

**Czech University of Life Sciences Prague**

**Faculty of Economics and Management**

**Department of Information Engineering**



**Diploma Thesis**

Development of desktop application for dental clinic management

**Bc. Bahaa Farag**

© 2021 CULS Prague

# CZECH UNIVERSITY OF LIFE SCIENCES PRAGUE

Faculty of Economics and Management

## DIPLOMA THESIS ASSIGNMENT

Bc. Bahaa Farag

Systems Engineering and Informatics  
Informatics

Thesis title

**Development of desktop application for dental clinic management**

---

### Objectives of thesis

This diploma thesis focuses on designing and implementing a desktop application for a dental clinic. The main goal of this application is to store, automate, organize and manage all the clinic's paperwork in one convenient and user-friendly system that facilitates the doctor's job which saves time and effort.

The partial goals of the thesis are:

- Describe application development procedures
- Describe the technologies used

### Methodology

The thesis will consist of two parts, the theoretical part will be based on the study of professional information sources.

The practical part will start with analyzing the requirements, based on the requirements the database will be implemented using SQL Server, then the application will be designed, implemented and finally tested. The standard tools and methods of software engineering will be used during the whole process. Based on both theoretical and practical parts, the conclusion will be formulated and possible future development and improvements proposed.

**The proposed extent of the thesis**

60-80 pages

**Keywords**

C#, SQL server, visual studio, software development, desktop application, management system, database.

---

**Recommended information sources**

Clare Churcher. Beginning database design from novice to professional. 2012. ISBN 978-1-4302-4210-9.

Jason Price. Mastering C# database programming. 2003. ISBN 0-7821-4183-8.

Jon Skeet. C# in depth. 2014. ISBN 9781617291340

Ross Mistry, Stacia Misner. Introducing Microsoft SQL Server 2012. ISBN 978-0-7356-6515-6.

Simon Kendal. Object Oriented Programming using C#. 2011. ISBN 978-87-7681-814-2.

Svetlin Nakov & Co. Fundamentals of computer programming with C#. 2013. ISBN 978-954-400-773-7.

---

**Expected date of thesis defence**

2020/21 SS – FEM

**The Diploma Thesis Supervisor**

Ing. Jiří Brožek, Ph.D.

**Supervising department**

Department of Information Engineering

Electronic approval: 19. 11. 2020

**Ing. Martin Pelikán, Ph.D.**

Head of department

Electronic approval: 19. 11. 2020

**Ing. Martin Pelikán, Ph.D.**

Dean

Prague on 08. 03. 2021

---

### **Declaration**

I declare that I have worked on my diploma thesis titled "Development of desktop application for dental clinic management" by myself and I have used only the sources mentioned at the end of the thesis. As the author of the diploma thesis, I declare that the thesis does not break copyrights of any their person.

In Prague on 30.03.2021

---

## **Acknowledgement**

I would like to thank Ing. Jiří Brožek, Ph.D. for all advices, help and support in the whole work from the very first step of choosing an appropriate thesis topic, till the last day of thesis submission, during this hard time of pandemic.

I would also like to thank Alfa Dental s.r.o represented by MDDr. Majd Farag to give me the chance to deploy and test the application on their end and provide all the needed support during the development lifecycle.

I'm also very grateful to my beloved family and friends at both home and Czech Republic, for giving me all love, power, will to and support not only during this work, but at all levels of my studies and life, I sincerely thank you!

I would also like to show my gratitude to the Global Service Delivery team in SAP concur Prague represented by my manager Mr. Oskar Hlinká and director Mr. Martin Lašek for giving me a great chance to join their team and cooperate among the greatest team members for an internship, and giving me a constant support and motivation always.

# Development of desktop application for dental clinic management

## Abstract

This diploma thesis focuses on implanting a desktop application for managing a dental clinic using Visual Studio development environment and MS SQL server, The goal of this thesis is to create a functional and user-friendly application that automates and organizes the unautomated parts in a dental clinic (employees, tools, equipment, ... etc.) in one place to facilitate the job, save time and effort and eliminate paperwork as much as possible.

The first part deals with object-oriented programming and C# language as well as WinForms applications and describes the used tools and technologies like MS Visual Studio, MS SQL Server, DevExpress, ClickOnce and Bunifu Framework.

The practical part starts with analysing the requirements and designing the database, then designing the application's forms, and then programming the application's functions.

After implementing the application's functions, the application will be tested in a dental clinic and the feedback will be collected.

Based on both theoretical and practical parts, the conclusion will be formulated, and possible future development and improvements proposed.

**Keywords:** C#, SQL server, visual studio, software development, desktop application, management system, database.

# Vývoj desktopové aplikace pro správu zubní kliniky

## Abstrakt

Tato diplomová práce se zaměřuje na implantaci desktopové aplikace pro správu zubní kliniky pomocí vývojového prostředí Visual Studio a MS SQL Server.

Cílem této práce je, vytvořit funkční a uživatelsky přívětivou aplikaci, která automatizuje a organizuje neautomatizované části v zubní klinice (zaměstnanci, nástroje, vybavení, atd.) na jednom místě, aby co nejvíce usnadnila práci, ušetřila čas a úsilí, a eliminovala papírování.

První část se zabývá objektově orientovaným programováním a jazykem C # i aplikacemi WinForm a popisuje použité nástroje a technologie jako MS Visual Studio, MS SQL server, DevExpress, ClickOnce a Bunifu Framework.

Praktická část začíná analýzou požadavků a návrhem databáze, návrhem formulářů aplikace a následné programováním funkcí aplikace. Po implementaci funkcí aplikace bude aplikace otestována na zubní klinice a bude shromážděna zpětná vazba. Na základě teoretické i praktické části bude formulován závěr a navržen možný budoucí vývoj a vylepšení

**Klíčová slova:** C#, SQL server, visual studio, vývoj softwaru, desktopová aplikace, systém řízení, databáze.

# Table of content

<b>Introduction.....</b>	<b>1</b>
<b>1 Objectives and Methodology .....</b>	<b>2</b>
1.1 Objectives .....	2
1.2 Methodology .....	2
<b>2 Literature Review .....</b>	<b>4</b>
2.1 C# programming language .....	4
2.1.1 WinForms .....	4
2.2 Object oriented programming (OOP) in C# .....	5
2.2.1 Encapsulation.....	7
2.2.2 Inheritance .....	8
2.2.3 Polymorphism.....	10
2.3 MS visual studio.....	11
2.3.1 Code Editor.....	13
2.3.2 Debugger .....	13
2.3.3 Designer.....	14
2.4 ClickOnce .....	14
2.5 Database.....	16
2.6 MS SQL Server.....	17
2.7 Bunifu UI Framework .....	19
2.8 DevExpress .....	20
2.9 Vonage API developer .....	22
<b>3 Practical part .....</b>	<b>24</b>
3.1 Requirements analysis .....	24
3.1.1 SWOT analysis .....	26
3.1.2 Use case diagram .....	26
3.2 Database Design.....	27
3.2.1 Used Tables .....	27
3.2.2 Tables attributes .....	28
3.3 Application implementation .....	29
3.3.1 Application layers .....	29
3.3.2 Communication with database.....	30
3.3.3 Language choosing.....	33
3.3.4 Login process.....	34
3.3.5 Dashboard.....	36
3.3.6 Tools section .....	36
3.3.7 Staff section .....	39



3.3.8	Suppliers section .....	43
3.3.9	Equipment section.....	45
3.3.10	Backup.....	45
3.3.11	Reporting .....	47
3.3.12	Exporting .....	50
3.4	Application testing .....	51
3.5	Application deployment .....	51
<b>5</b>	<b>Conclusion .....</b>	<b>52</b>
<b>6</b>	<b>References.....</b>	<b>53</b>

## List of pictures

Picture 1	- Public class implementation - Source: author.....	6
Picture 2	- Static class implementation - Source: author.....	7
Picture 3	- Inheritance example - Source: author.....	9
Picture 4	- polymorphism types - source: <a href="https://www.studytonight.com/post/csharp-polymorphism">https://www.studytonight.com/post/csharp-polymorphism</a> .....	10
Picture 5	- MS visual studio interface - source: author .....	12
Picture 6	- MS SQL server structure - source: <a href="https://rdbmsknowledge.blogspot.com/2015/06/sql-server-architecture.html">https://rdbmsknowledge.blogspot.com/2015/06/sql-server-architecture.html</a> .....	19
Picture 7	- Bunifu framework installation - Source: author .....	20
Picture 8	- DevExpress reporting - Source: <a href="http://devexpress.com/subscriptions/reporting/">devexpress.com/subscriptions/reporting/</a> .....	22
Picture 9	- SMS API config - source: <a href="https://dashboard.nexmo.com/getting-started/sms">https://dashboard.nexmo.com/getting-started/sms</a> .....	23
Picture 10	- SWOT analysis - Source: author .....	26
Picture 11	- Use Case Diagram - Source: author .....	27
Picture 12	- Application layers - Source: author.....	29
Picture 13	- language form - Source: author.....	33
Picture 14	- Login Form - Source: author.....	35
Picture 15	- User Dashboard - Source: author .....	36
Picture 16	- Tools Management - Source: author .....	37
Picture 17	- Adding new tool - source: author.....	38
Picture 18	- Updating tool's info - Source: author .....	39
Picture 19	- SMS Form - Source: author .....	40
Picture 20	- SMS reminder - Source: author.....	40
Picture 21	- Adding new supplier - Source: author.....	43
Picture 22	- Backup form - Source: author.....	45
Picture 23	- Successful backup - Source: author.....	46
Picture 24	- Reporting layers - source: author .....	48
Picture 25	- Report Designer - Source: author.....	49

## List of tables

Table 1	- Database's tables - Source: author.....	27
Table 2	- Table's attributes - source: author.....	29

## **Introduction**

Information technology has important effects on our general life where we use it directly or indirectly in our daily routine, moreover, it also has a very big effect on business nowadays regardless the location or size of the enterprise.

Nowadays the formula for business success has become clear and simple: drive innovation with information technology, so for that reason, any entrepreneur or enterprise should use all available information if it wants to succeed in today's global environment, and affects the culture, efficiency and relationships of the business.

As almost all applications and systems related to the medical field and especially to dentists, are related to patients, appointments and payments management, many more different things are still treated in the normal way in a paper form and needs a lot of efforts and time for managing and tracking.

For this reason, this diploma thesis proposes a design and implementation of a software that can store and manage all unautomated paperwork in a dental clinic such as managing all the used tools and materials, managing employees' affairs etc. in one convenient and user friendly software so that a lot of time and work can be saved, required tasks can be accomplished easier and the paperwork is eliminated.

# **1 Objectives and Methodology**

## **1.1 Objectives**

This diploma thesis focuses on designing and implementing a desktop application for a dental clinic. The main goal of this application is to store, automate, organize, and manage all the clinic's paperwork in one convenient and user-friendly system that facilitates the doctor's job which saves time and effort.

The partial goals of the thesis are:

- Describe application development procedures.
- Describe the technologies used.

## **1.2 Methodology**

Based on information obtained from professional sources and practice, technologies and tools that can be used in the development of applications for the Windows operating system in the windows form environment will be described.

The knowledge gained from the theoretical part will be applied in the practical part, where the goal is to design and implement an application for dental clinic management, that automates and manages all the paperwork in the clinic.

First, the requirements will be gathered and analysed, based on the analysis, it will be decided what functions the application will have and how the application will look like.

Also, a use case diagram will be used to explain the system's behaviour with 2 types of users admin and normal user.

After analysing the requirements and defining the application's functions, the design of the database using Microsoft SQL Server 2019 will be explained by describing the needed tables, the attributes of each table and the data type of each attribute as well the keys (primary and foreign keys).

Designing the database with its attributes will be followed by entity diagram, in order to ensure a useful way of accessing the data and eliminating redundant data which saves disk space.

the application implementation phase will take place using Microsoft Visual Studio 2019, where the application will consist of 3 main layers, Application, Data access, and Presentation layer, each layer will be responsible for set of functions and tasks.

During implementation, the application's individual parts will be tested, and after implementation the final version will be tested on client device with real data. Which is much more efficient than using randomly generated monotonous data for testing. After testing, the application will be deployed using ClickOnce technology

## 2 Literature Review

### 2.1 C# programming language

C # is a high-level object-oriented programming based on older C, C ++ programming languages and moreover it's also very similar to Java, developed by Microsoft by Anders Hejlsberg and his team within the .Net initiative, this programming language was introduced along with the developmental .NET and later approved by ECMA and ISO standardization committees. The main features of C# are based on .NET features and used mainly to create forms, database or web applications as well as web services. (Nagel *et al.*,2010)

*“Nowadays C# is one of the most popular programming languages. It is used by millions of developers worldwide. Because C# is developed by Microsoft as part of their modern platform for development and execution of applications, the .NET Framework, the language is widely spread among Microsoft-oriented companies, organizations and individual developers”*(Co., 2013).

Moreover, C# is an easy language to start as it's closer to the other popular programming languages such like Java and C++, which make it easy language to learn, moreover, it's widely used for developing desktop and web applications and especially for desktop applications whitest it's the first choice when someone wants to develop Microsoft apps.

*“There are a great many programming languages around, during your career you will have to learn more than just one. C# is a great language to start programming in, but do not think that it is the only language you will ever need”*(Miles, 2009).

Other features such like object-oriented, interfaces, Automatic Garbage Collection, etc. makes C# common language for gaming development as well.

#### 2.1.1 WinForms

Windows Forms (WinForms) is an open source and free graphical user interface (GUI) class library which is considered as a part of Microsoft .Net framework, where it provides a platform to code applications for many platforms such as laptops, desktops, and tablets (*Your First Application: Take Two | Getting Started with .NET Development Using C# | InformIT*, 2013).

