DEVELOPMENT OF DESKTOP APPLICATION FOR SALES REGISTRATION OF MOBILE OPERATOR IN REPUBLIC OF IRAQ

GRADUATE QUALIFICATION WORK
SUSU–02.04.02.2019.308-643.GQW

Supervisors,
Cand. Sci., Assoc. Prof.
___________ O.N. Ivanova

Author,
the student of the group CE-229
___________ M.B. Aquly

Normative control
___________ O.N. Ivanova
“___”___________ 2019

Chelyabinsk–2019
TASK of the master graduate qualification work
For the student of the group CE-219
Aquli Mustafa Bahaa
in master direction 02.04.02
“Fundamental Informatics and Information Technologies”
(master program “Database Technologies”)

1. The topic (approved by the order of the rector from 25.04.2019 No. 899)
Development of desktop application for sales registration of mobile operator in Republic of Iraq

2. The deadline for the completion of the work: 01.06.2019.

3. The source data for the work

4. The list of the development issues
4.1. To define the problem statement
4.2. To make a comparative analysis of analogues
4.3. To choose development tools
4.4. To determine functional and non-functional requirements
4.5. To design the database
4.6. To design and to implement the application
4.7. To test the system.

5. **Issuance date of the task:** 09.02.2019.

**Supervisor**
Cand. Sci, Assoc. Prof. O.N. Ivanova

**The task is taken to perform**
M.B. Aquly
# TABLE OF CONTENTS

INTRODUCTION ........................................................................................................... 5

1. THE ANALYSIS OF THE SUBJECT AREA ......................................................... 8

1.1. The problem statement ..................................................................................... 8

1.2. Comparative analysis of the existing analogical applications ....................... 9

1.3. The used development tools ............................................................................. 11

2. DESIGN OF THE APPLICATION .......................................................................... 15

2.1. Functional and non-functional requirements .................................................. 15

2.2. Use case diagram ............................................................................................ 16

2.3. Development of the database .......................................................................... 17

2.4. Development of the interface .......................................................................... 25

3. IMPLEMENTATION OF THE APPLICATION WITH A WEB-INTERFACE ......... 26

3.1. Architecture of the system ................................................................................. 26

3.2. Several fragments of C#-code for implementing the basic functionality ....... 27

3.3. The implementation of the algorithm of the total sales calculation .......... 31

4. TESTING OF THE DESKTOP APPLICATION .................................................. 33

4.1. The used methods of testing ............................................................................. 33

4.2. Screenshots of the application ........................................................................ 39

CONCLUSIONS ......................................................................................................... 53

REFERENCE .............................................................................................................. 55
INTRODUCTION

Topicality of Research

We all know that each company has a special system to storage data and retrieval when needed to these programs vary from one company to another until today we may see that in Iraq some companies use ordinary paper in the storage of data and other companies use the Excel system to storage of data but the desktop application The advantages of many where and most importantly you see the fields of graphic color data entry assistant helps to complete the work without errors and a large speed in the input and retrieval when needed and even giant companies Our application designed for telecommunications companies and mobile phones, which is the largest in the world in terms of data The of this application is very, very useful and avoid the company repeated mistakes.

Aims and Objectives of this Project

The aim of the project is to develop a desktop application for sales registration of mobile operator in Republic of Iraq.

The objectives of this project are:

1) to make a comparative analysis of the existing solutions;
2) to design database and to design a project due to the functional requirements;
3) to implement the project;
4) to perform functional testing.

The specific objectives of the work were to: design an easy to learn and use sales management system suitable for company; build a prototype which can be used as a model for future projects; reduce cost of employing ICT in company; expose company to the usefulness of ICT to their business processes; make ICT more adaptable and appealing to entrepreneurs of SMEs within the region; help SMEs introduce more formalized sales transaction procedures into their businesses; and help SMEs maximize profits through a more effective sales and stock tracking system [1].
Practical significance of the project is considered to be the following:

The benefit of the company from the use of this application is multiple, which is summarized below.

1. Huge database: The huge data that each company owns from the contacts of customers and their employees needs to have such a program to accommodate them as well as the ability to retrieve them and access them when needed.

2. 2-time saving: all of us know that time is valuable to everyone. This application saves a lot of time for the responsible employee and makes his performance very fast. For example, when the manager asks for a report on sales yesterday, in this case, the employee needs only one minute to get to the data and prepare the report. It is faster than Hue in the rest of the spreadsheet programs.

3. 3-Saving Money: The Objective of this review is to analyze and design sales software and true database which will be used in the day to day activities of the company, let us take a look at the computer and the manual computation, Computer has being built in such a way that the things which are going to be computed will be easier for the user rather than manual computation, these has created the need to build a software for computational analysis, as in financial department. It will be easier to calculate with the computer rather than using manual which may result to a lot of mistakes in the computation [2].

Structure of the thesis

The thesis consists of four chapters, introduction, conclusion and reference list.

In the first chapter, the problem statement is given in details, the comparative analysis of the existing analogical applications is given. We describe the used development took as well there.
In chapter two, there is a description of functional and non-functional requirements use case diagram, database scheme and the design of the application’s interfaces.

In chapter three, we show several fragments of source code for implementing the basic functionality of the system and the algorithm of sales calculation and its implementation in the desktop-application.

Chapter Four is devoted to the testing of the application. It contains the results of functional, and usability testing.

The thesis has 56 pages; the list of references contains 20 resources.
1. THE ANALYSIS OF THE SUBJECT AREA

1.1. The problem statement

Presently, due to the great development in the world of modern technology and the proliferation of mobile phones and the internet also to increase the communications companies and compete with each other.

Each company owns a number of employees and a number of other customers and their data regardless of whether it is a telecommunications company or not. Majority of the company in Iraq use the traditional daily sales books for their business transactions.

Only a few of them have formalized sales tracking systems which are mostly accounting software or spreadsheet applications such as Microsoft excel, Microsoft Access or very uncommonly oracle and other proprietary software applications. While a few of the company find the existing sales management systems to be problem-free and a key to their continuous growth, most of them quit using these systems after a while for various reasons [10].

