Programming in C# with SQL

Realization

Internschip

Ruben Boone 3APP02

Acadamic year 2021-2022

Campus Geel, Kleinhoefstraat 4, BE-2440 Geel

LID VAN ASSOCIATIE KU LEUVEN

THOMAS MORE

TABLE OF CONTENTS

1 INTRODUCTION

In this paper I am going to prove everything I have learned and created during my internship at the hospital in Geel. Now you might be thinking, why in a hospital? I have chosen for this because of 2 reasons. The first reason is because the hospital in Geel had an internship on offer in programming in C# and SQL. My personal preference lies in creating applications in C#, Java, ... This does not mean that I do not like working with web-based applications, because with ASP you also have the opportunity to work with this. The second reason was mainly because I wanted to take a look behind the scenes of a hospital. How does the IT in a hospital work, how much does IT have to do with the current state of affairs in a hospital? I would like to thank my internship mentor Mr. Stefan de Wilde and internship supervisor Hajar Ghaem Sigarchian for the great support and tips during my internship and the hospital for giving me the opportunity to do my internship with them.

2 COMAPNY STRCUTURE

2.1 3-TIER

The applications in the hospital are built using a "3-tier" structure. So there is a data layer, an application/business layer and a presentation layer.

2.1.1 Data-layer

The data layer contains all the SQL queries of the application. This is a static class in the application that can be called anywhere.

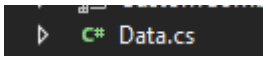


Figure 1: data class

2.1.2 Application- or Business-layer

In these classes the objects are created. These contain all the properties of an object and methods that are needed. Also present here are the various methods to populate the object, store it in the database, modify it in the database and delete it.

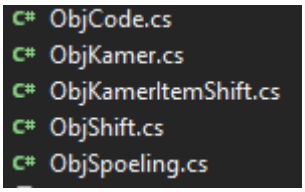


Figure 2: object classes

2.1.3 Presentation layer

In this layer all screens are provided, this is the visual layer where all buttons, selection lists, input fields are present. This layer contains very little self written code.

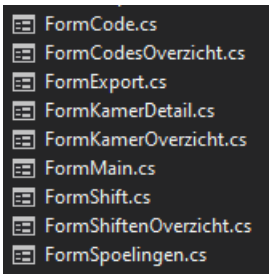


Figure 3: presentation classes

2.1.4 Helper classes

This is not really in the 3-tier system but it is used a lot. An example of a method in such a helper class is to fill a selection list.

Figure 4: extra helper classes

3 ASSIGNMENTS

3.1 ECONOMATLABELS

3.1.1 Description

This is an existing application that reads an Excel file to print labels to stick in the warehouse with the corresponding articles. On such a label is the article number, the quantity of a minimum order, a description, a reference number, a barcode and finally the date and time of the creation of the label. As soon as the barcode is scanned, the article is automatically ready to be ordered.

Figure 5: economat Label

3.1.2 Goal

The current printing process is not ideal. Once the data is read it is placed into a Word template using mail references, this Word document is then printed. The problem here is that when the printer finishes printing the Word process is not always completed correctly. If a lot of labels are printed it could be that a lot of Word processes are left open and that the computer is slow or even breaks down in the long run.

This gives us the following goals:

1. Look for a way to get to a label without making use of Word.
2. Look for a way to print the label without Word.
3. The barcode will need to be generated in a new way
4.

3.1.3 Research

After some searching and the already delivered information found by the mentor, I came up with the following solution.

Word can be replaced by the C# class "PrintDocument". This immediately gave me the possibility to use a PrintPage, such a printPage corresponds to the page where you normally write the text.

The Barcode can be generated using a Nuget package.

3.1.4 Realization

To keep it neat, we chose to create a static class called "Label" where everything related to the label can be placed.

In the new class, the method "PrintEconomate" is present. It takes a "KeyValuePair" object as a parameter so that we have all the necessary data that needs to be on the label available in the class. Thus we also immediately fill in the barcode and the date with time. Finally, we call the "PrintEtiket()" method to continue.

```
public static void PrintEconomaat(List<KeyValuePair<string, string>> fields)
{
    Zen.Barcode.Code128BarcodeDraw barcode = Zen.Barcode.BarcodeDrawFactory.Code128WithChecksum;

    barcodeCijfers = fields.FirstOrDefault(key => key.Key == "BARCODETEKST").Value.ToString();
    barcodeImg = barcode.Draw(barcodeCijfers, 32);
    artikelLabel = fields.FirstOrDefault(key => key.Key == "ARTIKEL").Value.ToString();
    aantalLabel = fields.FirstOrDefault(key => key.Key == "AANTAL").Value.ToString();
    beschrijving = fields.FirstOrDefault(key => key.Key == "ARTIKELOMSCHRIJVING").Value.ToString();
    referentieLabel = fields.FirstOrDefault(key => key.Key == "REFERENTIE").Value.ToString();
    systTime = DateTime.Now.ToString("MM/dd/yyyy  HH:mm");

    PrintEtiket();

}
```

Figure 6: checking keyvaluepair

This is where the "PrintDocument" is created, followed by the creation and padding of the "PrintPage". Because we are not really working in a document and are actually drawing on a sheet we have to draw everything with "Graphics". Thinking about possible future changes, several constant variables have been created so that later changes only have to be made in one place and it is not necessary to search for everything.

```
private static void CreateEtiket(object sender, PrintPageEventArgs printPageEventArgs)
{

    printPageEventArgs.Graphics.DrawString(artikelLabel, DefaultFontBold, textColor, Lijn1ArtikelX, Lijn1Y);
    printPageEventArgs.Graphics.DrawString(aantalLabel, DefaultFont, textColor, Lijn1AantalX, Lijn1Y);

    if (beschrijving.Length > maxCharachterOnLine)
    {
        printPageEventArgs.Graphics.DrawString(beschrijving.Substring(0, maxCharachterOnLine), DefaultFont, textColor, Lijn2BeschrijvingX, Lijn2BeschrijvingY);
        printPageEventArgs.Graphics.DrawString(beschrijving.Substring(maxCharachterOnLine), DefaultFont, textColor, Lijn2BeschrijvingX, Lijn2BeschrijvingY2);
    }
    else
    {
        printPageEventArgs.Graphics.DrawString(beschrijving, DefaultFont, textColor, Lijn2BeschrijvingX, Lijn2BeschrijvingY);
    }

    printPageEventArgs.Graphics.DrawString(referentieLabel, DefaultFont, textColor, Lijn3ReferentieX, Lijn3ReferentieY);

    printPageEventArgs.Graphics.DrawImage(barcodeImg, Lijn4BarcodeX, Lijn4BarcodeY);

    printPageEventArgs.Graphics.DrawString(barcodeCijfers, DefaultFont, textColor, Lijn5BarcodeTekstX, Lijn5BarcodeTekstY);
    printPageEventArgs.Graphics.DrawString(systTime, DefaultFontSmall, textColor, SystTimeX, SystTimeY);
}
```

Figure 7: constructing the label

```
private static readonly Font DefaultFont = new Font("Arial", 7);
private static readonly Font DefaultFontBold = new Font("Arial", 7, FontStyle.Bold);
private static readonly Font DefaultFontSmall = new Font("Arial", 4, FontStyle.Bold);

private static readonly SolidBrush textColor = new SolidBrush(Color.Black);

private static readonly int maxCharachterOnLine = 34;

private const int StartPunt = 110;

private const int Lijn1ArtikelX = StartPunt;
private const int Lijn1AantalX = StartPunt + 80;
private const int Lijn1Y = 0;

private const int Lijn2BeschrijvingX = StartPunt;
private const int Lijn2BeschrijvingY = Lijn1Y + 10;
private const int Lijn2BeschrijvingY2 = Lijn2BeschrijvingY + 10;

private const int Lijn3ReferentieX = StartPunt;
private const int Lijn3ReferentieY = Lijn2BeschrijvingY2 + 10;

private const int Lijn4BarcodeX = StartPunt + 35;
private const int Lijn4BarcodeY = Lijn3ReferentieY + 12;

private const int Lijn5BarcodeTekstX = StartPunt + 55;
private const int Lijn5BarcodeTekstY = Lijn4BarcodeY + 40;

private const int SystTimeX = StartPunt + 135;
private const int SystTimeY = Lijn4BarcodeY + 58;
```

Figure 8: setting values of the variables

Finally, a printer must be chosen where the document is to be printed. To be future-proof, we thought of doing this with the "App.config" so that the whole application does not have to be re-released and it is possible to work with a different printer in a different location. This also gives us the possibility to provide multiple printers and thus let the user choose which printer to use.

```
private static string GetSelectedPrinter(PrintDocument pDoc)
{
    string printNames = ConfigurationManager.AppSettings["Printers"].ToString();
    string[] printers = printNames.Split(';');

    string selectedPrinter = string.Empty;
    if (printers.Length == 1)
    {
        selectedPrinter = printNames; //Of printers[0] was ook ok
    }
    else
    {
        //Meerder printers in config wil zeggen dat we die negeren.
        //User dient printer zelf te kiezen
        //(kan andere zijn dan in de config staat).
        PrintDialog pd = new PrintDialog();
        pd.Document = pDoc;
        DialogResult dr = pd.ShowDialog();
        if (dr == DialogResult.OK)
        {
            selectedPrinter = pd.PrinterSettings.PrinterName;
        }
    }

    return selectedPrinter;
}
```

Figure 9: looking for the printer

3.2 URGENT-LABELS

3.2.1 Description

For the emergency department one has to print labels to stick on administered medication. This is mandatory by Belgian law.