Microsoft has announced releasing Windows Forms as an open source project on GitHub on December 4, 2018, under the MIT License allowing WinForms to become available for projects targeting the .NET Core framework. Moreover, WinForms is yet available on the Mono's incomplete implementation of WinForms remains the only cross-platform implementation and Windows platform.

Windows Forms is like a Microsoft Foundation Class (MFC) library in terms of developing client applications. Where it provides a wrapper consisting of a set of C++ classes for development of Windows applications where every control in a Windows Forms application is a concrete instance of a class. However the main difference is that it does not provide a default application framework like the MFC (*What is Windows Forms - Windows Forms .NET / Microsoft Docs, 2021*).

The main properties of WinForms are, that all visual elements class library derive from the Control class which provides the minimal functionality of a user interface element such as size, font, colour, text, and location as well as common events, such like click and drag/drop. Moreover, the Control class also has docking support to let a control rearrange its position under its parent. (*Windows Forms Designer for .NET Core Released | .NET Blog, 2021*).

*“Besides providing access to native Windows controls like button, textbox, checkbox and listview, Windows Forms added its own controls for ActiveX hosting, layout arrangement, validation and rich data binding. Those controls are rendered using GD”* (Griffiths et al., 2002)

## **2.2 Object oriented programming (OOP) in C#**

Object oriented programming (OOP) is a programming model based on the concept of Object that contains data and methods, programs in OOP are organized around data and methods rather than logic and action, this model is suitable for complex, large and actively updated programs, moreover, it's also convenient for collaborative development where the projects are being divided into groups which allows code reusability, scalability and efficiency.(Kendal, 1997)

The basic component in OOP is the object which is the first thing that we think about when designing a program or software, object can be described as anything in our real life (car,

house, mobile, dog, etc.) and each object has its own properties and methods. Each object is an instance of a class or subclass within class's own procedures and methods.

Object properties are defined or represented in the program in form of variables that the object holds, for example object house can have properties like area, location, price, etc (Budd, 2001).

```
public class House
{
    public string Location { get; set; } // properties
    public string Owner { get; set; }
    public House(string location, string owner) // constructor
    {
        this.Location = location;
        this.Owner = owner;
    }

    public void HouseOwner() // Method
    {
        Console.WriteLine(this.Location + " " + this.Owner);
    }
}
House firstHouse = new House("Praha 6", "Bahaa Farag"); // creating instance
```

**Picture 1 - Public class implementation - Source: author**

A special type of classes exists in C# is called a static class using “static” keyword, which is a class that can’t be instantiated, or in other words, we can’t use the “new” operator to create a variable of the class type, simply because there’s no instance variable, the class members are accessible using the class name itself.

To simplify the static class, can be used as a box for the methods that operates on input parameters and do not have to get or set any internal instance fields.

*“A class can be marked static, indicating that it must be composed solely of static members and cannot be subclassed. The System.Console and System.Math classes are good examples of static classes”* (Albahari, Joseph & Albahari, 2012).

```

public static class MyStaticClass
{
    public static int myStaticVariable = 0;

    public static void MyStaticMethod()
    {
        Console.WriteLine("This is a static method.");
    }

    public static int MyStaticProperty { get; set; }
}

```

Picture 2 - Static class implementation - Source: author

### 2.2.1 Encapsulation

One of the most important principles of OOP is encapsulation, which is a mechanism that wraps code and the data it manipulates together, moreover, encapsulation is a protective shield that avoids the data from being accessed by the code outside this shield, in other words, it enables group of properties, members and methods to be as a single object or unit ((6) *Object-Oriented Programming*, 2020).

Encapsulation of data is allowed in C# through accessor to get the data, and mutators to modify the data so that the private data can be manipulated without making it public in an indirect way.

*“Encapsulation is the process in which no direct access is granted to the data, instead, it is hidden. If you want to gain access to the data, you have to interact with the object responsible for the data”.* (Churcher, 2007)

Encapsulation protects the data from unexpected damage or corruption instead of defining it in a way form public, this field can be declared as private, Properties provide a single point of access to an object (by get or set values), moreover, properties are accessed in either read only mode or read write mode and they are an alternate mechanism for the private data to be encapsulated, Properties provide a single point of access to an object (by get or set values), moreover, properties are accessed in either read only mode or read write



mode and they are an alternate mechanism for the private data to be encapsulated. (Guéhéneuc, 2013)

Encapsulation can be implemented in a different level of accessing data using following access modifiers:

- Public access: Access to the whole code in the program
- Private access: Access to the members of the same class only
- Internal access: Access to current assembly
- Protected Internal access: Access to current assembly and types derived from containing class.
- Protected access: Access to members of same class and its derived classes

Encapsulation has many advantages such as:

- Code testing: testing the encapsulated code is easier for unit testing
- Flexibility: based on the requirements, variables of the class can be read-only using Get accessor or read-write using Set Accessor.
- Reusability: encapsulation makes changes the requirements easy and improves the reusability. (Guéhéneuc, 2013)

### **2.2.2 Inheritance**

*“Inheritance is the process by which one object can acquire the properties of another object. This is important because it supports the concept of hierarchical classification”.*(Herbert Schildt, 2010)

Inheritance provides allowing to create or define a class based on another class or on terms of another class to extend or customize the original class, the basic existing class is called the base, and the new created class are referred to as a derived class (Kendal, 1997).

Once a class is derived from another class, it immediately inherits all its properties and methods of the parent class, however, the child class can also define its own methods and behaviour if required (Griffiths *et al.*, 2002)

Inheriting from a class allows to reuse the functionality in that class rather than building it from scratch, moreover, a class can inherit from a single class only, but can itself be inherited by many classes.

*“You use inheritance in OOP to classify the objects in your programs according to common characteristics and function. This makes working with the objects easier and more intuitive. It also makes programming easier because it enables you to combine general characteristics into a parent object and inherit these characteristics in the child objects”.*(Churcher, 2007)

In the example below, the derived class (car) inherits the fields and methods from the base class (Vehicle).

```
class Vehicle // base class (parent)
{
    public string brand = "Toyota"; // Vehicle field
    public void horn() // Vehicle method
    {
        Console.WriteLine("peep, peep!");
    }
}

class Car : Vehicle // derived class (child)
{
    public string modelName = "Camry"; // Car field
}

class Program
{
    static void Main(string[] args)
    {
        Car myCar = new Car(); // Create a myCar object

        myCar.horn(); // Call the horn method on the myCar object

        // Display the value of the brand field (from the Vehicle class)
        //and the value of the modelName from the Car class

        Console.WriteLine(myCar.brand + " " + myCar.modelName);
    }
}
```

**Picture 3 - Inheritance example - Source: author**

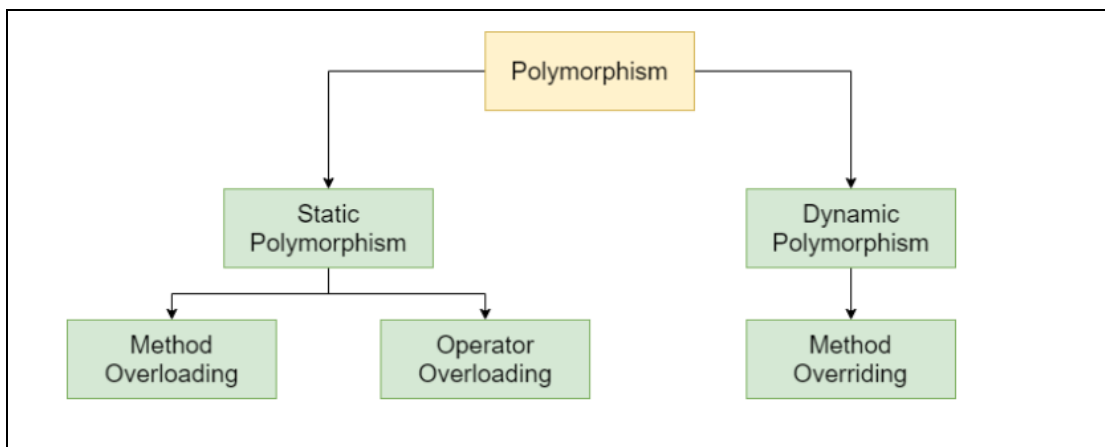
### 2.2.3 Polymorphism

Polymorphism means many forms, its combined from two words “poly” which means many “morphs” which means forms, polymorphism happens when there are many classes related to each other by inheritance, it also refers to ‘one interface, multiple functions’.

Polymorphism represents the ability of representing the same interface for multiple forms, while the main concept of polymorphism is same in all programming languages, however, its implementation differs from one programming language to another.(Ronald *et al.*, 2009)

*“Polymorphism is a fancy name for a common idea. Someone who knows how to drive can get into and drive most cars because they have a set of shared key characteristics - steering wheel, gear stick, pedals for clutch, brake and accelerator etc - which the driver knows how to use. There will be lots of differences between any two cars, but you can think of them as subclasses of a superclass which defines these crucial shared operations”.*  
(Kendal, 1997)

Based on the function response, polymorphism is divided into two types: static and dynamic polymorphism, in static polymorphism the response is determined at the compile time because the decision of which method is to be called is made at compile time, it’s also called static binding and it has two ways of implementing it, function overloading and operator overloading (Guéhéneuc, 2013)



Picture 4 - polymorphism types - source: <https://www.studytonight.com/post/csharp-polymorphism>

Dynamic polymorphism is also known as late binding or run time polymorphism because calling an overridden function is being resolved at the run time. Method overriding means having two or more methods with same names, same signature but different implementation.

In this case the method is being called by reference variable of a superclass, the decision of calling a method is based on the object being referred to by reference variable. (Kendal, 1997)

### **2.3 MS visual studio**

Microsoft visual studio is an incorporated environment for development, developed by Microsoft.

it is used for developing desktop application (console and UI apps), websites, web apps, web services and mobile applications. (*Overview of Visual Studio | Microsoft Docs,2021*)

36 programming languages are supported in MS visual studio, furthermore, it allows the debugger and coder to support almost any programming language, built in languages are C, C#, C++, C++/CLI, Visual Basic .NET, F#, XML, XSLT, JavaScript, TypeScript, CSS, and HTML. Moreover ,the support for more programming languages like as Python, Ruby, Node.js, and M among others is also available in Visual Studio via plug-ins. And in the past, Java and J# were also supported (*Visual Studio IDE documentation | Microsoft Docs, 2021*).

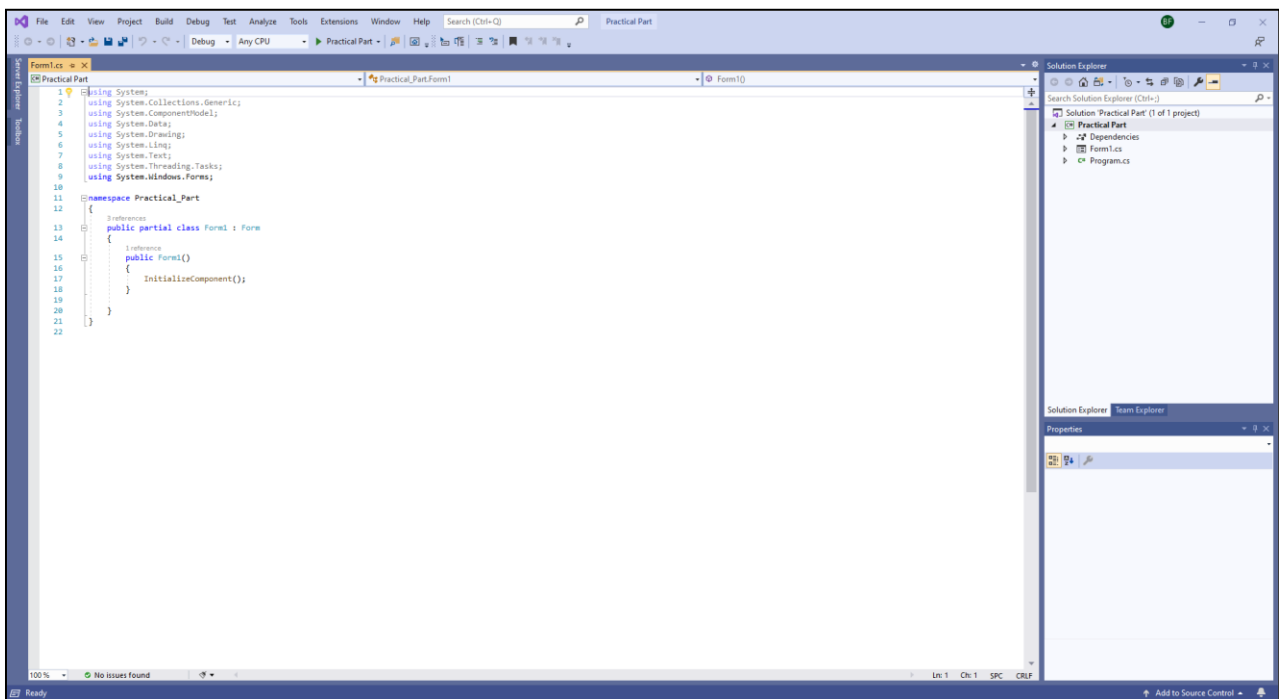
Visual studio community edition is the most basic edition, which was announced in November 12<sup>th</sup> 2014 as a new and free of charge edition with similar functions of the professional edition, community edition is suitable for individual developers and who have no restriction on the use of this edition, where enterprise organizations with more than 6 users needs to buy a commercial licence (*Install Visual Studio | Microsoft Docs, 2021*).

Professional edition is the entry level commercial version of visual studio, basically it includes tools like Server Explorer and integration with Microsoft SQL Server also. Moreover, It also supports XSLT and XML , also can create deployment packages that only use ClickOnce and MSI.

