmd.Parameters.Add("@memphone", OleDbDbType.VarChar, 15);
updateCmd.Parameters["@memphone"].SourceColumn = "m_phone";

updateCmd.Parameters.Add("@email", OleDbDbType.VarChar, 20);
updateCmd.Parameters["@email"].SourceColumn = "m_email";

updateCmd.Parameters.Add("@memberid",OleDbType.Numeric,3);
updateCmd.Parameters["@memberid"].SourceColumn = "member_id";

member_uadp.UpdateCommand = updateCmd;

OleDbCommand deleteCmd = new OleDbCommand("DELETE FROM members WHERE " +
"member_id=@memberid",member_conn);

deleteCmd.Parameters.Add("@memberid",OleDbType.Numeric);
deleteCmd.Parameters["@memberid"].SourceColumn = "member_id";

member_uadp.DeleteCommand = deleteCmd;

OleDbCommand insertCmd = new OleDbCommand("INSERT INTO members (member_id, m_fname, m_lname,
m_address, m_city, m_state, m_zip, m_phone, m_email)" +
"VALUES (@memberid, @firstname, @lastname, @address, @city, @state, @zip, @mphonem, @email)", member_conn);

insertCmd.Parameters.Add("@memberid",OleDbType.Numeric, 3);
insertCmd.Parameters["@memberid"].SourceColumn = "member_id";

insertCmd.Parameters.Add("@firstname",OleDbType.VarChar, 20);
insertCmd.Parameters["@firstname"].SourceColumn = "m_fname";

insertCmd.Parameters.Add("@lastname",OleDbType.VarChar, 20);
insertCmd.Parameters["@lastname"].SourceColumn = "m_lname";

insertCmd.Parameters.Add("@address",OleDbType.VarChar, 50);
insertCmd.Parameters["@address"].SourceColumn = "m_address";

insertCmd.Parameters.Add("@city",OleDbType.VarChar, 20);
insertCmd.Parameters["@city"].SourceColumn = "m_city";
insertCmd.Parameters.Add("@state", OleDbType.VarChar, 2);
insertCmd.Parameters["@state"].SourceColumn = "m_state";

insertCmd.Parameters.Add("@zip", OleDbType.Numeric, 5);
insertCmd.Parameters["@zip"].SourceColumn = "m_zip";

insertCmd.Parameters.Add("@memphone", OleDbType.VarChar, 15);
insertCmd.Parameters["@memphone"].SourceColumn = "m_phone";

insertCmd.Parameters.Add("@email", OleDbType.VarChar, 20);
insertCmd.Parameters["@email"].SourceColumn = "m_email";

member_uadp.InsertCommand = insertCmd;

member_uds = new DataSet();
member_uadp.Fill(member_uds, "members");
UgridMembers.DataSource = member_uds.Tables["members"].DefaultView;
UgridMembers.DataBind();

Session.Add("member_uadp", member_uadp);
Session.Add("member_uds", member_uds);

} catch (OleDbException OleDbEx)
{
    Response.Write("OleDbException: " + OleDbEx.ToString());
} catch (Exception ex)
{
    Response.Write("Exception: " + ex.ToString());
} else
{
    member_uds = (DataSet)Session["member_uds"]; member_uadp = (OleDbDataAdapter)Session["member_uadp"]; }
else
{
    Response.Redirect("bratInv.aspx");
} // Put user code to initialize the page here

private void Page_Init(object sender, EventArgs e)
{

}
InitializeComponent();

#endregion Web Form Designer generated code
// override protected void OnInit(EventArgs e)
// {
// // CODEGEN: This call is required by the ASP.NET Web
// Form Designer.
// //
// // InitializeComponent();
// // base.OnInit(e);
// // }

/// <summary>
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
    this.UgridMembers.ItemCommand += new
    System.Web.UI.WebControls.DataGridCommandEventHandler(this.OnItemClicked);

    this.UgridMembers.SelectedIndexChanged += new
    System.EventHandler(this.UgridMembers_SelectedIndexChanged);

    this.btnUpdateMember.Click += new
    System.EventHandler(this.btnUpdateMember_Click);

    this.btnMemberAdd.Click += new
    System.EventHandler(this.btnMemberAdd_Click);

    this.Load += new System.EventHandler(this.Page_Load);
}

protected void OnItemClicked(object source,
DataGridCommandEventArgs e)
{
    int itemindex = (int)e.Item.ItemIndex;

    if (e.CommandName == "Select")
    {
        ViewState["SelectedItemIndex"] = itemindex;