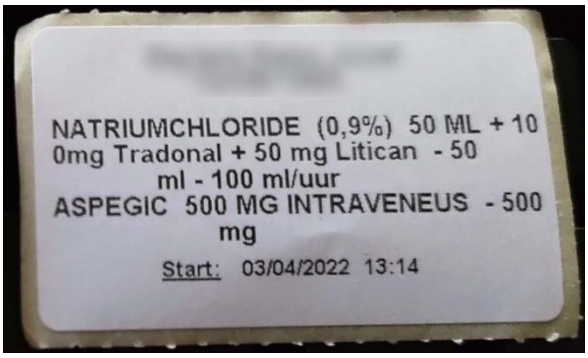


Figure 10: medication label

This is also an existing application but one that is structured slightly differently. Here the data comes through external software called ChipSoft - HiX. HiX provides, in short, information from electronic patient records. The selected patient ID and file number are then forwarded on request to an ASPX web page using arguments in the URL. There all additional information is retrieved using the ID. All the information on the label is placed in a text file and read by in-house made software. This data is then, just like the Economate labels, transferred via mail references to a Word template for subsequent printing.

3.2.2 Goal

As with the economy labels, the way of printing and the use of Word must be adjusted. In this command, no bar code should be provided on the label.

3.2.3 Research

Since this task is almost identical to the previous task, we can therefore reuse some code in this application. Thus, additional Research should not be done.

3.2.4 Realization

Again, a Label class is provided. So again the data is given in a "KeyValuePair" to the class, now the only real difference between these two assignments is just layout of the labels and the constants. So the printing is done just the same. The constants are here placed in the App.config because otherwise with every change in the label the web page has to be re-released and that is not ideal in an urgency department.

```
private static void CreateEtiketSpoed(object sender, PrintPageEventArgs printPageEventArgs)
{

    printPageEventArgs.Graphics.DrawString(CenterText(patientNaam), DefaultFont, textColor, NaamX, NaamY);
    printPageEventArgs.Graphics.DrawString(CenterText(patientGebDatum), DefaultFont, textColor, GeboorteDatumX, GeboorteDatumY);

    extraPosition = startHoogteMedicatie;
    UseExtraLine(printPageEventArgs, hoofdProduct);

    if (bijkomendProduct != " ")
    {
        UseExtraLine(printPageEventArgs, bijkomendProduct);
    }

    extraPosition -= margeSysTime;

    printPageEventArgs.Graphics.DrawString("Start: ", DefaultFontBoldUnderline, textColor, StartX, ProdY - extraPosition);
    printPageEventArgs.Graphics.DrawString(systTime, DefaultFontSmallBold, textColor, TimeX, ProdY - extraPosition);

}
```

Figure 11: constructing the label

3.3 SPOOL REGISTRATION

3.3.1 Description

In the hospital, there are several items that need to be rinsed. Such an item may be a shower, sink, toilet, ... These items can be rinsed by a different performer. There are currently 3 different types of performers. The nursing staff, the technical staff and of course the cleaning staff. Now we have been asked if an application can be made to improve this process. Currently everything is noted on paper and this takes a lot of time.

3.3.2 Goal

A few things need to be provided and prepared.

1. A configuration part in the application so that it can be entered which items are there, which rooms are there, which shifts are there and which shift should flush a particular item in which room.
2. A way for the performers to mark an item as completed.
    - Because the cleaning crew does not have access to a laptop, they will have to indicate this in another way. Especially for them we provide a web page where an item can be marked as completed.

3. A way to export all collected data to an Excel file. The data that can be exported can be filtered in different ways so you can generate specific tables

3.3.3 Realization

To begin with, a step-by-step plan was made. In this way we can work on the application in a structured way. To start in a logical way we start with the Configuration part.

3.3.3.1 step-by-step plan

Database

1. setting up the Tables
    a. Rooms table
    a. RoomsItems table
    a. RoomsItemsShiften table

Data-layer

1. Writing query's
    a. Room
        i.SELECT Room

i.SELECT Rooms
i.INSERT Room
i.UPDATE Room
i.DEL Room

a. RoomItem
i.SELECT RoomItems
i.INSERT RoomItem
i.DELETE RoomItem
i.(optioneel) UPDATE RoomItem

a. RoomItemShift
i.SELECT RoomItemShift
i.INSERT RoomItemShift
i.DELETE RoomItemShift
i.(optioneel) UPDATE RoomItemShift

a. general query's (dropdowns)
i.SELECT Items
i.SELECT Spool locationen
i.SELECT Shiften
i.SELECT SpoelUitvoerdersShiften (uitvoerderID vanuit AD)

Applicatie/Business-layer

1. Objecten aanmaken
   a. Spool location
      i.Properties
      - ID
      - Name

   a. Item
      i.Properties
      - ID
      - Name

   a. Room
      i.Properties
      - ID
      - Name
      - Spool location - Spool location
      - Items – List<Item>

      i.Methodes
      - Load - void
      - Save - void
      - Delete - void
      - IsValid – bool (out errormsg)

   a. RoomItemShift
      i.Properties
      - ID - int
      - RoomID - int
      - ItemID - int
      - ShiftID - int

Presentation layer

1. general
   a. Ribbon
      i.Config tab

1. Room overview (config)
   a. Insert button
   a. Filters
      i.Spool location
      i.Room
      i.Item
      i.Shift
      i.Uitvoerder

   a. Datagrid
      i.Roomname
      i.Delete button
      i.Detail button

1. Room Details (config)
    a. Room name
        i.Label
        i.Textbox
    a. Spool location
        i.Label
        i.Dropdown met spool locationen
    a. Datagrid
        i.Items (dropdown)
        i.Shift (dropdown)
    a. Add ItemShift Button
        i.Button
    a. Delete ItemShift Button
        i.Button

3.3.4 Configuration part

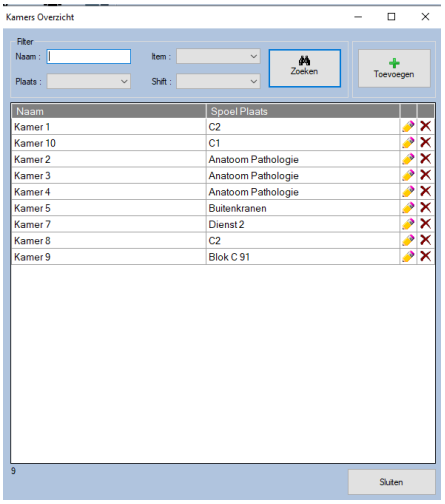


Figure 13: config overview screen of the rooms

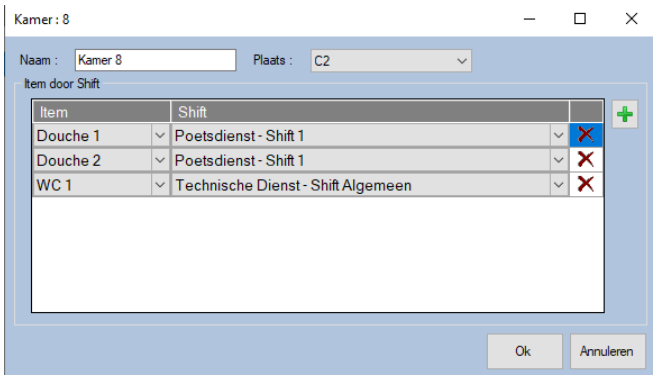


Figure 12: config detail screen of a room

3.3.4.1 Data layer

We start by getting the database in order; without correct tables, we can hardly proceed. Once the database is in order we can begin our 3-tier system. The first step is to write our database queries so we can retrieve, create, modify or delete our data.