Visual Studio also provides an IDE for all supported development languages. The support for MSDN is also available as MSDN Essentials or the full MSDN library and it's depending on licensing (*Install Visual Studio | Microsoft Docs*, 2021).

in the standard edition of Visual Studio 2005, Windows Mobile development support was included, however, and with Visual Studio 2008, it is only available in Professional and higher editions. Moreover, the Windows Phone 7 development support was also included in all editions in Visual Studio 2010.

Yet another two more editions of MS visual studio are available, one is the enterprise edition which provides a group of tools for software and database development, collaboration, testing and reporting tools. And the second one is test professional edition which is mainly for dedicated testers role and it supports management and testing environment (*Introduction to projects and solutions - Visual Studio | Microsoft Docs*, 2021).



**Picture 5 - MS visual studio interface - source: author**

### 2.3.1 Code Editor

The code editor in MS visual studio supports syntax highlighting and auto-completion code using IntelliSense, which suggests code from the .NET Framework library, or even more from other available libraries, and from all the source code in the project.

Pressing shortcuts Ctrl + space to turn on whisper. IntelliSense completes the code as you type the variable, functions, methods, cycles and queries. The editor includes bookmark support, code collapsing or refactoring. Refactoring is making changes to code that do not affect behaviour other external code (*Introduction to editing in the code editor - Visual Studio | Microsoft Docs, 2021*).

Refactoring improves the quality, readability, and cleanliness of the code. On the background is running a compilation of code to notify about syntax errors

In MS Visual Studio, the background compilation was firstly provided, but now it has been included and expanded for all included languages (*Visual Studio IDE, Code Editor, Azure DevOps, & App Center - Visual Studio, 2021*).

### 2.3.2 Debugger

Debugger is a software tool for finding errors in application development. Breakpoints are used to detect errors that stop a program from running at a specific location (*First look at the debugger - Visual Studio | Microsoft Docs, 2021*).

It is also possible to use conditional breakpoints which are activated when a certain condition is met. Another way to track errors is to use watch which displays the values of the variables while the application is running. Stepping is also used. As we step through, the code stops after each line and the next code is not executed until we continue to step further. In Visual Studio, the code can be edited during debugging, using the Edit and Continue functions. (*Debug using the Just-In-Time Debugger - Visual Studio | Microsoft Docs, 2021*)

### 2.3.3 Designer

*“One thing that is integrated into an Integrated Development Environment is a way to edit files graphically, sometimes called a Graphic Development Environment or designer. Visual Studio allows you to graphically edit five different types of code bases and provides adjunct tools for the further management of said code bases.”(Davis, 2010)*

Visual Studio supports various designers, depending on the type of application being created. The available designers are:

- Windows Forms Designer: used to build GUI applications using Windows Forms
- WPF Designer: has been added to Visual Studio 2008 and can handle all the features of the Windows Presentation Foundation.
- Web designer/development: used for developing ASP.NET applications and supports HTML, CSS and JavaScript
- Class designer: used to author and edit the classes including its members and their access using UML modelling
- Data designer: used to graphically edit database schemas, including typed tables, primary and foreign keys and constraints.
- Mapping designer: used by LINQ to SQL to design the mapping between database schemas and the classes that encapsulate the data, used from Visual Studio 2008 onwards(*Debug using the Just-In-Time Debugger - Visual Studio | Microsoft Docs, 2021*).

## 2.4 ClickOnce

If you have deployed your projects using visual C#, Microsoft Visual Studio provides a full comprehensive support for updating and publishing the applications using ClickOnce technology which is a deployment technology that allows you to create a self-updating windows based apps that can be managed and installed with minimum user interaction.

Applications are mainly divided into desktop and web applications. Each one of them has its advantages and disadvantages (*ClickOnce Security and Deployment - Visual Studio | Microsoft Docs, 2021*).

Nowadays, companies prefer to invest more in web applications. Their advantage is obvious, as a web application runs on all computers that are connected to the Internet and nothing is needed but a web browser which is a strong advantage, and often overlooks

some of the disadvantages such as network dependency, slower data flow and application work, or possible security risks of data leakage in the case of a poor ISP. (*ClickOnce Reference - Visual Studio | Microsoft Docs*, 2021)

Moreover, web applications have another key advantage which is very easy to deploy the application or deploy a new version. For example, if the developer of a web application detects an error, after fixing it, he can upload the solution to the server that evening. Thus, users who use the application will notice the change as soon as they start working with the application again and do not have to worry about anything. Unlike desktop applications, where the user would first have to notice that a new version has been released, then download and install it (*ClickOnce Deployment for Windows Forms Applications | Microsoft Docs*, 2021).

For desktop applications, the user works with a version that was made sometime in the past, and there is no guarantee that it is the latest version. This cannot happen with web applications. Here the user uses the version that is currently running on the server. Thus, the biggest advantages of desktop applications unconditionally include high user comfort, known as the "rich user experience", or very fast application response.

To overcome this issue, using ClickOnce will resolve 3 main issues:

- **Difficulties in updating applications:** where the updates will be provided automatically making only those parts of the application that have changed are downloaded, and then the full, updated application is reinstalled from a new side-by-side folder.
- **Impact to the user's computer:** each app with ClickOnce cannot interfere with other apps and is self-contained.
- **Security permissions:** ClickOnce deployment enables non-administrative users to install, and only the Code Access Security permissions necessary for the application will be granted (*Testing ClickOnce Applications - Overview | TestComplete Documentation*, 2021).

Before, these limits made developers choose to make Web applications rather than Windows-based applications, giving up a rich UI for simplicity of establishment. By utilizing applications sent utilizing ClickOnce, you can have the best of the two advances.



## 2.5 Database

In applications development, the data should be stored and retrieved, for this reason a database is used which is a collection of data organized in a way that it can be easily accessed, managed and updated. Databases usually contain collections of data records or files, containing data about sales, customers, transactions or financial data etc.

usually, a database manager gives users with the ability to control access and analyse the usage, however some databases offer atomicity, consistency, isolation and durability (ACID) compliance in order to ensure that data is consistent and that transactions are completed.

Database management system (DBMS) is software used to allow users to access, manipulate, manage, and retrieve the data in the database. There are many DBMS examples like Oracle, SQL server, MySQL etc.

*“A database management system, or DBMS, is software designed to assist in maintaining and utilizing large collections of data. The need for such systems, as well as their use, is growing rapidly. The alternative to using a DBMS is to store the data in files and write application-specific code to manage it”.* (Ramakrishnan and Gehrke, 2003)

There are 4 major types of DBMS:

- Hierarchal DBMS
- Network DBMS
- Relational DBMS
- Object oriented relation DBMS

Relational databases are based on relational model, which provides a standard way of representing and querying data that could be used by any application. Moreover, relational database type stores and provides access to data parts that are related to one another, it is a straightforward way of representing and organizing data in tables, where each and every row in the table is distinguished as a record with a unique specified ID called the key. each column of the table has attributes of the data, and each attribute has a value, which makes it easy to create the relationships among data points.

There are many examples of relational database management systems such as MySQL, Oracle, and Microsoft SQL Server database, etc.

The primary key (PK) is a column or set of columns in a table that contain values that uniquely differentiate each row in the table. The primary key constraints guarantee unique data, they are frequently defined on an identity column of the table and strengths the entity integrity of the table.

*“When an attribute or set of attributes is declared as the primary key, then the attribute will not accept NULL value moreover it will not accept duplicate values. It is to be noted that only one primary key can be defined for each table.”*(Sumathi and Esakkirajan, 2007)

A foreign key (FK) is a column or combination of columns that is used to create and enforce a link between the data in two tables, so the data that can be stored in the second table (foreign key table) can be controlled. In a foreign key reference, a link is created between two tables when the column or columns that has the primary key value for one table are referenced by the column or columns in the second one, and this column becomes a foreign key in the second table.

*“Whenever the values in an attribute column in one table “point to” primary keys in another (or the same) table, the attribute column is said to be a foreign key. Columns containing foreign keys are subject to an integrity constraint: any value present as a foreign key must also be present as a primary key”.*(Robbins, 2018)

## **2.6 MS SQL Server**

*“Microsoft SQL Server is a relational database management system (RDBMS) developed by Microsoft, it is used for storing and retrieving data as requested by other software applications that runs over the same computer or on another computer across a network”* (SQL Server technical documentation - SQL Server | Microsoft Docs,2021).

Moreover, it also provides a suite of tools that help to build, change, and manage the data. As well as tools for report writing, data (import & export), and data analysis tools.

MS SQL server, as the other relational database management system software has been built on top of structured query language (SQL), which is the programming languages used by the database admins in order to manage the database and the data that contains.

MS SQL server is entitled to deal with SQL, or T-SQL, however, Microsoft's implementation of SQL that adds a set of proprietary programming constructs (*What's new in SQL Server 2019 - SQL Server / Microsoft Docs, 2021*).

*"Microsoft has made major investments in the SQL Server product as a whole; however, the new features and breakthrough capabilities that should interest database administrators (DBAs) are divided in the chapter into the following categories: Availability, Manageability, Programmability, Scalability and Performance, and Security."* (Simpkins and Simpkins, 2016)

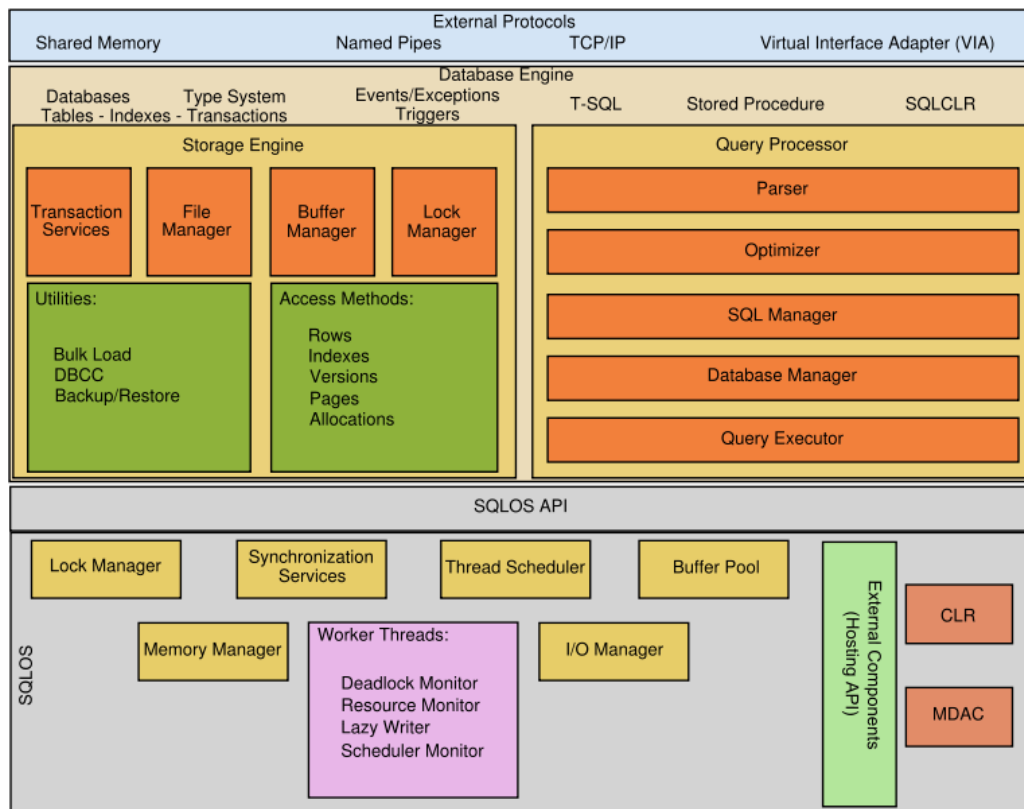
MS SQL server has two main components: database engine and SQLOS (figure 4), the database engine is the core component that consists of relational engine for processing the queries and the storage engine for managing database files and pages (*Editions and supported features of SQL Server 2019 - SQL Server / Microsoft Docs, 2021*).

Under the relational and storage engines, there is the SQL Server Operating System (SQLOS) which is responsible for providing many operating system services such as memory and I/O management. Other services may include exception handling and synchronization services.

For more than 20 years, MS SQL server has been available on for Windows operating systems, however, in 2016, Microsoft made it available on Linux. SQL Server 2017 became widely available in October 2016 that ran on both Windows and Linux (*Microsoft SQL samples - SQL Server / Microsoft Docs, 2021*).

SQL Server has four primary editions that have different bundled services and tools, however, two of them are available free of charge:

- SQL Server Developer edition for use in database development and testing.
- SQL Server Expression for small databases with the size up to 10 GB of disk storage capacity.
- For larger and more critical applications, SQL Server offers the Enterprise edition that includes all SQL server's features.
- SQL Server Standard Edition has partial feature sets of the Enterprise Edition and limits on the Server regarding the numbers of processor core and memory that can be configured (*Editions and supported features of SQL Server 2019 - SQL Server / Microsoft Docs, 2021*).