The opinion of those who discontinued the computerized systems is either that, the systems reduced their daily sales or that they were too complex and expensive to use. The case is opposite for those who continue to use the systems. It is important that the wide spread of information and communication technologies should enable every willing business to find an appropriate solution to its particular problems among the various solutions provided. However, it is not uncommon to overlook some of the challenges company in developing countries still grapple with when it comes to sales management.

While some businesses have had their problems solved by using computerized systems, others have to bear high cost if they wish to meet their sales target with the systems they wish to or are currently implementing. The common challenges faced by the company includes misrepresentation of sales, loss of goods and profit, high cost of sales tracking, customer dissatisfaction,
difficulty implementing marketing strategies, and high cost of acquiring software licenses.

Though some of the challenges listed above have been tackled by various computer applications and sales management systems, many companies are still confronted with them. One of these problems is that most of the existing solutions have not been created to provide the information that most of the company need to keep their businesses going. In summary, the challenges with the existing systems include high cost of acquiring full software packages, special hardware requirements of some functions, systems not satisfying unique goals of company, requirement for understanding some accounting principles, and the cost of paying for extra functions that are not needed. The purpose of this work therefore was to design and develop a desktop sales management system for company in Republic of Iraq. These data must be managed by a competent staff member who can be consulted when necessary. In this project, we decided to work on the desktop application of the Telecommunications Services Company, this program stores customer information in the company and sales and processing sales reports.

We have to mention the sales of mobile phones and Wi-Fi devices, where at the present time constitute an increase in sales in the world where in every country there are many mobile phone companies and telecommunications companies where only in the Republic of Iraq we have 10 giant telecommunications companies and Countless companies for sells mobile phones.

1.2. Comparative analysis of the existing analogical applications

Such companies usually use Excel to perform all operations – to register sales, returns and orders. But I see that it has a lot of disadvantages.

1. **Time Consuming:** In Microsoft Excel, when we entering the data manually it can take a very long time especially if you have a lot of data to enter.
2. **Viruses:** In Microsoft Excel File, the viruses can be attached through macros. Macros are mini-programs that are written into an MS Excel spreadsheet.

3. **Customizable Graphs:** This is a big Disadvantage of Microsoft Excel, it does offer a good variety of graph capabilities. The customization of the standard formats, and combinations of the different data sets and different types of graph is awkward and not natural.

4. **Hard to use:** MS Excel training programs, it is still hard to use for some users might not get the hang of it. While it is easy to create the formulas, the reference cells, copy and paste the data and the spreadsheets become more difficult to change and manage.

5. **Difficult to analyze:** The spreadsheets are absolute for creating one-time analysis. Especially for some people, a spreadsheet is difficult to analyze. It becomes problematic as the data grows and emerge over time. As is parked new rows and columns get added or new ones created, the data and the formulas are not consistently updated[15]

These mistakes lead to bad results and decisions we decided to use the desktop application instead of the web site for several reasons [11]:

1. Cost factor - web application development and its maintenance involve higher costs and mostly recurring in nature. Desktop applications are purchased one time and there are not continually occurring charges. However, in certain cases, maintenance fees may be charged. Web based applications need to be installed only once whereas desktop applications are to be installed separately on each computer. Also updating the applications is cumbersome with desktop applications as it needs to be done on every single computer which is not the case with web applications.

2. A company should pay salary to administrator and for hosting.

3. Security - web applications are exposed to more security risks than desktop applications. You can have a total control over the standalone
applications and protect it from various vulnerabilities. This may not be the case with web applications as they are open to a large number of users in the Internet community thus widening the threat.[7]

4. Connectivity - web application development relies significantly on Internet connectivity and speed. Absence of Internet or its poor connectivity can cause performance issues with web applications. Desktop applications are standalone in nature and hence do not face any hindrances resulting from Internet connectivity. Connectivity also significantly affects the speed at which desktop and web applications operate. As web applications are internet dependent, they cost more bandwidth usage than desktop applications do [7].

5. Moreover, the target audience of the project is represented by small, not very rich companies, which do not have many offices in different parts of the country. They usually have only one or two offices, so that they do not need to maintain a web-application at all.

So, it is obvious that for the problem solution it is necessary to develop a desktop application which will provide all functions in a relevant way for a company sailing Wi-Fi routers, SIM-cards, and cell phones.

1.3. The used development tools

I have chosen C# as a programming language for the implementation of my project. C# is highly expressive, yet it is also simple and easy to learn. The curly-brace syntax of C# will be instantly recognizable to anyone familiar with C, C++ or Java. Developers who know any of these languages are typically able to begin to work productively in C# within a very short time. C# syntax simplifies many of the complexities of C++ and provides powerful features such as null able value types, enumerations, delegates, lambda expressions and direct memory access, which are not found in Java. C# supports generic methods and types, which provide increased type safety and performance, and iterators, which enable implementers of collection classes to define custom iteration
behaviors that are simple to use by client code. Language-Integrated Query (LINQ) expressions make the strongly-typed query a first-class language construct [5].

C# is an object oriented programming language. OOP includes classes, objects, overloading, encapsulation, data hiding, and inheritance. In this article, we will learn how to write code for Classes and Objects, Constructor and Destructor, Function Overloading, Encapsulation, Inheritance, Interface, Polymorphism in C# and .NET [1].

Also c# it provides a lot of the following features.

Simple. C# is a simple language in the sense that it provides structured approach (to break the problem into parts), rich set of library functions, data types etc.

Modern Programming Language. C# programming is based upon the current trend and it is very powerful and simple for building scalable, interoperable and robust applications.

Object Oriented. C# is object oriented programming language. OOPs makes development and maintenance easier whereas in Procedure-oriented programming language it is not easy to manage if code grows as project size grow.

Type Safe. C# type safe code can only access the memory location that it has permission to execute. Therefore it improves a security of the program.

Interoperability. Interoperability process enables the C# programs to do almost anything that a native C++ application can do.

Scalable and Updateable. C# is automatic scalable and updateable programming language. For updating our application we delete the old files and update them with new ones.

Component Oriented. C# is component oriented programming language. It is the predominant software development methodology used to develop more robust and highly scalable applications.
Structured Programming Language. C# is a structured programming language in the sense that we can break the program into parts using functions. So, it is easy to understand and modify.