Figure 14: tables in the database

3.3.4.2 Application layer

After the Data layer, the object classes are created also called the application or business layer. In the objects we keep track of the properties, e.g. the object "Room" has the properties "Name", "ID", ... Similarly, there is a "Save" method to save an instance of this object to the database or modify it, a "Delete" to delete an instance. An "IsValid" method is also provided to check if the properties meet the predefined requirements. If they do not, they may not be saved or modified.

```
public class ObjKamer
{
    14 references
    public int? Id { get; set; }
    7 references
    public string Naam { get; set; }
    6 references
    public int SpoelPlaatsId { get; set; }

    public List<ObjKamerItemShift> KamerItemShiften = new List<ObjKamerItemShift>();

    1 reference
    public ObjKamer(int? id, string naam, int spoelPlaatsId)
    {
        Id = id;
        Naam = naam;
        SpoelPlaatsId = spoelPlaatsId;
    }
```

Figure 17: properties of a room object

```
{
    bool succeeded = false;

    try
    {
        DataRow dr = Data.GetKamer((int)Id);
        Naam = dr["Naam"].ToString().Trim();
        SpoelPlaatsId = (int)dr["spoelplaatsid"];
    }
    catch (Exception error)
    {
        string errorMessage = error.Message;
    }

    return succeeded;
}
```

Figure 16: trying to get a room from the database

```
    bool valid = true;
    error = string.Empty;

    if (Naam == "" || Naam == null)
    {
        valid = false;
        error += "- Het veld \"Naam\" is niet ingevuld.\n";
    }

    if (SpoelPlaatsId <= 0)
    {
        valid = false;
        error += "- Er is geen spoelplaats geselecteerd\n";
    }

    string errorText = string.Empty;
    foreach (ObjKamerItemShift kamerItemShift in KamerItemShiften)
    {
        kamerItemShift.IsValid(out errorText);
    }
    error += errorText;

    return valid;
}
```

Figure 15: a function that checks the input fields

3.3.4.3 Presentation layer

Finally, we end with the presentation layer, these are our screens with our button pen, choice lists, checkboxes, ... These controls all contain their own events that hardly contain real code but rather refer to a method in another place. For example, there are several "Helper" classes that are privileged, filling a combo box.

A presentation layer also usually contains the same methods. For example, in the event when the window is loaded we have our "Init" methods.

These methods are executed only once, so the datagrid is prepared preferentially. This means that columns that should not be visible are made invisible. Another important method is the "InitControls" method, this method is used more often. Every time a button is pressed this method can be called. This method ensures that all controls in the window are set correctly. A small example of this is that the Room picker may not be used until a shift is selected.

```
public void InitForm()
{
    InitFormText();
    InitComboBoxes();

    InitFilterDefaults();
    kamerDataGridView.Columns.Clear();
}
```

Figure 19: first function that gets called when a screen is opened

```
private void InitComboBoxes()
{
    bool addEmptyRow = true;
    bool useSpoelUitvoerderID = false;

    ControlCreator.CreateSpoelPlaatsenComboBox(plaatsComboBox, addEmptyRow);
    ControlCreator.CreateSpoelItemsComboBox(itemComboBox, addEmptyRow);
    ControlCreator.CreateShiftenComboBox(shiftComboBox, addEmptyRow, useSpoelUitvoerderID);
}
```

Figure 18: filling in a dropdown

```
// Add new kamer
1 reference
private void insertButton_Click(object sender, EventArgs e)
{
        FormKamerDetail kamerDetail = new FormKamerDetail();
        kamerDetail.ShowDialog();

    FillDataGridViewOverzicht();
}

// Search with new filters
1 reference
private void zoekButton_Click(object sender, EventArgs e)
{
    FillDataGridViewOverzicht();
}

// Exit
1 reference
private void buttonSluiten_Click(object sender, EventArgs e)
{
    Close();
}
```

Figure 20: controling events

For example, each method has its own functionality so they can be reused anywhere and therefore only perform 1 thing.

```
4 references
private void FillDataGridViewOverzicht()
```

Figure 21: example function

These steps also repeat themselves in Data-in and Export. Many of the methods can be often adopted with minor modifications.

3.3.5 Data-in part

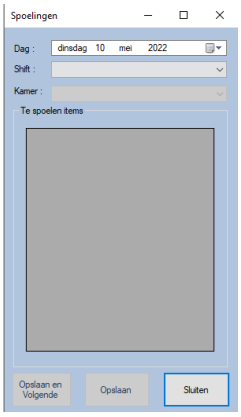


Figure 24: empty windows application screen

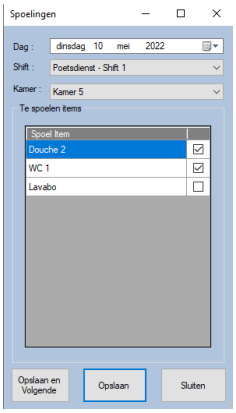


Figure 26: filled in screen of the windows application

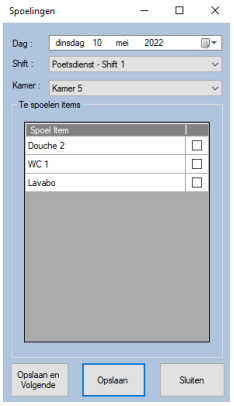


Figure 25: not yet filled in screen of the windows application

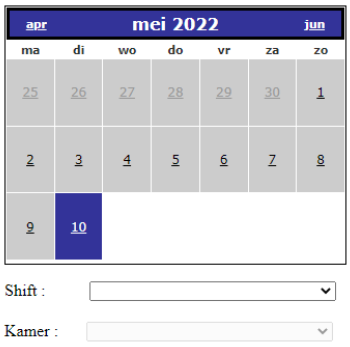


Figure 23: empty web screen

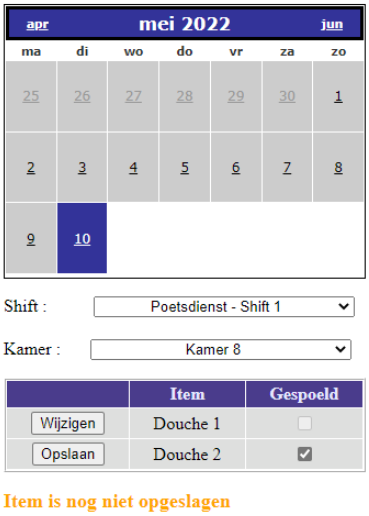


Figure 22: filled in web screen

The data-in part consists of 2 parts. The Windows form application and the ASP web page. Because not all functionalities work or are present in the web page, some things have to be done differently.

3.3.5.1 Data layer

The WHERE in the SQL query contains 3 pieces. The date of the flush, what shift the user is on and what Room the user is in to request the items that need to be rinsed.

If no flush has been done yet then the application shows the configured items. As soon as an item with the correct Room, shift and date combination is found it will be displayed.

```
public static DataTable GetSpoelingen(DateTime? date, int? shiftId, int? kamerId )
{
    SqlConnection sqlcon = new SqlConnection(databaseConnectionString);
    sqlcon.Open();

    string selectString = @"if(
                            (SELECT COUNT(*)
                            FROM Spoelingen
                            WHERE KamerID = @kamerid and ShiftID = @shiftId
                                        AND (DATEPART(yy, dbo.Spoelingen.Datum) = @year
                                        AND DATEPART(mm, dbo.Spoelingen.Datum) = @month
                                        AND DATEPART(dd, dbo.Spoelingen.Datum) = @day)) != 0)
                                SELECT dbo.Spoelingen.ID, dbo.Kamers.id as Kamer, dbo.SpoelItems.SpoelItem, dbo.SpoelItems.id as spoelitemid,
                                            dbo.SpoelUitvoerdersShiften.id as SpoelUitvoerdersShiftID, dbo.Spoelingen.Gespoeld
                                FROM dbo.Spoelingen INNER JOIN
                                            dbo.SpoelUitvoerdersShiften ON dbo.Spoelingen.ShiftID = dbo.SpoelUitvoerdersShiften.ID INNER JOIN
                                            dbo.SpoelItems ON dbo.Spoelingen.SpoelItemID = dbo.SpoelItems.ID INNER JOIN
                                            dbo.Kamers ON dbo.Spoelingen.KamerID = dbo.Kamers.ID
                                WHERE DATEPART(yy, dbo.Spoelingen.Datum) = @year
                                            AND DATEPART(mm, dbo.Spoelingen.Datum) = @month
                                            AND DATEPART(dd, dbo.Spoelingen.Datum) = @day
                                            AND KamerID = @kamerid
                                            AND ShiftID = @shiftid
                            else
                                SELECT NULL as ID, kis.kamerid as kamer, si.SpoelItem,kis.spoeluitvoerdersshiftid as SpoelUitvoerdersShiftID, kis.spoelitemid ,cast('False' as bit) as Gespoeld from dbo.KamerItemShift kis
                                JOIN SpoelItems si on si.ID = kis.SpoelItemID
                                WHERE kis.KamerID = @kamerid
                                AND kis.SpoelUitvoerdersShiftID = @shiftId";

    SqlCommand selectCommand = new SqlCommand(selectString, sqlcon);

    if (date != null)
    {
        DateTime dateTime = (DateTime)date;

        string year = dateTime.ToString("yyyy");
        string month = dateTime.ToString("MM");
        string day = dateTime.ToString("dd");

        selectCommand.Parameters.AddWithValue("year", year);
        selectCommand.Parameters.AddWithValue("month", month);
        selectCommand.Parameters.AddWithValue("day", day);
    }
    else
    {
        selectCommand.Parameters.AddWithValue("year", DBNull.Value);
        selectCommand.Parameters.AddWithValue("month", DBNull.Value);
        selectCommand.Parameters.AddWithValue("day", DBNull.Value);
    }
    selectCommand.Parameters.AddWithValue("kamerId", DataHelper.CNullToDbNull(kamerId));
    selectCommand.Parameters.AddWithValue("shiftId", DataHelper.CNullToDbNull(shiftId));
```

