

A REPORT

ON

**1. DESIGN AND DEVELOPMENT
OF
BRANCH COMMUNICATION SOFTWARE**

BY

Shivang Patel

2001U7PS098

AT

**ETA –MELCO ELEVATOR CO. L.L.C.
DUBAI, UAE**

A Practice School – II station of



Birla Institute of Technology and Science, Pilani-Dubai Campus
Knowledge Village, Dubai
UAE

(FEBRUARY 2005 - JULY 2005)

TABLE OF CONTENTS

Acknowledgements	I
Abstract	II
Summary	IV
Chapter 1	
1.1 About ETA Melco	1
1.2 Introduction to the Project	7
1.3 Scope and Purpose	8
1.4 Limitations of DDBCS	10
Chapter 2	
2.1 Data Collection	12
2.2 System Requirements	13
2.3 Application Used	14
Chapter 3	
3.1 Methodology	17
3.2 Layout of DDBCS	19
3.3 Update Status	20
3.4 Create, Edit and Delete Record	22
3.4.1 Create Contract	23
3.4.2 Create Unit	26
3.4.3 Edit Contract	28
3.4.4 Edit Unit	30
3.4.5 Delete Contract	33
3.4.6 Delete Unit	35

Chapter 4	
4.1 Results	37
4.2 Recommendation for Further Improvement	38
4.3 Conclusion	39
References	41
Appendix	43

ACKNOWLEDGEMENTS

We are highly grateful to the following people who have spent their valuable time and effort in helping us to successfully complete the project:-

- | | |
|---------------------------------|----------------------------------------|
| ■ Mr. V.R. Kumar | ED,ETA-Melco |
| ■ Mr. A.S.A. Basheer | GM,ETA-Melco |
| ■ Mr. V. Devarajan | AGM, Maintenance,ETA-Melco |
| ■ Mr. Shankaran | AGM,Service Sales,ETA-Melco |
| ■ Mr. Ramanujam | IT Coordinator,ETA-Melco |
| ■ Mr. Fareed | IT Coordinator,ETA-Melco |
| ■ Mr. Malaikannn | Material Engineer(Stores),ETA-Melco |
| ■ Prof. Dr. Ramachandran | Director, BITS-Pilani Dubai Campus |
| ■ Dr. Nagendra Parashar | PS-2 Coordinator, BITS-Pilani Dubai |
| ■ Mrs. Mubeena Rahman | PS-2 Faculty, BITS-Pilani Dubai Campus |
| ■ Mrs. Sujala Shetty | PS-2 Mentor, BITS-Pilani Dubai Campus |
| ■ Dr. Vadivel | PS-2 Faculty,BITS-Pilani Dubai Campus |

Also I would like to thank all the staff of ETA-MELCO and my fellow friends for co-operating and supporting throughout the project

A REPORT
ON
DESIGN AND DEVELOPMENT
OF
BRANCH COMMUNICATION SOFTWARE

BY

Shivang Patel

2001U7PS098

CS

**Prepared in Partial Fulfillment of the
Practice School –II Course**

AT

**ETA –MELCO ELEVATOR CO. L.L.C.
DUBAI, UAE**

A Practice School – II station of



Birla Institute of Technology and Science, Pilani-Dubai Campus
Knowledge Village, Dubai
UAE

(FEBRUARY 2005 - JULY 2005)

Birla Institute of Technology and Science, Pilani-Dubai Campus
Knowledge Village, Dubai

Station: ETA- MELCO

Centre: DUBAI

Duration: 01.02.2005 – 14.07.2005

Date of Start: 01.02.2005

Date of Submission: 10.07.2005

Title of the Project: 1. Design and Development Of
Branch Communication Software

ID No. / Name of the student: 2001U7PS098 / Shivang Patel

Discipline of Student: B.E (Hons) Computer Science

Name(s) and Designation(s) of the Expert(s):

Mr. Ramanujam, IT Coordinator

Mr. A.S.A Basheer, General Manager

Name of the PS Faculty: Mrs. Mubeena Rahman / Dr. Vadivel

Key Words: 1.Maintenance and Installation

Project Area(s): F.O.D. (Field Operation Department) & IT Department

Abstract:

- The **Design and Implementation of Branch Communication Software** at ETA Melco is developed in VB.Net to enable capturing and editing of information for all the existing as well as new contracts .The changes made will be directed to the Melco database without any redundancy.

Signature of Student
Date:

Signature of PS Faculty
Date:

SUMMARY

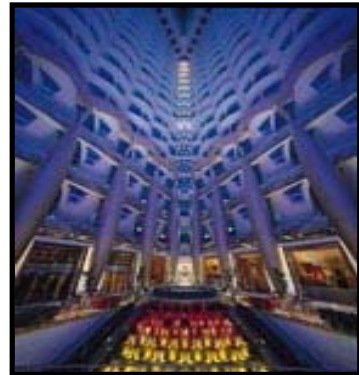
DDBCS (Design and Development of Branch Communication Software) is standalone software developed for easy and reliable data entry process for the installation and maintenance department of ETA Melco. The software has been designed in Visual Basic.Net. The application has been designed using Microsoft Visual Studio .NET Framework 2003, enterprise edition. The software was designed based on the working procedure of both the Installation and Maintenance department of ETA Melco. The main function of DDBCS is to enable the user to enter the contract details in a systematic way, starting from the details of the contract and then further going on to the units under that particular contract. The Software also provides the provision of editing the details of an existing contract in a structured way, thus saving time as well as avoids errors in data entry. Using DDBCS, the details of any contract can be viewed and updated at any point of time avoiding the hassles to search for details in the entire database.

CHAPTER

I

1.1 ABOUT ETA-MELCO

ETA-MELCO Elevator Co. L.L.C., was formed as a joint venture between **Emirates Trading Agency**, Dubai and **Mitsubishi Electric Corporation**, Japan. ETA-MELCO started its operations in 1975 in Dubai and expanded its operations to 13 countries including the U.A.E., the State of Qatar, Sri Lanka, the Sultanate of Oman, Kuwait, Turkey, Maldives, India, Bangladesh, Azerbaijan, Kyrgyzstan, Kazakhstan and Turkmenistan. They have grown to be market leaders in the region and earned the title "**Elevator People**" of the Gulf. There has been a steady growth of ETA MELCO who has installed elevators in the most prestigious projects which includes:

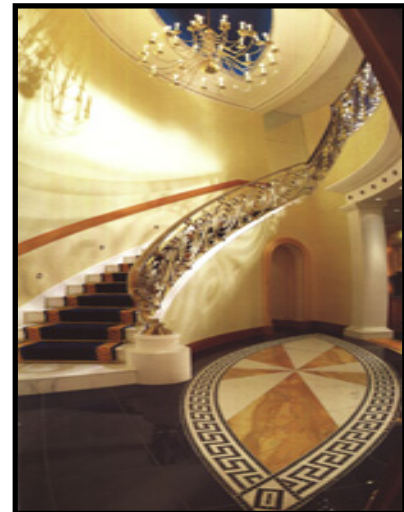


- **Fastest Passenger Elevator** in the whole of Middle East & Europe with a speed of 7M/Sec. In the world's tallest hotel (321 Meters) "**Burj Al Arab**", Dubai.
- Fastest passenger elevator with a speed of 7M/Sec in the world's 9th tallest building (350 Meters) "**Emirates Towers**", Dubai.
- Installed the maximum number of 121 units (35 elevators, 30 moving walks and 56 escalators) at Dubai International Airport.
- Sold more than 14,000 units including 10,000 units alone in the U.A.E.
- Having more than 1600 employees in the company which is the largest of its kind in the Middle East.

- Obtained the prestigious **ISO-9001 Certificate** in 1999.

ETA - MELCO follows a no-compromise on quality and service policy, which is synonymous to both parent companies - The ETA Group and Mitsubishi Electric Corporation. ETA MELCO assures that their elevators provide absolute safety and optimum reliability. Every individual member of the staff is trained to give utmost care to the quality of their work.

Over the last century, Mitsubishi has come to signify quality and cutting edge technology in all of its diverse range of products, be it industrial, electric, auto-mobile or domestic. **MELCO (Mitsubishi Electric)** was formed in 1920's to target primarily at the fast growing elevator market. In due course of time, it went on to become one of the world's leading makers of elevators, escalators and other hoisting devices. Today, Mitsubishi Electric has earned a reputation as one of the most advanced elevator and escalator manufacturers in the world, offering many '**Firsts**', in the Technology, including:



- The pioneers in introduction of **Variable Voltage Variable Frequency (VVVF)** Technology in the field of elevators.
- The world's fastest passenger elevator with a speed of 45 kms per hour, which has earned them a place in the '**Guinness Book of World Records**'.
- The only manufacturer in the world offering state-of-the-art technology with its **Spiral Escalator** considered an engineering genius in the industry.
- The First '**Zigzag**' Elevators manufactured in the world.
- Created **MELART II**, a full color graphic painting finish to create works of art for the artistic designs applied for passenger elevators.

- Supplied more than 320,000 units with latest production technology of delivering 12,000 units annually.
- Has a labor force of 117,000 with 120 facilities in 34 countries worldwide.

About the time when MELCO was celebrating its 50th anniversary, there was an unprecedented construction boom in the Middle East. The Al Ghurairs saw a vast untapped market in the construction segment. Thus came into existence the **Emirates Trading Agency (E T A)**. Where high rise buildings on a ‘rise’, there was tremendous scope for high quality and easily available elevators. Thus ETA proposed to be sole agents for Mitsubishi Electric Corporation of Japan, and launched Mitsubishi Elevators and Escalators in the UAE Market. In its initial stages of development, Western suppliers dominated the UAE market and the majority of equipment then was of Conventional type.



Based on initial success achieved in the UAE market, in 1980, the ETA-MELCO Elevator Co. LLC was formed as a joint venture between Emirates Trading Agency, Dubai and Mitsubishi Electric Corporation of Japan. Since then, the company has been carrying out Marketing, Selling, Design & Engineering, Installation and Maintenance of Mitsubishi’s Vertical Transportation Equipment. With the growth of the operations in the UAE market, ETA-MELCO expanded its operations in other countries Viz. State of Qatar, Sultanate of Oman, Kuwait, Turkey, Sri Lanka, Maldives, India, Bangladesh and the CIS countries.

Presently ETA - MELCO has grown to be the market specialists in the region and enjoys 55% of the elevator market and has earned the title, **Elevator People of the Gulf**.

1.1.1 DEPARTEMENTS OF ETA MELCO

SALES AND MARKETING:-

The purpose of this department is the promotion of products through marketing and sales by targeting premium segment. The activities that take place in sales are presale, enquiry and quotation, negotiation and finalizing and post contract. The activities that take place in marketing are exhibitions, advertisements, catalogues, presentation tools and factory visits.

DESIGN DEPARTMENT:-

This department supports all the activities related to layout drawing preparation and processing of the design to all the 14 branches. We are fully computerized with all the latest equipments. The design engineers are trained at our factory in Japan from time to time in order to acquaint themselves to the latest technology.

They have executed major projects like Dubai International Airport, Al Ghurair Centre, Burjuman Centre, Burj Al Arab, Emirates Tower Hotel, Dubai and Baynunah Hilton Tower, Etisalat Headquarters, Abu Dhabi Trade Centre, Abu Dhabi Investment Authority HQ., and Conference Palace Hotel At Abu Dhabi. They have installed spiral escalators at Abu Dhabi New Gold Souq.

SERVICE SALES:-

The Service Sales Department of ETA Melco is basically responsible for the maintenance of contracts of the equipments after the defects liability period. It also includes renewal of maintenance contract as and when they fall due. It also involves itself in recovery of maintenance contract from competitors. Some of the other activities of the service sales department include Repair Orders, Spare Parts Order, and Maintain customer relationship.

INSTALLATION DEPARTMENT:-

The department deals with the complete installation, testing and commissioning of the elevators, escalators and moving walks from the handover of the contracts from the sales department to the final handover of the units to the maintenance department.

It also has an **Internal Quality Control Section**, which ensures the quality of the elevators, escalators and moving walks are as per the requirements laid in the quality assurance plan. It also ensures that the checks and audits are conducted and checks list are maintained for each project as per the requirement of **ISO 9001:2000**. It also ensures that site safety requirements are enforced in the project sites and regular toolbox talks are conducted to create safety awareness at site. It also coordinates with the site safety team to give a safe working environment.

MAINTENANCE DEPARTMENT:-

This department is responsible for the maintenance of elevators, escalators, moving walks, and dumb waiters installed in various buildings. In addition to the regular maintenance, it also carries out repair works and attends to breakdown calls 24 hours a day throughout the year.

CREDIT CONTROL DEPARTEMENT:-

This department essentially receives money for the company. The department generates invoices for the company. Once the contract is signed and the bank guarantee is given, the sales department hands over the control to the Credit Control Department.

COMMERCIAL & SHIPPING DEPARTEMENT:-

Commercial and Shipping Department deals in the preparation of purchase order, maintenance of shipping details of contracts, payment of custom duty, clearing and deals with all commercial aspects.

STORES:-

The Stores Department works in conjunction with the maintenance and installation departments. About 80% of the inventory at stores is maintenance spare parts. The activity at stores is directly monitored by the maintenance department by means of an Oracle Database System called ORBITS. The store department directly reports to the head office.

PERSONAL DEPARTEMENT:-

It caters to the need of the workmen and is the driving force behind the performance of the firm. They ensure optimum coordination amongst the workforce and offers scope for individual development and healthy competition. They also deal with administrative issues, transport related issues, office management, workmen accommodation and arranging travel and tours.

1.2 INTRODUCTION TO THE PROJECT

The project, DDBCS (Design and Development of Branch Communication Software), is developed to meet the reporting and database requirements of the various departments in ETA Melco.

We started with the development of a module for the FOD (Fields Operation Division) which covers working of installation and maintenance. The module requires the official to enter the details of the contract and its status along with all the details of unit under a particular contract. The software provides a common format for maintaining the information across branches and departments.

The BCS software was developed keeping in mind all the above-mentioned problems faced in the company for contract details.

1.3 SCOPE AND PURPOSE

The main objective of the BCS software is to allow **easy and systematic data entry** of the contract details into the BCS Database.