Picture 6 - MS SQL server structure - source: <https://rdbmsknowledge.blogspot.com/2015/06/sql-server-architecture.html>

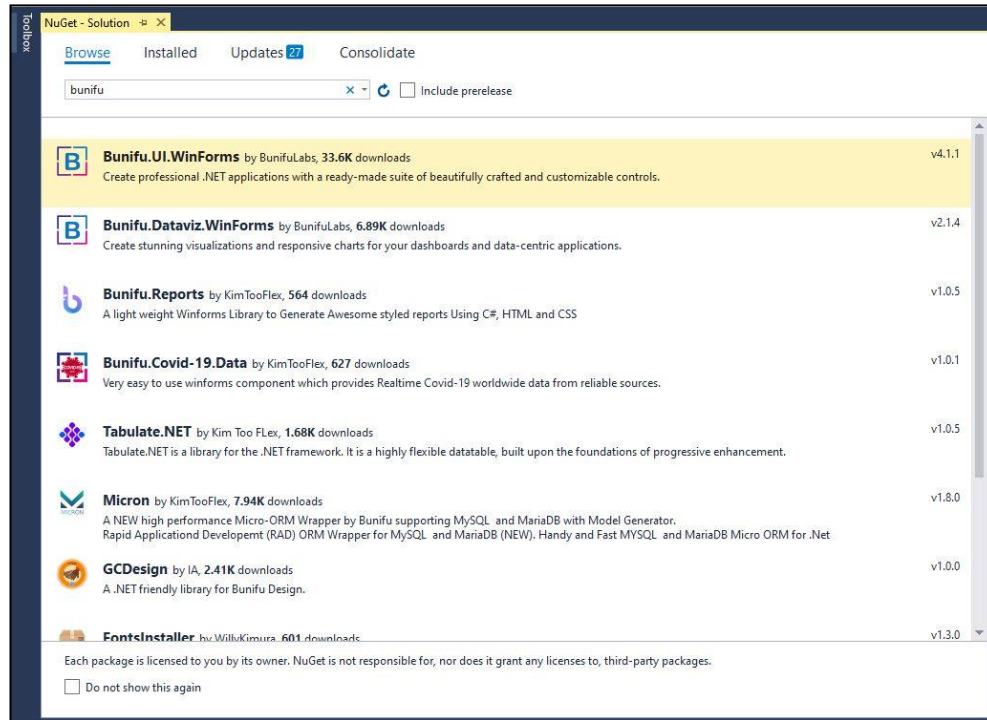
## 2.7 Bunifu UI Framework

*“Bunifu Framework is a product of Bunifu Technologies Limited (Bunifu). Bunifu was founded in 2011 and later incorporated as a limited company is 2014.*

*Bunifu has been developing custom software solutions since 2011 and with years of experience in coding, in-house custom software developer tools became part of development process. Later in 2016, Bunifu saw the need to empower other developers around the world by sharing their in-house tools at an affordable cost”(About Us - Bunifu Framework | Empowering software developers craft great user experiences in less time. Productivity tools for C# & VB.NET UX/UI design, 2021).*

Bunifu UI is an open source DLL, it provides a wide user interface. Its intended to make desktop Windows application looks more attractive and elegant, the DLLs are installed in the .NET environment and are built on top of WinForms. It also enables developers to drag and drop hence faster coding.

Downloading Bunifu framework is done through visual studio by clicking manage NuGet packages from the solution explorer searching for it. (*Bunifu Framework UI tools - Crafting stunning UI and data visualizations fast!*, 2021)



**Picture 7 - Bunifu framework installation - Source: author**

*“Bunifu UI Controls are DLL driven tools to help you build awesome desktop application interfaces. It guarantees great user experience in your apps and reduces development time for Microsoft Visual Studio .NET software developers. It empowers developers to Improve productivity Build modern stunning designs. And it targets to individual software developers, software companies and industrial companies requiring automation” (Bunifu Framework - Visual Studio .NET UI tools, 2021).*

## 2.8 DevExpress

DevExpress is a component library for the .NET Framework and a reporting tool, it provides feature-complete UI controls, business application frameworks, automated web testing tools and enterprise-ready reporting systems.

DevExpress includes:

- Desktop Controls like WinForms, WPF, UWP and Desktop Reporting.
- Enterprise & Server Tools like Dashboard and Office File APIs.

- Frameworks & Productivity Tools like XAF - Cross-Platform .NET App UI, XPO - ORM Library and CodeRush for Visual Studio.
- Web Controls like ASP.NET, ASP.NET MVC and Core, Bootstrap Web Forms, JavaScript - jQuery, Angular, React, Vue, Web Reporting and Blazor (*.NET Reporting Tools - Core, Blazor, WinForms, MVC / DevExpress, 2020*).

Moreover, it also includes TestCafe Studio, Native Mobile UI Controls, Xamarin.Forms UI Controls, Priority Support and Source code for all controls and libraries, Icon Library.

For desktop controls, DevExpress provides:

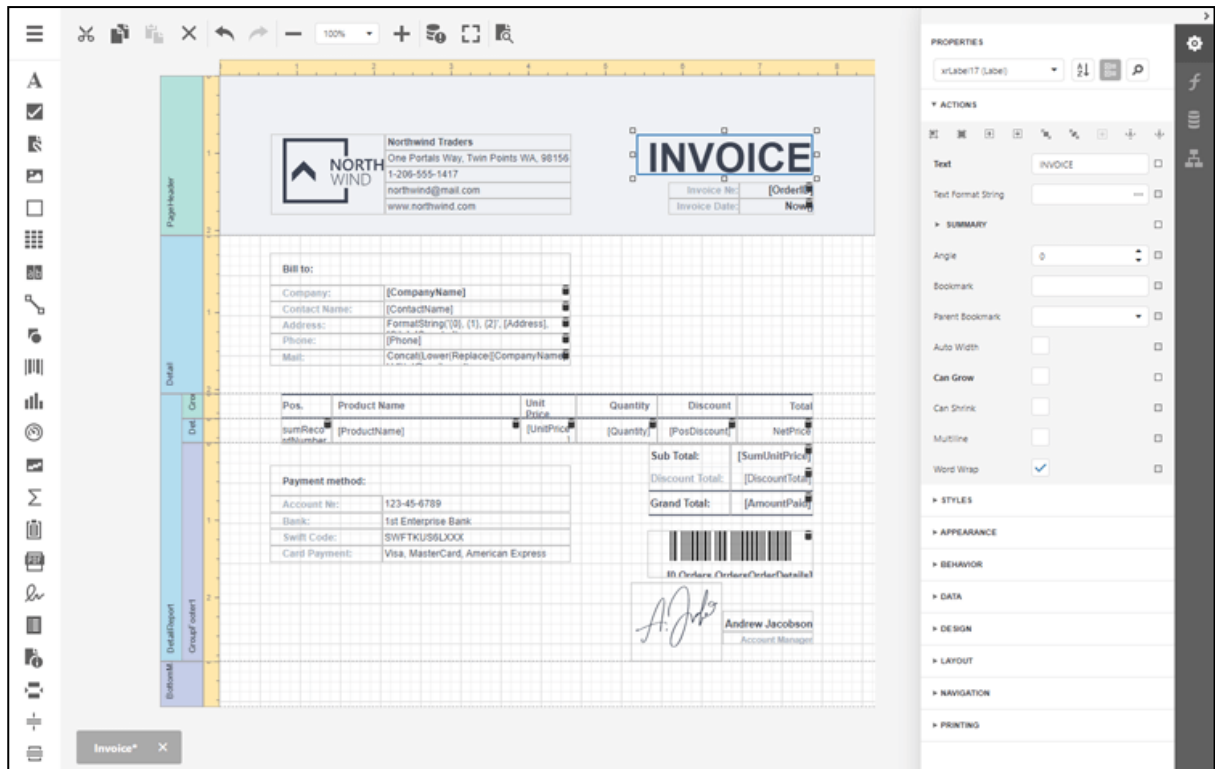
- WinForms: the DevExpress WinForms has everything the developer's need to create high-impact business solutions for the Windows Forms platform With over 190 controls.
- WPF: The DevExpress WPF provides with over 120 UI Controls and Libraries which allows the developer to build Microsoft Office inspired, as well as data analysis application.
- UWP: DevExpress Delivers an elegant, touch-enabled form of applications on the famous and know Windows.
- Desktop Reporting - The DevExpress Reporting provides elegant and easy-to-use customization options and a wide set of report controls, styles, colours and many more.

DevExpress Reports come with an impulsive Visual Studio report designer, runtime report designers for WinForms/WPF/Web, and a wide set of reporting controls, including cross tabs and charts so the developers and the users can create reports of informational clarity and extraordinary elegance(*Reporting / Reporting / DevExpress Documentation, 2020*).

*“DevExpress Reports ship with the fully integrated Visual Studio Report Designer, report wizards, pre-built report templates, and end-user report designers so you can build your best inside your favourite IDE, without limits or compromise”*(*.NET Reporting Tools - Core, Blazor, WinForms, MVC / DevExpress, 2020*).

Using DevExpress Reports provides creating a diversity of report types - from simple mail-merge, table and vertical reports up to hierarchical or (master-detail) and crosstab reports, moreover, it provides business users with the information required to monitor business outcomes and make intelligent, real-time decisions.

DevExpress also Merges multiple PDF / DOCX files together within the report where the XRPdfContent and XRRichTextreport controls embed PDF and DOCX documents alongside the report and preview the complete report within the Document Viewer (*Create a Report from A to Z | Reporting | DevExpress Documentation, 2020*).



Picture 8 - DevExpress reporting - Source: [devexpress.com/subscriptions/reporting/](https://devexpress.com/subscriptions/reporting/)

## 2.9 Vonage API developer

Vonage API includes everything the developer needs to build connected applications with Vonage APIs, including SMS, video & voice calls, verifying the user's identity and many more.

Vonage's SMS API enables the developer to send and receive text messages to and from users worldwide, using REST APIs, it also provides

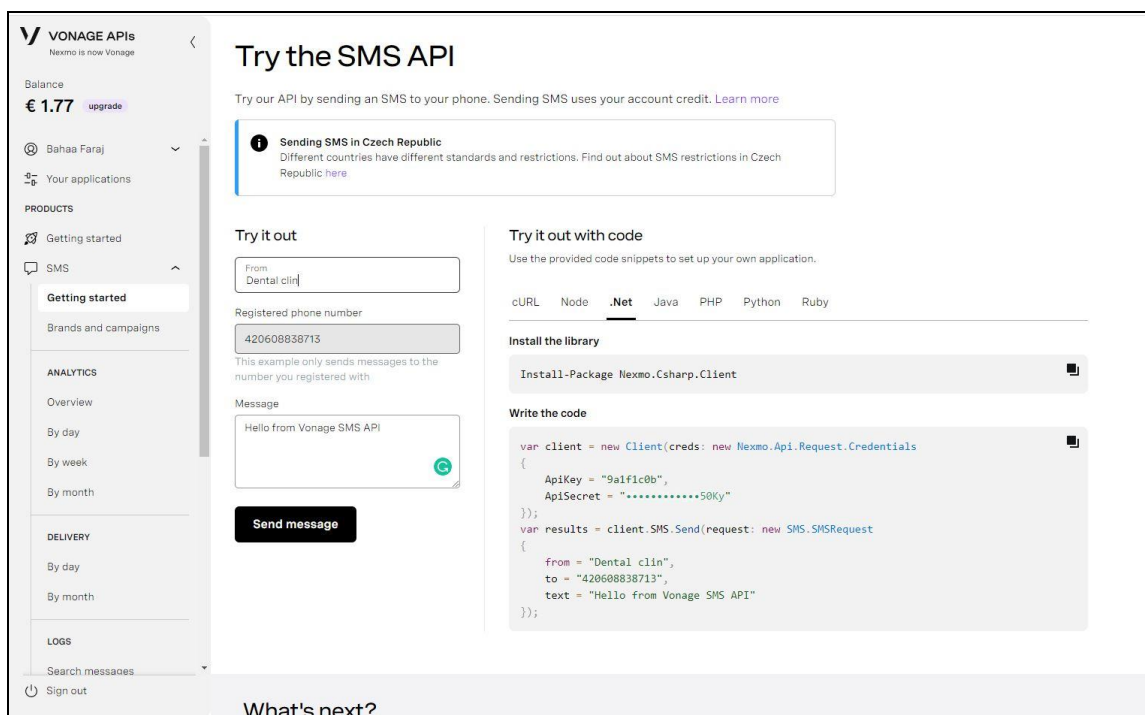
- Programmatically send and receive high volumes of SMS globally.
- Send SMS with low latency and high delivery rates.

- Receive SMS using local numbers.
- Scale the applications with familiar web technologies.
- Auto-redact feature to protect privacy (*SMS API > Send and receive SMS with the SMS API / Vonage API Developer, 2020*).

In order to start sending SMSs, you need to login into your account or signup for a new one, creating a new account will create a 1.77 € as initial credit, and some free features, most important parts are two things, a key and secret to be saved for use

After that, the SMS navigation shows the further steps and details to be specified such like the number, message and the sender as shown below (*Vonage API Developer, 2020*).

As also shown in the below picture, the website provides the code to use in many programming languages like .Net, PHP, Node and many more.



Picture 9 - SMS API config - source: <https://dashboard.nexmo.com/getting-started/sms>



### 3 Practical part

The first part of this work has introduced the tools and technologies that will be used in the practical part of creating the software application.

In this part, the process of designing, implementing, and testing the software application for windows operating system will be introduce.

The resulting software application will be intended for automating all the paperwork in a dental clinic in one convenient and user-friendly system that makes the process easier and faster and eliminate all the paperwork.

#### 3.1 Requirements analysis

As there are many applications, systems and tools that offer many functions to store and manage the patient's information, their medical profiles, payments and appointments, this software application will not treat the same aspects.

The main objective of the this application is to automate all the unautomated parts of doctor's and staff's daily tools and tasks in order to save their efforts and time and avoid all the issues and confusion caused by using the legacy method of treating and managing the daily tasks by paper or other manual work.

The first window will be the login window where the user is asked to provide his credentials, the application will have 2 types of roles for users: admin with full access to all functions, and a normal user that will have some restrictions to modify and review.

After successful login, the application will consist of the following sections

- **Tools Management:** where all the used tools will be stored in order to manage them easily and accurately (view, add, remove, edit, etc).
- **Staff Management:** where the employees' information will be stored (personal info, contracts, working hours, etc) to manage the employee's affairs easier and faster and avoid any kind of issues.
- **Suppliers Management:** this section will be like an index for all the suppliers that the company deals with, and the ability to manage them (add, remove, edit).
- **Equipment Management:** this section will contain all the matters about the equipment in the clinic (doctor's chair, x-ray machines, Sterilization equipment, etc) to manage them.