Figure 27: example of the data class

3.3.5.2 Application layer

The application layer has the same structure as the Rooms object.

```
public class ObjSpoeling
{
    5 references
    public int? ID { get; set; }
    4 references
    public int? KamerID { get; set; }
    4 references
    public int? ItemID { get; set; }
    4 references
    public int? ShiftID { get; set; }
    3 references
    public bool Gespoeld { get; set; }
    4 references
    public DateTime? Datum { get; set; }

    2 references
    public ObjSpoeling(int? id, DateTime? datum, int? shiftID, int? kamerID, int? itemID, bool gespoeld)
    {
        ID = id;
        Datum = datum;
        ShiftID = shiftID;
        KamerID = kamerID;
        ItemID = itemID;
        Gespoeld = gespoeld;
    }

    2 references
    public bool Save(out string errorTxt)
    {
        bool succeeded = false;
        errorTxt = string.Empty;
        int affected = 0;

        if (succeeded = IsValid(out errorTxt))
        {
            if (ID == null)
            {
                ID = Data.InsertSpoeling((DateTime)Datum, (int)ShiftID, (int)KamerID, (int)ItemID, Gespoeld);
                affected = 1;
            }
            else
            {
                affected = Data.UpdateSpoeling((int)ID, (DateTime)Datum, (int)ShiftID, (int)KamerID, (int)ItemID, Gespoeld);
            }
        }
        else
        {
            succeeded = false;
        }

        if (affected == 1)
        {
            succeeded = true;
        }
        else
```

Figure 28: example of a object

3.3.5.3 Presentation layer

The "InitControls" method is often used in this screen, namely when a shift is selected, a Room is selected, and in opening this screen.

```
3 references
private void InitControls()
{
    long? idShift = Helper.GetIDFromCboSelectedValue(shiftComboBox);
    if (idShift != null)
    {
        kamerComboBox.Enabled = true;
    }
    else
    {
        kamerComboBox.Enabled = false;
        kamerComboBox.SelectedIndex = -1;
    }

    long? idKamer = Helper.GetIDFromCboSelectedValue(kamerComboBox);
    if (idKamer != null)
    {
        teSpoelenItemsGridView.Enabled = true;

        saveAndNextButton.Enabled = true;
        okButton.Enabled = true;
    }
    else
    {
        teSpoelenItemsGridView.Enabled = false;
        teSpoelenItemsGridView.DataSource = null;

        saveAndNextButton.Enabled = false;
        okButton.Enabled = false;
        IsInitDataGridView = false;
    }
}
```

Figure 29: example of a presentation class

3.3.6 Data-out

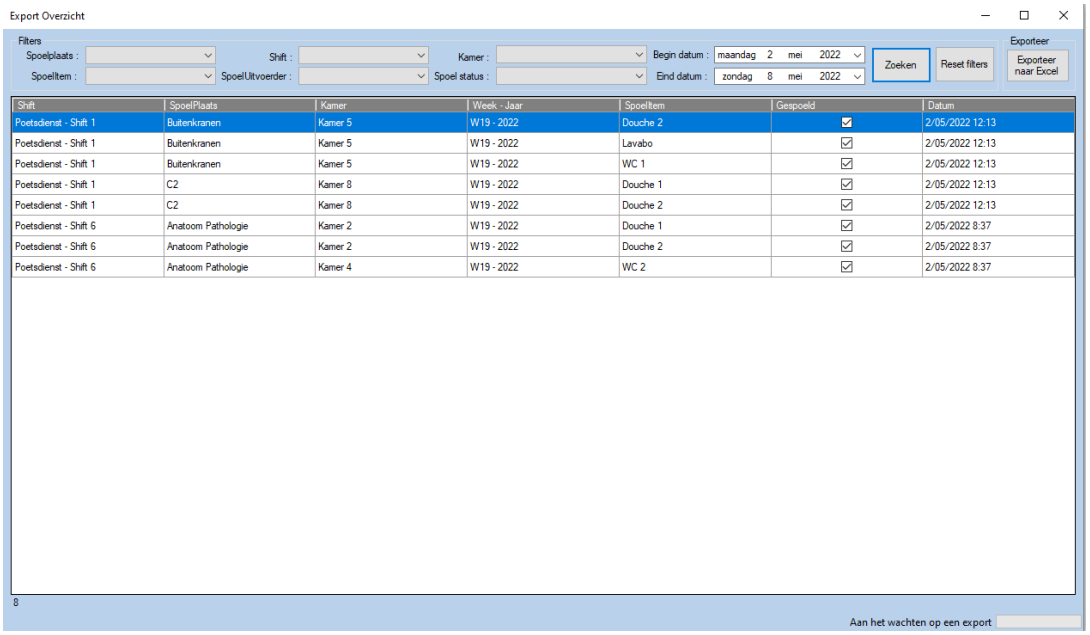


Figure 30: default screen without any filters of the export

When the screen is opened you can filter by different properties of a flush. For example: show me all showers that have not been rinsed between this particular time.

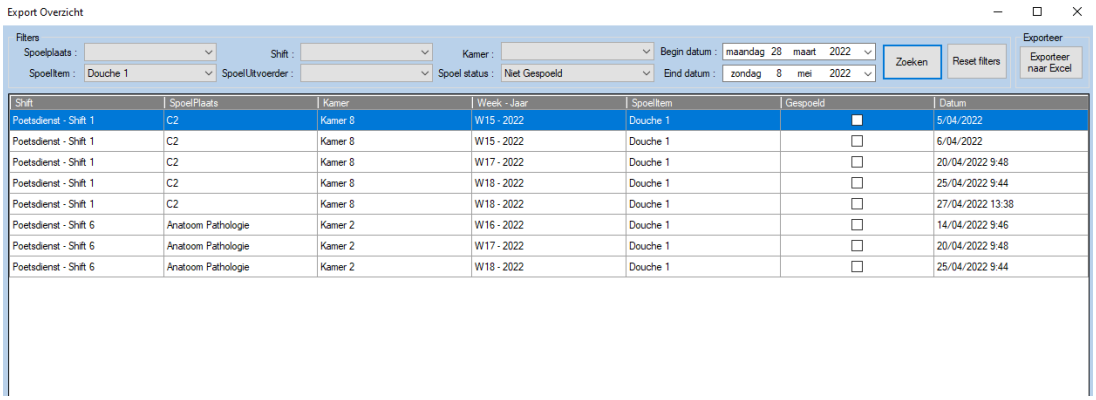


Figure 31: export screen with the "not yet rinsed" filter

After requesting the desired data, you can be asked to export the data to a Excel file. A loading bar shows the progress made in generating the Excel.