The purpose of DDBCS is:

- To **speed up the data entry process**. The contract details will be entered in a uniform way, starting from the details of the contract and then further going on to the jobs which are included in the contract. The Software also provides the provision of editing the details of an existing contract in a structured way, thus saving time. Using BCS, the details of any contract can be viewed and updated. Saving time spent on searching through a bulk of records in the database.
- The software also **prevents redundant entries** for the same contract. For example, if a particular contract already exists in the Melco Database, no other contract can have the same contract number. The software will prompt a message to warn the user about it.
- The software also has a search field which allows the user to search for a particular contract including all the sub details of the contract in a very interactive way. The user does not have to search the whole database on its own and then find the records, which can be time consuming sometimes if the database is really huge .Thus it **saves the precious time** of the user and provides **accurate information**.
- To **reduce cost**. Man hours spent on entering and editing details of a contract can sometimes be very expensive. The software through its interactive user interface allows the user to enter or edit information easily and quickly, thus saving a lot of man-hours and in turn reducing man hour costs

1.4 LIMITATIONS OF BCS-FOD Module

All software's have certain drawbacks. Similarly the **BCS-FOD Module** has the following limitations:

- Module for only one department has been completed as of now. Modules for other nine departments will have to be designed to reap full benefits.
- The Program cannot generate crystal reports as of now this service component will have to be added
- The details of a unit have to be deleted before deleting the contract as a whole. Thus making the delete operation pretty time consuming but can be regarded as a blessing in disguise

CHAPTER II

2.1 DATA COLLECTION

The data collection process involved the following:-

- The guidelines given by the Installation Department regarding the procedure that is followed when a new contract is signed and the follow up procedures.
- Flow charts depicting the systematic flow of control between various departments of ETA Melco
- Manual for the FOD Department was a great source of information .Based on the guidelines mentioned in the Manual; the BCS Software was designed to match its requirements.
- The time ticket database which contained information about all the contracts, their jobs, contract type and maintenance service requirements.(melco.mdb)
- The format in which the forms had to be generated was also gathered along with the information about the key fields to be included in the software.

2.2 SYSTEM REQUIREMENTS

The software requirements for DDBCS are:-

- Microsoft XP Professional/Server 2003
- Microsoft Access
- Microsoft Visual Basic .Net Framework 1.0 or greater

The system should have all of the above-mentioned software to successfully execute BCS.

The database for BCS is created in Microsoft Access that consists of tables – areas, job details, unitsdetails, status, status of units, unit type.

2.3 APPLICATION USED

The BCS is designed and developed in Microsoft Visual Basic .Net framework 2003, Microsoft Access with SQL Commands.

Microsoft Visual Basic .Net is a language rapid application development environment that gives fast, easy, and intuitive tools to quickly develop Windows applications. Using Visual Basic, simple utilities or sophisticated applications can be developed. Data access features allow creating databases, front-end applications, and scalable server-side components for most popular database formats. ActiveX technologies allow using the functionality provided by other applications, and even automate applications and objects created using the Professional or Enterprise editions of Visual Basic.Net.

VB is an event-driven, hybrid development environment consisting of an integrated development environment (IDE) and a language based loosely on the original BASIC programming language. The IDE consists of a menu, a Forms window, a Toolbox, a Properties window (that changes context based on the windows object that is selected), a Form positioning window, a Code window, and an advanced Debugger window. VB allows the programmer to work in a "visual workshop" where the programmer can drag and drop different window elements into their programs before defining their meaning with the aid of a few drop down boxes.

The following are the reasons for developing the BCS in Visual Basic.NET Framework:-

- The structure of the Basic programming language is very user friendly, particularly as to the executable code.
- VB.Net is not only a language but also primarily an integrated, interactive development environment ("IDE").
- The VB-IDE has been highly optimized to support rapid application development ("RAD"). It is particularly easy to develop graphical user interfaces and to connect them to handler functions provided by the application.
- The graphical user interface of the VB-IDE provides intuitively appealing views for the management of the program structure in the large and the various types of entities (classes, modules, procedures, forms ...).
- VB.Net provides a comprehensive interactive and context-sensitive online help system.
- When editing program texts the "IntelliSense" technology informs you in a little popup window about the types of constructs that may be entered at the current cursor location.
- VB.Net is a component integration language which is attuned to Microsoft's Component Object Model ("COM").
- COM components can be written in different languages and then integrated using VB.Net.
- Interfaces of COM components can be easily called remotely via Distributed COM ("DCOM"), which makes it easy to construct distributed applications.
- COM components can be embedded in / linked to your application's user interface and also in/to stored documents (Object Linking and Embedding "OLE", "Compound Documents").
- There is a wealth of readily available COM components for many different purposes.

CHAPTER III

3.1 METHODOLOGY

The software was designed based on the working procedure of both the Installation and Maintenance department of ETA Melco. The approach that was followed to develop and design the software is strictly based on the guidelines given by both the departments.

Working Procedure followed by the Installation Department:-

- ❖ The installation takes over from the sales department. The installation department actually steps in once the order is placed with the principal by the processing department.
- ❖ Installation is done only after the structure is completed. Except in some rare cases where a two stage installation is carried out.
- ❖ At every stage the credit control department is informed for the collection of the payment.
- ❖ At least 80% of the total money has to be collected before the installation department hands over the project to the client.
- ❖ Third party consultants are employed at times.

Working Procedure followed by the Maintenance Department:-

- ❖ Information is received from the installation department once the unit is handed over.
- ❖ Joint Inspection is carried out with the installation engineer.
- ❖ The customer acceptance is received and a maintenance file is opened.
- ❖ Job is allotted to a specific route.
- ❖ Any job from installation is checked for snags and cleared and the list of units for which the free maintenance is expiring is sent to the service sales on a monthly basis.

- ❖ For execution of free and paid maintenance, the job to be services is identified.
- ❖ Required spare parts are received and replacement is carried out.
- ❖ The checklist is filled up and the weekly checked time is forwarded to the accounts department.
- ❖ For repair work (annual repairs/ received from service sales), the content of repair work to be carried out is determined.
- ❖ Manpower requirement is determined and required spares and tools are arranged
- ❖ The repair work is carried out and is recorded.
- ❖ In case of a break down call from the client, details are recorded in daily CBS record and the specific route technician is informed.
- ❖ The break down is attended to and the call back report is made.

3.2 Layout of BCS-FOD Module

Main Page of BCS – FOD Module:-

The form given below is the main page of BCS-FOD Module, which has two tab pages at the top left corner of the form. The Tab pages are:-

1. Update Status.
2. Create, Edit and Delete Records.

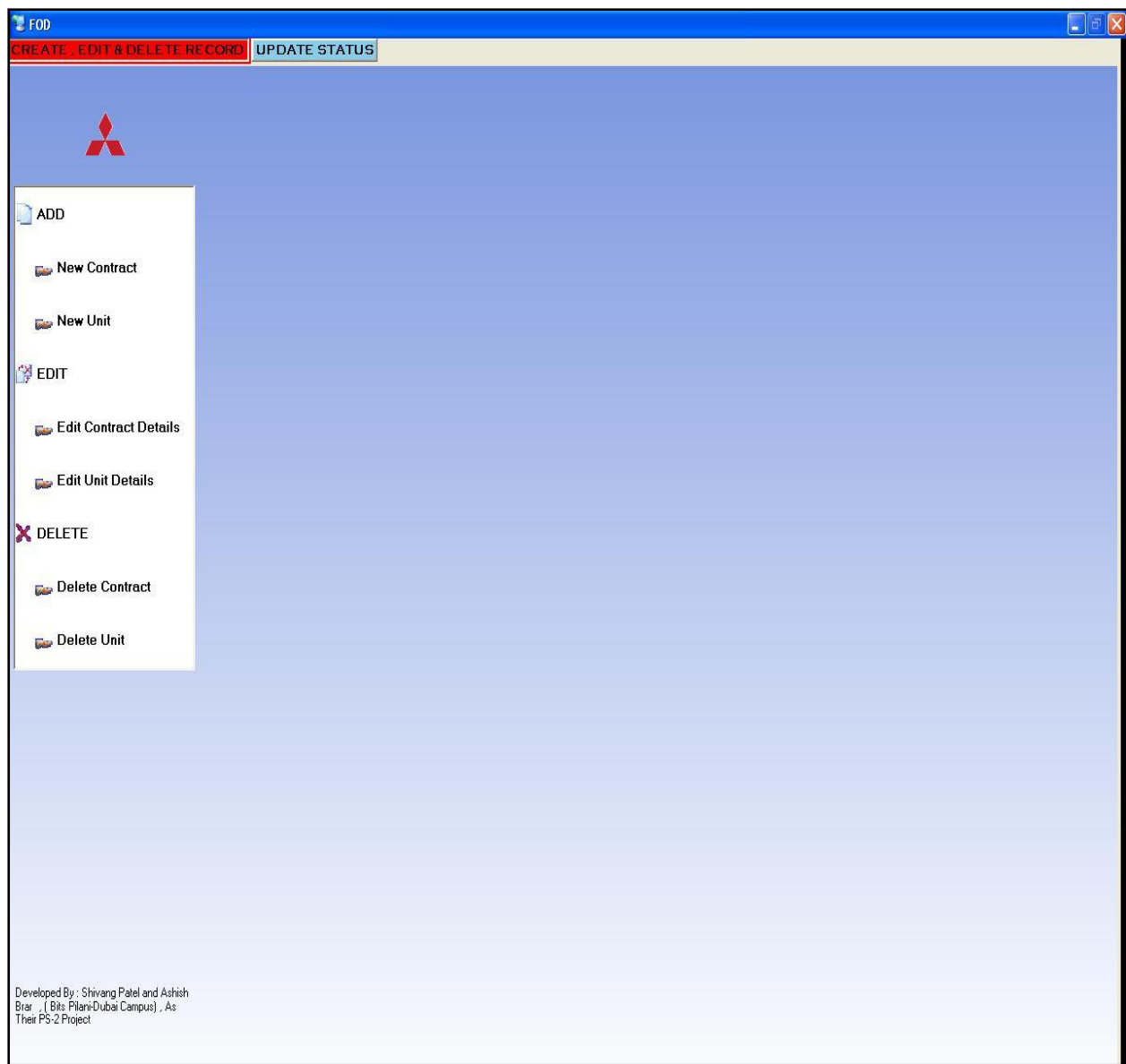


Fig. 3.1

3.3 UPDATE STATUS

Clicking on the first tab on the form which is update status, displays the following page:-

FOD

CREATE , EDIT & DELETE RECORD **UPDATE STATUS**

SEARCH RECORDS

CONTRACT NUMBER : A0438/29

CONTRACT NUMBER : A0438/29 PROJECT NAME : ABUDHABI TRADE CENTER PHASE-II

START DATE : 12/1/2000 12:00:00 AM LOCATION : ABUDHABI

STATUS : Ongoing

UNIT NO : 01 JOB UNIT : 3

UNIT DETAILS

3
160
277
349
444
478
498
512
840
866
890
912
932
950
968
983
997
1011
1213
1225
1236
1246
1254

TYPE : AC-GD (HELICAL)

CONTROL : VFDH

SPEED : 210

No. OF STOPS : 16

BASE HOURS : 1650

CAPACITY : 1600

PRESENT STATUS

STATUS DATE : 5/11/2005 12:00:00 AM

CURRENT STATUS : EXTENDED FREE MAINTENANCE

1 of 29

Developed By : Shivang Patel and Ashish Brar , (Bits Pilani-Dubai Campus) , As Their P5-2 Project

Fig. 3.2

The options that are available on the update status form are:-

1. **Search Records:** - The search field provides the user with the privilege to search for a particular contract based on the contract number entered by the user. Once the search button is clicked, every information related to the project is displayed on the form .All the jobs that are contained in a particular contract are also listed in the left side of the form. The information or details of each of the jobs can also be further viewed along with the status if each job as well as of the contract as a whole.

2. **Unit Details:-**This group box enables the user to get information related to the technical specifications of a particular elevator design. It displays the information which is contained in job unit specifications under a particular contact number.
 - a) Control
 - b) Speed
 - c) No of Stops
 - d) Base Hours
 - e) Capacity

3. **Present Status:-**This group box includes the status date and the current status of the contract and the jobs under it. After the search button is pressed, all the contracts along with their status are displayed in this group box.

It also has an Edit Status button which shows the date on which the status was changed along with the current status of the job. Using it the current status of the job can be changed to extended free maintenance or paid maintenance. Once the edit button is clicked another group box is displayed on the form which allows the user to change the status of the job.

3.4 CREATE, EDIT AND DELETE RECORD:-

This is the tab page which is displayed on the BCS FOD Module form. It can further be categorized into three different forms, i.e. Create Form, Edit Form, and Delete Records Form.