- **Users Management:** this section will contain list available users with actions for editing them or add new users.
- **Backup:** this section will allow the user to create a backup for the data in a specified location on the device.

The dashboard will allow add a new entry for each section from the mentioned above separately by a special button as well.

Based on the requirements above, it was decided to develop the application as a desktop application and moreover for the following reasons:

- The application will be mainly used on one PC intended for this purpose (reception's PC) with many user's access.
- the application doesn't need an internet connection to be accessed, so the application will be always available in case of any internet outage or cloud break.
- As a desktop application it's more secure than web application, since it has a business information, security is an important aspect to be considered.
- Since the application is running locally, any feature or function can be added.

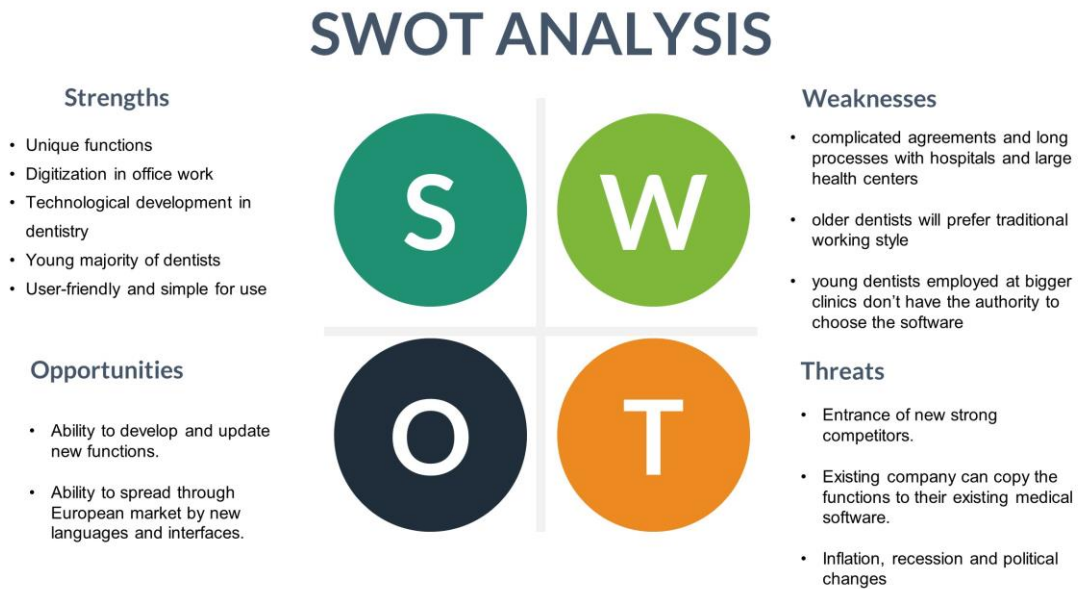
the application is considered as client – server application, the server's code (insert, update, delete,...) will be running on server's level by stored procedures to be called by the application, this way will prevent hacker's attacks, and ensures that the application will run fast since the commands execution on server level is fast and efficient.

the application will be divided into 3 layers, where each layer is responsible for set of tasks which makes the code more organized, reusable, and easier to maintain later, these layers are following:

- **Data access layer:** this layer will be responsible for:
  - Communication with the database (open / close connection)
  - Read the data from the database
  - Insert / delete / update / select the data from the database.
- **Forms:** this will present or contain all the final forms and will interact with end user.
- **Presentation layer (PL):** will contain the classes of the project for each section (suppliers, tools, users, ...) as well as functions determination and sending and receiving parameters from and to the forms.
- **Reporting:** will be for creating the reports for the reports for each section.

### 3.1.1 SWOT analysis

SWOT analysis shows Strengths, weaknesses, Opportunities and Threats, picture (10)



Picture 10 - SWOT analysis - Source: author

### 3.1.2 Use case diagram

use case diagram is a diagram created using UML (Unified Modelling Language) to show the system's behaviour, it also describes the functionality of the system and how the system will perform with one or more external users which are called actors.

The main purpose of the use case diagram is to show what the system can do not how the system is performing its functions, knowing how the system is performing is being known after designing the use case diagram.

Dental Clinic Manager Use Case Diagram



Picture 11 - Use Case Diagram - Source: author

### 3.2 Database Design

#### 3.2.1 Used Tables

The application will be communicating with SQL database that will store all the data needed, the following table (1) shows the database's tables with their content:

Table Name	Content
<b>Users_table</b>	Used to contain users and their PIN codes.
<b>Tools_table</b>	Used to store the tools for daily use.
<b>Staff_table</b>	Used to store employee's personal information.
<b>Suppliers_table</b>	Used to store supplier's information.
<b>Equipment_table</b>	Used to store equipment's details and information.

Table 1 - Database's tables - Source: author

### 3.2.2 Tables attributes

The following table (2) shows the attributes, their types and sizes per each table:

Attribute name	Table	Data type	Size	Other
<b>Tool_id</b>	Tools	Int	18	Primary key
<b>Tool_name</b>	Tools	Nvarchar	50	
<b>Tool_category</b>	Tools	Nvarchar	50	
<b>Tool_supplier</b>	Tools	Nvarchar	50	
<b>Tool_price</b>	Tools	Real	Default	
<b>Tool_validity</b>	Tools	Date	Default	
<b>Tool_notification</b>	Tools	Nvarchar	50	
<b>Tool_quantity</b>	Tools	Nvarchar	50	
<b>ID</b>	Staff	Int	18	Primary key
<b>Name</b>	Staff	Nvarchar	50	
<b>Surname</b>	Staff	Nvarchar	50	
<b>Date_of_birth</b>	Staff	Date	Default	
<b>Email</b>	Staff	Nvarchar	50	
<b>Phone_number</b>	Staff	Nvarchar	50	
<b>Address</b>	Staff	Nvarchar	50	
<b>Contract_type</b>	Staff	Nvarchar	50	
<b>Starting_date</b>	Staff	Date	Default	
<b>Ending_date</b>	Staff	Date	Default	
<b>Working_hours</b>	Staff	Nvarchar	50	
<b>notification</b>	Staff	Nvarchar	50	
<b>ID</b>	Users	Int	18	Primary key
<b>User_name</b>	Users	Nvarchar	50	
<b>User_password</b>	Users	Nvarchar	4	
<b>User_type</b>	Users	Nvarchar		
<b>ID</b>	Suppliers	Int	18	Primary key
<b>Name</b>	Suppliers	Nvarchar	50	
<b>Address</b>	Suppliers	Nvarchar	50	
<b>website</b>	Suppliers	Nvarchar	50	
<b>Email</b>	Suppliers	Nvarchar	50	
<b>Phone_number</b>	Suppliers	Nvarchar	50	
<b>Contact_person</b>	Suppliers	Nvarchar	50	
<b>Serial_number</b>	Equipment	Nvarchar	50	
<b>Name</b>	Equipment	Nvarchar	50	Primary key

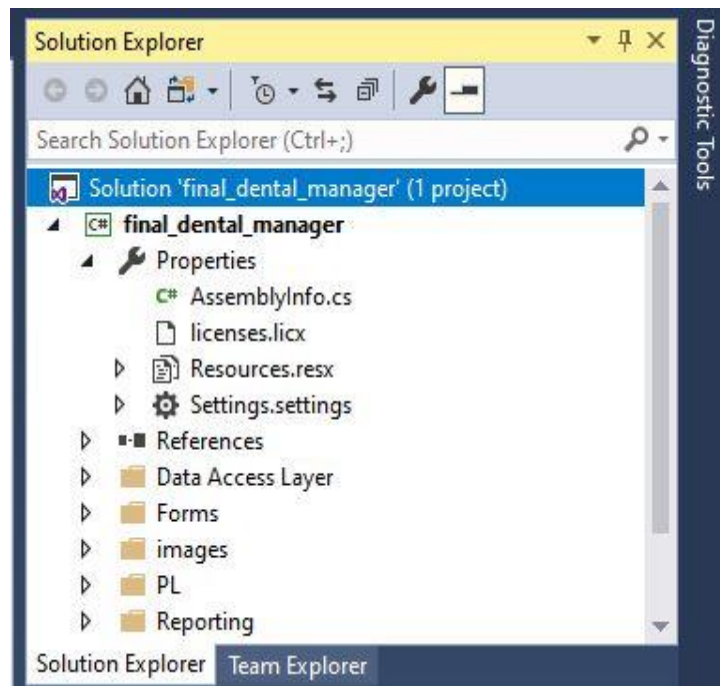
<b>Supplier</b>	Equipment	Nvarchar	50	
<b>Service_contact</b>	Equipment	Nvarchar	50	
<b>price</b>	Equipment	Float	Default	
<b>Purchasing_date</b>	Equipment	Date	Default	
<b>notification</b>	Equipment	Nvarchar	50	

**Table 2 - Table's attributes - source: author**

### 3.3 Application implementation

#### 3.3.1 Application layers

The application will consist of the following layers, picture (12), Data Access Layer (DAL),Forms, images, presentation Layer (PL) and reporting where each one of them will be represented by a folder in the project to contain all the content of each layer.



**Picture 12 - Application layers - Source: author**

Data Access Layer will allow the access to the database and will contain all the stored procedures on the server's level (update, select, delete, ...)

### **3.3.2 Communication with database**

In order to connect SQL database with visual studio, couple of libraries need to be used in the name space to enable the processes with the database, the libraries are System.Data.SqlClient and System.Data.

The folder DAL will contain the first-class (DAL) that will be responsible for initializing, opening and closing the connection with the database, read and write the information from the database.

Opening the connection with the database needs to specify 3 parameters: server name that would be copied from SQL server, database name which would be taken in the same way and integrated security which has the value of true that means that accessing the database server is by windows authentication without any special credentials.

A method will be checking the connection status, if it's not opened it opens it, and a method for closing the connection if it's already opened.

Another method will be for reading the data coming from the stored procedure on server's level, the method will take 2 parameters the stored procedure and the parameters that belongs to the stored procedure.

Another method will be used to execute the stored procedure (update, select, insert) will also take 2 parameters the stored procedure name and its parameters

```

using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.SqlClient;
using System.Data;

namespace dental_manager.DAL
{
    class DAL
    {
        SqlCommand cmd;
        SqlDataAdapter da;
        SqlConnection cn;
        DataTable dt = new DataTable();

        public DAL() //initiallizing the connection
        {
            cn = new SqlConnection(@"server=PRGN34262711A;database=Dental ;integrated
security=true");
        }

        public void open() //opening the connection
        {
            if (cn.State == ConnectionState.Closed)
            {
                cn.Open();
            }
        }
    }
}

```



```

public void close() // closing the connection
{
    if (cn.State == ConnectionState.Open)
    {
        cn.Close();
    }
}

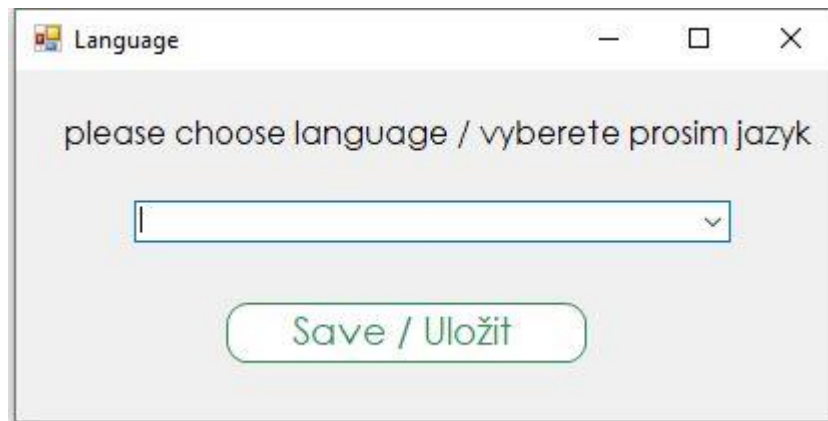
//reading the data from the DB
public DataTable Reader(string sp, SqlParameter[] p)
{
    cmd = new SqlCommand();
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.CommandText = sp;
    cmd.Connection = cn;
    if (p != null)
    {
        cmd.Parameters.AddRange(p);
    }
    da = new SqlDataAdapter(cmd);
    da.Fill(dt);
    return dt;
}

//Remove ,update ,add from the DB
public void RUA(string sp, SqlParameter[] p)
{
    cmd = new SqlCommand();

```

### 3.3.3 Language choosing

The application will be available in 2 languages, English, and Czech languages where the user will choose the language once he starts the application as shown in picture (13).



Picture 13 - language form - Source: author

In order to create language configuration, first thing is to use Globalization library and change the localizable property to true, after that the other language to be chosen by from the language property.

After setting up this configuration, the controls on forms need to be changed to the Czech language, and if we choose English from the language form's options, the controls will be switched to English back, and both languages are successfully added.

```
using System.Globalization;
using System.Linq;
using System.Text;
using System.Threading;
using System.Threading.Tasks;
using System.Windows.Forms;

private void btn_lang_Click(object sender, EventArgs e)
{
    string lang = cmb_lang.Text;
    CultureInfo cultureInfo = new CultureInfo(lang);
    Thread.CurrentThread.CurrentCulture = cultureInfo;
    Thread.CurrentThread.CurrentUICulture = cultureInfo;
    login frm = new login();
    frm.ShowDialog();
    this.Hide(); }

```

### 3.3.4 Login process

the users table in the database consists of 4 columns, user ID, username, password and type, the user will be entering his credentials and clicks login, the values from the form will be sent to the stored procedure that checks the values from the database, if the values match the user will be able to login to the main page.