Figure 32: progressbar and label

A1 Shift

	A	B	C	D	E	F	G	H
1	Shift	SpoelPlaats	Kamer	Week - Jaar	SpoelItem	Gespoeld	Datum	
2	Poetsdienst - Shift 1	C2	Kamer 8	W15 - 2022	Douche 1	False	5/04/2022 0:00:00	
3	Poetsdienst - Shift 1	C2	Kamer 8	W15 - 2022	Douche 1	False	6/04/2022 0:00:00	
4	Poetsdienst - Shift 1	C2	Kamer 8	W16 - 2022	Douche 1	True	14/04/2022 9:46:15	
5	Poetsdienst - Shift 1	C2	Kamer 8	W16 - 2022	Douche 1	True	15/04/2022 9:21:16	
6	Poetsdienst - Shift 1	C2	Kamer 8	W17 - 2022	Douche 1	True	19/04/2022 11:04:18	
7	Poetsdienst - Shift 1	C2	Kamer 8	W17 - 2022	Douche 1	False	20/04/2022 9:48:59	
8	Poetsdienst - Shift 1	C2	Kamer 8	W17 - 2022	Douche 1	True	21/04/2022 0:00:00	
9	Poetsdienst - Shift 1	C2	Kamer 8	W18 - 2022	Douche 1	False	25/04/2022 9:44:17	
10	Poetsdienst - Shift 1	C2	Kamer 8	W18 - 2022	Douche 1	False	27/04/2022 13:38:21	
11	Poetsdienst - Shift 1	C2	Kamer 8	W18 - 2022	Douche 1	True	29/04/2022 0:00:00	
12	Poetsdienst - Shift 1	C2	Kamer 8	W19 - 2022	Douche 1	True	2/05/2022 12:13:11	
13	Poetsdienst - Shift 6	Anatoom Pathologie	Kamer 2	W16 - 2022	Douche 1	True	13/04/2022 7:58:01	
14	Poetsdienst - Shift 6	Anatoom Pathologie	Kamer 2	W16 - 2022	Douche 1	False	14/04/2022 9:46:15	
15	Poetsdienst - Shift 6	Anatoom Pathologie	Kamer 2	W16 - 2022	Douche 1	True	15/04/2022 9:21:16	
16	Poetsdienst - Shift 6	Anatoom Pathologie	Kamer 2	W17 - 2022	Douche 1	False	20/04/2022 9:48:59	
17	Poetsdienst - Shift 6	Anatoom Pathologie	Kamer 2	W18 - 2022	Douche 1	False	25/04/2022 9:44:17	
18	Poetsdienst - Shift 6	Anatoom Pathologie	Kamer 2	W18 - 2022	Douche 1	True	27/04/2022 13:50:09	
19	Poetsdienst - Shift 6	Anatoom Pathologie	Kamer 2	W19 - 2022	Douche 1	True	2/05/2022 8:37:21	
20								
21								
22								

Figure 33: result in the excel

3.3.6.1 Data layer

The data is retrieved with the following query. The columns are formatted by joining multiple tables.

```
public static DataTable GetOutput(int? spoelPlaatsID, int? spoelItemID, int? spoelUitvoerdersShiftID, int? spoelUitvoerdersID, int? kamerID, int? spoelStatus, DateTime? beginDate, DateTime? eindDate, bool isCount)
{
    SqlConnection sqlcon = new SqlConnection(databaseConnectionString);
    sqlcon.Open();

    string selectString = string.Empty;
    string orderString = string.Empty;

    if (isCount)
    {
        selectString += @"SELECT COUNT (*)";
    }
    else
    {
        selectString += @"SELECT TOP (100) PERCENT dbo.Spoelingen.ID, dbo.SpoelUitvoerders.SpoelUitvoerder + ' - ' + dbo.SpoelUitvoerdersShiften.SpoelUitvoerdersShift AS Shift, dbo.SpoelPlaatsen.SpoelPlaats, dbo.Kamers.Naam AS Kamer, 'W' + CONVERT(VARCHAR,
            DATEPART(ww, dbo.Spoelingen.Datum)) + ' - ' + CONVERT(VARCHAR, DATEPART(yyyy, dbo.Spoelingen.Datum)) AS 'Week - Jaar', dbo.SpoelItems.SpoelItem, dbo.Spoelingen.Gespoeld, dbo.Spoelingen.Datum";
        orderString = @"ORDER BY dbo.SpoelUitvoerders.SpoelUitvoerder, dbo.SpoelUitvoerdersShiften.SpoelUitvoerdersShift, Kamer, dbo.Spoelingen.Datum, dbo.SpoelItems.SpoelItem";
    }

    selectString += @" FROM     dbo.SpoelUitvoerders INNER JOIN
                      dbo.SpoelUitvoerdersShiften ON dbo.SpoelUitvoerders.ID = dbo.SpoelUitvoerdersShiften.SpoelUitvoerderID INNER JOIN
                      dbo.Spoelingen ON dbo.SpoelUitvoerdersShiften.ID = dbo.Spoelingen.ShiftID INNER JOIN
                      dbo.SpoelItems ON dbo.Spoelingen.SpoelItemID = dbo.SpoelItems.ID INNER JOIN
                      dbo.Kamers ON dbo.Spoelingen.KamerID = dbo.Kamers.ID INNER JOIN
                      dbo.SpoelPlaatsen ON dbo.Kamers.SpoelplaatsID = dbo.SpoelPlaatsen.ID";

    string whereString = BuildWhereOutput(spoelPlaatsID, spoelItemID, spoelUitvoerdersShiftID, spoelUitvoerdersID, kamerID, spoelStatus, beginDate, eindDate);

    SqlCommand sqlCommand = new SqlCommand($"{selectString} {whereString} {orderString}", sqlcon);
    SqlDataAdapter da = new SqlDataAdapter(sqlCommand);
    DataTable dt = new DataTable();
    da.Fill(dt);

    sqlcon.Close();

    return dt;
}
```

Figure 34: example of the data class

3.3.6.2 Application layer

For the export, no object is created so there is no Application layer present for this part.

3.3.6.3 Presentation layer

As with the configuration and data-in section, the structure of the presentation layer is the same here. The only noticeable thing in this piece is that we use an enumeration for the status of the items. Since there is an informational label in the main screen of the application that can be double clicked and directly filtered for items not rinsed from the previous week, it should also be directly populated in the filters.

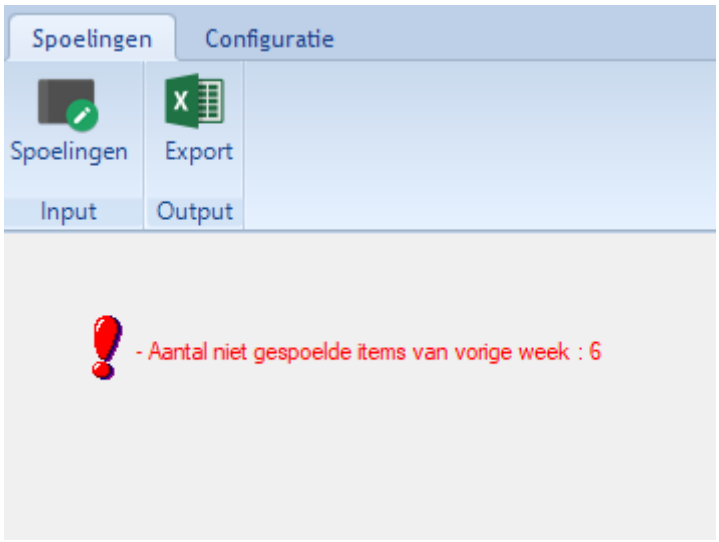


Figure 35: alert on the mainscreen of the application

```
2 references
public FormExport(Enums.SpoelStatuses? spoelStatusFilter, bool isFillMode = false)
{
    InitializeComponent();
    IsFillMode = isFillMode;
    SpoelStatusFilter = spoelStatusFilter;
}
```

Figure 36: setting the default values when opening a export screen

3.3.7 Excel creation

The Excel file is created in a static class. The Excel class also has its own helper class, which allows, for example, the creation of a method to thicken the border of a certain range in the file. Several methods have been created to change the background color, add text decoration and create different types of borders.

```
2 references
public static void ColorBackgroundCell(int row, int col, Color color, Excel.Worksheet worksheet)
{
    worksheet.Cells[row, col].Interior.Color = color;
}

// Make Text in cell Bold
1 reference
public static void AddBoldStyleToCell(int row, int col, Excel.Worksheet worksheet)
{
    worksheet.Cells[row, col].Font.Bold = true;
}

// Make Text in cell italic
0 references
public static void AddItalicStyleToCell(int row, int col, Excel.Worksheet worksheet)
{
    worksheet.Cells[row, col].Font.Italic = true;
}

1 reference
public static void ColorBooleanCell(int row, int col, string data ,Excel.Worksheet worksheet)
{
    switch (data)
    {
        case "False":
            ExcelHelper.ColorBackgroundCell(row, col, Color.Salmon, worksheet);
            break;
        case "True":
            ExcelHelper.ColorBackgroundCell(row, col, Color.PaleGreen, worksheet);
            break;
        default:
            break;
    }
}
```