Rich Library. C# provides a lot of inbuilt functions that makes the development fast.

Fast Speed. The compilation and execution time of C# language is fast [12].

I have chosen ASP.NET as a platform for the implementation of my project. C# programs run on the .NET Framework, an integral component of Windows that includes a virtual execution system called the common language runtime (CLR) and a unified set of class libraries [4].

The CLR is the commercial implementation by Microsoft of the common language infrastructure (CLI), an international standard that is the basis for creating execution and development environments in which languages and libraries work together seamlessly. In addition to the run time services, the .NET Framework also includes an extensive library of over 4000 classes organized into namespaces that provide a wide variety of useful functionality for everything from file input and output to string manipulation to XML parsing, to Windows Forms controls. The typical C# application uses the .NET Framework class library extensively to handle common "plumbing" chores. Also c# it provides a lot of features that are given below [5].

- Interoperability;
- Common Language Runtime engine (CLR);
- Language independence;
- Base Class Library;
- Simplified deployment;
- Security;
- Portability.
I have chosen SQL server as DBMS for the implementation of my project. SQL Server is a data engine introduced by Microsoft. It provides an environment used to create and manage databases. It allows secure and efficient storage. It provides other components and services that support the business intelligence platform to generate reports and help analyse the data.

The features provided by the SQL Server are as mentioned below [19]:

1. Scalability: It allows distributing data in the large tables into different file groups. The server can access the file groups simultaneously.

2. CLR integration: It allows user to use the CLR features of the .NET Framework into the server database.

3. Service oriented architecture: It provides distributed, asynchronous application framework for large scale applications.

4. Web services support: It allows direct access to the data from web services by implementing the HTTP endpoints.

5. High security: It implements high security by adding policies for log on and passwords.

6. Support for data migration and analysis: It provides tools to migrate data from data sources to a common database.

7. Policy based management: It is used to define a set of policies for configuring and managing data.

8. Resource governor: It is used to manage the workload of the server by allocating and managing resources.
2. DESIGN OF THE APPLICATION

2.1. Functional and non-functional requirements

Functional requirements are something any project must have in order not to get fail during the work.

The functions those are available for a system.

1. Function “sales” is available for the Customer care so he will be responsible for fully managing of sales (CRUD).

2. Function “Customer” is available for the Customer care so he will be responsible for fully managing of himself (CRUD).

3. Function “Type” is available for the Customer care so he will be responsible for fully managing of items types (CRUD).

4. Function “Item” is available for the Customer care so he will be responsible for fully managing of items (CRUD).

5. Function “Guarantee return” is available for the Customer care so he will be responsible for fully managing of guarantee returns (CRUD).

6. Function “Unblock SIM” is available for the Customer care so he will be responsible for fully managing of unblock SIM (CRUD).

7. Function “get statistical reports” is available for the Customer care so he will be responsible for generate any report he wants.

As for non-functional, it should be present in each project as well. We will divide it into two parts, the first operating systems:

• as you know, we have many operating systems in our times, each one has its own features and has its own programs in this program. It has been created in order to reach the Windows operating system where it does not work on other operating systems such as Linux and Mac etc.;

• also databases programs. There are many programs for databases where it varies from one program to another It is known that desktop applications rely on a special program in the storage and retrieval of information. In this program we chose (SQL server) and do not work on others programs.
2.2. Use case diagram

Unified Modeling Language (UML) enables IT professionals to model computer applications [20].

Use case diagrams are valuable for visualizing the functional requirements of a system that will translate into design choices and development priorities [17].

This diagram shows nine use cases. There is one type of user: customer care, represented as actors. These actors are connected with these use cases by relationships to show the capability for each user in the system. Fig. 1 shows the Use case diagram for the sales management functions.

The use case “CRUD Sale” is extended different available operations depicted as various use cases: “Sell Wi-Fi router”, “Sell SIM-card”, and “Sell Phone”. All other use cases follow the user’s goals and are not connected with each other.

![Use case diagram](image-url)

Fig. 1. Use case diagram
2.3. Development of the database

Database management systems (DBMS) are computer programs designed to manage the storage, maintenance, and retrieval of information from a computer system, to optimize and manage the storage and retrieval of data from databases. DBMS offers a systematic approach to manage databases via an interface for users as well as workloads accessing the databases via apps.

A database system is its structure described in a formal language supported by the database management system DBMS and refers to the organization of data as a blueprint of how a database is constructed (divided into database tables in the case of relational databases) [2].

The formal definition of database schema is a set of formulas sentences called integrity constraints imposed on a database. All constraints are expressible in the same language [13].

MySQL, launched in 1995, has become the most popular open source DBMS. Another root cause of MySQL's popularity has been the ongoing success of phpMyAdmin [14], a well-established MySQL web-based interface.

SQL Server contains a number of components. Each component is provided with the specific services and support to the clients connected to the server.

The server contains the following components [19]:

a) **Database Engine:** The component provides support to store query, process and secure data on the server. It allows user to create and manage database objects;

b) **Integration Services:** The service allows gathering and integrating varied data in a consistent format in a common database. The database is known as data warehouse. The warehouse contains integrated databases, text files or flat files;
c) Analysis Services: The warehouses are designed to facilitate reporting and analysis. The applications are widely using this data store for analytical purpose. The applications used for this purpose are known as BI applications;

d) Reporting Services: They provide support to generate complete reports on data in the database engine in the data warehouse.

These services provide a set of tools that help in creating and managing reports in different formats.

We decided to specify the following tables of the database:
- the table for customer that contain the details of the customer in our company like (ID Customer, First Name, Second Name, Last Name, Address City, Address Street, Address House, and Phone);
- the table for the type that saved the information of type like (ID Type and name);
- the items table that contain the different type of category (ID Item, ID Type, serial NO, Cost, Model, Quantity, Guarantee Return.);
- sales table which has all information of sales like (ID Sales, ID Customer, ID Item, date and Quantity);
- the table for the Unblock SIM that saved the information of Unblock SIM like (ID Unblock, ID Item and ID Customer);
- the offer table that contain the different type of category (ID Offer, ID Customer, ID Item, Type and Date);
- the Guarantee Return table that contain the different type of category (ID Return, ID Customer, ID Item, Date and status);

The scheme of the database is shown in the fig. 2. It consists of 8 tables, described below.

