

# Documentation

# Warehouse- Management- Software

## Inhaltsverzeichnis

1. Technical specifications.....	3
System environment requirements .....	3
2. Configuration.....	4
2.1 Active Directory Configuration.....	4
2.2 Configuration of the SQL-Database.....	6
2.2.1 Creating the DB .....	6
2.2.2 Tables.....	7
2.2.3 Functions and Procedures .....	9
2.4 Storage structure.....	11
3. User Manual .....	12
3.1 How to Log On.....	12
3.2 Menu .....	13
3.3 Booking.....	14
3.3.1 Register.....	14
3.3.2 Create a new article.....	14
3.4 Booking out .....	15
3.5 Statistic .....	16
4. Project Structure .....	17
5. Program Structure .....	18
6. Hardware peripherals.....	23
7. Description of program code.....	24
7.1 Login with AD.....	24
7.2 Booking code .....	25
7.3 Statistics “PieAuslastung” .....	25
8. List of shortcuts .....	26
9. Sources .....	26

## 1. Technical specifications

Brand	Eugen-Reintjes-Schule Hameln
Software name	WHM
Software category	Warehouse management software
Version	1.0 (11.02.2021)
Programming language	C#
GUI Language	German
Operating system	Microsoft Windows

### System environment requirements
















- Microsoft Active Directory (Windows Server 2019)
- Microsoft SQL Server (MSSQL 2017)
- User Client (Microsoft Windows 10 Client)
  - o Barcode Scanner
- Warehouse

## 2. Configuration

### 2.1 Active Directory Configuration

After the basic setup of a Windows Server 2019 and the installation of a domain with the Active Directory Service (AD) there are some configurations to do. The domain name needs to be WHM.local to work. All users that wants to use this program needs to be located in the structure below the WHM.local. All Users that are inside this domain can login into the WHM-Software but they only see options of they are assigned to a specific Active-Directory group. The devices on which the software is used needs to be in the domain because the role assignment works with Active Directory Groups for easier administration.

#### Active Directory-Benutzer und -Computer

Datei	Aktion	Ansicht	?
			
 Active Directory-Benutzer und -C			
>  Gespeicherte Abfragen			
▼  WHM.local			
>  Built-in			
>  Computers			
>  DBUser			
>  Domain Controllers			
>  ForeignSecurityPrincipals			
>  Group			
>  Managed Service Account			
>  Users			
	Name	Typ	Beschreibung
	 WHM_Admin	Benutzer	
	 WHM_Einkauf	Benutzer	
	 WHM_Lager	Benutzer	

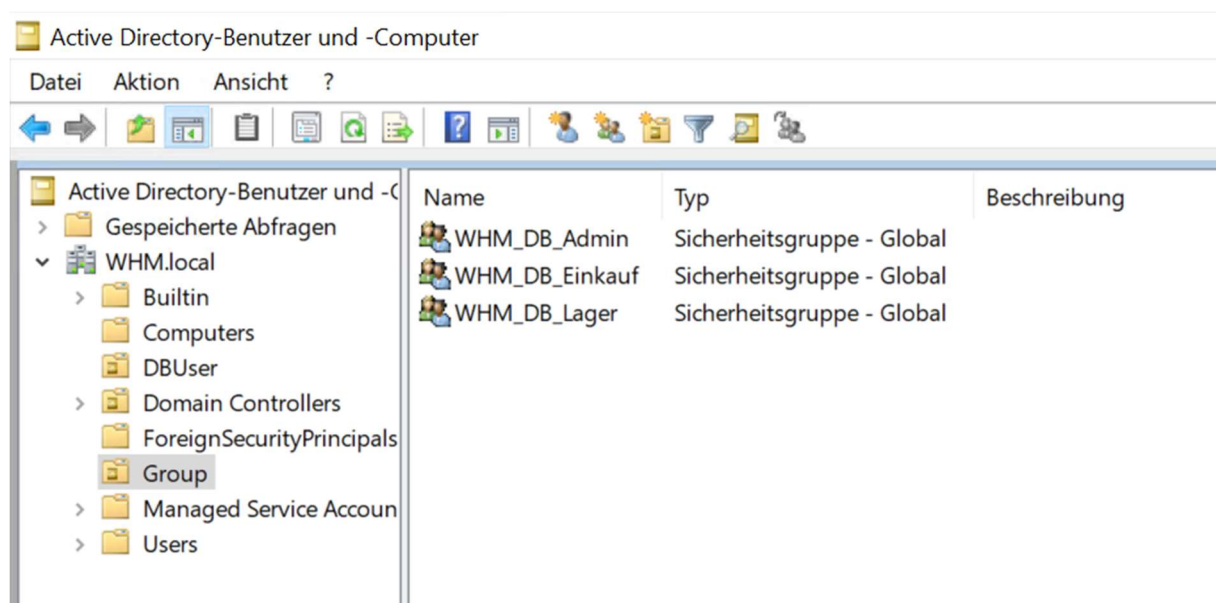
Viewing and using permissions are bound to the groups pictured below. In order to get the right permissions, they need to be created in the OU "Group" in the Active Directory. The program will check if the Users are part of these groups. Users that need permissions need to be added to that group. If the user is in none of these groups, he can sign in into the program, but the menu will be empty.