Figure 37: example of the excel-helper class

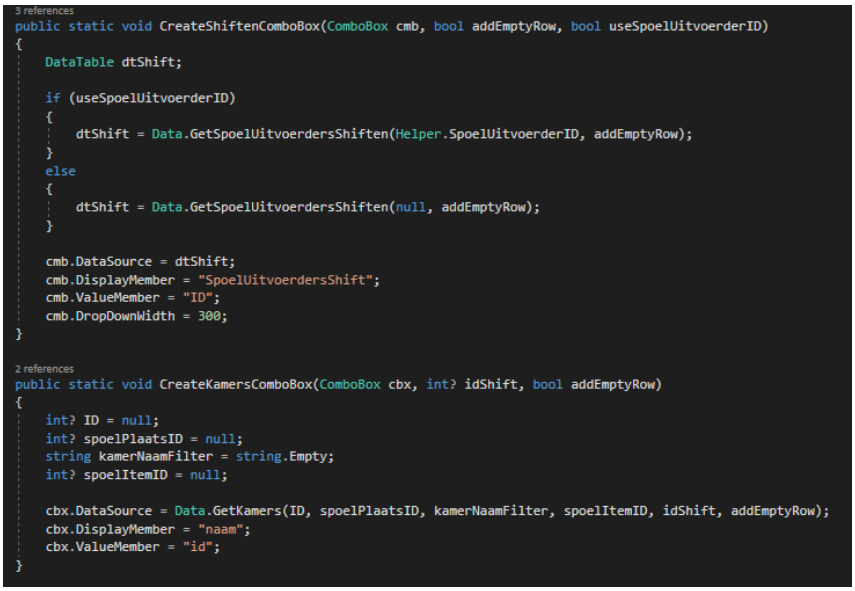
```
public static void DrawThickBorder(int startRow, int startCol, int endRow, int endCol, Excel.Worksheet sheet)
{
    Excel.Range ranges = sheet.Range[sheet.Cells[startRow, startCol + 1], sheet.Cells[endRow, endCol]].Cells;

    ranges.BorderAround2(Excel.XlLineStyle.xlContinuous, Excel.XlBorderWeight.xlThick, Excel.XlColorIndex.xlColorIndexAutomatic, Color.Black);
}
```

Figure 38: example to draw a thick border around a range of cells

3.3.8 Helper classes

As already mentioned, several helper classes have been created. For example, there is a helper to populate drop-down lists.



```
3 references
public static void CreateShiftenComboBox(ComboBox cmb, bool addEmptyRow, bool useSpoelUitvoerderID)
{
    DataTable dtShift;

    if (useSpoelUitvoerderID)
    {
        dtShift = Data.GetSpoelUitvoerdersShiften(Helper.SpoelUitvoerderID, addEmptyRow);
    }
    else
    {
        dtShift = Data.GetSpoelUitvoerdersShiften(null, addEmptyRow);
    }

    cmb.DataSource = dtShift;
    cmb.DisplayMember = "SpoelUitvoerdersShift";
    cmb.ValueMember = "ID";
    cmb.DropDownWidth = 300;
}

2 references
public static void CreateKamersComboBox(ComboBox cbx, int? idShift, bool addEmptyRow)
{
    int? ID = null;
    int? spoelPlaatsID = null;
    string kamerNaamFilter = string.Empty;
    int? spoelItemID = null;

    cbx.DataSource = Data.GetKamers(ID, spoelPlaatsID, kamerNaamFilter, spoelItemID, idShift, addEmptyRow);
    cbx.DisplayMember = "naam";
    cbx.ValueMember = "id";
}
```

Figure 39: Helper class to fill a dropdown

Thus, there are also certain functions that only users in a certain Active Directory group are allowed to perform. Therefore, there is a method that looks at which AD group the user is from and what things this user may and may not see and/or perform.

```
2 references
public static void  SelectADMembership()
{
    foreach (int group in Enum.GetValues(typeof(Enums.AdGroups)))
    {
        string name = Enum.GetName(typeof(Enums.AdGroups), group);
        bool isMember = IsMemberOf(Environment.UserDomainName, Environment.UserName, name);

        if (isMember)
        {
            if (name == "G_SpoelRegistratie_PowerUsers")
            {
                isPowerUser = true;
            }
            else
            {
                SpoelUitvoerderID = group;
                break;
            }
        }
    }
}

1 reference
private static bool IsMemberOf(string domain, string userName, string groupName)
{
    PrincipalContext ctx = new PrincipalContext(ContextType.Domain, domain);

    // find a user
    UserPrincipal user = UserPrincipal.FindByIdentity(ctx, userName);

    // find the group in question
    GroupPrincipal group = GroupPrincipal.FindByIdentity(ctx, groupName);

    // get members
    object foundUsers = null;
    if (user != null && group != null)
    {
        foundUsers = group.GetMembers(true).Where(p => p.SamAccountName.Equals(user.SamAccountName, StringComparison.InvariantCultureIgnoreCase)).FirstOrDefault();
    }

    // check if user is member of that group
    if (foundUsers != null)
    {
        return true;
    }
    else
    {
        return false;
    }
}
```

Figure 40: example to determine the AD-group

For example, the application uses a datagrid a lot. To give the columns a nice header, we have a function that makes the name nice for the user.

```
6 references
public static string BeautifyTableName(string DatabaseTabel, bool enkelvoud)
{
    string newName = DatabaseTabel[0].ToString().ToUpper() + DatabaseTabel.Substring(1);

    if (enkelvoud)
    {
        if (newName.Substring(newName.Length - 2).Contains("en"))
        {
            newName = newName.Substring(0, newName.Length - 2);
        }
        else if (newName.Substring(newName.Length - 1).Contains("s"))
        {
            newName = newName.Substring(0, newName.Length - 1);
        }
    }

    return newName;
}
```

Figure 41: example to make the column name nicer to read

Thus, there are many other small methods that perform simple tasks without having to write duplicate code.

3.3.9 Extra label

Near the end of my internship, I got the question if I could add a label like the “not rinsed items” label. With this new label they wanted to see how many rooms got skipped of the past week. Therefore the following query got called in existence:

```
SELECT        HuidigeKamersVorigeWeekJaar.KamerID,
HuidigeKamersVorigeWeekJaar.SpoelDag
FROM
(SELECT        TOP (100) PERCENT HuidigeKamers.KamerID, VorigeWeekJaar.SpoelDag
FROM            (SELECT        ID AS KamerID
FROM            Kamers) as HuidigeKamers CROSS JOIN
                         (SELECT        SpoelDag
FROM            (SELECT        CONVERT(INT, CONVERT(VARCHAR, DATEPART(ww,
CONVERT(datetime, '20220509'))) + CONVERT(VARCHAR, DATEPART(yyyy, CONVERT(datetime,
'20220509')))) AS SpoelDag) AS dt) as VorigeWeekJaar
ORDER BY VorigeWeekJaar.SpoelDag) as HuidigeKamersVorigeWeekJaar
 LEFT OUTER JOIN
                        (SELECT DISTINCT TOP (100) PERCENT KamerID,
CONVERT(VARCHAR(10), CONVERT(VARCHAR(10), DATEPART(ww, CONVERT(datetime, Datum))) +
CONVERT(VARCHAR, DATEPART(yyyy, CONVERT(datetime, Datum)))) AS Spoeldag
FROM            Spoelingen
WHERE        (CONVERT(INT, CONVERT(VARCHAR, DATEPART(ww, CONVERT(datetime, Datum))) +
CONVERT(VARCHAR, DATEPART(yyyy, CONVERT(datetime, Datum)))) = CONVERT(INT,
CONVERT(VARCHAR, DATEPART(ww, CONVERT(datetime,
                         '20220509'))) + CONVERT(VARCHAR, DATEPART(yyyy,
CONVERT(datetime, '20220509')))))
ORDER BY Spoeldag) as SpoelingenOpWeekNr ON HuidigeKamersVorigeWeekJaar.KamerID =
SpoelingenOpWeekNr.KamerID AND HuidigeKamersVorigeWeekJaar.SpoelDag =
SpoelingenOpWeekNr.Spoeldag
WHERE        (SpoelingenOpWeekNr.KamerID IS NULL)
```

The query consists out of a few parts.