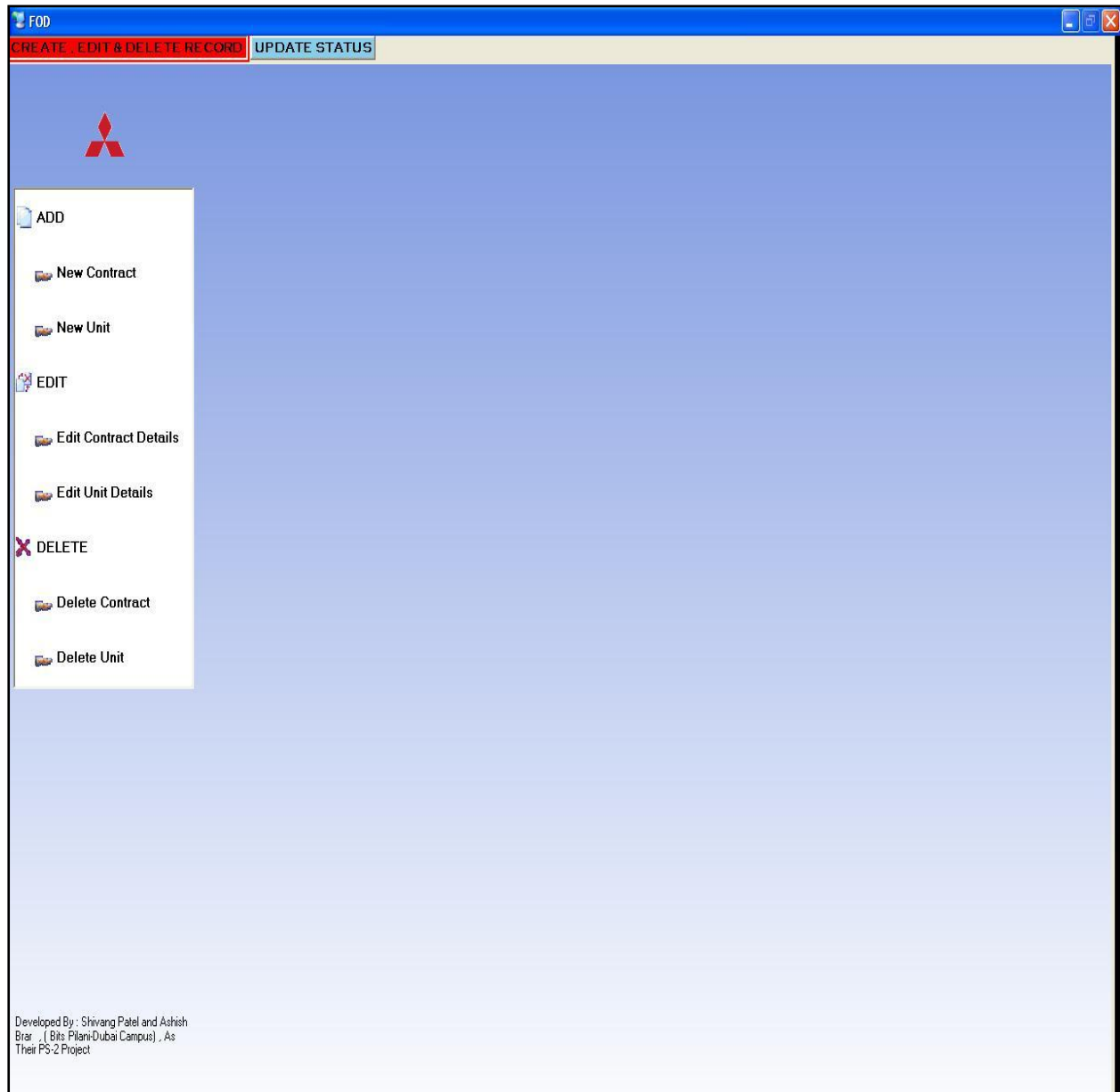


Fig. 3.3

3.4.1 CREATE CONTRACT

This is the form that is used when a new contract has to be added into the database along with the jobs that are contained in the contract. This form has a lot more functions in it.

The screenshot shows a web application window titled "FOD" with a navigation bar containing "CREATE, EDIT & DELETE RECORD" and "UPDATE STATUS". The main content area is titled "CREATE CONTRACT" and features a sidebar on the left with the following menu items:

- ADD
 - New Contract
 - New Unit
- EDIT
 - Edit Contract Details
 - Edit Unit Details
- DELETE
 - Delete Contract
 - Delete Unit

The main form contains the following fields:

- CONTRACT NUMBER :
- PROJECT NAME :
- START DATE : 7/ 6/2005 (dropdown)
- LOCATION : DUBAI (dropdown)
- STATUS :
- PROJECT DETAILS :

A "CREATE CONTRACT" button is located at the bottom center of the form.

Developed By : Shivang Patel and Ashish Brar , (BITS Pilani-Dubai Campus) , As Their PS-2 Project



Fig. 3.4(a)

- The contract is created by entering the contract number, project number, start date, location, and status and project details.

Fields contained in the new contract form:-

1. Contract number
2. Project number
3. Start date
4. Location
5. Status
6. Project details

NOTE: -

-  If any one of the above mentioned details are missing, the software will give an error stating the user to enter complete and unambiguous information.
-  Moreover, if the database already contains a contract with the same contact number, then again an error message will be popped up on the screen, probing the user to re check its values.

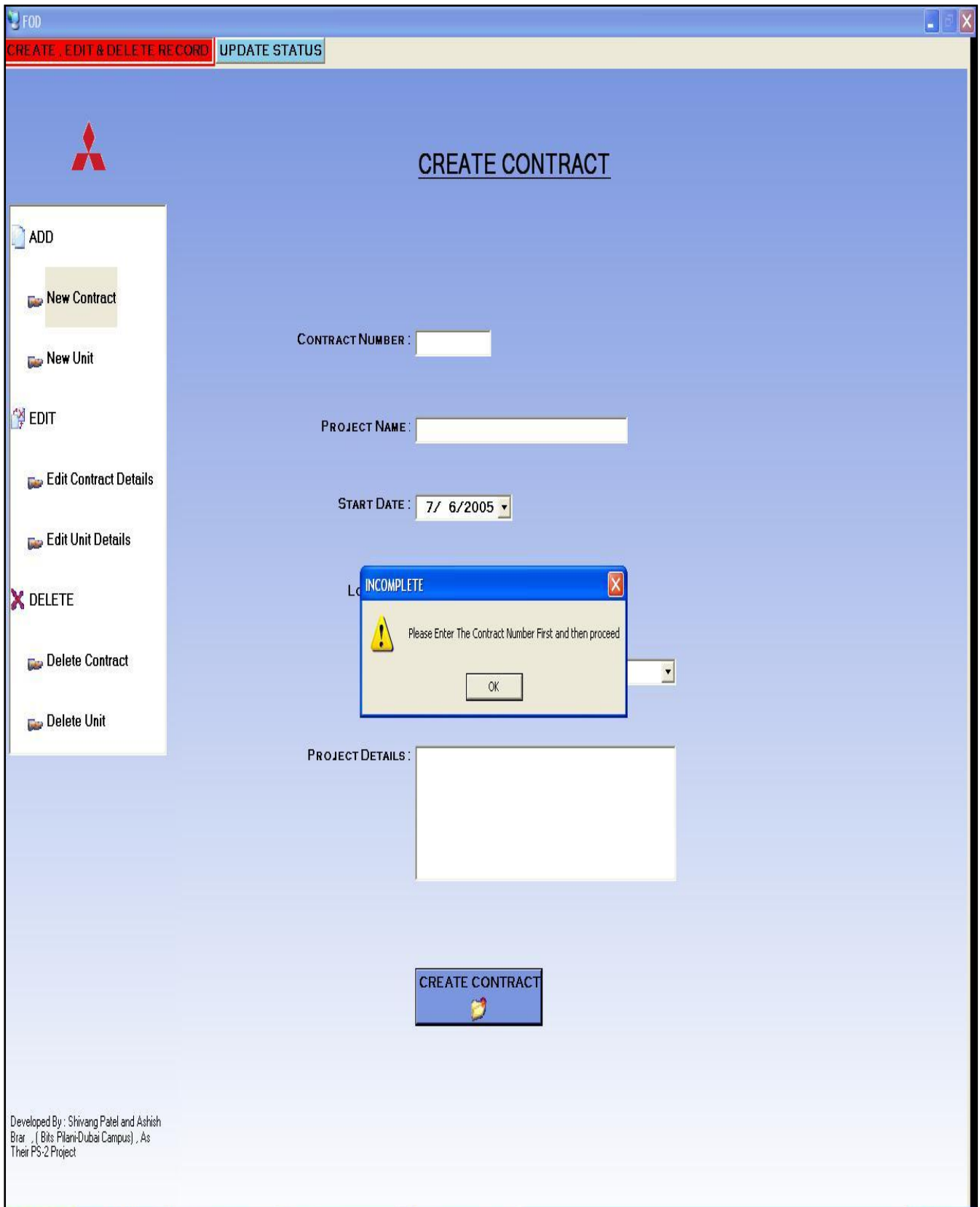


Fig. 3.4(b)

3.4.2 CREATE UNIT

Once the contact has been created, the user can now enter the details of the jobs that are included in the contract along with all the details of the job.

The screenshot shows a web application window titled 'FOD' with a menu bar containing 'CREATE, EDIT & DELETE RECORD' and 'UPDATE STATUS'. The main content area is titled 'CREATE UNIT' and features a Mitsubishi logo in the top left. A navigation menu on the left is organized into three sections: 'ADD' (with 'New Contract' and 'New Unit' buttons), 'EDIT' (with 'Edit Contract Details' and 'Edit Unit Details' buttons), and 'DELETE' (with 'Delete Contract' and 'Delete Unit' buttons). The 'DELETE' section is currently selected, indicated by a red 'X' icon. The main form area contains the following fields and controls:

- CONTRACT NUMBER :
- UNIT No :
- UNIT ID :
- DATE :
- STATUS :
- TYPE :
- CONTROL :
- SPEED :
- No. OF STOPS :
- BASE HOURS :
- CAPACITY :

A 'CREATE UNIT' button with a small icon is located at the bottom center of the form area.

Developed By : Shivang Patel and Ashish Brar , (Bits Pilani-Dubai Campus) , As Their PS-2 Project

Fig. 3.5

The fields that are included in the create unit form are:-

1. Contract Number
2. Unit Number
3. Unit Id
4. Date
5. Status
6. Type
7. Control
8. Speed
9. Number of Stops
10. Base hours
11. Capacity

Once all the information is entered in the form, a new unit will be created under that particular contract name as entered by the user. The changes made in the form will be updated directly in the Melco database.

NOTE: - Based on the type of design selected by the user from the drop down combo box, the speed as well as the control type of the elevator will be automatically displayed on the form.

3.4.3 EDIT CONTRACT

Similarly, the contract details can also be updated or edited based on the selection made by the user.

The screenshot shows a web application window titled 'FOD' with a menu bar containing 'CREATE', 'EDIT & DELETE RECORD', and 'UPDATE STATUS'. The main content area is titled 'EDIT CONTRACT' and features a Mitsubishi logo. A left-hand navigation menu includes sections for 'ADD' (New Contract, New Unit), 'EDIT' (Edit Contract Details, Edit Unit Details), and 'DELETE' (Delete Contract, Delete Unit). The 'EDIT Contract Details' option is highlighted. The main form area contains a search bar with 'A0438/29' and a 'SEARCH' button. Below this, the 'CONTRACT NUMBER' is displayed as 'A0438/29'. Other fields include 'PROJECT NAME' (ABUDHABI TRADE CENTER PHASE-II), 'START DATE' (12/ 1/2000), 'LOCATION' (ABUDHABI), and 'STATUS' (ONGOING). A 'PROJECT DETAILS' section is present but empty. A 'SUBMIT CHANGES' button is located at the bottom right. Footer text at the bottom left reads: 'Developed By : Shivang Patel and Ashish Brar , (Bits Pilani-Dubai Campus) , As Their PS-2 Project'.

Fig. 3.6

First of all, the user can search for a particular contract based on either the contract number or the project number. Once the search results are displayed the user can edit the details of the contract. The fields that are displayed on the contract edit form are:-

1. **Contract Number:** - This cannot be edited by the user directly. In such a case the user will have to completely remove the contract and all its details and even the job details and then submit the complete details of the contract as a new contract. This can be accomplished with the create contract form which has been explained before.
 2. **Project Name**
 3. **Start Date**
 4. **Location**
 5. **Status**
 6. **Project Details**
- Once all the changes are made, the moment user clicks the **SUBMIT CHANGES** button; the changes will be made directly into the database. These changes can again be verified by the user by searching for that particular contract number in the search contract field of the form.

3.4.4 EDIT UNIT

EDIT UNIT

CONTRACT

CONTRACT No: A0438/29

UNIT No: **EDIT**

UNIT Id:

UNIT DETAILS

DATE: 5/11/2005 1 **EDIT**

STATUS: EXTENDED FREE MAINTENANCE

TYPE: AC-GD (HELICAL)

CONTROL: VFDH

SPEED: 210

UNIT SUB-DETAILS

No. OF STOPS:

BASE HOURS: **EDIT**

CAPACITY:

SEARCH

BY:

CONTRACT NUMBER

A0438/29

SEARCH RESULT :

- 3
- 160
- 277
- 349
- 444
- 478
- 498
- 512
- 840
- 866
- 890
- 912
- 932
- 950
- 968
- 983
- 997
- 1011
- 1213
- 1225
- 1236
- 1246
- 1254
- 1262

ADD

- New Contract
- New Unit

EDIT

- Edit Contract Details
- Edit Unit Details**

DELETE

- Delete Contract
- Delete Unit

Developed By : Shivang Patel and Ashish Brar , (Bits Pilani-Dubai Campus) , As Their PS-2 Project

1 OF 29

Fig. 3.7(a)

Since each contract will have numerous jobs inside it. **The procedure that is followed to edit job unit details is as follows:**

- Search is made based on the contract number or project number of a particular project.
- Search results are displayed in the left corner of the form.
- Based on the user's choice, the unit is selected.
- All the details that are relevant to the unit are displayed in their respective fields on the form.
- The details of a unit can be modified by clicking on the **EDIT** button in the respective group box.
- After the changes are made, the user clicks on the **SUBMIT CHANGES** button and again the changes are directed to the database without any redundancy.
- And similarly changes can be incorporated to all the units under a particular contact.


NOTE: -

1. The user will not be able to change any of the details until and unless the **EDIT** button on the form is pressed.
2. The form will also not allow the user to submit incomplete details about the unit and an error message will be displayed on the screen prompting the user to enter the details properly.
3. In case there are no records under a particular contact number or project number, an error message will be popped up informing the user that the particular contract number has no units in it.
4. Please enter another contract number to proceed.

FOD

CREATE, EDIT & DELETE RECORD UPDATE STATUS

EDIT UNIT



ADD

- New Contract
- New Unit


EDIT

- Edit Contract Details
- Edit Unit Details

DELETE

- Delete Contract
- Delete Unit

SEARCH



BY:

CONTRACT NUMBER

FDGDFG

SEARCH RESULT :

CONTRACT

CONTRACT No:

UNIT No :

UNIT ID :

EDIT

UNIT DETAILS

DATE :

STATUS :

TYPE :

EDIT

ERROR

No records were found for CONTRACT NUMBER:FDGDFG
Please enter another contract number and try again

OK

UNIT SUB-DETAILS

No. OF STOPS :

BASE HOURS :

CAPACITY

EDIT

SUBMIT CHANGES

Developed By : Shivang Patel and Ashish Brar , (Bits Pilani-Dubai Campus) , As Their PS-2 Project

0 of 0

Fig. 3.7(b)

3.4.5 DELETE CONTRACT

The delete contract form also works on the same principal as explained earlier. Search is made based on the contract number or the project number and all the results are displayed on the form.