        DataView selectedRow =
        member_uds.Tables["members"].Rows[itemindex];
        txtMember_fnameUpdate.Text =
        selectedRow["m_fname"].ToString();
        txtMember_lnameUpdate.Text =
        selectedRow["m_lname"].ToString();
        txtMember_addressUpdate.Text =
        selectedRow["m_address"].ToString();
    }
}
txtMember_cityUpdate.Text =
selectedRow["m_city"].ToString();
txtMember_stateUpdate.Text =
selectedRow["m_state"].ToString();
txtMember_zipUpdate.Text =
selectedRow["m_zip"].ToString();
txtMember_phUpdate.Text =
selectedRow["m_phone"].ToString();
txtMember_emailUpdate.Text =
selectedRow["m_email"].ToString();

txtMember_fnameUpdate.Enabled = true;
txtMember_lnameUpdate.Enabled = true;
txtMember_addressUpdate.Enabled = true;
txtMember_cityUpdate.Enabled = true;
txtMember_stateUpdate.Enabled = true;
txtMember_zipUpdate.Enabled = true;
txtMember_phUpdate.Enabled = true;
txtMember_emailUpdate.Enabled = true;
btnUpdateMember.Enabled = true;
}
else
{
try
{
member_uds.Tables["members"].Rows[itemindex].Delete();

member_uadp.Update(member_uds,"members");

Session["member_uds"] = member_uds;

UgridMembers.DataSource = member_uds.Tables["members"].DefaultView;
UgridMembers.DataBind();
}
catch (OleDbException OleDbEx)
{
Response.Write("OleDbException: " + OleDbEx.ToString());
}
catch (Exception ex)
{
Response.Write("Exception: "+ ex.ToString());
}
}
}
private void btnMemberAdd_Click(object sender,
System.EventArgs e)
{
try
{
DataRow newRow =
member_uds.Tables["members"].NewRow();

117
newRow["member_id"] = txtMember_IdAdd.Text;
newRow["m_fname"] = txtMember_fNameAdd.Text;
newRow["m_lname"] = txtMember_lNameAdd.Text;
newRow["m_address"] =
txtMember_addressAdd.Text;
newRow["m_city"] = txtMember_cityAdd.Text;
newRow["m_state"] = txtMember_stateAdd.Text;
newRow["m_zip"] = txtMember_zipAdd.Text;
newRow["m_phone"] = txtMember_phAdd.Text;
newRow["m_email"] = txtMember_emailAdd.Text;

member_uds.Tables["members"].Rows.Add(newRow);
member_uadp.Update(member_uds, "members");
member_uds.Clear();
member_uadp.Fill(member_uds, "members");
Session["member_uds"] = member_uds;
UgridMembers.DataSource =
member_uds.Tables["members"].DefaultView;
UgridMembers.DataBind();
txtMember_IdAdd.Text = ";
txtMember_fNameAdd.Text = ";
txtMember_lNameAdd.Text = ";
txtMember_addressAdd.Text = ";
txtMember_cityAdd.Text = ";
txtMember_stateAdd.Text = ";
txtMember_zipAdd.Text = ";
txtMember_phAdd.Text = ";
txtMember_emailAdd.Text = ";
}
catch (OleDbException OleDbEx)
{
    Response.Write("OleDbException: "+
OleDbEx.ToString());
}
catch (Exception ex)
{
    Response.Write("Exception: "+ ex.ToString());
}

private void btnUpdateMember_Click(object sender,
System.EventArgs e)
{
try
{
    DataRow selectedRow =
member_uds.Tables["members"].Rows[(int)(ViewState["SelectedItemIndex"])];

    selectedRow["m_fname"] =
txtMember_fNameUpdate.Text;
selectedRow["m_lname"] =
txtMember_lnameUpdate.Text;
selectedRow["m_address"] =
txtMember_addressUpdate.Text;
selectedRow["m_city"] =
txtMember_cityUpdate.Text;
selectedRow["m_state"] =
txtMember_stateUpdate.Text;
selectedRow["m_zip"] =
txtMember_zipUpdate.Text;
selectedRow["m_phone"] =
txtMember_phUpdate.Text;
selectedRow["m_email"] =
txtMember_emailUpdate.Text;

member_uadp.Update(member_uds, "members");
Session["member_uds"] = member_uds;