The first table is the "customer" where the table can be viewed by all the information that belongs to the customer as well as the data type used for each
field. Each field contains a specific data type for example the fields as shown below (fig. 3).

Fig. 2. The scheme of the database

The structure of the table:

- First name – nvarchar
- Second name – nvarchar
- Last name – nvarchar
- Address city – nvarchar
- Address street – nvarchar
- Address house – nvarchar
- Phone – int

Fig. 3. Structure of the table “Customer”
In this table you can see the data to all customers that was previously entered as stored in the database as shown below (fig 4).

![Customer Table](image)

**Fig. 4. Data of the table “Customer”**

In the sales table we can see all the information about the sale and the dates of sale. Each field contains a specific data type for example the fields as shown below (fig 5).

- ID-Sale – int
- ID-Customer – int
- ID-Item – int
- Quantity – int
- Date – date

![Sales Table](image)

**Fig. 5. Structure of the table “Sales”**

In this table you can see the data to the all sales that was previously entered as stored in the database as shown below (fig 6).
In the "Type" table we can see all the information about the type. Each field contains a specific data type for example the fields as shown below (fig 7).

ID-Type – int
Name – nvarchar

In this table you can see the data to the all sales that was previously entered as stored in the database as shown below (fig 8).
In the "items" table we can see all the information about the items and each field contains a specific data type for example the fields as shown below (fig 9).

- ID-item – int
- Serial NO – int
- ID-Type – int
- Quantity – int
- Cost – int
- Model – nvarchar
- Guarantee period – nvarchar

![Fig. 9. Structure of the table “Item”](image)

In this table you can see the data to the all sales that was previously entered as stored in the database as shown below (fig 10).

![Fig. 10. Data of the table “Item”](image)
In the "Unblock SIM" table we can see all the information about the unblock SIM and each field contains a specific data type for example the fields as shown below (fig 11).

- ID-Unblock – int
- ID-Item – int
- ID-Customer – int

Fig. 11. Structure of the table “Unblock-SIM”

In this table you can see the data to the all unblocked SIM that was previously entered as stored in the database as shown below (fig 12).

Fig. 12. Data of the table “Unblock-SIM”

In the "Offer" table we can see all the information about the offers and each field contains a specific data type for example the fields as shown below (fig. 13).

- ID-Offer – int
- ID-Item – int
- ID-Customer – int
- Type – nvarchar
- Date – date
In this table you can see the data to the all offers that was previously entered as stored in the database as shown below (fig 14).

In the "guarantee return" table we can see all the information about the guarantee and each field contains a specific data type for example the fields as shown below (fig 15).

- ID-return – int
- ID-Item – int
- ID-Customer – int
- status – nvarchar
- Date – date
In this table you can see the data to the all unblocked SIM that was previously entered as stored in the database as shown below (fig 16).

<table>
<thead>
<tr>
<th>id_return</th>
<th>id_item</th>
<th>id_customer</th>
<th>date</th>
<th>status</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1004</td>
<td>1</td>
<td>2019-03-02</td>
<td></td>
</tr>
<tr>
<td>1003</td>
<td>3002</td>
<td>1006</td>
<td>2019-03-04</td>
<td></td>
</tr>
<tr>
<td>3002</td>
<td>3003</td>
<td>1</td>
<td>2019-03-11</td>
<td></td>
</tr>
<tr>
<td>3003</td>
<td>3003</td>
<td>2004</td>
<td>2019-03-11</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 16. Data of the table “Guarantee return”

2.4. Development of the interface

Each desktop application contains many interfaces and allows the user to move between the interfaces and to summarize the interfaces in our program to you this tree it's a Schema of available interfaces for customer care as shown below (fig 17). it contains all the elements in the application and also explains the process of moving from one to another.

Fig. 17. Schema of available interfaces for customer care
3. IMPLEMENTATION OF THE APPLICATION WITH A WEB-INTERFACE

3.1. Architecture of the system

Component diagrams are used to visualize the organization of system components and the dependency relationships between them. They provide a high-level view of the components within a system [3].

UML Component diagrams are used in modeling the physical aspects of object-oriented systems that are used for visualizing, specifying, and documenting component-based systems and also for constructing executable systems through forward and reverse engineering [18].

The architecture of the system is shown in fig.18 shows that we have four component design, main, source and database .the first component "source" it is depending on "design" and he "main "has dependency relationship with "database".

Also we have dependency relationship between the "source "and "main". The component “Main” is devoted to the connection to database.
The component “Source” contains the main code of the project.
The component “Design” contains files with the description of GUI.

![Component diagram of the system](image-url)
3.2. Several fragments of C#-code for implementing the basic functionality

Within the system, there are a lot of fragment code, some of which are simple and others are rather complex. Many of them duplicate each other.

**The variables.** Fig. 19 shows how we assign the variables to each section of our program, as you see each part containing more than one variable.

```csharp
//customer variables
public static bool cusEdit = false;
public static int cusID;
public static string cusFname = "";
public static string cusSname = "";
public static string cusLname = "";
public static string cusAddcity = "";
public static string cusAddstreet = "";
public static string cusAddhouse = "";
public static string cusPhon = "";

//type variables
public static bool typeEdit = false;
public static int typeID;
public static string typeName = "";

//item variables
public static bool itemEdit = false;
public static int id_item;
public static int id_type;
public static string serialNo;
public static string cost;
public static string model;
public static int quantity;
public static string garanty;

//offer variables
public static bool offerEdit = false;
public static int id_offer;
public static string ofrType;
```

Fig. 19. Function for The variables

**Check quantity.** Fig. 20 shows where in this function we can know the number of items that exist and coordinate them with the sale.

```csharp
//check Quantity
if (noQuantity.Value == 0)
{
    errorProvider1.SetError(noQuantity, "Quantity Required");
    noQuantity.Select();
    return;
}
```