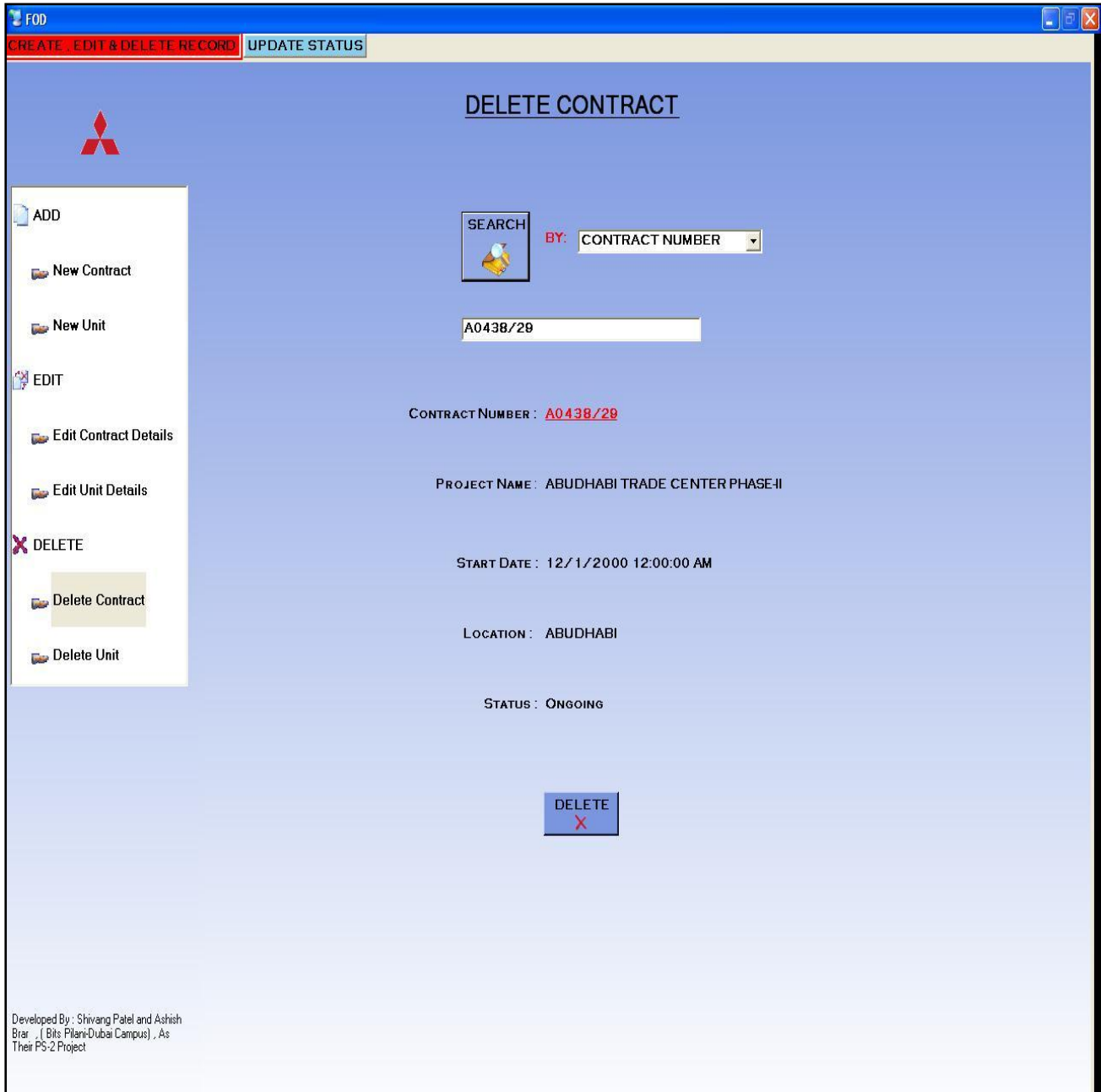


Fig 3.8

NOTE: - The contract details cannot be deleted until and unless the units that are contained in a particular contract are deleted. By doing so, possibility of deleting the records by mistake is removed.

The user will have to remove all the units one by one and only after all the jobs under the contract are deleted, the user can delete the contract.

3.4.6 DELETE UNIT

The delete unit form also works on the same principal as explained earlier. Search is made based on the contract number or the project number and all the results are displayed on the form with all the details pertaining to the unit.

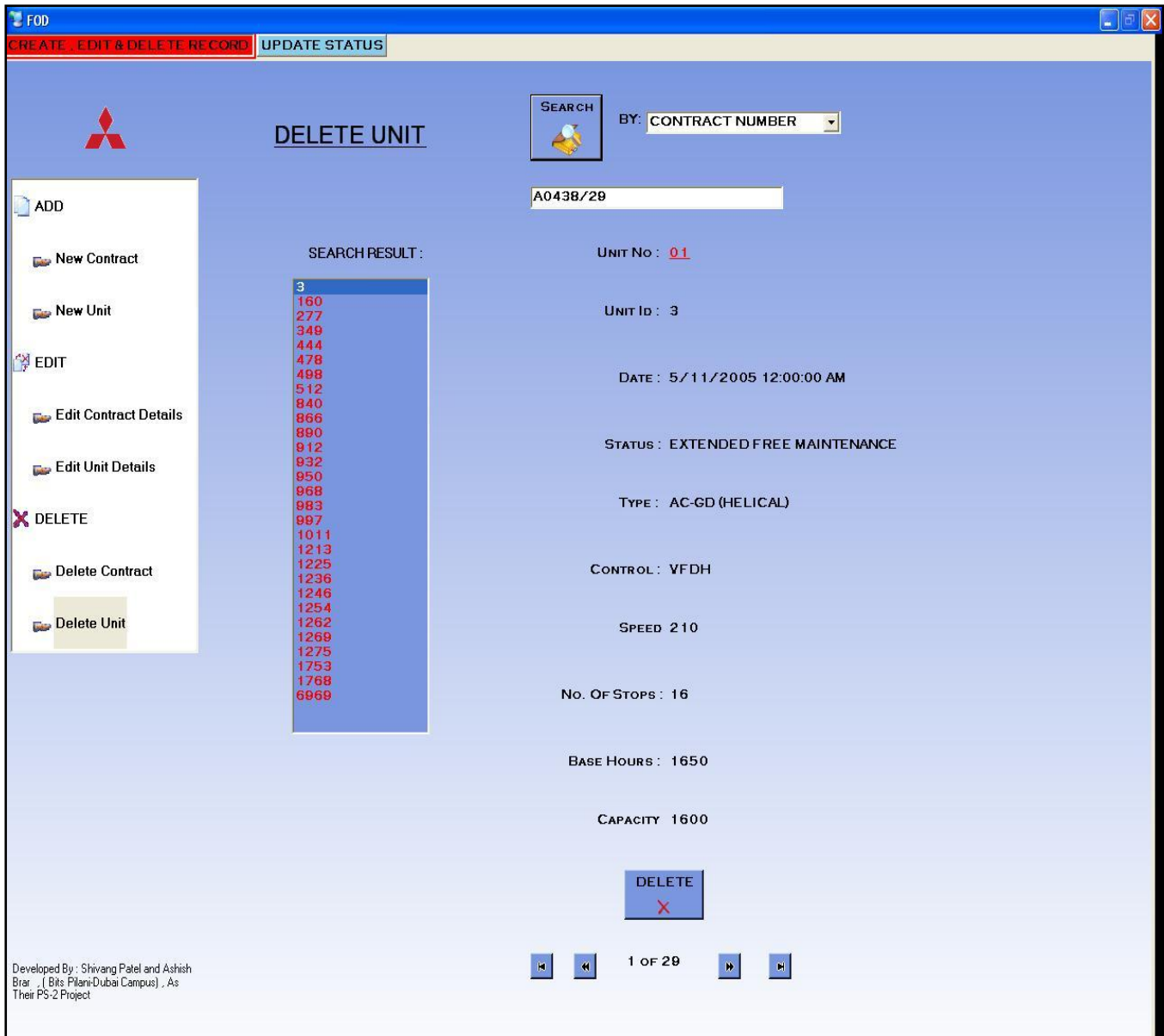


Fig. 3.9

CHAPTER IV

4.1 RESULTS

✚ **Speed Up Data Entry Process For Installation and Maintenance Department of ETA Melco**

As Mentioned earlier, with the use of the software the contract details will be entered and updated in a uniform way, reducing the possibility of any error during the process. The software takes care of all the **Referential Integrity** issues because of proper definition of primary and foreign keys.

✚ The **Error Reporting Mechanism** has been fully developed in order to warn the user for any mistakes in manipulation of data. Also errors in data entry such as null fields are also avoided by **Proper Error Messages**.

✚ The deletion of accidental records by the user is also avoided by prompting user **Confirmation Messages** before the actual submission of data into the database.

✚ The software does not allow the user to enter details of the contract which has already been entered in the database. A message is prompted by the software and thus **preventing redundant entries** into the database.

4.2 RECOMMENDATION FOR FURTHER IMPROVEMENT

If required the following improvements can be established into the DDIBCS

Software:-

- ✚ An Additional Crystal Report Generation service Module can be added on to the software to facilitate automatic generation of monthly reports for the maintenance and installation department of ETA Melco.

- ✚ The program has been designed only for one department (FOD) whereas ETA Melco has nine departments. Further changes can be incorporated into the BCS Software to cater to the needs of all the departments of ETA Melco along with report generation facilities.

4.3 CONCLUSION

The DDBCS (Design and Development of Branch Communication Software) is surely a boon to the company's procedure for the data entry process and follow up by the maintenance department. The projects undertaken will go a long way in reducing the response time of each department .The implementation of the software is a step in the right direction to incorporate various changes in the current work environment of ETA Melco.

Implementation of BCS in the organization will make the task of the concerned officials less tedious, thus saving a considerable amount of time and cost for the organization.

It will not only make work flow more streamlined but also emphasizes on quality work, efficiency and excellence. The BCS will play a major role in being an indispensable asset for the organization and its employees.

REFERENCES

WEBSITES

- **TEXTBOOK REFERENCES**

Nagendra Parashar & RK Mittal, (2003) Elements of Manufacturing Processes, 1st Edition, pp 7-13, Prentice Hall of India, New Delhi.

- [Wrox - Beginning Visual Basic .NET Database Programming](#)
- [Sams - A Programmers Introduction to Visual Basic.NET](#)
- [OReilly's Visual Basic .NET Language in a Nutshell](#)
- [Wrox - Professional ADO.NET](#)
- [MS Press - Microsoft ADO.NET Step by Step](#)
- [Sybex - Mastering Visual Basic .NET](#)
- [Visual Studio .NET Understanding Visual Basic Syntax and Structure](#)

- **WEBSITE REFERENCES**

- www.msdn.microsoft.com/vbasic/
- www.visualbasic.about.com/
- www.programmersheaven.com
- www.vbcode.com
- www.vbnet.mvps.org/
- www.a1vbcode.com
- www.vbnetheaven.com
- www.vbaccelerator.com
- www.extreme-vb.net
- www.codeproject.com/vb/net/
- www.developers.net
- www.experts-exchange.com/Programming

APPENDIX

PROGRAMMING CODE

SOURCE CODE FOR BCS-FOD Module

```
Imports System
Imports System.Data.OleDb
Imports System.IO
Imports System.Data
```

```
Public Class Form3
    Inherits System.Windows.Forms.Form

    Dim str(30) As String

    Dim date11 As Date

    Dim objcurrencymanager As CurrencyManager

    Dim ocm As CurrencyManager

    Dim ocml As CurrencyManager
```

WINDOWS GENERATED CODE NOT INCLUDED

```
Private Sub Form3_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

    objcurrencymanager = CType(Me.BindingContext(ds1.Tables(0)),
CurrencyManager)

    ocm = CType(Me.BindingContext(dsp.jobunits), CurrencyManager)
    ocml = CType(Me.BindingContext(dspl.jobunits), CurrencyManager)

    menu1.ExpandAll()

    'loads all the list boxes

    loadstatus()

    loadtype()
```

```

        loadcontrol(addunit2.SelectedValue)

        loadspeed(addunit2.SelectedValue, addunit3.SelectedValue)

End Sub

Private Sub insertunits()

    Dim str As String

    Dim sqlcommand2 As New OleDbCommand

    ' parameters for sqlcommand2 used to insert units under a job
    sqlcommand2.Parameters.Add("@unit_no", OleDbType.VarChar, 50)
    sqlcommand2.Parameters.Add("@job_no", OleDbType.VarChar, 50)
    sqlcommand2.Parameters.Add("@type", OleDbType.VarChar, 50)
    sqlcommand2.Parameters.Add("@stop", OleDbType.Integer)
    sqlcommand2.Parameters.Add("@speed", OleDbType.Integer)
    sqlcommand2.Parameters.Add("@capacity", OleDbType.Integer)
    sqlcommand2.Parameters.Add("@base_hours", OleDbType.Integer)
    sqlcommand2.Parameters.Add("@status", OleDbType.VarChar, 50)
    sqlcommand2.Parameters.Add("@statusdate", OleDbType.VarChar)
    sqlcommand2.Parameters.Add("@JobUnitId", OleDbType.Integer)
    sqlcommand2.Parameters.Add("@control", OleDbType.VarChar, 50)

    ' command text of the sql command

    sqlcommand2.CommandText = "INSERT INTO jobunits
(unit_no, job_no, type, stop, speed, capacity, base_hours, status, statusDate, J
obUnitId, control) Values(?,?,?,?,?,?,?,?,?,?)"

    ' connection to be used by the sqlcommand

    sqlcommand2.Connection = cn1

    ' values of the paramters in the sqlcommand
    sqlcommand2.Parameters(0).Value = addunit1.Text

    sqlcommand2.Parameters(1).Value = TextBox17.Text

    sqlcommand2.Parameters(2).Value = addunit2.Text

    sqlcommand2.Parameters(3).Value = addunit6.Text

    sqlcommand2.Parameters(4).Value = addunit5.Text