UgridMembers.DataSource =
member_uds.Tables["members"].DefaultView;
UgridMembers.DataBind();

txtMember_fnameUpdate.Text = "";
txtMember_fnameUpdate.Enabled = false;

txtMember_lnameUpdate.Text = "";
txtMember_lnameUpdate.Enabled = false;

txtMember_addressUpdate.Text = "";
txtMember_addressUpdate.Enabled = false;

txtMember_cityUpdate.Text = "";
txtMember_cityUpdate.Enabled = false;

txtMember_stateUpdate.Text = "";
txtMember_stateUpdate.Enabled = false;

txtMember_zipUpdate.Text = "";
txtMember_zipUpdate.Enabled = false;

txtMember_phUpdate.Text = "";
txtMember_phUpdate.Enabled = false;

txtMember_emailUpdate.Text = "";
txtMember_emailUpdate.Enabled = false;

btnUpdateMember.Enabled = false;
}
catch (OleDbException OleDbEx)
{
    Response.Write("OleDbException: "+
    OleDbEx.ToString());
}
catch (Exception ex)
{
    Response.Write("Exception: "+ ex.ToString());
}
private void LinkButton1_Click(object sender, System.EventArgs e)
{
    Session["uid"] = "";
    Response.Redirect("bratInv.aspx");
}

private void LinkButton2_Click(object sender, System.EventArgs e)
{
    Response.Redirect("bratInvUpdate.aspx");
}

private void UgridMembers_SelectedIndexChanged(object sender, System.EventArgs e)
{
}

UbratUsers.aspx.cs
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;
namespace BratInventory
{
    /// <summary>
    /// Summary description for UbratUsers.
    /// </summary>
    public class UbratUsers : System.Web.UI.Page
    {
        private DataSet user_ds;
        private OleDbDataAdapter user_adp;
        protected System.Web.UI.WebControls.TextBox txtUserIdAdd;
        protected System.Web.UI.WebControls.DataGrid UgridUsers;
        protected System.Web.UI.WebControls.Label Label2;
        protected System.Web.UI.WebControls.Label Label3;
        protected System.Web.UI.WebControls.Label Label5;
        protected System.Web.UI.WebControls.Label Label6;
        protected System.Web.UI.WebControls.TextBox txtPwdAdd;
        protected System.Web.UI.WebControls.TextBox txtUpdate_pwd;
protected System.Web.UI.WebControls.Label Label1;
protected System.Web.UI.WebControls.LinkButton LinkButton1;
protected System.Web.UI.WebControls.LinkButton LinkButton2;
protected System.Web.UI.WebControls.Label Label7;

public UbratUsers()
{
    Page.Init += new System.EventHandler(Page_Init);
}

private void Page_Load(object sender, System.EventArgs e)
{
    if ((Session["uid"].ToString()) != "")
    {
        if (!IsPostBack)
        {
            try
            {
                OleDbConnection conn = new OleDbConnection("Provider=Microsoft.JET.OLEDB.4.0;" + "DataSource=E:\ak\bratTest\BratInventory.mdb;Jet OLEDB:Engine Type=5;" + "Jet OLEDB:Password=abc");
                OleDbConnection user_conn = new OleDbConnection("Provider=Microsoft.JET.OLEDB.4.0;Password="";UserID=Admin;Data Source=E:\ak\bratTest\BratInventory.mdb;Mode=ReadWrite;Extended Properties="";Jet OLEDB:System database="";Jet OLEDB:Registry Path="";Jet OLEDB:Database Password="";Jet OLEDB:Engine Type=5;Jet OLEDB:Database Locking Mode=1;Jet OLEDB:Global Partial Bulk Ops=2;Jet OLEDB:Global Bulk Transactions=1;Jet OLEDB:New Database Password="";Jet OLEDB:Create System Database=False;Jet OLEDB:Encrypt Database=False;Jet OLEDB:Don't Copy Locale on Compact=False;Jet OLEDB:Compact Without Replica Repair=False;Jet OLEDB:SFP=False");
                OleDbDataAdapter user_adp = new OleDbDataAdapter();
                OleDbCommand selectCmd = new OleDbCommand("Select * FROM users", user_conn);
                user_adp.SelectCommand = selectCmd;
                OleDbCommand updateCmd = new OleDbCommand("UPDATE users SET pwd=@password" + " WHERE user_id=@userid", user_conn);
                user_adp.UpdateCommand = updateCmd;