Fig. 20. Function for Check quantity
**Check type name.** Fig. 21 shows, the customer care is required to enter the name field because this is required and not optional and he will receive massage tell him that is required.

```csharp
//check type name
if (txtTypeName.Text == "")
{
    errorProvider1.SetError(txtTypeName, "Type Name Required");
txtTypeName.Select();
    return;
}
```

Fig. 21. Function for Check type name

**Delete sale.** Fig. 22 shows, the customer care can delete any sales he want if he has some problem or some mistake when he sold some item.

```csharp
//delete record from sales
SqlCommand cmdSales = new SqlCommand("delete from sales where id_sale=@id_sale", con);
    cmdSales.Parameters.AddWithValue("@id_sale", salesGridView.SelectedCells[1].Value);
    con.Open();
    cmdSales.ExecuteNonQuery();
    con.Close();
    MessageBox.Show("Returned Successfully", "Sales Management System", MessageBoxButtons.OK, MessageBoxIcon.Information);
    btnLoad_Click(sender, e);
```

Fig. 22. Function for Delete sale

**Insert to guarantee.** Fig. 23 shows in this function we can enter any item that has been sold in the database of guarantee where it is automatically entered into the guarantee section.

```csharp
//check type name
if (txtTypeName.Text == "")
{
    errorProvider1.SetError(txtTypeName, "Type Name Required");
txtTypeName.Select();
    return;
}
```

Fig. 23. Function for Insert to guarantee
Check if the item not SIM. Fig. 24 shows in this function we can check the items which sold if it is SIM or not because if this item is a SIM then we will not add it to the guarantee section. The user can enter different spelling; this function handles all possible variants.

```csharp
//check if item is not sim
if (!Regex.IsMatch(txtbxType.Text, "Sim|sim|SIM"))
{
    //insert into guaranty
    SqlCommand cmdGrnty = new SqlCommand("insert into [garantees] values (@id_item,@id_customer,@date,@status)", con);
    cmdGrnty.Parameters.AddWithValue("@id_item", txtbxItem.SelectedValue);
    cmdGrnty.Parameters.AddWithValue("@id_customer", txtbxCustomer.SelectedValue);
    cmdGrnty.Parameters.AddWithValue("@date", DateTime.Now.ToString("yyyy-MM-dd");
    cmdGrnty.Parameters.AddWithValue("@status","";
    con.Open();
    cmdGrnty.ExecuteNonQuery();
    con.Close();
}
else
```

Fig. 24. Function for Check if the item not SIM

Check all information of user. Fig. 2 shows in this function we cannot add any new customer without information and we did all the information is required.

```csharp
//check first name
if (txt FName.Text == "")
{
    errorProvider1.SetError(txt FName, "First Name Required");
    txt FName.Select();
    return;
}
//check second name
if (txt SName.Text == "")
{
    errorProvider1.SetError(txt SName, "Second Name Required");
    txt SName.Select();
    return;
}
//check last name
if (txt LName.Text == "")
{
    errorProvider1.SetError(txt LName, "Last Name Required");
    txt LName.Select();
    return;
}
//check city
if (txt AddsCity.Text == "")
{
    errorProvider1.SetError(txt AddsCity, "City Required");
    txt AddsCity.Select();
    return;
```

Fig. 25. Function for Check all information of user
**Connection with our database.** Fig. 26 shows in this function which makes the program able to access to our database which we already create it.

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data;
using System.Data.SqlClient;
using System.Windows.Forms;

namespace Sales_Management_System
{
    public class publicCode
    {
        //connection string
        public static string DB = @"Data Source=MUSTAFAC-4QQLTSQL;Initial Catalog=SalesDB;Persist Security Info=True;User ID=mustafa;password=12345;";
    }
}
```

**Fig. 26. Function for Connection with our database**

**Create list of items.** Fig. 27 shows in this function the customer care able to make list of any item he wants to add.

```csharp
//create list of item
List<item> itemList = new List<item>();
itemList.Clear();
itemList.Add(new item() { id_item = 0, model = "" });
SqlCommand cmdItem = new SqlCommand("select * from [item] where id_type=@id_type", con);
cmdItem.Parameters.AddWithValue("@id_type", cboxType.SelectedValue);
try
{
    
}
```

**Fig. 27. Function for Create list of item**

**Unblock SIM.** Fig. 28 shows the code for unblocking the SIM-card that when the customer care will sale any (sim) automatically will update in sim-unblock.
3.3. The implementation of the algorithm of the total sales calculation

The algorithm of the total sales calculation for the users is shown in fig. 29. It calculates the number of currently items in our company by the system (depicted as C.quantity in a scheme). It calculates the number of items which we need to sell in our company by the system (depicted as S.quantity in a scheme). It compares between the number of items which we need to sell and the number of currently items in our company by the system.
The code implementing this algorithm is shown in fig. 30.

```csharp
//get quantity of selected item
SqlCommand cmd = new SqlCommand("select * from [item] where id_item = @id_item", con);
cml.Parameters.AddWithValue("@id_item", cmbItem.SelectedValue);
try {
    con.Open();
    SqlDataReader dreader = cmd.ExecuteReader();
    if (dreader.Read()) {
        qtm = (int)dreader("quantity");
        if (qtm > (int)(noQuantity.Value)) {
            con.Close();

            //insert sales
            SqlCommand cmdSales = new SqlCommand("INSERT INTO [sales] VALUES(@id_customer, @id_type, @id_item, @date, @quantity)", con);
            cmdSales.Parameters.AddWithValue("@id_customer", cmbCustomer.SelectedValue);
            cmdSales.Parameters.AddWithValue("@id_type", cmbType.SelectedValue);
            cmdSales.Parameters.AddWithValue("@id_item", cmbItem.SelectedValue);
            cmdSales.Parameters.AddWithValue("@date", DateTime.Now.ToString("yyyy-MM-dd"));
            cmdSales.Parameters.AddWithValue("@quantity", qtm, noQuantity.Value);
            con.Open();
            cmdSales.ExecuteNonQuery();
            con.Close();