The code below shows the stored procedure that will be responsible for login process

```
Create proc user_login
@user_name nvarchar (50),
@user_password nvarchar (50),
@user_type nvarchar (50)
as
select * from users_table
where
user_name = @user_name and user_password = @user_password and user_type = @user_type
```

In visual Studio, the Login class will be created in presentation Layer (PL), the class will contain a procedure which will be responsible for executing the stored procedure above, the procedure will have the same parameters that the stored procedure takes, the username and password code as shown below.

```
namespace dental_manager.BL
{
class login
{
public DataTable user_login(string username, string password, string
type)
{
Data_Access_Layer.DAL ob = new Data_Access_Layer.DAL();
DataTable dt = new DataTable();
SqlParameter[] p = new SqlParameter[3];
p[0] = new SqlParameter("@user_name", SqlDbType.NVarChar, 50);
p[0].Value = username;
p[1] = new SqlParameter("@user_password", SqlDbType.NVarChar, 50);
p[1].Value = password;
p[2] = new SqlParameter("@user_type", SqlDbType.NVarChar, 50);
p[2].Value = type;
ob.open();
dt = ob.Reader("user_login", p);
ob.close();
return dt;
}
}
```

```

private void btn_login_Click(object sender, EventArgs e)
{
    if (rd_admin.Checked)
    {
        type = "admin";

    } else
    {
        type = "user";
        Forms.dashboard ds = new Forms.dashboard();
        ds.btn_backup.Enabled = false;
        ds.btn_settings.Enabled = false;

    }

    DataTable dt = log.user_login(txt_usr.Text, txt_pss.Text, type);
    if (dt.Rows.Count > 0)
    {

        Forms.dashboard frm = new Forms.dashboard();
        frm.ShowDialog();
        this.Close();

    } else
    {
        MessageBox.Show("Username, password or type are incorrect, please
check your input", "Wrong Credentials", MessageBoxButtons.OK,
MessageBoxIcon.Warning);
    }
}

```

After programming the login process, the login form will be designed, and as mentioned before that the presentation layer (PL) will contain all the forms of the application, thanks to Metro UI the form looks more attractive than the typical Win Forms, the picture (14) below shows the login form

The screenshot shows a login window with a title bar containing 'Login' and standard window controls. On the left side, there is a stylized illustration of a man in a blue shirt and red tie, with a green checkmark in a circle next to him. To the right of the icon are two text input fields labeled 'User Name' and 'Password'. Below the 'Password' field is a checkbox labeled 'Show Password'. At the bottom, there are two radio buttons: 'Admin' and 'User'. At the very bottom, there are two buttons: 'Login' and 'Cancel'.

Picture 14 - Login Form - Source: author

### 3.3.5 Dashboard

Once the user is logged in successfully, he will be redirected to the main dashboard that contains all the application's function as shown in the picture (15) below.

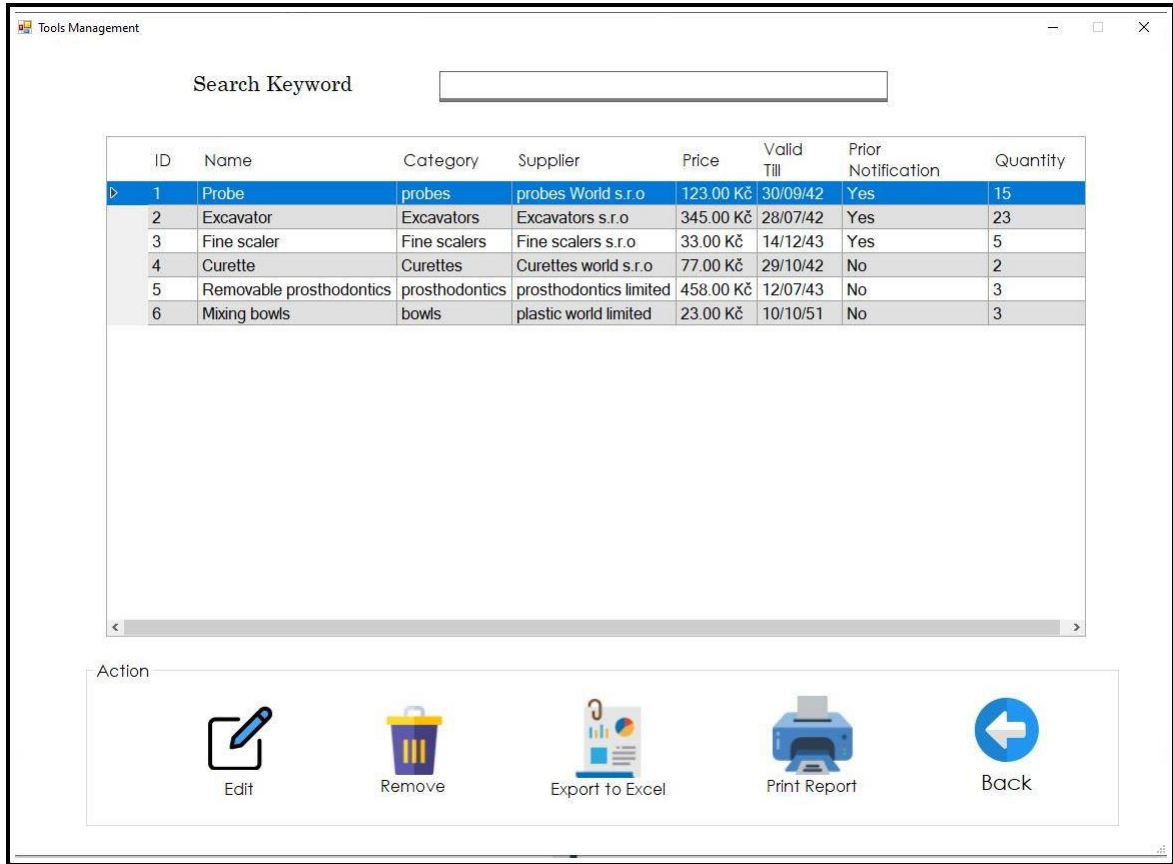


Picture 15 - User Dashboard - Source: author

### 3.3.6 Tools section

As the tools are being used daily, they may cause some issues by running out of them in an unappropriated time, or running out of their validity for some of them, the tools section will avoid such issues by storing the tools with their details including their quantity and validity and many more details in order to avoid any kind of future issues for any reason, moreover, the UI will also enable the user to make an order before the quantity runs out by sending an SMS to the supplier that specifies the tool and the quantity.

Figure(16) below shows the tools management window with some test data, where the user can search for a tool, edit or remove it, exporting the list of tools to Excel file, order from supplier via SMS and print an elegant report for the available tools.



**Picture 16 - Tools Management - Source: author**

Clicking add new tool from the dashboard will allow the user to add a new tool to the database with auto increase of the tool's ID in the ID's field as shown in the figure below, once the tool is added a confirmation message will confirm the adding process.

New\_Tool

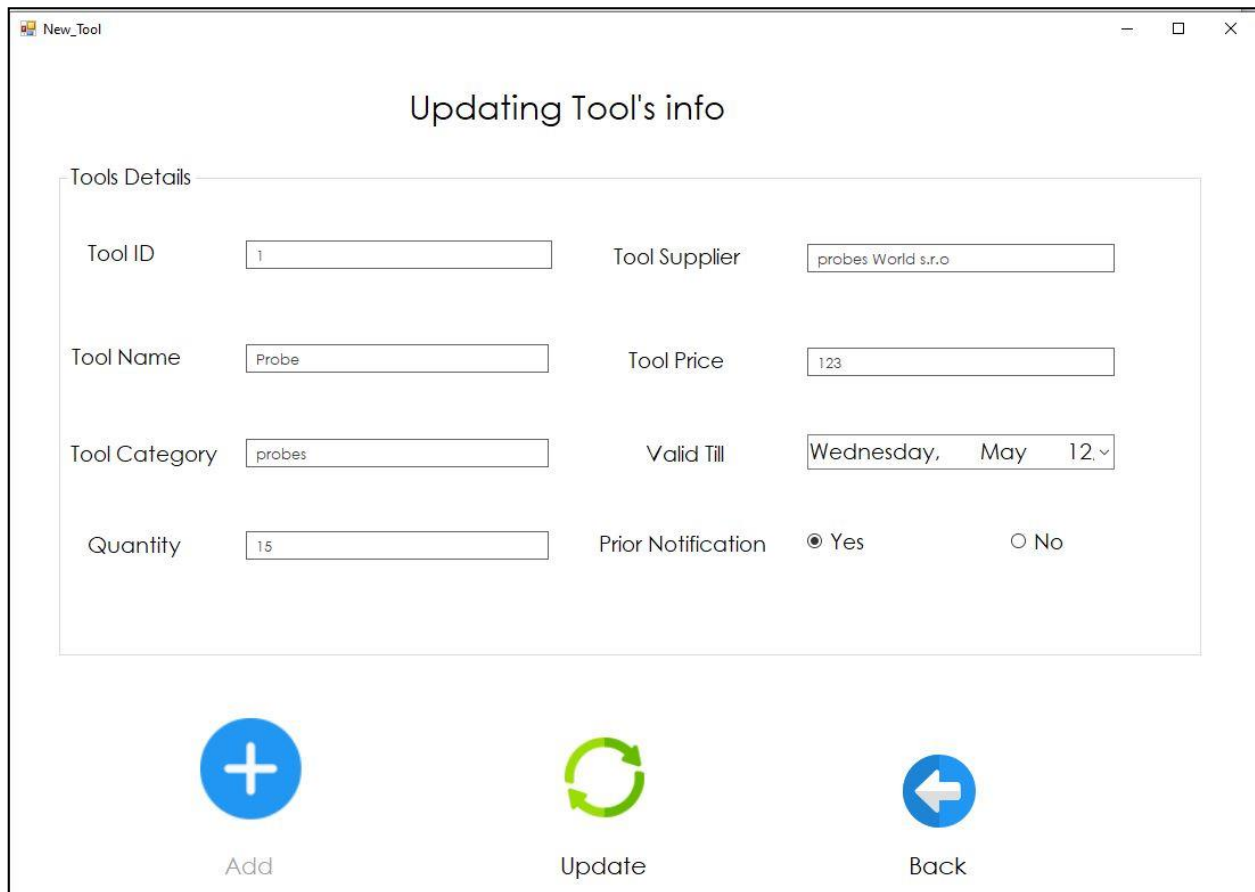
### Adding New Tool

Tools Details

Tool ID	<input type="text" value="7"/>	Tool Supplier	<input type="text"/>
Tool Name	<input type="text"/>	Tool Price	<input type="text"/>
Tool Category	<input type="text"/>	Valid Till	<input type="text" value="Tuesday , March 2"/>
Quantity	<input type="text"/>	Prior Notification	<input type="radio"/> Yes <input type="radio"/> No

**Picture 17 - Adding new tool - source: author**




by clicking edit button, the window of adding new tool will show up filled with the tools info to allow the user to edit the needed fields, the add button will be disabled and the update button will be enabled instead, as shown in the picture (18) below.



**Updating Tool's info**

Tools Details

Tool ID	<input type="text" value="1"/>	Tool Supplier	<input type="text" value="probes World s.r.o"/>
Tool Name	<input type="text" value="Probe"/>	Tool Price	<input type="text" value="123"/>
Tool Category	<input type="text" value="probes"/>	Valid Till	<input type="text" value="Wednesday, May 12, v"/>
Quantity	<input type="text" value="15"/>	Prior Notification	<input checked="" type="radio"/> Yes <input type="radio"/> No

Add
Update
Back

**Picture 18 - Updating tool's info - Source: author**

### 3.3.7 Staff section

Other issue that might be happening in the dental clinic / centre is in case of having a lot of doctors with their own staff each, each doctor's staff have different roles, contract types, working hours, addresses and many more, the staff section will store all these details on one place to view or update them just like the options for the tools, moreover, prior to the ending date of an employee, the application will show a notification to remind the employee with the expiring date of his/her contract, and it allows the user to send an SMS to the employee as a reminder. pictures (19,20).

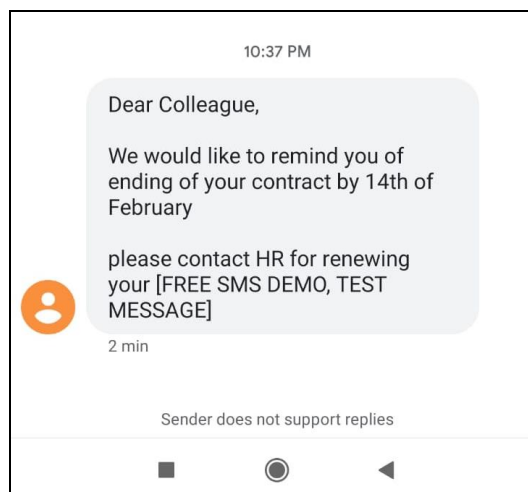


The image shows a screenshot of an SMS form titled "SMS". It contains the following fields and content:

- To:** A text input field containing "John".
- Phone Number:** A text input field containing "+420 256 663 120".
- From:** A text input field containing "Dental Manager".
- Message:** A large text area containing the following text:  
Dear Colleague,  
We would like to remind you of ending of your contract by 14th of February  
please contact HR for renewing your contract.  
thank you!

At the bottom of the form, there are two buttons: "Send" and "Cancel".