The groups include the following permissions:

WHM\_DB\_Admin: Einbuchen, Ausbuchen, Statistik

WHM\_DB\_Einkauf: Statistik

WHM\_DB\_Lager: Einbuchen, Ausbuchen

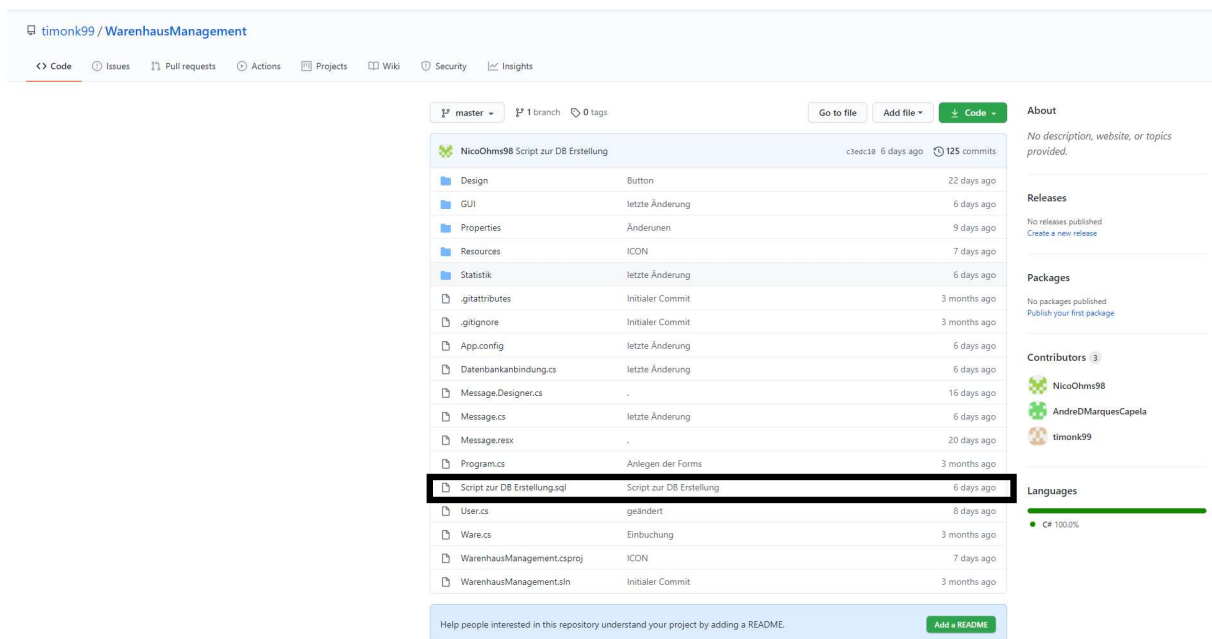


The assignment of rights is managed by a domain administrator.

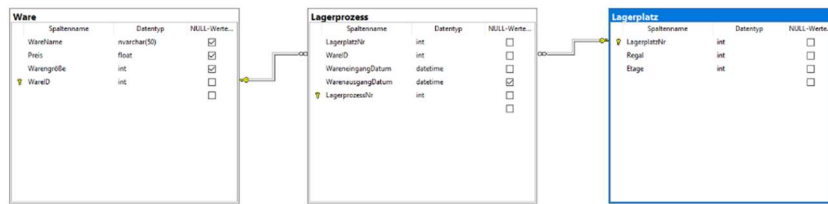
## 2.2 Configuration of the SQL-Database

### 2.2.1 Creating the DB

In the project the MSSQL Database is used. To create the database, you need to install the Microsoft SQL Server 2017 on the wanted device on which the database is gonna run in the future. If the SQL Server is installed, you need to copy the whole SQL statements that are inside the “Script zur DB Erstellung.sql” file inside the repository (<https://github.com/timonk99/WarenhausManagement>) and put it into the created database and start the process. The whole database is gonna build up and afterwards the database part should be completed. The database needs to be online during the use of the WHM Software otherwise the program won't work.



## 2.2.2 Tables



Here are the Tables “Lagerprozess”, “Lagerplatz” and Ware. The Primary key from the Tables is “WareID” and this is one one-to-n relation. Many of Ware and one ‘Lagerprozess’. The PK from the ‘Lagerplatz’-Table is the “LagerplatzNr” and this is a one-to-n relation. Many of “Lagerplatz” and one Lagerprozess. For the statistic in the table “Lagerprozess” are the datetime data's “WareneingangsDatum” and “WareneingangsDatum”.