            //decrease item quantity
            SqlCommand cmdItem = new SqlCommand("UPDATE [item] SET quantity = quantity - @quantity WHERE Id_item = @id_item", con);
            cmdItem.Parameters.AddWithValue("@quantity", qtm - (int)(noQuantity.Value));
            cmdItem.Parameters.AddWithValue("@id_item", cmbItem.SelectedValue);
            con.Open();
            cmdItem.ExecuteNonQuery();
            con.Close();
        }
    }
}
```
4. TESTING OF THE DESKTOP APPLICATION

4.1. The used methods of testing

Software testing is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is Defect free. It involves execution of a software component or system component to evaluate one or more properties of interest [8].

Software testing is the art of investigating a software in a systematic fashion so as to find deep-rooted defects in it. In addition to that, software testing also checks the quality and correctness of the software. After the errors are identified, it becomes easier to develop a bug-free and user-friendly software [6].

As software applications get ever more complex and intertwined and with the large number of different platforms and devices that need to get tested, it is more important than ever to have a robust testing methodology for making sure that software products/systems being developed have been fully tested to make sure they meet their specified requirements and can successfully operate in all the anticipated environments with the required usability and security [11].

Therefore, we compare the actual results and the expected results (tab. 1).

Table 1. The protocol of functional testing of the site

<table>
<thead>
<tr>
<th>No.</th>
<th>Function</th>
<th>Expected result</th>
<th>Obtained result</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The customer-care can create new sale</td>
<td>The customer will be able to add new sale</td>
<td>The customer able to add new sale</td>
<td>The function works</td>
</tr>
<tr>
<td>2</td>
<td>The customer-care can read all information which saved in list of sales.</td>
<td>The customer will be able to see all table of sale</td>
<td>The customer able to see all table of sale</td>
<td>The function works</td>
</tr>
<tr>
<td>3</td>
<td>The customer-care can update any record</td>
<td>The customer will be able to change any</td>
<td>The customer able to change</td>
<td>The function works</td>
</tr>
<tr>
<td>No.</td>
<td>Function</td>
<td>Expected result</td>
<td>Obtained result</td>
<td>Conclusion</td>
</tr>
<tr>
<td>-----</td>
<td>--------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td></td>
<td>in the list of sales</td>
<td>information in the table of sale</td>
<td>any information in the table of sale</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>The customer-care can delete any sales in the previous.</td>
<td>The customer will be able to delete one or more record from the table of sale</td>
<td>The customer able to delete one or more record from the table of sale</td>
<td>The function works</td>
</tr>
<tr>
<td>5</td>
<td>The customer-care can create new item</td>
<td>The customer will be able to add new item</td>
<td>The customer able to add new item</td>
<td>The function works</td>
</tr>
<tr>
<td></td>
<td>The customer-care can read all items in the list of item.</td>
<td>The customer will be able to see all table of item</td>
<td>The customer able to see all table of item</td>
<td>The function works</td>
</tr>
<tr>
<td>7</td>
<td>The customer-care can update any item he want or change the name.</td>
<td>The customer will be able to change any information in the table of item</td>
<td>The customer able to change any information in the table of item</td>
<td>The function works</td>
</tr>
<tr>
<td>8</td>
<td>The customer-care can delete any item in the list of item.</td>
<td>The customer will be able to delete one or more record from the table of item</td>
<td>The customer able to delete one or more record from the table of item</td>
<td>The function works</td>
</tr>
<tr>
<td>No.</td>
<td>Function</td>
<td>Expected result</td>
<td>Obtained result</td>
<td>Conclusion</td>
</tr>
<tr>
<td>-----</td>
<td>--------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>9</td>
<td>The customer-care can create new type in our application.</td>
<td>The customer will be able to add new type</td>
<td>The customer able to add new type</td>
<td>The function works</td>
</tr>
<tr>
<td>10</td>
<td>The customer-care can read all types which we have, and show it by a table.</td>
<td>The customer will be able to see all table of type</td>
<td>The customer able to see all table of type</td>
<td>The function works</td>
</tr>
<tr>
<td>12</td>
<td>The customer-care can update any record of type or change the name or date.</td>
<td>The customer will be able to change any information in the table of type</td>
<td>The customer able to change any information in the table of type</td>
<td>The function works</td>
</tr>
<tr>
<td>13</td>
<td>The customer-care can delete any record in the list of type.</td>
<td>The customer will be able to delete one or more record from the table of type</td>
<td>The customer able to delete one or more record from the table of type</td>
<td>The function works</td>
</tr>
<tr>
<td>14</td>
<td>The customer-care can create and add a new customer to our list of customer.</td>
<td>The customer will be able to add new customer</td>
<td>The customer will be able to add new customer</td>
<td>The function works</td>
</tr>
<tr>
<td>15</td>
<td>The customer-care can read all the information of the customers by show it in the table.</td>
<td>The customer will be able to see all table of customer</td>
<td>The customer able to see all table of customer</td>
<td>The function works</td>
</tr>
<tr>
<td>No.</td>
<td>Function</td>
<td>Expected result</td>
<td>Obtained result</td>
<td>Conclusion</td>
</tr>
<tr>
<td>-----</td>
<td>--------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>16</td>
<td>The customer-care can update any record in the list of customer</td>
<td>The customer will be able to change any information in the table of customer</td>
<td>The customer able to change any information in the table of customer</td>
<td>The function works</td>
</tr>
<tr>
<td>17</td>
<td>The customer-care can delete any record in the list of customer</td>
<td>The customer will be able to delete one or more record from the table of customer</td>
<td>The customer able to delete one or more record from the table of customer</td>
<td>The function works</td>
</tr>
<tr>
<td>18</td>
<td>The customer-care can create and add all items which sales I the list of guarantee-return</td>
<td>The customer will be able to add new guarantee _return</td>
<td>The customer able to add new guarantee _return</td>
<td>The function works</td>
</tr>
<tr>
<td>19</td>
<td>The customer-care can read all information and show it as a table guarantee-return</td>
<td>The customer will be able to see all table of guarantee _return</td>
<td>The customer able to see all table of guarantee _return</td>
<td>The function works</td>
</tr>
<tr>
<td>20</td>
<td>The customer-care can update any record and change it in the list of guarantee-return</td>
<td>The customer will be able to change any information in the table of guarantee _return</td>
<td>The customer able to change any information in the table of guarantee</td>
<td>The function works</td>
</tr>
<tr>
<td>No.</td>
<td>Function</td>
<td>Expected result</td>
<td>Obtained result</td>
<td>Conclusion</td>
</tr>
<tr>
<td>-----</td>