```

```

sqlcommand2.Parameters(5).Value = addunit7.Text

sqlcommand2.Parameters(6).Value = addunit8.Text

sqlcommand2.Parameters(7).Value = addunit10.Text

sqlcommand2.Parameters(8).Value = d22.Value.ToShortDateString

sqlcommand2.Parameters(9).Value = addunit4.Text

sqlcommand2.Parameters(10).Value = addunit3.Text

Try
    cn1.Open()

    sqlcommand2.ExecuteNonQuery()

    cn1.Close()

Catch ex As Exception

    MessageBox.Show(ex.ToString)

End Try

If str = Nothing Then

    MessageBox.Show("Contract Was Successfully Created",
"Success", MessageBoxButtons.OK, MessageBoxIcon.Information)

    TextBox17.Clear()
    addunit1.Clear()
    addunit4.Clear()
    d22.ResetText()
    addunit10.ResetText()
    addunit2.ResetText()
    addunit3.ResetText()
    addunit5.ResetText()
    addunit6.ResetText()
    addunit8.ResetText()
    addunit7.ResetText()

Else

    MessageBox.Show(str & _
ControlChars.NewLine & _
" Please Correct the error and try again", "ERROR",
MessageBoxButtons.OK, MessageBoxIcon.Error)

End If

End Sub

```

```

Private Sub insertjob()

    ' this is used to create a new unit

    Dim sqlcommand1 As New OleDbCommand

    Dim str As String

    sqlcommand1.Connection = cn1

    sqlcommand1.CommandText = "INSERT INTO jobdetails
(job_no,job_start_date,project_name,project_details,location,status,are
aid) Values (?, ?, ?, ?, ?, ?, ?)"

    sqlcommand1.Parameters.Add("@job_no", OleDbType.VarChar, 50)
    sqlcommand1.Parameters.Add("@job_start_date",
OleDbType.VarChar, 50)
    sqlcommand1.Parameters.Add("@project_name", OleDbType.VarChar,
255)
    sqlcommand1.Parameters.Add("@project_details",
OleDbType.VarChar)
    sqlcommand1.Parameters.Add("@location", OleDbType.VarChar, 50)
    sqlcommand1.Parameters.Add("@status", OleDbType.VarChar, 50)
    sqlcommand1.Parameters.Add("@areaid", OleDbType.Integer)

    sqlcommand1.Parameters(0).Value = TextBox2.Text ' job no
    sqlcommand1.Parameters(1).Value = d11.Value.ToShortDateString '
start date
    sqlcommand1.Parameters(2).Value = TextBox1.Text ' Project name
    sqlcommand1.Parameters(3).Value = TextBox5.Text ' project
details
    sqlcommand1.Parameters(4).Value = ComboBox1.Text ' location
    sqlcommand1.Parameters(5).Value = ComboBox2.Text ' status
    sqlcommand1.Parameters(6).Value = ComboBox1.SelectedValue

    cn1.Open()
    Try
        sqlcommand1.ExecuteNonQuery()
    Catch ex As Exception
        MessageBox.Show(ex.Message)
        str = ex.ToString

    End Try
    cn1.Close()
    If str = Nothing Then
        MessageBox.Show("Contract Was Successfully Created",
"Success", MessageBoxButtons.OK, MessageBoxIcon.Information)
        TextBox2.Clear()
        TextBox1.Clear()
    End If
End Sub

```

```

        d11.ResetText()
        ComboBox1.ResetText()
        ComboBox2.ResetText()
        TextBox5.Clear()

    Else
        MessageBox.Show("Record With This Job Number already Exits
, Please try with another number", "ERROR", MessageBoxButtons.OK,
MessageBoxIcon.Asterisk)
    End If

End Sub

Private Sub rec_search()

    ' this is used to search most of the records in all the panels

    OleDbSelectCommand6.CommandText = "SELECT job_no,
job_start_date, project_name, project_details, location, status, areaid
FROM jobdetails WHERE (job_no = ?)"

    OleDbSelectCommand6.Parameters(0).Value = TextBox4.Text

    Try

        searchadapter.Fill(ds1.Tables(1))

    Catch ex As Exception

        MessageBox.Show(ex.ToString)

    End Try

    OleDbSelectCommand6.CommandText = "SELECT unit_no, JobUnitId,
type, control, speed, capacity, stop, base_hours, statusDate, status
FROM jobunits WHERE (job_no = ?)"

    searchadapter.Fill(ds1.Tables(0))

    showposition(Label73, objcurrencymanager)

End Sub

Private Sub Button9_Click_1(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button9.Click

    ' tabpage1 moves the current record to the last position

    objcurrencymanager.Position = objcurrencymanager.Count - 1
    ListBox1.SelectedIndex = objcurrencymanager.Position
    showposition(Label73, objcurrencymanager)

End Sub

```

```
Private Sub Button10_Click_1(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button10.Click
```

```
'tabpagel moves the current record one forward
```

```
objcurrencymanager.Position += 1
ListBox1.SelectedIndex = objcurrencymanager.Position
showposition(Label73, objcurrencymanager)
```

```
End Sub
```

```
Private Sub Button8_Click_1(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button8.Click
```

```
'tabpagel moves the current record one back
```

```
objcurrencymanager.Position -= 1
ListBox1.SelectedIndex = objcurrencymanager.Position
showposition(Label73, objcurrencymanager)
```

```
End Sub
```

```
Private Sub Button3_Click_1(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button3.Click
```

```
'tabpage 1 moves the record to the first position
```

```
objcurrencymanager.Position = 0
ListBox1.SelectedIndex = objcurrencymanager.Position
showposition(Label73, objcurrencymanager)
```

```
End Sub
```

```
Private Sub Button6_Click_1(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button6.Click
```

```
' serch button of tabpagel
```

```
' ds1.Tables(0).Reset()
```

```
ds1.Tables(1).Clear()
```

```
ds1.Tables(0).Clear()
```

```
clearbindings()
```

```
bindings()
```

```
rec_search()
```

```
End Sub
```

```
Private Sub showposition(ByVal lb As Label, ByVal ob As
CurrencyManager)
```

```
' this funtion is used by all pages to diaply the position of
the records u need to pass the label and the currecy manager
```

```
lb.Text = ob.Position + 1 & _
" of " & ob.Count()
```

```

End Sub

Private Sub bindings()

    ' bindings of tab page 1 loaded when the search button is hit

    Label58.DataBindings.Add("text", ds1.Tables(1), "job_no")
    Label59.DataBindings.Add("text", ds1.Tables(1), "project_name")
    Label60.DataBindings.Add("text", ds1.Tables(1),
"job_start_date")
    Label61.DataBindings.Add("text", ds1.Tables(1), "location")
    Label62.DataBindings.Add("text", ds1.Tables(1), "status")

    Label119.DataBindings.Add("text", ds1.Tables(0), "unit_no")
    Label120.DataBindings.Add("text", ds1.Tables(0), "JobUnitId")
    Label116.DataBindings.Add("text", ds1.Tables(0), "Type")
    Label68.DataBindings.Add("text", ds1.Tables(0), "control")
    Label69.DataBindings.Add("text", ds1.Tables(0), "speed")
    Label72.DataBindings.Add("text", ds1.Tables(0), "capacity")
    Label70.DataBindings.Add("text", ds1.Tables(0), "stop")
    Label71.DataBindings.Add("text", ds1.Tables(0), "base_hours")
    Label102.DataBindings.Add("text", ds1.Tables(0), "statusDate")
    Label66.DataBindings.Add("text", ds1.Tables(0), "status")

End Sub

Private Sub clearbindings()

    ' clears the bindings of tab page 1

    Label58.DataBindings.Clear()
    Label59.DataBindings.Clear()
    Label60.DataBindings.Clear()
    Label61.DataBindings.Clear()
    Label62.DataBindings.Clear()
    Label119.DataBindings.Clear()
    Label120.DataBindings.Clear()
    Label66.DataBindings.Clear()
    Label116.DataBindings.Clear()
    Label68.DataBindings.Clear()
    Label69.DataBindings.Clear()
    Label70.DataBindings.Clear()
    Label71.DataBindings.Clear()
    Label72.DataBindings.Clear()
    Label102.DataBindings.Clear()

End Sub

Private Sub ListBox1_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ListBox1.SelectedIndexChanged
    ' moves the currency manager when the reocod is seleted from
the listbox

    'tabpage1

```

```

objcurrencymanager.Position = ListBox1.SelectedIndex

showposition(Label73, objcurrencymanager)

End Sub

Private Sub loadstatus()

    ' loads all the comboboxes showing the status

    lc(ComboBox2, "SELECT status FROM status", 3)
    lc(addunit10, "SELECT status FROM status", 3)
    lc(ComboBox7, "SELECT status FROM status", 3)
    lc(ComboBox4, "SELECT status FROM status", 3)

End Sub

Private Sub lc(ByVal cb As ComboBox, ByVal ct As String, ByVal i As
Integer)
    Dim temprow As DataRow

    ' need to pass a combobox bobo loaded and a commandtext

    ' loads different listboxes list box

    ds2.Tables(i).Clear()

    OleDbSelectCommand4.CommandText = ct

    adapter1.Fill(ds2.Tables(i))

    For Each temprow In ds2.Tables(i).Rows

        cb.Items.Add(temprow.Item(0))

    Next

End Sub

Private Sub loadcontrol(ByVal x As String)

    'function is used by edit and create unit panels
    'loads the control list box

    ds2.Tables(1).Clear()

    OleDbSelectCommand4.CommandText = "SELECT DISTINCT control FROM
Unittype WHERE type = @type"

    OleDbSelectCommand4.Parameters.Clear()

```

```
50) OleDbSelectCommand4.Parameters.Add("@type", OleDbType.VarChar,
OleDbSelectCommand4.Parameters(0).Value = x
```

```
adapter1.Fill(ds2.Tables(1))
```

```
End Sub
```

```
Private Sub loadspeed(ByVal x As String, ByVal y As String)
```

```
'loads the speed list box
```

```
ds2.Tables(2).Clear()
```

```
OleDbSelectCommand4.CommandText = "SELECT DISTINCT speed FROM
Unittype WHERE (type = @type) AND (control = @control)"
```

```
OleDbSelectCommand4.Parameters.Clear()
```

```
OleDbSelectCommand4.Parameters.Add("@type", OleDbType.VarChar,
```

```
50)
```

```
OleDbSelectCommand4.Parameters.Add("@control",
```

```
OleDbType.VarChar, 50)
```

```
OleDbSelectCommand4.Parameters(0).Value = x
```

```
OleDbSelectCommand4.Parameters(1).Value = y
```

```
adapter1.Fill(ds2.Tables(2))
```

```
End Sub
```

```
Private Sub newstatus()
```

```
Dim i As Integer
```

```
Dim sqlcommand3 As OleDbCommand
```

```
Dim sqlcommand4 As OleDbCommand
```

```
sqlcommand3.Connection = cn1
```

```
sqlcommand4.Connection = cn1
```

```
' is used to change the status of the unit under a contract
TRANSACTION
```

```
sqlcommand3.CommandText = "INSERT INTO statusofunits
(contractno,unitno,statusdate,status) Values (?, ?, ?, ?)"
```

```
sqlcommand4.CommandText = "UPDATE jobunits SET statusDate = ? ,
status= ? WHERE job_no = ? AND unit_no = ?"
```

```
i = objcurrencymanager.Position
```

```
datell = DateTimePicker1.Value
```

```
str(20) = datell.ToShortDateString
```

```

' paramter collections for the sqlcommand command

' parameters for sqlcommand3 used to insert new status in teh
statusofunits table

sqlcommand3.Parameters.Add("@job_no", OleDbType.VarChar, 50)
sqlcommand3.Parameters.Add("@unit_no", OleDbType.Integer)
sqlcommand3.Parameters.Add("@date", OleDbType.Date)
sqlcommand3.Parameters.Add("@status", OleDbType.VarChar, 50)

' parameters for sqlcommand4 used to update the status in the
jobunits table

sqlcommand4.Parameters.Add("@statusDate", OleDbType.Date)
sqlcommand4.Parameters.Add("@status", OleDbType.VarChar, 50)
sqlcommand4.Parameters.Add("@job_no", OleDbType.VarChar, 50)
sqlcommand4.Parameters.Add("@unit_no", OleDbType.VarChar, 50)

'values of the paramters

sqlcommand3.Parameters(0).Value = Label58.Text
sqlcommand3.Parameters(1).Value = Label119.Text
sqlcommand3.Parameters(2).Value = str(20)
sqlcommand3.Parameters(3).Value = ComboBox3.Text

sqlcommand4.Parameters(0).Value = str(20)
sqlcommand4.Parameters(1).Value = ComboBox3.Text
sqlcommand4.Parameters(2).Value = Label120.Text
sqlcommand4.Parameters(3).Value = Label119.Text

cn1.Open()
Try
    sqlcommand3.ExecuteNonQuery()
Catch ex As Exception
    MessageBox.Show(ex.ToString)
End Try
Try
    sqlcommand4.ExecuteNonQuery()
Catch ex1 As Exception
    MessageBox.Show(ex1.ToString)
End Try

cn1.Close()

rec_search()

objcurrencymanager.Position = i

showposition(Label73, objcurrencymanager)

End Sub

Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button4.Click
    newstatus()
End Sub