**Picture 19 - SMS Form - Source: author**



**Picture 20 - SMS reminder - Source: author**

```

using System.Data;
using System.Data.SqlClient;
namespace final_dental_manager.PL
{
    class staff
    {
        public void new_staff(int staff_id, string staff_name, string
staff_surname, DateTime birth, string email, string phone, string address,
string contract, DateTime start, DateTime end, string hours, string notifi) {
            Data_Access_Layer.DAL ob = new Data_Access_Layer.DAL();
            ob.open();
            SqlParameter[] p = new SqlParameter[12];
            p[0] = new SqlParameter("@id", SqlDbType.Int);
            p[0].Value = staff_id;
            p[1] = new SqlParameter("@name", SqlDbType.NVarChar, 50);
            p[1].Value = staff_name;
            p[2] = new SqlParameter("@surname", SqlDbType.NVarChar, 50);
            p[2].Value = staff_surname;
            p[3] = new SqlParameter("@date_of_bith", SqlDbType.DateTime);
            p[3].Value = birth;
            p[4] = new SqlParameter("@email", SqlDbType.NVarChar, 50);
            p[4].Value = email;
            p[5] = new SqlParameter("@phone", SqlDbType.NVarChar, 50);
            p[5].Value = phone;
            p[6] = new SqlParameter("@address", SqlDbType.NVarChar, 50);
            p[6].Value = address;
            p[7] = new SqlParameter("@contract", SqlDbType.NVarChar, 50);
            p[7].Value = contract;
            p[8] = new SqlParameter("@start", SqlDbType.DateTime);
            p[8].Value = start;
            p[9] = new SqlParameter("@end", SqlDbType.DateTime);
            p[9].Value = end;
            p[10] = new SqlParameter("@work_hours", SqlDbType.NVarChar, 50);
            p[10].Value = hours;
            p[11] = new SqlParameter("@nofiti", SqlDbType.NVarChar, 50);
            p[11].Value = notifi;
            ob.RUA("new_staff", p);
            ob.close(); }
    }
}

```

```

//viewing the staff in the table

public DataTable view_all_staff()
{
    DataTable dt = new DataTable();
    Data_Access_Layer.DAL ob = new Data_Access_Layer.DAL();
    ob.open();
    dt = ob.Reader("view_staff", null);
    ob.close();
    return dt;
}

//auto increment for the ID field
public DataTable show_staff_id()
{
    DataTable dt = new DataTable();
    Data_Access_Layer.DAL ob = new Data_Access_Layer.DAL();
    ob.open();
    dt = ob.Reader("show_staff_id", null);
    ob.close();
    return dt;
}

//Searching for staff
public DataTable search_staff(string STAFF_ID)
{
    DataTable dt = new DataTable();
    Data_Access_Layer.DAL ob = new Data_Access_Layer.DAL();
    ob.open();
    SqlParameter[] p = new SqlParameter[1];
    p[0] = new SqlParameter("@staff_id", SqlDbType.NVarChar,50);
    p[0].Value = STAFF_ID;
    dt = ob.Reader("search_staff", p);
    ob.close();
    return dt;
}

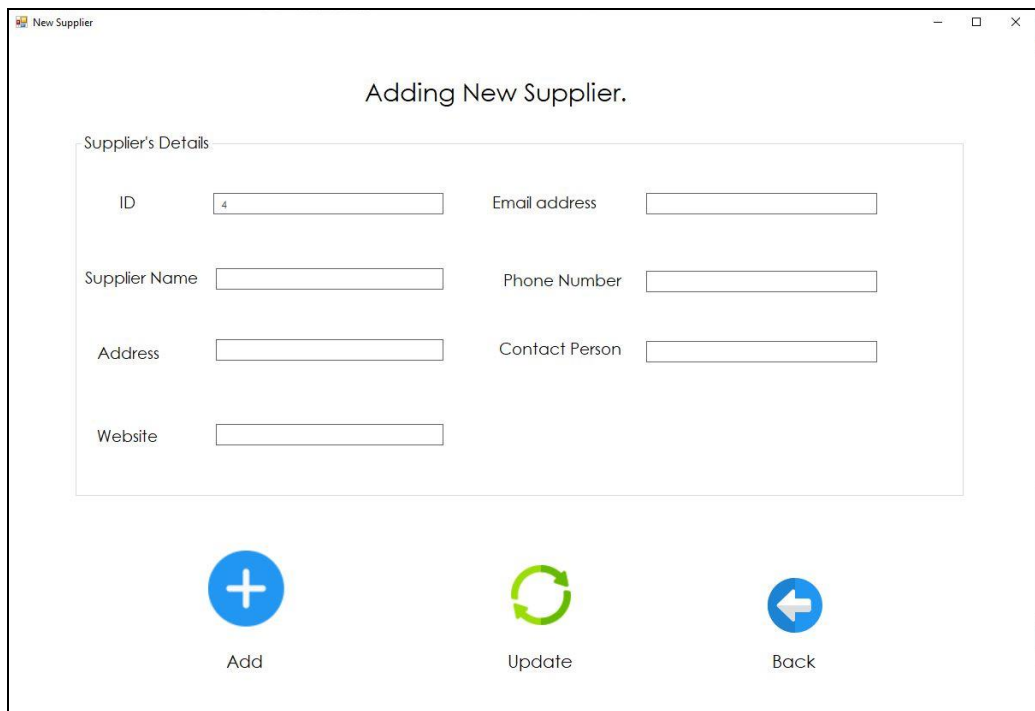
```

### 3.3.8 Suppliers section

By accessing the suppliers management section, the user will find a list of the suppliers that the dental centre deals with usually, with contact people for it and many other details without the need to store some visit cards or written phone numbers anywhere, that will also facilitate the ordering process if the user doesn't want to go back to the tools page/window, he/she can order from the suppliers management page by specifying the tool needed after choosing the wanted supplier.

Supplier management also enables editing and removing suppliers, exporting the suppliers list to an Excel file, or creating an elegant report for printing just like the previous sections for staff and tools, and of course, an instant search for a specific supplier.

Adding a new supplier is available with a new button in the dashboard as previous sections with an auto increment for the supplier ID, as shown in picture (21).



The screenshot shows a web browser window titled "New Supplier" with a form titled "Adding New Supplier." The form is enclosed in a box labeled "Supplier's Details" and contains the following fields:

- ID: 4
- Email address: [empty]
- Supplier Name: [empty]
- Phone Number: [empty]
- Address: [empty]
- Contact Person: [empty]
- Website: [empty]

Below the form are three buttons:

- Add**: A blue circular button with a white plus sign.
- Update**: A green circular button with a white refresh arrow.
- Back**: A blue circular button with a white left-pointing arrow.

**Picture 21 - Adding new supplier - Source: author**

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.SqlClient;
using System.Data;

namespace final_dental_manager.PL
{
    class suppliers
    {
        public void new_supplier(int supp_id, string supp_name, string supp_address,
string supp_web, string supp_email, string phone, string person)

        {
            Data_Access_Layer.DAL ob = new Data_Access_Layer.DAL();
            ob.open();
            SqlParameter[] p = new SqlParameter[7];
            p[0] = new SqlParameter("@id", SqlDbType.Int);
            p[0].Value = supp_id;
            p[1] = new SqlParameter("@name", SqlDbType.NVarChar, 50);
            p[1].Value = supp_name;
            p[2] = new SqlParameter("@address", SqlDbType.NVarChar, 50);
            p[2].Value = supp_address;
            p[3] = new SqlParameter("@website", SqlDbType.NVarChar, 50);
            p[3].Value = supp_web;
            p[4] = new SqlParameter("@email", SqlDbType.NVarChar, 50);
            p[4].Value = supp_email;
            p[5] = new SqlParameter("@phone", SqlDbType.NVarChar, 50);
            p[5].Value = phone;
            p[6] = new SqlParameter("@person", SqlDbType.NVarChar, 50);
            p[6].Value = person;

            ob.RUA("new_supp", p);
            ob.close();
        }
    }
}

```

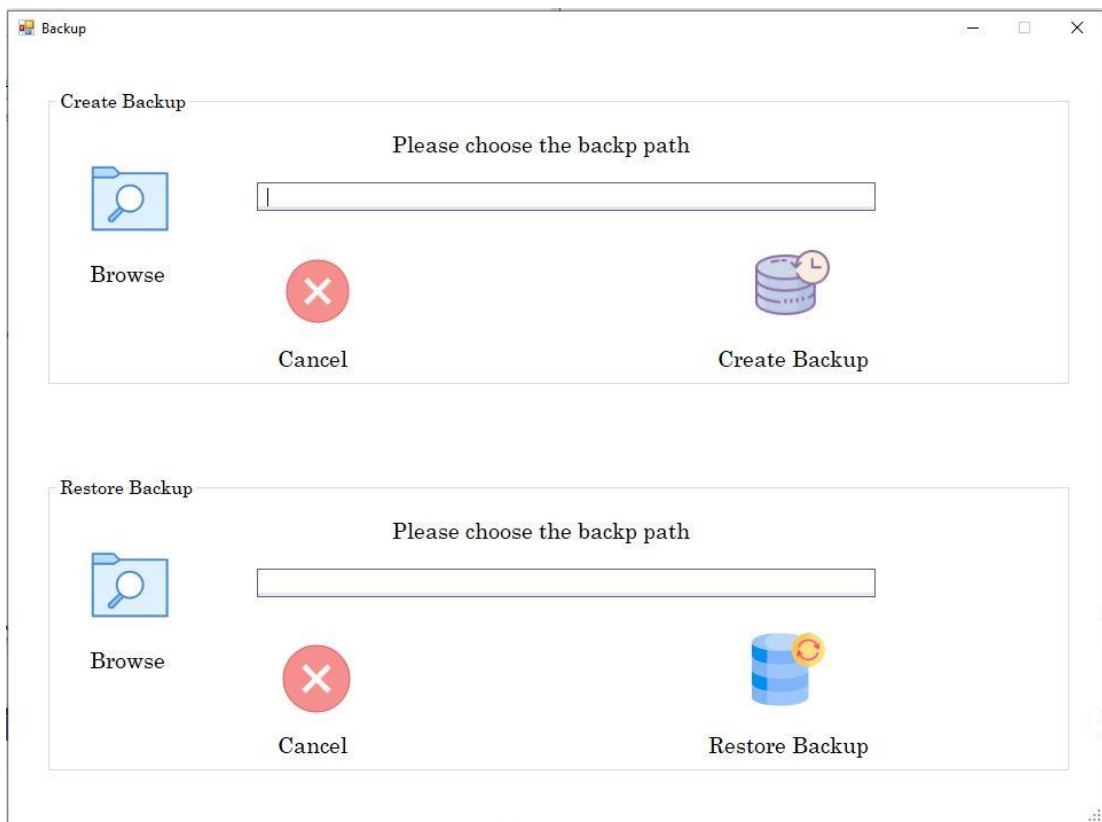
### 3.3.9 Equipment section

Equipment is one of the most important aspects to be automated in the dental clinic, clinic's equipment differ from sizes, uses, prices, suppliers and many other details which usually are being printed on papers and saved in files, which makes searching process manual and slow in order to find equipment's files and find the needed info, moreover, the periodical maintenance might not happen on time due to the same issue of not following the dates specified in the equipment's files.

For those reasons, equipment's section stores and facilitates all the clinic's equipment with their details and important dates as well as provides the option for reminding the service for the coming service check-up.

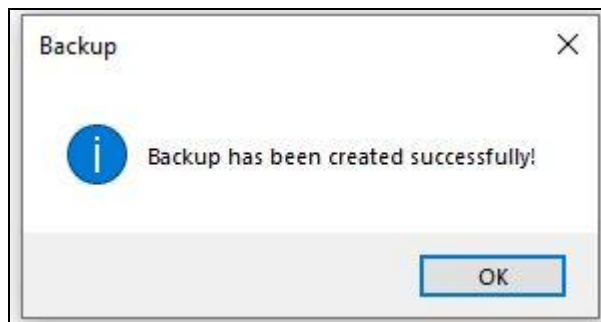
### 3.3.10 Backup

To avoid data lose for any reason, the user will have a section for backup, by clicking backup, the user will chose the path for creatin a full backup of his data, the application will create a .bak backup file that could be imported in the settings section. Pictures (22,)



Picture 22 - Backup form - Source: author

After creating the backup successfully, a confirmation message will confirm that to the user



Picture 23 - Successful backup - Source: author

```
private void btn_creat_backup_Click(object sender, EventArgs e)
{
    string filename = txt_path.Text + "\\Dental " +
DateTime.Now.ToShortDateString().Replace('/', '.') + "-" +
DateTime.Now.ToShortTimeString().Replace(':', '.');
    string query = "Backup Database Dental to Disk = '" + filename +
".bak'";
    cmd = new SqlCommand(query, cn);
    cn.Open();
    cmd.ExecuteNonQuery();
    cn.Close();
    MessageBox.Show("Backup has been created successfully!", "Backup",
MessageBoxButtons.OK, MessageBoxIcon.Information);
}
```

```

private void btn_import_backup_Click_1(object sender, EventArgs e)
{
    string query = "ALTER DATABASE Dental SET OFFLINE WITH ROLLBACK
IMMEDIATE; Restore Database Dental From Disk  =' + txt_res_path.Text + "' WITH
REPLACE";

    cmd = new SqlCommand(query, cn);
    cn.Open();
    cmd.ExecuteNonQuery();
    cn.Close();
    MessageBox.Show("Backup has been Restorted successfully!", "Backup",
MessageBoxButtons.OK, MessageBoxIcon.Information);
}