                OleDbConnection conn = new OleDbConnection("Provider=Microsoft.JET.OLEDB.4.0;" + "DataSource=E:\ak\bratTest\BratInventory.mdb;Jet OLEDB:Engine Type=5;" + "Jet OLEDB:Password=abc");
                OleDbConnection user_conn = new OleDbConnection("Provider=Microsoft.JET.OLEDB.4.0;Password="";UserID=Admin;Data Source=E:\ak\bratTest\BratInventory.mdb;Mode=ReadWrite;Extended Properties="";Jet OLEDB:System database="";Jet OLEDB:Registry Path="";Jet OLEDB:Database Password="";Jet OLEDB:Engine Type=5;Jet OLEDB:Database Locking Mode=1;Jet OLEDB:Global Partial Bulk Ops=2;Jet OLEDB:Global Bulk Transactions=1;Jet OLEDB:New Database Password="";Jet OLEDB:Create System Database=False;Jet OLEDB:Encrypt Database=False;Jet OLEDB:Don't Copy Locale on Compact=False;Jet OLEDB:Compact Without Replica Repair=False;Jet OLEDB:SFP=False");
                OleDbDataAdapter user_adp = new OleDbDataAdapter();
                OleDbCommand selectCmd = new OleDbCommand("Select * FROM users", user_conn);
                user_adp.SelectCommand = selectCmd;
                OleDbCommand updateCmd = new OleDbCommand("UPDATE users SET pwd=@password" + " WHERE user_id=@userid", user_conn);
                updateCmd.Parameters.Add("@password", OleDbType.VarChar, 10);
                updateCmd.Parameters["@password"].SourceColumn = "pwd";
                updateCmd.Parameters.Add("@userid", OleDbType.VarChar, 10);
                updateCmd.Parameters["@userid"].SourceColumn = "user_id";
                user_adp.UpdateCommand = updateCmd;
            }
            catch (Exception)
            {
                // catch any exceptions
            }
        }
    }
}
OleDbCommand deleteCmd = new OleDbCommand("DELETE FROM users WHERE " +
    "user_id=@userid", user_conn);

deleteCmd.Parameters.Add("@userid", OleDbType.VarChar);
deleteCmd.Parameters["@userid"].SourceColumn = "user_id";

user_adp.DeleteCommand = deleteCmd;

OleDbCommand insertCmd = new OleDbCommand("INSERT INTO users (user_id, pwd) " +
    "VALUES (@userid, @password)", user_conn);

insertCmd.Parameters.Add("@userid", OleDbType.VarChar, 10);
insertCmd.Parameters["@userid"].SourceColumn = "user_id";

insertCmd.Parameters.Add("@password", OleDbType.VarChar, 10);
insertCmd.Parameters["@password"].SourceColumn = "pwd";

user_adp.InsertCommand = insertCmd;

user_ds = new DataSet();
user_adp.Fill(user_ds, "users");

UgridUsers.DataSource = user_ds.Tables["users"].DefaultView;
UgridUsers.DataBind();

Session.Add("user_adp", user_adp);
Session.Add("user_ds", user_ds);

} catch (OleDbException OleDbEx) {
    Response.Write("OleDbException: "+
    OleDbEx.ToString());
} catch (Exception ex) {
    Response.Write("Exception: "+ex.ToString());
}
else {
    user_ds = (DataSet)Session["user_ds"]; 
    user_adp = (OleDbDataAdapter)Session["user_adp"]; 
}
private void Page_Init(object sender, EventArgs e)
{
    InitializeComponent();
}

#region Web Form Designer generated code
// override protected void OnInit(EventArgs e)
// {
//     // CODEGEN: This call is required by the ASP.NET Web Form Designer.
//     //
//     InitializeComponent();
//     base.OnInit(e);
// }