```

```
Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
```

```
    panel2nullcheck()
```

```
End Sub
```

```
Private Sub Button7_Click_1(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button7.Click
```

```
    panel3nullcheck()
```

```
End Sub
```

```
Private Sub menu1_AfterSelect(ByVal sender As System.Object, ByVal e As System.Windows.Forms.TreeViewEventArgs) Handles menu1.AfterSelect  
    str(5) = menu1.SelectedNode.Text
```

```
    ' this is the code which drives the menu of the create edit and delete tabpage
```

```
    Select Case str(5)
```

```
        Case "New Contract"
```

```
            Panel4.Visible = False
```

```
            Panel11.Visible = False
```

```
            Panel3.Visible = False
```

```
            Panel5.Visible = False
```

```
            Panel6.Visible = False
```

```
            Panel2.Visible = True
```

```
            area1.Clear()
```

```
            area.Fill(area1, "areas")
```

```
        Case "New Unit"
```

```
            Panel4.Visible = False
```

```
            Panel11.Visible = False
```

```
            Panel2.Visible = False
```

```
            Panel5.Visible = False
```

```
            Panel6.Visible = False
```

```
            Panel3.Visible = True
```

```
        Case "Edit Contract Details"
```

```
            Panel4.Visible = False
```

```
            Panel2.Visible = False
```

```
            Panel3.Visible = False
```

```
            Panel5.Visible = False
```

```
            Panel6.Visible = False
```

```
            Panel11.Visible = True
```

```
            area1.Clear()
```

```
            area.Fill(area1, "areas")
```

```
            ComboBox6.SelectedIndex = 0
```

```
            OleDbSelectCommand1.CommandText = "SELECT job_no,  
project_name, job_start_date, location, status FROM jobdetails WHERE  
(job_no = ?)"
```

```
        Case "Edit Unit Details"
```

```
            Panel2.Visible = False
```

```

        Panel3.Visible = False
        Panel11.Visible = False
        Panel5.Visible = False
        Panel6.Visible = False
        Panel4.Visible = True
        ComboBox11.SelectedIndex = 0
        OleDbSelectCommand1.CommandText = "SELECT
unit_no,JobUnitId,type,control,speed, capacity, stop, base_hours,
statusDate, status,job_no FROM jobunits WHERE (job_no = ?)"

        Case "Delete Contract"
            Panel2.Visible = False
            Panel3.Visible = False
            Panel11.Visible = False
            Panel4.Visible = False
            Panel6.Visible = False
            Panel5.Visible = True
            ComboBox12.SelectedIndex = 0
            OleDbSelectCommand1.CommandText = "SELECT job_no,
job_start_date, project_name, location, status FROM jobdetails WHERE
(job_no = ?)"

        Case "Delete Unit"
            Panel2.Visible = False
            Panel3.Visible = False
            Panel11.Visible = False
            Panel4.Visible = False
            Panel5.Visible = False
            Panel6.Visible = True
            ComboBox13.SelectedIndex = 0
            OleDbSelectCommand1.CommandText = "SELECT
unit_no,JobUnitId,type,control,speed, capacity, stop, base_hours,
statusDate, status,job_no FROM jobunits WHERE (job_no = ?)"
            End Select

    End Sub

    Private Sub ComboBox13_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ComboBox13.SelectedIndexChanged

        str(13) = ComboBox13.SelectedItem

        Select Case str(13)

            Case "CONTRACT NUMBER"

                OleDbSelectCommand1.CommandText = "SELECT
unit_no,JobUnitId,type,control,speed, capacity, stop, base_hours,
statusDate, status,job_no FROM jobunits WHERE (job_no = ?)"

            Case "UNIT ID"

```

```

        OleDbSelectCommand1.CommandText = "SELECT
unit_no,JobUnitId,type,control,speed, capacity, stop, base_hours,
statusDate, status,job_no FROM jobunits WHERE (JobUnitId = ?)"

        End Select

    End Sub

    Private Sub Button23_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button23.Click

        panel6search()

    End Sub

    Private Sub Panel6_Visiblechanged(ByVal sender As Object, ByVal e
As System.EventArgs) Handles Panel6.VisibleChanged
        If Panel6.Visible = True Then

            panel6bindingsclear()
            dsp1.jobunits.Clear()
            panel6bindings()
            Label92.ResetText()
            Label93.ResetText()
            Label96.ResetText()
            Label97.ResetText()
            Label98.ResetText()
            Label101.ResetText()
            Label99.ResetText()
            Label100.ResetText()
            Label94.ResetText()
            Label95.ResetText()
            showposition(Label80, ocml)

        End If

        If Panel6.Visible = False Then
            panel6bindingsclear()
            dsp1.jobunits.Clear()
        End If

    End Sub

    Private Sub panel6bindings()

        Label92.DataBindings.Add("text", dsp1.jobunits, "unit_no")
        Label93.DataBindings.Add("text", dsp1.jobunits, "JobUnitId")
        Label96.DataBindings.Add("text", dsp1.jobunits, "type")
        Label97.DataBindings.Add("text", dsp1.jobunits, "control")
        Label98.DataBindings.Add("text", dsp1.jobunits, "speed")
        Label101.DataBindings.Add("text", dsp1.jobunits, "capacity")
        Label99.DataBindings.Add("text", dsp1.jobunits, "stop")
        Label100.DataBindings.Add("text", dsp1.jobunits, "base_hours")
        Label94.DataBindings.Add("text", dsp1.jobunits, "statusDate")
    End Sub

```

```

Label95.DataBindings.Add("text", dsp1.jobunits, "status")

End Sub

Private Sub Button21_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button21.Click
    ocml.Position = 0
    ListBox2.SelectedIndex = ocml.Position
    showposition(Label73, ocml)

End Sub

Private Sub Button20_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button20.Click
    ocml.Position -= 1
    ListBox2.SelectedIndex = ocml.Position
    showposition(Label80, ocml)

End Sub

Private Sub Button19_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button19.Click
    ocml.Position += 1
    ListBox2.SelectedIndex = ocml.Position
    showposition(Label80, ocml)

End Sub

Private Sub Button18_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button18.Click
    ocml.Position = ocml.Count - 1
    ListBox2.SelectedIndex = ocml.Position
    showposition(Label80, ocml)

End Sub

Private Sub ListBox2_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ListBox2.SelectedIndexChanged
    ocml.Position = ListBox2.SelectedIndex
    showposition(Label80, ocml)

End Sub

Private Sub panel6bindingsclear()

'clears all the bindings on panel6

Label92.DataBindings.Clear()
Label93.DataBindings.Clear()
Label94.DataBindings.Clear()
Label95.DataBindings.Clear()
Label96.DataBindings.Clear()
Label97.DataBindings.Clear()
Label98.DataBindings.Clear()
Label99.DataBindings.Clear()
Label100.DataBindings.Clear()
Label101.DataBindings.Clear()

```

End Sub

```
Private Sub panel5bindings()  
    'assigns all bidngins on panel 5  
  
    Label65.DataBindings.Add("text", dsp.jobdetails, "job_no")  
    Label35.DataBindings.Add("text", dsp.jobdetails,  
"project_name")  
    Label34.DataBindings.Add("text", dsp.jobdetails,  
"job_start_date")  
    Label33.DataBindings.Add("text", dsp.jobdetails, "location")  
    Label32.DataBindings.Add("text", dsp.jobdetails, "status")
```

End Sub

```
Private Sub panel5bindingsclear()  
  
    'clears all the bindings on panel 5  
  
    Label65.DataBindings.Clear()  
    Label35.DataBindings.Clear()  
    Label34.DataBindings.Clear()  
    Label33.DataBindings.Clear()  
    Label32.DataBindings.Clear()
```

End Sub

```
Private Sub ComboBox12_SelectedIndexChanged(ByVal sender As  
System.Object, ByVal e As System.EventArgs) Handles  
ComboBox12.SelectedIndexChanged  
  
    'changes the search criteria  
  
    str(13) = ComboBox12.SelectedItem  
  
    Select Case str(13)  
  
        Case "CONTRACT NUMBER"  
  
            OleDbSelectCommand1.CommandText = "SELECT job_no,  
job_start_date, project_name, location, status FROM jobdetails WHERE  
(job_no = ?)"  
  
        Case "PROJECT NAME"  
  
            OleDbSelectCommand1.CommandText = "SELECT job_no,  
job_start_date, project_name, location, status FROM jobdetails WHERE  
(project_name = ?)"  
  
    End Select  
  
End Sub
```

```
Private Sub Button5_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button5.Click
```

```
    panel5search()
```

```
End Sub
```

```
Private Sub Panel5_VisibleChanged(ByVal sender As Object, ByVal e As System.EventArgs) Handles Panel5.VisibleChanged
```

```
    If Panel5.Visible = True Then
```

```
        panel5bindingsclear()
```

```
        panel5bindings()
```

```
        dsp.jobdetails.Clear()
```

```
    End If
```

```
End Sub
```

```
Private Sub Panell1_VisibleChanged(ByVal sender As Object, ByVal e As System.EventArgs) Handles Panell1.VisibleChanged
```

```
    If Panell1.Visible = True Then
```

```
        panell1bindingsclear()
```

```
        bindingspanell1()
```

```
        dsp.jobdetails.Clear()
```

```
    End If
```

```
End Sub
```

```
Private Sub bindingspanell1()
```

```
    'activates bindings of panel 1
```

```
    Label105.DataBindings.Add("text", dsp.jobdetails, "job_no")
```

```
    TextBox6.DataBindings.Add("text", dsp.jobdetails, "project_name")
```

```
    d99.DataBindings.Add("text", dsp.jobdetails, "job_start_date")
```

```
    ComboBox5.DataBindings.Add("text", dsp.jobdetails, "location")
```

```
    ComboBox4.DataBindings.Add("text", dsp.jobdetails, "status")
```

```
End Sub
```

```
Private Sub panell1bindingsclear()
```

```
    'clear bindings of panel 1
```

```

Label105.DataBindings.Clear()
TextBox6.DataBindings.Clear()
d99.DataBindings.Clear()
ComboBox5.DataBindings.Clear()
ComboBox4.DataBindings.Clear()

End Sub

Private Sub panellsearch()

    'code which drives the search function of the panell

    Dim dr As DialogResult

    If TextBox8.Text = Nothing Then

        textboxnullcheck(TextBox8, "Search Filed is Empty, Plesase
Try again")

        Return

    End If

    dsp.jobdetails.Clear()

    'add the parameter to the collection

    OleDbSelectCommand1.Parameters.Add("@reader",
OleDbType.VarChar, 50)

    'define the value for the paramter

    OleDbSelectCommand1.Parameters(0).Value = TextBox8.Text

    paneladapter.Fill(dsp.jobdetails)

    If dsp.jobdetails.Rows.Count = 0 Then

        dr = MessageBox.Show("No records were found for " &
ComboBox6.Text & ":" & TextBox8.Text & _
        ControlChars.NewLine & _
        "Please enter another contract number and try again",
"ERROR", MessageBoxButtons.OK, MessageBoxIcon.Stop)

    End If

    Select Case dr

        Case DialogResult.OK

            Button1.Focus()

    End Select

```

```

End Sub

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click

    panellsearch()

End Sub

Private Sub ComboBox6_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ComboBox6.SelectedIndexChanged

    'changes search criteria

    str(13) = ComboBox6.SelectedItem

    Select Case str(13)

        Case "CONTRACT NUMBER"

            OleDbSelectCommand1.CommandText = "SELECT job_no,
project_name, job_start_date, location, status FROM jobdetails WHERE
(job_no = ?)"

        Case "PROJECT NAME"

            OleDbSelectCommand1.CommandText = "SELECT job_no,
project_name, job_start_date, location, status FROM jobdetails WHERE
(project_name = ?)"

    End Select

End Sub

Private Sub Panel4_VisibleChanged(ByVal sender As Object, ByVal e
As System.EventArgs) Handles Panel4.VisibleChanged

    If Panel4.Visible = True Then

        panel4bindingsclear()

        dsp.jobunits.Clear()

        panel4bindings()

        TextBox11.ResetText()

        TextBox10.ResetText()
        ComboBox10.ResetText()
        ComboBox9.ResetText()
        ComboBox8.ResetText()
        TextBox9.ResetText()
        TextBox13.ResetText()
        TextBox12.ResetText()

```

```

        d88.ResetText()
        ComboBox7.ResetText()
        Label106.ResetText()

        showposition(Label31, ocm)
End If

If Panel4.Visible = False Then

    panel4bindingsclear()

    dsp.jobunits.Clear()
End If

End Sub

Private Sub ComboBox11_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ComboBox11.SelectedIndexChanged

    str(13) = ComboBox11.SelectedItem

    Select Case str(13)

        Case "CONTRACT NUMBER"

            OleDbSelectCommand1.CommandText = "SELECT
unit_no,JobUnitId,type,control,speed, capacity, stop, base_hours,
statusDate, status,job_no FROM jobunits WHERE (job_no = ?)"

            Case "UNIT ID"

                OleDbSelectCommand1.CommandText = "SELECT SELECT
unit_no,JobUnitId,type,control,speed, capacity, stop, base_hours,
statusDate, status, job_no FROM jobunits WHERE (JobUnitId = ?)"

    End Select

End Sub

Private Sub SEARCH_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles SEARCH.Click

    panel4search()

End Sub

Private Sub panel4bindings()

    'activates bindings for panel 4

    TextBox11.DataBindings.Add("text", dsp.jobunits, "unit_no")
    TextBox10.DataBindings.Add("text", dsp.jobunits, "JobUnitId")
    Label109.DataBindings.Add("text", dsp.jobunits, "type")
    Label110.DataBindings.Add("text", dsp.jobunits, "control")
    Label111.DataBindings.Add("text", dsp.jobunits, "speed")
    TextBox9.DataBindings.Add("text", dsp.jobunits, "capacity")

```

```
TextBox13.DataBindings.Add("text", dsp.jobunits, "stop")
TextBox12.DataBindings.Add("text", dsp.jobunits, "base_hours")
Label107.DataBindings.Add("text", dsp.jobunits, "statusDate")
Label108.DataBindings.Add("text", dsp.jobunits, "status")
Label106.DataBindings.Add("text", dsp.jobunits, "job_no")
```

End Sub

```
Private Sub panel4bindingsclear()  
    'clears the bindings for panel 4
```

```
Label106.DataBindings.Clear()  
TextBox11.DataBindings.Clear()  
TextBox10.DataBindings.Clear()  
Label107.DataBindings.Clear()  
Label108.DataBindings.Clear()  
Label109.DataBindings.Clear()  
Label110.DataBindings.Clear()  
Label111.DataBindings.Clear()  
TextBox13.DataBindings.Clear()  
TextBox12.DataBindings.Clear()  
TextBox9.DataBindings.Clear()
```