1. All the configured rooms
   ```
   SELECT        ID AS KamerID
   FROM            dbo.Kamers
   ```
2. The query to get the week number and year combination of the previous
   ```
   SELECT        SpoelDag
   FROM            (SELECT        CONVERT(INT, CONVERT(VARCHAR, DATEPART(ww,
   CONVERT(datetime, '20220509'))) + CONVERT(VARCHAR, DATEPART(yyyy, CONVERT(datetime,
   '20220509')))) AS SpoelDag) AS dt
   ```
3. All the rooms with the week number combination
   ```
   SELECT        TOP (100) PERCENT dbo.HuidigeKamers.KamerID, dbo.VorigeWeekJaar.SpoelDag
   FROM            dbo.HuidigeKamers CROSS JOIN
                            dbo.VorigeWeekJaar
   ORDER BY dbo.VorigeWeekJaar.SpoelDag
   ```
4. All rinses with the week number combination

```
SELECT DISTINCT TOP (100) PERCENT KamerID, CONVERT(VARCHAR(10), CONVERT(VARCHAR(10),
DATEPART(ww, CONVERT(datetime, Datum))) + CONVERT(VARCHAR, DATEPART(yyyy,
CONVERT(datetime, Datum)))) AS Spoeldag
FROM            dbo.Spoelingen
WHERE        (CONVERT(INT, CONVERT(VARCHAR, DATEPART(ww, CONVERT(datetime, Datum))) +
CONVERT(VARCHAR, DATEPART(yyyy, CONVERT(datetime, Datum)))) = CONVERT(INT,
CONVERT(VARCHAR, DATEPART(ww, CONVERT(datetime,
                         '20220509'))) + CONVERT(VARCHAR, DATEPART(yyyy,
CONVERT(datetime, '20220509'))))))
ORDER BY Spoeldag
```

If we put this all together we get all the rooms that need to be rinsed, if we then add a "WHERE" where the room is "NULL" we get all rinses that are skipped.

3.3.10 Deploying the software

This is not something I was allowed to do myself but was allowed to be present for. The web page is released through IIS and the Forms application is put on the portal that the hospital uses to distribute software.

3.4 Extra assingments

The main assignments were finished sooner than anticipated but that did not cause me to do nothing. I have been busy with small assignments that needed to be done but were not project worthy.

- QR-code generator
- Datatype change in a column in the database
- Importing a Excel file to create new intern accounts.
- Small analyses exercises

3.4.1 QR-code generator

The communication department likes to use QR codes. And in order not to have to use a website every time, it was asked if a QR-generator could not be made.

The possibility is provided to save the QR code or to place it directly on the clipboard.

There is also a check that if no "https://" is given that it is automatically added to the link.

Figure 42: screen of the QR-generator

3.4.2 Datatype change

In the database, a column was marked as a VARCHAR and it needed to be changed to an INT. This could not be done just like that because nothing had been changed in the existing application.

Therefore I had to find a structural way of working so that no step would be skipped and the application would still work perfectly after the change in the database. So I started in the Data layer, so I could easily see the references of the methods where the field got used and they could be changed step by step. To give the mentor a clear overview of what got changed, each change in the code has a comment line with my name and the date of the change. This way, it was easy to find the changes using the "Find" option in Visual Studio. As an extra, I created a document where all the changes are listed.

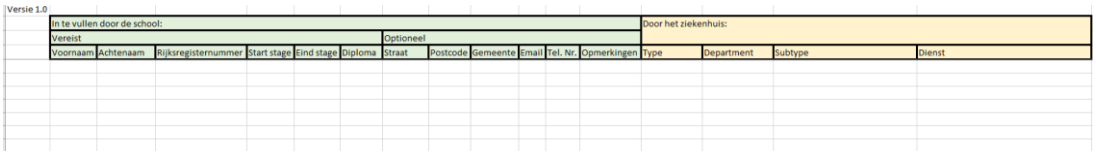
This document is called "DataType modification" and can be found on the portfolio.

3.4.3 Importing intern Excel

Because there are interns every year and this is not about one or two interns, it is envisioned that the school can fill out a Excel template and it can be put into an application so that the accounts that normally had to be created manually can be generated automatically.

My mentor sent me an email with all the necessary fields in the Excel and had me prepare the document. For example, there are fields that are required to be filled in, fields that are optional, and fields that are for the hospital staff itself. It was important that it was clear to both parties who must fill in what where. All the tricks in Excel were used to lock the header and lists of choices were made so that the correct values were

entered.



Versie 1.0																
	In te vullen door de school:												Door het ziekenhuis:			
	Vereist						Optioneel									
	Voornaam	Achtenaam	Rijksregisternummer	Start stage	Eind stage	Diploma	Straat	Postcode	Gemeente	Email	Tel. Nr.	Opmerkingen	Type	Department	Subtype	Dienst

Figure 43: header of the intern excel

For the hospital, the fields are actually already fixed, here the staff only has to choose from a selection list. These choice lists are laid out in a second sheet that has been made invisible to the schools that would get this document.

Type	Department	SubType		Dienst			
Student	Andere	ArtsSubtype	VerpleegkudigeSubType	VerpleegkundigeDienst	ArtsDienst	ParamediciDienst	AndereDienst
	Arts	Dokter Stagair	Student Zonder Medicatiebeheer	Adjunct Directie Verpleging	Dokters Algologie	Anapath	Accreditatie
	Paramedici	Dokter Assistent	Student Met Medicatiebeheer	Chirurgisch Daghospitaal	Dokters Anaesthesie	Dienst Zingeving	Apotheek
	Verpleegkundige			Diabetes Consultatie	Dokters Anathomopathologie	Dietisten	Archief
				Dialyse	Dokters Dermatologie	Endoscopie	Biotechnische Dienst
				Dienst 1	Dokters Endoscopie	Gipskamer	Boekhouding
				Dienst 10	Dokters Gynaecologie	Labo	Directie
				Dienst 2	Dokters Heelkunde	Medische Registratie	Directie Secretariaat
				Dienst 4	Dokters Inwendige	Nucleaire	Economaat
				Dienst 5	Dokters Klinisch Labo	Palliatief Support Team	Financieel Manager
				Dienst 6	Dokters MKG	Pijnkliniek	Financiele Dienst
				Dienst 7	Dokters Nefrologie	Poetsdienst	Personeelsdienst
				Dienst 8	Dokters Neurochirurgie	Poli Gynaecologie	Informatica
				Dienst 9	Dokters Neurologie	Poli Heelkunde - Orthopedie	Informatieveiligheid
				Dienst Intensieve	Dokters Nucleaire	Poli Inwendige	Keuken
				Dienst OK	Dokters Oftalmologie	Poli Neuro	Linnenkamer
				Dienst Pediatrie	Dokters Oncologie	Poli Onco	Magazijn
				Dienst SP	Dokters ORL	Poli ORL	Opname
				Dienst Spoed	Dokters Orthopedie	Poli Pediatrie	Preventieadviseur
				Dienst Vliegende	Dokters Pediatrie	Poli Urologie	Tarificatie - Facturatie
				Loopwacht	Dokters Radiologie	Psychologen	Technische Dienst
				Oncologisch Daghospitaal	Dokters Revalidatie	Radiologie	VTO
				Slaaplabo	Dokters Spoedgevallen	Dienst Revalidatie	Communicatie
				Dienst Materniteit	Dokters Stomatologie	Secretariaat Vliegende Ploeg	
				Geriatrisch Daghospitaal	Dokters Urologie	Sociale Dienst	
				Mobiele Equipe kritieke Diensten	Psychiatrie	Sterilisatie	
					Dokters Pneumologie	Tabakologen	
					Dokters Geriatrie	Vervoer	
					Dienst Gastro- enterologie	Wondzorg	
						Dienst Ergo	
						Poly Gastro	

Figure 44: list of all possible values in a cell

At the request of the mentor, a manual was created so that something can be adjusted later. So it is simple to undo my settings, adjust something and set it back.

The import will be done in an already existing application called “New Employees”. I added two extra classes called “ExcelImportHelper”, this is where my functions that help will go like “GetCellValue” which will give me the value of a specific cell.

```
10 references
public static string GetCellValue(int col, int row)
{
    var value = excelWorksheet.Cells[row, col].Value;
    if (value == null)
    {
        return null;
    }

    string result = value.ToString();
    if (!string.IsNullOrEmpty(result))
    {
        result = result.Trim();
    }
    return result;
}
```

Figure 45: helper function that return the value of a cell

The other class is the “ExcelImport” class. Here is the main flow of what needs to happen when a user pressed the import button.