/// <summary>
/// Required method for Designer support - do not modify the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
    this.UgridUsers.ItemCommand += new System.Web.UI.WebControls.DataGridCommandEventHandler(this.OnItemClicked);
    this.LinkButton1.Click += new System.EventHandler(this.LinkButton1_Click);
    this.btnUpdateUser.Click += new System.EventHandler(this.btnUpdateUser_Click);
    this.btnUserAdd.Click += new System.EventHandler(this.btnUserAdd_Click);
    this.LinkButton2.Click += new System.EventHandler(this.LinkButton2_Click);
    this.Load += new System.EventHandler(this.Page_Load);
}
#endregion

protected void OnItemClicked(object source, DataGridCommandEventArgs e)
{
    int itemindex = (int)e.Item.ItemIndex;
    if (e.CommandName == "Select")
    {
        ViewState["SelectedItemIndex"] = itemindex;
        DataRow selectedRow = user_ds.Tables["users"].Rows[itemindex];
        txtUpdate_pwd.Text = selectedRow["pwd"].ToString();
        // txtLastNameEdit.Text = selectedRow["last_name"].ToString();
    }
}
public void grid_User_SelectedIndexChanged(object sender, EventArgs e) {
    int itemIndex = Convert.ToInt32(grid_User_SelectedIndexChanged.SelectedIndex);
    txtUpdate_pwd.Enabled = true;
    // txtLastNameEdit.Enabled = true;
    btnUpdateUser.Enabled = true;
}
else {
    try {
        user_ds.Tables["users"].Rows[itemIndex].Delete();
        user_adp.Update(user_ds, "users");
        Session["user_ds"] = user_ds;
        UgridUsers.DataSource = user_ds.Tables["users"].DefaultView;
        UgridUsers.DataBind();
    }
    catch (OleDbException OleDbEx) {
        Response.Write("OleDbException: "+OleDbEx.ToString());
    }
    catch (Exception ex) {
        Response.Write("Exception: "+ex.ToString());
    }
}

private void btnUserAdd_Click(object sender, EventArgs e) {
    try {
        DataRow newRow = user_ds.Tables["users"].NewRow();
        newRow["user_id"] = txtUserIdAdd.Text;
        newRow["pwd"] = txtPwdAdd.Text;
        user_ds.Tables["users"].Rows.Add(newRow);
        user_adp.Update(user_ds, "users");
        user_ds.Clear();
        user_adp.Fill(user_ds, "users");
        Session["user_ds"] = user_ds;
        UgridUsers.DataSource = user_ds.Tables["users"].DefaultView;
        UgridUsers.DataBind();
    }
}
try
{
    try
    {
        DataRow selectedRow = user_ds.Tables["users"].Rows[(int)(ViewState["SelectedItemIndex"])];
        selectedRow["pwd"] = txtUpdate_pwd.Text;
        user_adp.Update(user_ds, "users");
        Session["user_ds"] = user_ds;
        UgridUsers.DataSource = user_ds.Tables["users"].DefaultView;
        UgridUsers.DataBind();
        txtUpdate_pwd.Text = "";
        txtUpdate_pwd.Enabled = false;
        btnUpdateUser.Enabled = false;
    }
    catch (OleDbException OleDbEx)
    {
        Response.Write("OleDbException: " + OleDbEx.ToString());
    }
    catch (Exception ex)
    {
        Response.Write("Exception: " + ex.ToString());
    }
}

private void btnUpdateUser_Click(object sender, System.EventArgs e)
{
    try
    {
        DataRow selectedRow = user_ds.Tables["users"].Rows[(int)(ViewState["SelectedItemIndex"])];
        selectedRow["pwd"] = txtUpdate_pwd.Text;
        user_adp.Update(user_ds, "users");
        Session["user_ds"] = user_ds;
        UgridUsers.DataSource = user_ds.Tables["users"].DefaultView;
        UgridUsers.DataBind();
        txtUpdate_pwd.Text = "";
        txtUpdate_pwd.Enabled = false;
        btnUpdateUser.Enabled = false;
    }
    catch (OleDbException OleDbEx)
    {
        Response.Write("OleDbException: " + OleDbEx.ToString());
    }
    catch (Exception ex)
    {
        Response.Write("Exception: " + ex.ToString());
    }
}

private void LinkButton1_Click(object sender, System.EventArgs e)
{
    Session["uid"] = "";
    Response.Redirect("bratInv.aspx");
}

private void LinkButton2_Click(object sender, System.EventArgs e)
{
}
Response.Redirect("bratInvUpdate.aspx");