End Sub

```
Private Sub Button17_Click(ByVal sender As System.Object, ByVal e  
As System.EventArgs) Handles Button17.Click
```

```
    ocm.Position = 0  
  
    ListBox3.SelectedIndex = ocm.Position  
  
    showposition(Label31, ocm)
```

End Sub

```
Private Sub Button16_Click(ByVal sender As System.Object, ByVal e  
As System.EventArgs) Handles Button16.Click
```

```
    ocm.Position -= 1  
  
    ListBox3.SelectedIndex = ocm.Position  
  
    showposition(Label31, ocm)
```

End Sub

```
Private Sub Button15_Click(ByVal sender As System.Object, ByVal e  
As System.EventArgs) Handles Button15.Click
```

```
    ocm.Position += 1
```

```

        ListBox3.SelectedIndex = ocm.Position

        showposition(Label31, ocm)

    End Sub

    Private Sub Button13_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button13.Click

        ocm.Position = ocm.Count

        ListBox3.SelectedIndex = ocm.Position

        showposition(Label31, ocm)

    End Sub

    Private Sub ListBox3_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ListBox3.SelectedIndexChanged

        ocm.Position = ListBox3.SelectedIndex

        panel4reset()

    End Sub

    Private Sub addunit2_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
addunit2.SelectedIndexChanged

        're loads the control list box

        ds2.Tables(1).Clear()

        loadcontrol(addunit2.SelectedValue)

        're loads the speed list box

        ds2.Tables(2).Clear()

        loadspeed(addunit2.SelectedValue, addunit3.SelectedValue)

    End Sub

    Private Sub ComboBox10_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ComboBox10.SelectedIndexChanged

        loadcontrol(ComboBox10.SelectedValue)

        Label109.Text = ComboBox10.Text

    End Sub

```

```

Private Sub addunit3_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
addunit3.SelectedIndexChanged

    'refreshes the speed list box when control is changed

    loadspeed(addunit2.SelectedValue, addunit3.SelectedValue)

End Sub

Private Sub Button14_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button14.Click

    panel4nullcheck()

End Sub

Private Sub Button12_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button12.Click

    panell1nullcheck()

End Sub

Private Sub Button22_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button22.Click
    Dim dr As DialogResult

    dr = MessageBox.Show("You are about to delete a unit from
Contract: " & TextBox16.Text & _
ControlChars.NewLine & _
"Having Unit Id : " & Label93.Text & _
ControlChars.NewLine & _
"Do you want to proceed ?", "Warning", MessageBoxButtons.YesNo,
MessageBoxIcon.Question)

    Select Case dr

        Case DialogResult.Yes

            unitdelete()

        Case DialogResult.No

            Button23.Focus()

    End Select

End Sub

Private Sub loadtype()

    ds2.Tables(0).Clear()

```

```

        OleDbSelectCommand4.CommandText = "SELECT DISTINCT type FROM
Unitttype"

        OleDbSelectCommand4.Parameters.Clear()

        adapter1.Fill(ds2.Tables(0))

    End Sub

    Private Sub Button11_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button11.Click

        GroupBox1.Visible = True

        GroupBox1.Enabled = True

        lc(ComboBox3, "SELECT status FROM status", 3)

    End Sub

    Private Sub panel2nullcheck()

        Dim dr As DialogResult
        Dim dr1 As DialogResult
        Dim str As String

        If TextBox2.Text = Nothing Then
            dr = MessageBox.Show("Please Enter The Contract Number
First and then proceed", "INCOMPLETE", MessageBoxButtons.OK,
MessageBoxIcon.Warning)
            Select Case dr
                Case DialogResult.OK
                    TextBox2.Focus()
            End Select
            Return
        End If

        If TextBox1.Text = Nothing Then
            dr = MessageBox.Show("Please Enter The name of the project
and then proceed", "INCOMPLETE", MessageBoxButtons.OK,
MessageBoxIcon.Warning)
            Select Case dr
                Case DialogResult.OK
                    TextBox1.Focus()
            End Select
            Return
        End If

        If ComboBox1.SelectedValue = Nothing Then
            dr = MessageBox.Show("Please Select the location of the
Project and Proceed", "INCOMPLETE", MessageBoxButtons.OK,
MessageBoxIcon.Warning)
            Select Case dr
                Case DialogResult.OK
                    ComboBox1.Focus()
            End Select
        End If
    End Sub

```

```

        Return
    End If

    If ComboBox2.SelectedItem = Nothing Then
        dr = MessageBox.Show("Plesage Select the status of contract
and proceed", "INCOMPLETE", MessageBoxButtons.OK,
MessageBoxIcon.Warning)
        Select Case dr
            Case DialogResult.OK
                ComboBox2.Focus()
        End Select
        Return
    End If

    dr1 = MessageBox.Show("You are about to create a new contract
with the following details :" & _
ControlChars.NewLine & _
"Contract NO. : " & TextBox2.Text & _
ControlChars.NewLine & _
"Project Name : " & TextBox1.Text & _
ControlChars.NewLine & _
"Start Date : " & d11.Value.ToShortDateString & _
ControlChars.NewLine & _
"Location : " & ComboBox1.Text & _
ControlChars.NewLine & _
"Status : " & ComboBox2.Text & _
ControlChars.NewLine & _
" Do you want to create the contract ?", "WARNING",
MessageBoxButtons.YesNo, MessageBoxIcon.Question)

    Select Case dr1
        Case DialogResult.Yes
            insertjob()
        Case DialogResult.No
            TextBox2.Focus()
    End Select

End Sub

Private Sub panellnullcheck()
    Dim dr As DialogResult

    Dim dr1 As DialogResult

    If TextBox6.Text = Nothing Then

        textboxnullcheck(TextBox6, "Please Enter The name of the
project and then proceed")

        Return

    End If

    If ComboBox5.Text = Nothing Then

```

```
        comboboxnullcheck(ComboBox5, "Please Select the location of  
the Project and Proceed")
```

```
        Return
```

```
    End If
```

```
    If ComboBox4.Text = Nothing Then
```

```
        comboboxnullcheck(ComboBox4, "Plesage Select the status of  
contract and proceed")
```

```
        Return
```

```
    End If
```

```
    dr1 = MessageBox.Show("Submit the following changes to contract  
NO : " & Label105.Text & _  
        ControlChars.NewLine & _  
        "Project Name : " & TextBox6.Text & _  
        ControlChars.NewLine & _  
        "Start date : " & d99.Value.ToShortDateString & _  
        ControlChars.NewLine & _  
        "Location : " & ComboBox5.Text & _  
        ControlChars.NewLine & _  
        "Status : " & ComboBox4.Text & _  
        ControlChars.NewLine & _  
        "Do u want to submit the changes ? ", "WARNING",  
    MessageBoxButtons.YesNo, MessageBoxIcon.Question)
```

```
    Select Case dr1
```

```
        Case DialogResult.Yes
```

```
            editcontract()
```

```
        Case DialogResult.No
```

```
            Button1.Focus()
```

```
    End Select
```

```
End Sub
```

```
Private Sub editcontract()
```

```
    Dim contractupdatecommand As New OleDbCommand
```

```
    Dim str As String
```

```
    contractupdatecommand.Connection = cn1
```

```
    ' contractupdatecommand command text
```

```
contractupdatecommand.CommandText = "UPDATE jobdetails SET
job_no=?,project_name=?,job_start_date=?,location=?,status=?,project_de
tails=? WHERE job_no=?"
```

```
'parameters for contract edit panel
```

```
contractupdatecommand.Parameters.Add("@job_no",
OleDbType.VarChar, 50)
contractupdatecommand.Parameters.Add("@project_name",
OleDbType.VarChar, 50)
contractupdatecommand.Parameters.Add("@job_start_date",
OleDbType.VarChar, 50)
contractupdatecommand.Parameters.Add("@location",
OleDbType.VarChar, 50)
contractupdatecommand.Parameters.Add("@status",
OleDbType.VarChar, 50)
contractupdatecommand.Parameters.Add("@project_details",
OleDbType.VarChar, 255)
contractupdatecommand.Parameters.Add("@search",
OleDbType.VarChar, 50)
```

```
' contractupdatecommand parameter values
```

```
contractupdatecommand.Parameters(0).Value = Label105.Text
contractupdatecommand.Parameters(1).Value = TextBox6.Text
contractupdatecommand.Parameters(2).Value =
d99.Value.ToShortDateString
contractupdatecommand.Parameters(3).Value = ComboBox5.Text
contractupdatecommand.Parameters(4).Value = ComboBox4.Text
contractupdatecommand.Parameters(5).Value = TextBox7.Text
contractupdatecommand.Parameters(6).Value = TextBox8.Text
```

```
Try
```

```
cn1.Open()
contractupdatecommand.ExecuteNonQuery()
cn1.Close()
```

```
Catch ex As Exception
```

```
str = ex.Message
```

```
End Try
```

```
If str = Nothing Then
```

```
MessageBox.Show("Changes Have Been Affected", "Changes
Made", MessageBoxButtons.OK, MessageBoxIcon.Information)
panellsearch()
```

```
Else
```

```
MessageBox.Show(str & _
ControlChars.NewLine & _
" Please Correct the error and try again", "ERROR",
MessageBoxButtons.OK, MessageBoxIcon.Error)
```

```
Return
```

```

        End If

    End Sub

    Private Sub panel4nullcheck()

        Dim dr As DialogResult
        Dim dr1 As DialogResult

        If TextBox11.Text = Nothing Then

            textboxnullcheck(TextBox11, "Please Enter The Unit NO. and
then proceed")

            Return
        End If

        If TextBox10.Text = Nothing Then
            textboxnullcheck(TextBox10, "Please Enter The Unit ID and
then proceed")

            Return
        End If

        If ComboBox7.Text = Nothing Then
            comboboxnullcheck(ComboBox7, "Please Select The Status and
than proceed")

            Return
        End If

        If ComboBox10.Text = Nothing Then
            comboboxnullcheck(ComboBox10, "Please Select The Status
and than proceed")
            Return
        End If

        If ComboBox9.Text = Nothing Then
            comboboxnullcheck(ComboBox9, "Please Enter the Control type
and proceed")
            Return
        End If

        If ComboBox8.Text = Nothing Then
            comboboxnullcheck(ComboBox8, "Please Enter the Speed of the
unit and proceed")
            Return
        End If

        If TextBox13.Text = Nothing Then
            textboxnullcheck(TextBox13, "Plesase Enter the No of Stops
for the Unit and Proceed")
            Return
        End If
    
```

```

        If TextBox12.Text = Nothing Then
            textboxnullcheck(TextBox12, "Please Enter the Base Hours
fort the Unit and Proceed")
            Return
        End If

        If TextBox9.Text = Nothing Then
            textboxnullcheck(TextBox9, "Please Enter the Capacity for
the Unit and Proceed")
            Return
        End If

        dr1 = MessageBox.Show("SUBMIT THE FOLLOWING:" & _
            ControlChars.NewLine & _
            "Unit NO: " & TextBox11.Text & _
            ControlChars.NewLine & _
            "Unit ID: " & TextBox10.Text & _
            ControlChars.NewLine & _
            "Date : " & d88.Value.ToShortDateString & _
            ControlChars.NewLine & _
            "Status : " & ComboBox7.Text & _
            ControlChars.NewLine & _
            "Type : " & ComboBox10.Text & _
            ControlChars.NewLine & _
            "Control : " & ComboBox9.Text & _
            ControlChars.NewLine & _
            "Speed : " & ComboBox8.Text & _
            ControlChars.NewLine & _
            "No of Stops : " & TextBox13.Text & _
            ControlChars.NewLine & _
            "Base Hours : " & TextBox12.Text & _
            ControlChars.NewLine & _
            "Capacity : " & TextBox9.Text & _
            ControlChars.NewLine & _
            "Proceed ?", "WARNING", MessageBoxButtons.YesNo,
MessageBoxIcon.Question)

        Select Case dr1
            Case DialogResult.Yes
                unitedit()
            Case DialogResult.No
                SEARCH.Focus()
        End Select

    End Sub

    Private Sub textboxnullcheck(ByVal tt As TextBox, ByVal msg As
String)

        Dim dr As DialogResult

```

```

        dr = MessageBox.Show(msg, "INCOMPLETE", MessageBoxButtons.OK,
        MessageBoxIcon.Warning)

        Select Case dr
            Case DialogResult.OK
                tt.Focus()
        End Select

    End Sub

    Private Sub comboboxnullcheck(ByVal cb As ComboBox, ByVal msg As
String)
        Dim dr As DialogResult

        dr = MessageBox.Show(msg, "INCOMPLETE", MessageBoxButtons.OK,
        MessageBoxIcon.Warning)

        Select Case dr

            Case DialogResult.OK
                cb.Focus()
        End Select

    End Sub

    Private Sub unitedit()

        Dim unitupdatecommand As New OleDbCommand

        Dim str As String

        unitupdatecommand.Connection = cn1

        unitupdatecommand.CommandText = "UPDATE jobunits SET
job_no=?,unit_no=?,JobUnitId=?,statusDate=?,status=?,type=?,control=?,s
peed=?,stop=?,base_hours=?,capacity=? WHERE JobUnitId=?"

        ' parameters for unit edit panel

        unitupdatecommand.Parameters.Add("@job_no", OleDbType.VarChar,
50)
        unitupdatecommand.Parameters.Add("@unit_no", OleDbType.VarChar,
50)
        unitupdatecommand.Parameters.Add("@unit_id", OleDbType.VarChar,
50)
        unitupdatecommand.Parameters.Add("@statusdate",
OleDbType.VarChar, 50)
        unitupdatecommand.Parameters.Add("@status", OleDbType.VarChar,
50)
        unitupdatecommand.Parameters.Add("@type", OleDbType.VarChar,
50)
        unitupdatecommand.Parameters.Add("@control", OleDbType.VarChar,
50)
        unitupdatecommand.Parameters.Add("@speed", OleDbType.VarChar,
50)
        unitupdatecommand.Parameters.Add("@stop", OleDbType.VarChar,
50)