<td>---------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>21</td>
<td>The customer-care can delete and remove it from the list of guarantee-return</td>
<td>The customer will be able to delete one or more record from the table of guarantee-return</td>
<td>The customer able to delete one or more record from the table of guarantee-return</td>
<td>The function works</td>
</tr>
<tr>
<td>22</td>
<td>The customer-care can create new unblock-sim</td>
<td>The customer will be able to add new unblock-sim</td>
<td>The customer able to add new unblock-sim</td>
<td>The function works</td>
</tr>
<tr>
<td>23</td>
<td>The customer-care can read all information and show it as a table unblock-sim</td>
<td>The customer will be able to see all table of unblock-sim</td>
<td>The customer able to see all table of unblock-sim</td>
<td>The function works</td>
</tr>
<tr>
<td>24</td>
<td>The customer-care can update any record and change it in the list of unblock-sim</td>
<td>The customer will be able to change any information in the table of unblock-sim</td>
<td>The customer able to change any information in the table of unblock-sim</td>
<td>The function works</td>
</tr>
<tr>
<td>25</td>
<td>The customer-care can delete and remove it from the list of unblock-sim</td>
<td>The customer will be able to delete one or more record from the table of unblock-sim</td>
<td>The customer able to delete one or more record from the table of</td>
<td>The function works</td>
</tr>
<tr>
<td>No.</td>
<td>Function</td>
<td>Expected result</td>
<td>Obtained result</td>
<td>Conclusion</td>
</tr>
<tr>
<td>-----</td>
<td>---------------------------------------------------------------------------</td>
<td>------------------------------------------------------</td>
<td>-----------------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>26</td>
<td>The Customer care so he will be responsible for generate daily report he wants.</td>
<td>The customer will be able to generate new report</td>
<td>The customer able to generate new report</td>
<td>The function works</td>
</tr>
<tr>
<td>27</td>
<td>The Customer care so he can generate current report.</td>
<td>The customer will be able to generate new report</td>
<td>The customer able to generate new report</td>
<td>The function works</td>
</tr>
<tr>
<td>28</td>
<td>The Customer care so he will be responsible for generate customize report.</td>
<td>The customer will be able to generate new report</td>
<td>The customer able to generate new report</td>
<td>The function works</td>
</tr>
<tr>
<td>29</td>
<td>The Customer care so he can create and get new offer.</td>
<td>The customer will be able to see all table of unblock-sim</td>
<td>The customer will be able to see all table of unblock-sim</td>
<td>The function works</td>
</tr>
<tr>
<td>30</td>
<td>The Customer care so he can read all the information inside tale of offer</td>
<td>The customer will be able to add new unblock-sim</td>
<td>The customer will be able to add new unblock-sim</td>
<td>The function works</td>
</tr>
<tr>
<td>31</td>
<td>The Customer care so he can update and change the offers.</td>
<td>change any information in the table of offer</td>
<td>change any information in the table of offer</td>
<td>The function works</td>
</tr>
<tr>
<td>32</td>
<td>The Customer care</td>
<td>The customer will be</td>
<td>The customer</td>
<td>The function</td>
</tr>
<tr>
<td>No.</td>
<td>Function</td>
<td>Expected result</td>
<td>Obtained result</td>
<td>Conclusion</td>
</tr>
<tr>
<td>-----</td>
<td>----------</td>
<td>-----------------</td>
<td>-----------------</td>
<td>------------</td>
</tr>
<tr>
<td></td>
<td>so he can update and change the offers.</td>
<td>able to delete one or more record from the table of offer</td>
<td>will be able to delete one or more record from the table of offer</td>
<td>works</td>
</tr>
</tbody>
</table>

4.2. Screenshots of the application

In the main interface of the program there are many buttons and each one is responsible for a specific functions as shown in the fig. below we see the first button is Customer, Type, Items, Guarantee return, unblock SIM and offer as shown below (fig. 31).

![Fig. 31. The home page](image)

When clicking on the "Customer" button we will notice the appearance of a new window with the survival of the original window of the main interface of the program and in this window will note the existence of many buttons as shown below (fig. 32).
The first button in the customer interface is the "load data". This button is responsible for fetching the data entered in advance and preparing it as a table divided into fields of each field containing information related to the customer dealing with our company as shown below (fig. 33).

The second button is "Add new". When clicking on this button, we will see a new window open in this window. There are seven fields. Each field gives information about the customer. We cannot leave one of these fields empty. After writing the information as "First name", "Second name", Last name",,
"Address city", "Address street", "Address house", "Phone", we press the "Save" button to send the information to our database and retrieve it when necessary as shown below (fig 34).

![Add new customer interface](image1)

**Fig. 34. The interface of "Add new" customer**

The third button is the "Edit" button. This button is responsible for modifying the previously entered data in the case of a mistake. The cursor must be placed on the field to be modified. After clicking the "Edit" button, we will opening a new window containing the same fields in "Add customer". We can edit all information which we need then clicking the "Save" button as shown below (fig. 35).

![Edit customer interface](image2)

**Fig. 35. The interface of "Edit" customer**
The last button is the "Delete" button where this button is responsible for deleting the customers by placing the cursor on the field to be deleted and after clicking the "Delete" button we will see a small window asking you if you are sure to delete this field as shown below (fig. 36).

![Fig. 36. The interface of "Delete" customer](image)

When clicking on the "Type" button we will notice the appearance of a new window with the survival of the original window of the main interface of the program and in this window will see the existence of many buttons as shown below (fig. 37).

![Fig. 37. The" Type" interface](image)
The first button in the type interface is the "load data". This button is responsible for fetching the data entered in advance and preparing it as a table divided into fields as NO, ID and Type Name as shown below (fig. 38).

![Fig. 38. The interface of "Load Data" type](image)

The second button is "Add new". When clicking on this button, we will see a new window open in this window. There is one field. We cannot leave it empty. After writing the new type as we press the "Save" button to send the information to our database and retrieve it when necessary as shown below (fig. 39).

![Fig. 39. The interface of "Add new" Type](image)
The third button is the "Edit" button. This button is responsible for modifying the previously entered data in the case of a mistake. The cursor must be placed on the field to be modified. After clicking the "Edit" button, we will opening a new window containing the same field in "Add new". We can edit the type which we need then clicking the "Save" button as shown below (fig. 40).

![Fig. 40. The interface of "Edit" Type](image)