The first step is checking if the person who filled in the Excel file found a way to break our security of the header. That’s why we still check if everything is in the right location.

```
1 reference
public static bool CheckHeaderCells(Excel.Worksheet sheet)
{

    bool succeeded = false;

    foreach (VerplichteKolommen col in Enum.GetValues(typeof(VerplichteKolommen)))
    {
        string cellValue = GetCellValue((int)col, rowHeader);

        cellValue = BeautifyCellValue(cellValue);

        if (cellValue != col.ToString())
        {
            return succeeded;
        }
    }

    foreach (OptioneleKolommen col in Enum.GetValues(typeof(OptioneleKolommen)))
    {
        string cellValue = GetCellValue((int)col, rowHeader);

        cellValue = BeautifyCellValue(cellValue);

        if (cellValue != col.ToString())
        {
            return succeeded;
        }
    }


    foreach (ZiekenhuisKolommen col in Enum.GetValues(typeof(ZiekenhuisKolommen)))
    {
        string cellValue = GetCellValue((int)col, rowHeader);

        cellValue = BeautifyCellValue(cellValue);

        if (cellValue != col.ToString())
        {
            return succeeded;
        }
    }

    succeeded = true;

    return succeeded;
}
```

Figure 46: function to check if the header is still correct

```
ExcelImportHelper.setLabel("...Excel nakijken", label);

if (ExcelImportHelper.CheckHeaderCells(ExcelImportHelper.excelWorksheet))
{
    int lastDataRow = ExcelImportHelper.SearchLastDataRow(ExcelImportHelper.excelWorksheet);
    int totalRows = lastDataRow - ExcelImportHelper.rowDataFirst;

    ExcelImportHelper.InitProgressBar(progressBar, totalRows);

    if (lastDataRow != 4)
    {
        for (int row = ExcelImportHelper.rowDataFirst; row <= lastDataRow; row++)
        {
            ExcelImportHelper.setLabel($"...Bezig aan rij {row - ExcelImportHelper.rowDataFirst} van de {totalRows}", label);

            string succeedCheck = ExcelImportHelper.GetCellValue(1, row);
            if (succeedCheck != "OK")
            {
```

Figure 47: changing the progressbar and label

Since the user does not get to see the Excel (because we open it invisible in the background), I added a progress bar and status label into the form. This way the user knows something is happening.

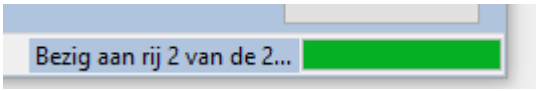


Figure 48: the progressbar and label

If a required field is not filled in the template file, the application will give us an error at the end of importing the file. It does not stop as soon as 1 row is not filled in correctly. If there is an error, there will be forwarded a mail to the bug-email from the hospital so they know something is wrong and can help the user if needed.

There is foreseen a validation on the required fields and a validation on the type of the properties. Like the property “RNR” needs to be a numeric value.

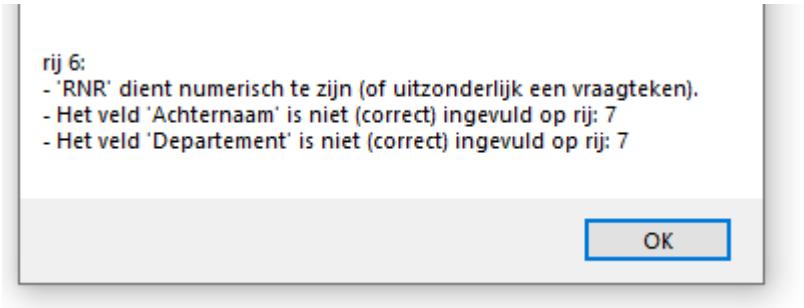


Figure 49: a error message

	...		?	...	Student

Figure 50: a second error message

I also foreseen something that puts a marker in the first cell of the row so if something went wrong, you can just use the same file and it will skip all rows containing the marked cell so we don't get a lot of duplicate entry errors.

5			
6	OK	RubenTest	Boone
7		StefTest	
8			

Figure 51: showing which row is completed and which are missing required data

If the cell contains "OK" it skips the row, the red and green color are there so the user can see which rows succeeded and which not.

3.4.4 Analyses exercises

The mentor sometimes sent an email with a text that needed to be analyzed. He then expected a diagram or clarification about what is present of objects, what properties these objects have, ...

4 SOFT SKILLS

4.1 COMMUNICATION

During my internship, I did not work from home for a single day. If there was a problem I could just stand up and tell the mentor. If he had time, he would come and think with me or administer help. If the mentor was not at the office, he could always be reached via email. We also had 2 team meetings to see how things were going.

Figure 52: teams meetings

Dag Stefan

Wat ik tot nu toe voor elkaar gekregen heb in de webpagina:

- De kalender heeft een default value van vandaag.
- Er kan geen datum in de toekomst gekozen worden.
- De shift dropdown (combobox) toont de juiste shifts (Door de AD-groepen)
- De juiste kamers worden getoond in hun dropdown
- De datagrid komt tevoorschijn met de juiste data, de overbodige kolommen worden onzichtbaar gemaakt

Het probleem waar ik nu al eventjes mee vast zit is dat de datagrid read only is. De datagrid bevat geen .ReadOnly property, heb al redelijk wat Google searches gedaan, maar zonder succes.

Figure 53: a informational mail to the mentor

4.2 SCHEDULING

A Excel file was created in which I could make a schedule. This way the mentor could always see what I was doing and what I planned to do next. There was also a column with all the subjects. Here I could visualize how long I expected to work on them and how long I actually worked on them.

Items	Estimate	Used
3-tier architecture + Ribbon		2
Aanwezig op Thomas More		8
Backup console app		4
Barcode test		12
Barcode test in EconomaatEtiketten		6
Barcode test documenteren		2
Barcode Test toepassen op spoedEtikett	8	10
Brug-dag		8
Bugs fixen		2
configuratietabellen schermpjes maken		16
Datatype database & devicemanager		8
Documenten voor Thomasmore opstellen		36
Herstructureren export + kleine details	16	24
import Excel nieuwe medewerkers	12	8
Klaarmaken deployment spoelregistratie		0

	28/02/2022	1/03/2022	2/03/2022	3/03/2022	4/03/2022
Voormiddag deel 1	Welkom	Barcode test	Barcode test	Barcode test in EconomaatEtiketten	Barcode Test toepassen op spoedEtiketten
Voormiddag deel 2	3-tier architecture + Ribbon	Barcode test	Barcode test	Barcode test in EconomaatEtiketten	Barcode Test toepassen op spoedEtiketten
Namiddag deel 1	Opdracht leren kennen / verwerken	Barcode test	Barcode test documenteren	Barcode Test toepassen op spoedEtiketten	Barcode Test toepassen op spoedEtiketten
Namiddag deel 2	Opdracht leren kennen / verwerken	Barcode test	Barcode test in EconomaatEtiketten	Barcode Test toepassen op spoedEtiketten	Rondleiding Ziekenhuis
	7/03/2022	8/03/2022	9/03/2022	10/03/2022	11/03/2022
Voormiddag deel 1	Spoed etiketten testen	Spoed etiketten testen	Spoed etiketten testen	SpoelRegistratie Data-Laag config	SpoelRegistratie Presentatie-Laag Config
Voormiddag deel 2	Nadenken over data spoelingRegistratie	SpoelingRegistratie meeting	Spoed etiketten testen	SpoelRegistratie Data-Laag config	SpoelRegistratie Presentatie-Laag Config
Namiddag deel 1	Nadenken over data spoelingRegistratie	SpoelingRegistratie analyse config	SpoelingRegistratie analyse config	SpoelRegistratie Presentatie-Laag Confi	SpoelRegistratie Presentatie-Laag Config
Namiddag deel 2	Spoed etiketten testen	SpoelingRegistratie analyse config	SpoelingRegistratie analyse config	SpoelRegistratie Presentatie-Laag Confi	SpoelRegistratie Presentatie-Laag Config
	14/03/2022	15/03/2022	16/03/2022	17/03/2022	18/03/2022
Voormiddag deel 1	SpoelRegistratie App/Bus-laag config	SpoelRegistratie Presentatie-Laag Confi	SpoelRegistratie App/Bus-laag config	SpoelRegistratie App/Bus-laag config	SpoelRegistratie Data-Laag dataIn
Voormiddag deel 2	SpoelRegistratie App/Bus-laag config	SpoelRegistratie Presentatie-Laag Confi	SpoelRegistratie App/Bus-laag config	SpoelRegistratie App/Bus-laag config	SpoelRegistratie Presentatie-laag dataIn
Namiddag deel 1	SpoelRegistratie Presentatie-Laag Confi	SpoelRegistratie App/Bus-laag config	SpoelRegistratie App/Bus-laag config	SpoelRegistratie App/Bus-laag config	SpoelRegistratie Presentatie-laag dataIn
Namiddag deel 2	SpoelRegistratie Presentatie-Laag Confi	SpoelRegistratie App/Bus-laag config	Documenten voor Thomasmore opstelle	Nadenken over benodigdheden dataIn	SpoelRegistratie App/Bus-laag dataIn
	21/03/2022	22/03/2022	23/03/2022	24/03/2022	25/03/2022

Figure 54: piece of the schedule