```

```

        unitupdatecommand.Parameters.Add("@base_hours",
OleDbType.VarChar, 50)
        unitupdatecommand.Parameters.Add("@capacity",
OleDbType.VarChar, 50)
        unitupdatecommand.Parameters.Add("@search", OleDbType.VarChar,
50)

' values of the paramters

unitupdatecommand.Parameters(0).Value = Label106.Text
unitupdatecommand.Parameters(1).Value = TextBox11.Text
unitupdatecommand.Parameters(2).Value = TextBox10.Text
unitupdatecommand.Parameters(3).Value = Label107.Text
unitupdatecommand.Parameters(4).Value = Label108.Text
unitupdatecommand.Parameters(5).Value = Label109.Text
unitupdatecommand.Parameters(6).Value = Label110.Text
unitupdatecommand.Parameters(7).Value = Label111.Text
unitupdatecommand.Parameters(8).Value = TextBox13.Text
unitupdatecommand.Parameters(9).Value = TextBox12.Text
unitupdatecommand.Parameters(10).Value = TextBox9.Text
unitupdatecommand.Parameters(11).Value = TextBox10.Text

Try
    cn1.Open()
    unitupdatecommand.ExecuteNonQuery()
    cn1.Close()

Catch ex As Exception
    str = ex.Message
End Try
If str = Nothing Then

    MessageBox.Show("Requested changes have been affected",
"Changes Made", MessageBoxButtons.OK, MessageBoxIcon.Information)
    panel4search()
    panel4reset()
Else
    MessageBox.Show(str & ",Please Correct the error and try
again", "ERROR", MessageBoxButtons.OK, MessageBoxIcon.Error)
    Return
End If

End Sub
Private Sub panel6search()

    If TextBox16.Text = Nothing Then

        textboxnullcheck(TextBox16, "Search Filed is Empty, Plesase
Try again")

        Return

    End If

' add the paramter to the collection

```

```

50) OleDbSelectCommand1.Parameters.Add("@value", OleDbType.VarChar,

' clear the dataset of earlier values if any
dsp1.jobunits.Clear()
'define the value for the paramter

OleDbSelectCommand1.Parameters(0).Value = TextBox16.Text

paneladapter.Fill(dsp1.jobunits)

showposition(Label80, ocml)

End Sub

Private Sub panel5search()

If TextBox15.Text = Nothing Then
textboxnullcheck(TextBox15, "Search Filed is Empty, Plesase
Try again")
Return
End If

'add the parameter to the collection
OleDbSelectCommand1.Parameters.Add("@value5",
OleDbType.VarChar, 50)

'clear the dataset of any earlier values
dsp.Clear()

'define the value for the paramter
OleDbSelectCommand1.Parameters(0).Value = TextBox15.Text
paneladapter.Fill(dsp.jobdetails)

End Sub

Private Sub panel4search()

Dim dr As DialogResult
Dim row As DataRow

If TextBox14.Text = Nothing Then
textboxnullcheck(TextBox14, "Search Filed is Empty, Plesase
Try again")
Return
End If

panel4bindingsclear()

dsp.jobunits.Clear()

panel4bindings()

'add the parameter to the collection

```

```

50) OleDbSelectCommand1.Parameters.Add("@xyz", OleDbType.VarChar,

'define the value for the parameter
OleDbSelectCommand1.Parameters(0).Value = TextBox14.Text
dsp.jobunits.BeginLoadData()

Try
    paneladapter.Fill(dsp.jobunits)
Catch ex As Exception
    MessageBox.Show(ex.ToString)
End Try
dsp.jobunits.EndLoadData()
showposition(Label31, ocm)
If dsp.jobunits.Rows.Count = 0 Then
    dr = MessageBox.Show("No records were found for " &
ComboBox11.Text & ":" & TextBox14.Text & _
        ControlChars.NewLine & _
        "Please enter another contract number and try
again", "ERROR", MessageBoxButtons.OK, MessageBoxIcon.Stop)
End If

Select Case dr
    Case DialogResult.OK
        SEARCH.Focus()
End Select

End Sub

Private Sub panel3nullcheck()

    Dim dr As DialogResult
    Dim dr1 As DialogResult

    If TextBox17.Text = Nothing Then

        textboxnullcheck(addunit1, "Please Enter The Unit NO. and
then proceed")

        Return
    End If

    If addunit1.Text = Nothing Then

        textboxnullcheck(addunit1, "Please Enter The Unit NO. and
then proceed")

        Return
    End If

    If addunit4.Text = Nothing Then
        textboxnullcheck(addunit4, "Please Enter The Unit ID and
then proceed")

        Return
    End If

```

```

        If addunit10.Text = Nothing Then
            comboboxnullcheck(addunit10, "Please Select The Status and
than proceed")

            Return
        End If

        If addunit2.Text = Nothing Then
            comboboxnullcheck(addunit2, "Please Select The Status and
than proceed")
            Return
        End If

        If addunit3.Text = Nothing Then
            comboboxnullcheck(addunit3, "Please Enter the Control type
and proceed")
            Return
        End If

        If addunit5.Text = Nothing Then
            comboboxnullcheck(addunit5, "Please Enter the Speed of the
unit and proceed")
            Return

        End If

        If addunit6.Text = Nothing Then
            textboxnullcheck(addunit6, "Plesase Enter the No of Stops
for the Unit and Proceed")
            Return

        End If

        If addunit8.Text = Nothing Then
            textboxnullcheck(addunit8, "Please Enter the Base Hours
fort the Unit and Proceed")
            Return
        End If

        If addunit7.Text = Nothing Then
            textboxnullcheck(addunit7, "Please Enter the Capacity for
the Unit and Proceed")
            Return
        End If

        dr1 = MessageBox.Show("Create a Unit with following details :")
& _
            ControlChars.NewLine & _
            "Contract No. : " & TextBox17.Text & _
            ControlChars.NewLine & _
            "Unit NO: " & addunit1.Text & _
            ControlChars.NewLine & _
            "Unit ID: " & addunit4.Text & _
            ControlChars.NewLine & _
            "Date : " & d22.Value.ToShortDateString & _
            ControlChars.NewLine & _

```

```

        "Status : " & addunit10.Text & _
        ControlChars.NewLine & _
        "Type : " & addunit2.Text & _
        ControlChars.NewLine & _
        "Control : " & addunit3.Text & _
        ControlChars.NewLine & _
        "Speed : " & addunit5.Text & _
        ControlChars.NewLine & _
        "No of Stops : " & addunit6.Text & _
        ControlChars.NewLine & _
        "Base Hours : " & addunit8.Text & _
        ControlChars.NewLine & _
        "Capacity : " & addunit7.Text & _
        ControlChars.NewLine & _
        "Proceed ?", "WARNING", MessageBoxButtons.YesNo,
MessageBoxIcon.Question)

    Select Case dr1
        Case DialogResult.Yes
            insertunits()
        Case DialogResult.No
            Button7.Focus()
    End Select

End Sub

Private Sub unitdelete()

    Dim unitdeletecommand As New OleDbCommand
    Dim str As String

    unitdeletecommand.CommandText = "DELETE jobunits.* FROM
jobunits WHERE JobUnitId=?"

    ' unit connection
    unitdeletecommand.Connection = cn1

    ' parameters for deleting a unit

    unitdeletecommand.Parameters.Add("@search", OleDbType.VarChar,
50)
    'paramter value
    unitdeletecommand.Parameters(0).Value = Label93.Text

    cn1.Open()
    Try
        unitdeletecommand.ExecuteNonQuery()
    Catch ex As Exception
        str = ex.Message
    End Try
    cn1.Close()

    If str = Nothing Then
        MessageBox.Show("The Unit was Deleted successfully",
"Successfully Deleted", MessageBoxButtons.OK,
MessageBoxIcon.Information)
        panel6search()
    End If
End Sub

```

```

        Else
            MessageBox.Show(str & "Please Correct the error and try
again", "ERROR", MessageBoxButtons.OK, MessageBoxIcon.Error)
            Return
        End If
    End Sub

Private Sub contractdelete()

    Dim checkunit As New OleDbCommand
    Dim contractdeletecommand As New OleDbCommand
    Dim rd As OleDbDataReader
    Dim str As String
    Dim i As Integer

    contractdeletecommand.Connection = cn1
    checkunit.Connection = cn1

    contractdeletecommand.CommandText = "DELETE jobdetails.* FROM
jobdetails WHERE job_no=?"

    checkunit.CommandText = "SELECT jobunits.* FROM jobunits WHERE
job_no=?"

    'paramters for deleteing a contract

    contractdeletecommand.Parameters.Add("@search",
OleDbType.VarChar, 50)

    'parameters for deleting all units under a contract

    checkunit.Parameters.Add("@search", OleDbType.VarChar, 50)

    'values of the parameters

    checkunit.Parameters(0).Value = Label65.Text
    contractdeletecommand.Parameters(0).Value = Label65.Text
    i = 0

Try
    cn1.Open()
    rd = checkunit.ExecuteReader()

    While rd.Read
        i += 1
    End While
    rd.Close()
    cn1.Close()
    If i <> 0 Then
        MessageBox.Show("Please Delete The units under this
contract first and than try again", "Message", MessageBoxButtons.OK,
MessageBoxIcon.Error)
        Return
    Else
        cn1.Open()
        contractdeletecommand.ExecuteNonQuery()
        cn1.Close()
    End If
End Try

```

```

        End If
    Catch ex As Exception
        str = ex.Message

    End Try

    If str = Nothing Then

        MessageBox.Show("Contract Deleted", "Operation Complete",
        MessageBoxButtons.OK, MessageBoxIcon.Information)
        TextBox15.Clear()
        Label65.ResetText()
        Label35.ResetText()
        Label34.ResetText()
        Label33.ResetText()
        Label32.ResetText()
    Else
        MessageBox.Show(str & ",Please Correct the error and try
again", "ERROR", MessageBoxButtons.OK, MessageBoxIcon.Error)
    End If

End Sub

```

```

Private Sub DELETE_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles DELETE.Click
    Dim dr As DialogResult

    dr = MessageBox.Show("You are about to delete the Contract : "
& Label65.Text & _
ControlChars.NewLine & _
"Do you want to proceed ?", "WARNING", MessageBoxButtons.YesNo,
MessageBoxIcon.Question)

    Select Case dr

        Case DialogResult.Yes
            contractdelete()

        Case DialogResult.No
            Button5.Focus()
    End Select

End Sub

```

```

Private Sub d88_ValueChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles d88.ValueChanged

```

```

        Label107.Text = d88.Value.ToShortDateString

    End Sub

    Private Sub ComboBox7_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ComboBox7.SelectedIndexChanged

        Label108.Text = ComboBox7.Text

    End Sub

    Private Sub ComboBox9_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ComboBox9.SelectedIndexChanged

        Label110.Text = ComboBox9.Text

    End Sub

    Private Sub ComboBox8_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ComboBox8.SelectedIndexChanged

        Label111.Text = ComboBox8.Text

    End Sub

    Private Sub Button25_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button25.Click

        d88.Visible = True
        ComboBox7.Visible = True
        ComboBox10.Visible = True
        ComboBox9.Visible = True
        ComboBox8.Visible = True

    End Sub

    Private Sub Button24_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button24.Click
        TextBox11.ReadOnly = False
        TextBox10.ReadOnly = False
        TextBox11.BackColor = Color.White
        TextBox10.BackColor = Color.White

    End Sub

    Private Sub Button26_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button26.Click

        TextBox13.ReadOnly = False
        TextBox12.ReadOnly = False
        TextBox9.ReadOnly = False
        TextBox13.BackColor = Color.White
        TextBox12.BackColor = Color.White
        TextBox9.BackColor = Color.White

```

```

End Sub

Private Sub panel4reset()

    d88.Visible = False
    ComboBox7.Visible = False
    ComboBox10.Visible = False
    ComboBox9.Visible = False
    ComboBox8.Visible = False

    TextBox11.ReadOnly = True
    TextBox10.ReadOnly = True
    TextBox13.ReadOnly = True
    TextBox12.ReadOnly = True
    TextBox9.ReadOnly = True

End Sub
Private Sub TabControll1_DrawItem(ByVal sender As Object, ByVal e As
System.Windows.Forms.DrawItemEventArgs) Handles TabControll1.DrawItem

    'Firstly we'll define some parameters.

    Dim CurrentTab As TabPage = TabControll1.TabPages(e.Index)
    Dim ItemRect As Rectangle = TabControll1.GetTabRect(e.Index)
    Dim FillBrush As New SolidBrush(Color.SkyBlue)
    Dim TextBrush As New SolidBrush(Color.Black)
    Dim sf As New StringFormat
    sf.Alignment = StringAlignment.Center
    sf.LineAlignment = StringAlignment.Center

    'If we are currently painting the Selected TabItem we'll
    'change the brush colors and inflate the rectangle.

    If CBool(e.State And DrawItemState.Selected) Then
        FillBrush.Color = Color.Red
        TextBrush.Color = Color.Black
        ItemRect.Inflate(2, 2)
    End If

    'Set up rotation for left and right aligned tabs

    If TabControll1.Alignment = TabAlignment.Left Or
TabControll1.Alignment = TabAlignment.Right Then
        Dim RotateAngle As Single = 90
        If TabControll1.Alignment = TabAlignment.Left Then
RotateAngle = 270
        Dim cp As New PointF(ItemRect.Left + (ItemRect.Width \ 2),
ItemRect.Top + (ItemRect.Height \ 2))
        e.Graphics.TranslateTransform(cp.X, cp.Y)
        e.Graphics.RotateTransform(RotateAngle)
        ItemRect = New Rectangle(-(ItemRect.Height \ 2), -
(ItemRect.Width \ 2), ItemRect.Height, ItemRect.Width)
        End If

    'Next we'll paint the TabItem with our Fill Brush

```

```
e.Graphics.FillRectangle(FillBrush, ItemRect)

'Now draw the text.
e.Graphics.DrawString(CurrentTab.Text, e.Font, TextBrush,
RectangleF.op_Implicit(ItemRect), sf)

'Reset any Graphics rotation
e.Graphics.ResetTransform()

'Finally, we should Dispose of our brushes.
FillBrush.Dispose()
TextBrush.Dispose()

End Sub

End Class
```