The last button is the "Delete" button where this button is responsible for deleting the types by placing the cursor on the field to be deleted and after clicking the "Delete" button we will see a small window asking you if you are sure to delete this field as shown below (fig. 41).

![Fig. 41. The interface of "Delete" Type](image)
When clicking on the "Items" button we will notice the appearance of a new window with the survival of the original window of the main interface of the program and in this window will see the existence of many buttons and Drop list as shown below (fig. 42).

![Fig. 42. The interface of items](image)

We will see a drop list in the items interface is the "Choose Type". This drop list is responsible for selecting the type of list to know the number and details of each type. When clicking on it, we will see all the types we have already entered as shown below (fig. 43).

![Fig. 43. The interface of "Choose type" items](image)
The second button in the items interface is the "load data". This button is responsible for fetching the data entered in advance and preparing it as a table divided into fields as NO, ID, Type, Serial NO, Cost, Model, Quantity and Guarantee Period as shown below (fig. 44).

![Fig. 44. The interface of "Load Data" items](image)

The third button is "Add new". When clicking on this button, we will see a new window open in this window. There are six fields. Each field gives information about the type. We cannot leave one of these fields empty. After writing the information as "Choose Type", "Serial No", Cost", "Model" ,"Quantity", "Guarantee Period", we press the "Save" button to send the information to our database and retrieve it when necessary as shown below (fig. 45).

![Fig. 45. The interface of "Add new" items](image)
The forth button is the "Edit" button. This button is responsible for modifying the previously entered data in the case of a mistake. The cursor must be placed on the field to be modified. After clicking the "Edit" button, we will opening a new window containing the same fields in "Add new". We can edit all information which we need then clicking the "Save" button as shown below (fig. 46).

![Fig. 46. The interface of "Edit" items](image)

The last button is the "Delete" button where this button is responsible for deleting the items by placing the cursor on the field to be deleted and after clicking the "Delete" button we will see a small window asking you if you are sure to delete this field as shown below (fig 47).

![Fig. 47. The interface of "Delete" items](image)
When clicking on the "Sales" button we will notice the appearance of a new window with the survival of the original window of the main interface of the program and in this window will see the existence of many buttons as shown below (fig 48).

![Fig. 48. The interface of sales](image1)

The first button in the sales interface is the "load data". This button is responsible for fetching the data entered in advance and preparing it as a table divided into fields as NO, ID ,customer id, Customer Name, Item ID, Item Model, Date and Quantity as shown below (fig 49).

![Fig. 49. The interface of "Load Data" sales](image2)
The second button is "New Sell". When clicking on this button, we will see a new window open in this window. There are four fields. Each field has a drop list and gives information about the sales. We cannot leave one of these fields empty. After writing the information as "Choose Type", "Choose Item", "Choose Customer", and "Quantity", we press the "Save" button to send the information to our database and retrieve it when necessary as shown below (fig. 50).

![Fig. 50. The interface of "Add new" sales](image)

The last button is the "Return Sales" button where this button is responsible for deleting the sale by placing the cursor on the field to be deleted and after clicking the "Return Sales" button we will see a small window asking you if you are sure to delete this field as shown below (fig. 51).

![Fig. 51. The interface of "Return sales" sales](image)
When clicking on the "offer" button we will notice the appearance of a new window with the survival of the original window of the main interface of the program and in this window will see the existence of many buttons as shown below (fig. 52).

![Fig. 52. The interface of "offer"](image)

The first button in the offer interface is the "load data". This button is responsible for fetching the data entered in advance and preparing it as a table divided into fields. As shown below (fig. 53).

![Fig. 53. The interface of "load data" offer](image)
The second button is "New offer". When clicking on this button, we will see a new window open in this window. There are four fields. Each field has a drop list and gives information about the offer. We cannot leave one of these fields empty as shown below (fig. 54).

Fig. 54. The interface of "new offer" offer

The third button is the "Edit" button. This button is responsible for modifying the previously entered data in the case of a mistake. The cursor must be placed on the field to be modified. After clicking the "Edit" button, we will opening a new window containing the same fields in "Add new". We can edit all information which we need then clicking the "Save" button as shown below (fig. 55).

Fig. 55. The interface of "Edit" offer
Here we will see only one button "load data". This button is responsible for fetching the data entered in advance and preparing it as a table divided into fields and we will not add anything but by itself will take all information when we will add a new sale. As shown below (fig. 56).

![Fig. 56. The interface of "guarantee return"](image1)

Here we will see only one button "load data". This button is responsible for fetching the data entered in advance and preparing it as a table divided into fields and we will not add anything but by itself will take all information when we will add a new sale for sim. As shown below (fig. 57).

![Fig. 57. The interface of "Unblock SIM"](image2)
CONCLUSIONS

Management of sales of hi-tech products naturally needs and requires an electronic system for being used. Companies often use Excel-documents for this task, but it can be automated in a better way with the usage of modern desktop applications.

During the developing of the application, we solved the following tasks:

1) the modern tools of desktop application development were selected and the comparative analysis of the existing solutions for sales management with their disadvantages was performed;
2) the structure of the required database for sales management system was developed;
3) the desktop application was designed, implemented and tested.

The perspectives for the developed application. Always we can think out some function, which we can implement in the future for example, are the following.

1. To make username and password to all of customers care in our company.
2. The system can send the password to the email if one of customer care forgets the password.
3. The system must show the secret question on the page of recovering password in order to help employee who forgot his password to remember his secret answer.
4. The system must give the ability to change secret question and to update secret answer.
5. the system must provide some report to of the best-selling item during the month or year
6. The system must provide some report to calculate the money sales in month or year.
7. To provide special button to switch-on the program and switch-off the program.

8. The system must send notification to the email that the customer care was sign-in and the same when he will sign-out.
REFERENCE