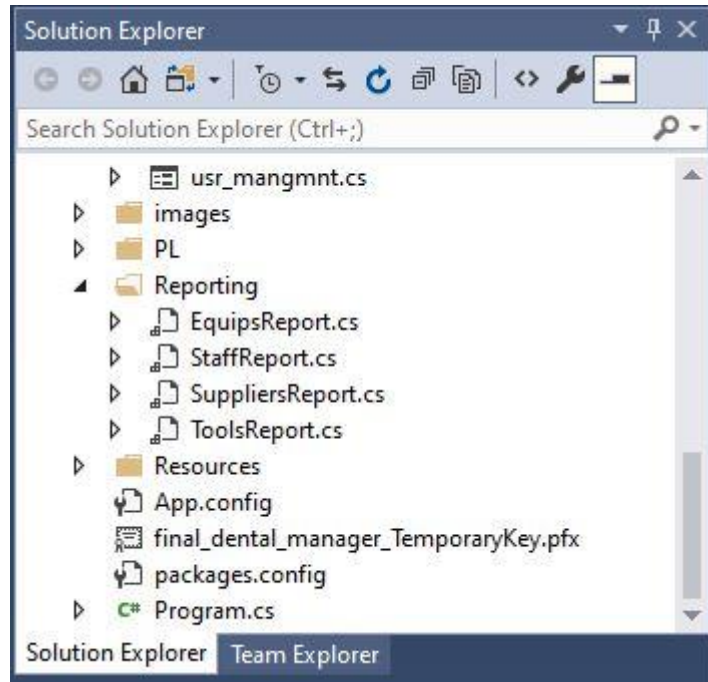
```

### 3.3.11 Reporting

Reporting is used in the 4 main sections in the application Tools, Suppliers, Staff and Equipment sections by clicking on print report button, to allow users to print out the list of available items in the selected section.

Reporting is done using DevExpress reporting, in the reporting layer 4 reports where created figure (24) in order to configure them with the database for the fields and tables and connect them with the sections and buttons later on.





**Picture 24 - Reporting layers - source: author**

Reports can be configured by clicking on each one, and adjusting the report's format from the bottom of the page and clicking designer tab, figure (20) where the report's header, name, logo and fields could be adjusted with more options like report's color and fields size.



Picture 25 - Report Designer - Source: author

```

using DevExpress.XtraReports.UI;

private void btn_print_report_Click(object sender, EventArgs e)
{
    Reporting.SuppliersReport ob = new Reporting.SuppliersReport();
    ob.ShowPreview();
}

```

### 3.3.12 Exporting

Another available option is to export the list of available items in a specific section into Excel file, to do so, Microsoft office library needs to be installed and defined within the used libraries in order to recognize its components, the code below shows the process of exporting the content of the DataGridView view into Excel file

```
using Microsoft.Office.Interop.Excel;

private void btn_exp_excel_Click(object sender, EventArgs e)
{
    Microsoft.Office.Interop.Excel.Application excel = new
Microsoft.Office.Interop.Excel.Application();
    Workbook wb = excel.Workbooks.Add(XLSheetType.xlWorksheet);
    Worksheet ws = (Worksheet)excel.ActiveSheet;
    excel.Visible = true;

    for (int i = 1; i < suppliers_grid.Columns.Count + 1; i++)
    {
        ws.Cells[1, i] = suppliers_grid.Columns[i - 1].HeaderText;
    }

    for (int i = 0; i < suppliers_grid.Columns.Count; i++)
    {
        for (int j = 0; j < suppliers_grid.Columns.Count; j++)
        {
            ws.Cells[i + 2, j + 1] =
suppliers_grid.Rows[i].Cells[j].Value.ToString();
        }
    }
}
```

### **3.4 Application testing**

Smaller projects do not need to spend as much time for testing as large projects, where several programmers work. However, it should not be completely neglected. In this application it was not neglected even during the development of this application. The application was continuously tested during the implementation of individual parts. The testing of the final version then took place on client computer in a dental clinic in Prague with real data. Testing with real data is much more efficient than using randomly generated monotonous data for testing. Errors detected during testing were subsequently corrected.

### **3.5 Application deployment**

For application development, ClickOnce technology was finally chosen. The main reason behind that is firstly the simplicity of the deployment itself, and the ability to further update the application, and for more other reasons and factors that were previously mentioned in the literature review above.

Since the program uses DevExpress components for reporting, it is necessary to set all DevExpress libraries used to be part of the installation package in order to avoid any issues or errors as these components will most likely not be installed on client computers. The application is based on the .NET Framework version 4.6.1 and it is therefore necessary to make sure that this version of the framework is installed on the client computer. Thanks to ClickOnce technology, it is possible to set the component were used in the project, and when the program is installed, they will be automatically downloaded and installed if necessary.

To develop the application itself, from solution explorer right-click on the project to display a menu in which the Publish option is selected. The Publish Wizard will then appear. The first step is to set the disk location where the application should be deployed. In the next step, the option to install from CD or DVD was selected. Since updated versions of the application are planned to take place, the location in which the application should check for updates needs to be specified so that each time you run the application, it will check for new versions. If it finds a new version, it informs the user via a dialog box. The new version will then be installed if the user agrees to the installation.

## **5 Conclusion**

The main objective of this diploma thesis is to implement a desktop application for dental clinic management, the theoretical part of the thesis introduced technologies and tools that were subsequently used in the development of application. First, the C# language and WinForms framework were described. Subsequently, an overview about object-oriented programming (OOP) with its fundamentals took place.

After that, visual studio and its components were also described and followed by description about ClickOnce, then database & MS SQL server were described.

And the last topics of the literature review were about the Bunifu UI framework which was used for designing the forms, DevExpress that was used for reporting, and last one was about Vonage API that was used for sending SMSs.

The practical part first analyses the requirements for the application features. Based on this analysis, the database was designed by defining all the needed tables with all their attributes, after designing the database, designing the forms took place using Bunifu Framework, and then the functions of the application were programmed.

The application was tested in a dental clinic and the feedback was collected, gradually, all phases of software development were followed, such as application analysis and design, implementation, testing and deployment.

## 6 References

1. *.NET Reporting Tools - Core, Blazor, WinForms, MVC / DevExpress* . Available at: <https://www.devexpress.com/subscriptions/reporting/> (Accessed: 2 March 2021).
2. *(6) Object-Oriented Programming* . Available at: [https://www.researchgate.net/publication/342592659\\_Object-Oriented\\_Programming](https://www.researchgate.net/publication/342592659_Object-Oriented_Programming) (Accessed: 24 March 2021).
3. *About Us - Bunifu Framework | Empowering software developers craft great user experiences in less time. Productivity tools for C# & VB.NET UX/UI design* . Available at: <https://bunifuframework.com/about-us/> (Accessed: 28 February 2021).
4. Albahari, Joseph & Albahari, B. (2012) *In a Nutshell, Naturschutz und Landschaftsplanung*.
5. Budd, T. A. . *An Introduction to Object-Oriented Programming 3rd Ed.*
6. *Bunifu Framework - Visual Studio .NET UI tools* . Available at: <https://appsource.microsoft.com/en-us/product/web-apps/bunifutechnologiesltd.bunifudevtools?tab=Overview> (Accessed: 28 February 2021).
7. *Bunifu Framework UI tools - Craft stunning UI and data visualizations fast!* . Available at: <https://bunifuframework.com/> (Accessed: 24 March 2021).
8. Churcher, C. (2007) 'Beginning Database Design. (J. Gennick, Ed.)', *New York, United States of America: Springer-Verlag New York, Inc.*
9. *ClickOnce Deployment for Windows Forms Applications | Microsoft Docs* . Available at: [https://docs.microsoft.com/en-us/previous-versions/visualstudio/visual-studio-2008/wh45kb66\(v=vs.90\)?redirectedfrom=MSDN](https://docs.microsoft.com/en-us/previous-versions/visualstudio/visual-studio-2008/wh45kb66(v=vs.90)?redirectedfrom=MSDN) (Accessed: 24 March 2021).
10. *ClickOnce Reference - Visual Studio | Microsoft Docs* . Available at: <https://docs.microsoft.com/en-us/visualstudio/deployment/clickonce-reference?view=vs-2019> (Accessed: 24 March 2021).
11. *ClickOnce Security and Deployment - Visual Studio | Microsoft Docs* . Available at: <https://docs.microsoft.com/en-us/visualstudio/deployment/clickonce-security-and-deployment?view=vs-2019> (Accessed: 24 March 2021).
12. Co., S. N. & (2013) *Fundamentals of Computer Programming with C# (The Bulgarian C# Programming Book) by Svetlin Nakov & Co.* <http://www.introprogramming.info>.

13. *Create a Report from A to Z / Reporting / DevExpress Documentation* . Available at: <https://docs.devexpress.com/XtraReports/2440/get-started-with-devexpress-reporting/create-a-report-from-a-to-z> (Accessed: 24 March 2021).
14. Davis, S. R. (2010) *C# 2010 for dummies*. Available at: [www.dummies.com/cheatsheet/csharp2010aio](http://www.dummies.com/cheatsheet/csharp2010aio) (Accessed: 3 April 2020).
15. *Debug using the Just-In-Time Debugger - Visual Studio / Microsoft Docs* . Available at: <https://docs.microsoft.com/en-us/visualstudio/debugger/debug-using-the-just-in-time-debugger?view=vs-2019> (Accessed: 24 March 2021).
16. *Editions and supported features of SQL Server 2019 - SQL Server / Microsoft Docs* . Available at: <https://docs.microsoft.com/en-us/sql/sql-server/editions-and-components-of-sql-server-version-15?view=sql-server-ver15> (Accessed: 24 March 2021).
17. *First look at the debugger - Visual Studio / Microsoft Docs* . Available at: <https://docs.microsoft.com/en-us/visualstudio/debugger/debugger-feature-tour?view=vs-2019> (Accessed: 24 March 2021).
18. Griffiths, I. *et al.* (2002) *.NET WINDOWS FORMS IN A NUTSHELL*.
19. Guéhéneuc, Y.-G. (2013) *CSE3009: (Software Architecture and Design) Encapsulation, inheritance and types, polymorphism (overriding/overloading), abstraction and types*.
20. Herbert Schildt (2010) *C# 4.0: the complete reference*.
21. *Install Visual Studio / Microsoft Docs* . Available at: <https://docs.microsoft.com/en-us/visualstudio/install/install-visual-studio?view=vs-2019> (Accessed: 24 March 2021).
22. *Introduction to editing in the code editor - Visual Studio / Microsoft Docs* . Available at: <https://docs.microsoft.com/en-us/visualstudio/get-started/tutorial-editor?view=vs-2019> (Accessed: 24 March 2021).
23. *Introduction to projects and solutions - Visual Studio / Microsoft Docs* . Available at: <https://docs.microsoft.com/en-us/visualstudio/get-started/tutorial-projects-solutions?view=vs-2019> (Accessed: 24 March 2021).
24. Kendal, S. (1997) *Object Oriented Programming Using C++, Object Oriented Programming Using C++*. doi: 10.1007/978-1-349-14449-5.
25. *Microsoft SQL samples - SQL Server / Microsoft Docs* . Available at: <https://docs.microsoft.com/en-us/sql/samples/sql-samples-where-are?view=sql-server-ver15> (Accessed: 24 March 2021).
26. Miles, R. (2009) *C# Development*.
27. Nagel, C. *et al.* . *C # 4 and . NET 4, Security*.

28. *Overview of Visual Studio | Microsoft Docs* . Available at: <https://docs.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?view=vs-2019> (Accessed: 24 March 2021).
29. Ramakrishnan, R. and Gehrke, J. (2003) *Database Management System*.
30. *Reporting | Reporting | DevExpress Documentation* . Available at: <https://docs.devexpress.com/XtraReports/2162/reporting> (Accessed: 24 March 2021)
31. Robbins, R. J. (2018) ‘Database Fundamentals ( DBF510S )’, (March), pp. 1–4.
32. Ronald, J. *et al.* (2009) ‘Object- Oriented Programming: Polymorphism’.
33. Simpkins, S. and Simpkins, S. (2016) *SQL Server, Building a SharePoint 2016 Home Lab*. doi: 10.1007/978-1-4842-2170-9\_9.
34. *SMS API > Send and receive SMS with the SMS API | Vonage API Developer* . Available at: <https://developer.nexmo.com/messaging/sms/overview> (Accessed: 24 March 2021).
35. *SQL Server technical documentation - SQL Server | Microsoft Docs* . Available at: <https://docs.microsoft.com/en-us/sql/sql-server/?view=sql-server-ver15> (Accessed: 24 March 2021).
36. Sumathi, S. and Esakkirajan, S. (2007) *Fundamentals of relational database management systems*.
37. *Testing ClickOnce Applications - Overview | TestComplete Documentation* . Available at: <https://support.smartbear.com/testcomplete/docs/app-testing/desktop/clickonce/about.html> (Accessed: 24 March 2021).
38. *Visual Studio IDE, Code Editor, Azure DevOps, & App Center - Visual Studio* . Available at: <https://visualstudio.microsoft.com/> (Accessed: 24 March 2021).
39. *Visual Studio IDE documentation | Microsoft Docs* . Available at: <https://docs.microsoft.com/en-us/visualstudio/ide/?view=vs-2019> (Accessed: 24 March 2021).
40. *Vonage API Developer* (2021). Available at: <https://developer.nexmo.com/documentation> (Accessed: 24 March 2021).
41. *What’s new in SQL Server 2019 - SQL Server | Microsoft Docs* . Available at: <https://docs.microsoft.com/en-us/sql/sql-server/what-s-new-in-sql-server-ver15?view=sql-server-ver15> (Accessed: 24 March 2021).
42. *What is Windows Forms - Windows Forms .NET | Microsoft Docs* . Available at: <https://docs.microsoft.com/en-us/dotnet/desktop/winforms/overview/?view=netdesktop-5.0> (Accessed: 24 March 2021).



43. *Windows Forms Designer for .NET Core Released* / *.NET Blog* . Available at: <https://devblogs.microsoft.com/dotnet/windows-forms-designer-for-net-core-released/> (Accessed: 24 March 2021).
44. *Your First Application: Take Two* / *Getting Started with .NET Development Using C#* | *InformIT* (2013). Available at: <https://www.informit.com/articles/article.aspx?p=2048355&seqNum=4> (Accessed: 24 March 2021).