	ABC WareName	123 Preis	123 Waregröße	123 WareID
1	Kola Kola	0,99	4	0
2	American Kola	1,54	4	1
3	Tofu	2,43	2	2
4	Ravioli Angelo	1,24	1	3
5	Chocolade	1,51	1	4
6	Schoggi Schokolade	1,76	1	5
7	Teatime Chocolate Biscuits	2,53	1	6
8	Scottish Longbreads	0,99	2	7
9	Raclette Courdavault	4,99	2	8
10	Manjimup Dried Apples	0,78	2	9
11	Longlife Tofu	2,78	2	10
12	Ipoh Coffee	1,35	2	11
13	Nord-Ost Matjeshering	2,32	2	12
14	Aniseed Syrup	1,78	1	13
15	Uncle Bob's Organic Dried Pears	2,1	1	14
16	Northwoods Cranberry Sauce	1,45	1	15
17	Chef Anton's Gumbo Mix	2,78	4	16
18	Chai	1,48	1	17
19	test	9	1	18
20	Test	1	1	19
21	Test6	1,25	1	20
22	test021	2,99	1	22
23	Brot	1,99	1	23
24	Kiste Wasser	3,65	1	24
25	Kiste Cola	4,99	1	25
26	Kiste Fanta	4,99	1	26
27	Kiste Sprite	4,99	1	27
28	Kiste Sprite light	4,99	1	28
29	Kiste Fanta light	4,99	1	29
30	Kiste Waser mt Kohlensäure	3,5	1	30
31	Kiste Wasser Medium	3,5	1	31

Here is the example from the table Ware with example data. The “Waregröße” describes the square that need the goods. The “WareID” is an enlisting list.

## Warehouse-Management-Software

[illegible]

This is the table “WareID”. Here you can see the connection to the “Ware” in the “Lagerplatz”. In the table you see when the goods are registered and when there are booked out.

	123 LagerplatzNr	123 Regal	123 Etage
1	10.101	1	1
2	10.102	1	2
3	10.103	1	3
4	10.104	1	4
5	10.105	1	5
6	10.106	1	6
7	10.107	1	7
8	10.108	1	8
9	10.109	1	9
10	10.110	1	10
11	10.201	2	1
12	10.202	2	2
13	10.203	2	3
14	10.204	2	4
15	10.205	2	5
16	10.206	2	6
17	10.207	2	7
18	10.208	2	8
19	10.209	2	9
20	10.210	2	10
21	10.301	3	1
22	10.302	3	2
23	10.303	3	3
24	10.304	3	4

Here are the table “Lagerplatz”. In this table you see the number of a place. The first number get the location of the goods and the other numbers describes the location of the good in which line and on which floor are the goods.



## 2.2.3 Functions and Procedures

In this function we get the “WareName” to get information's about this special good. This function is a simple select command.

```
CREATE FUNCTION dbo.WareID (@WareName nvarchar(50))
RETURNS int as
Begin
    return (select WareID
            from dbo.Ware
            where WareName = @WareName)
END
```

This function read from the database when the goods was registered and booked out.

```
Create Function dbo.f_get_menge_ware(@ts_startdate datetime, @ts_enddate datetime)
RETURNS @rtnTable TABLE
(
    Menge int,
    Ware nvarchar(50)
)
Begin
    insert into @rtnTable
        Select Count(l.WareID), w.WareName from Lagerprozess l
        join Ware w on w.WareID = l.WareID
        where l.WareneingangDatum < @ts_startdate
        and l.WarenausgangDatum > @ts_enddate
        group by w.WareName

    return
END
```

In this procedure we create new goods. We can't use for this a function because the MS-SQL-server can't handle the SQL-command *insert into* form a function.

```
CREATE PROCEDURE dbo.NeuerArtikel @Warename nvarchar(50), @Preis varchar(50), @Warengroße int
AS
INSERT INTO Ware(Warename, Preis, Warengroße)
VALUES (@Warename, convert(float,@Preis), @Warengroße)
```

This is the function to get the information of one good

```
Create Function dbo.f_get_auslastung( @ts_startdate datetime, @ts_enddate datetime, @i_regal int)
Returns int as
BEGIN
    return(Select Count(l.LagerplatzNr) from Lagerplatz l
        join Lagerprozess l2 on l2.LagerplatzNr = l.LagerplatzNr
        where l2.WareneingangDatum < @ts_startdate
        and l2.WarenausgangDatum > @ts_enddate
        and l.Regal = @i_regal)
END
```

In this procedure the programm delete one entry in the database

```
CREATE PROCEDURE dbo.Ausbuchen @LagerplatzNr int, @WareID int
AS
Update Lagerprozess set Warenausgangdatum = GETDATE()
where LagerprozessNr = (Select LagerprozessNr from Lagerprozess where
    LagerplatzNr = @LagerplatzNr and WareID = @WareID and Warenausgangdatum is NULL)
```

```

create function f_einbuchen(@i_waren_id int)
returns @rtn_table TABLE
(
    beschreibung nvarchar(50),
    speicherbedarf int,
    preis float
)
Begin

    insert into @rtn_table
        select w.WareName, w.Warengröße, w.Preis from Ware w
        where w.WareID = @i_waren_id;
    return
END

```

```

CREATE FUNCTION dbo.fn_diagramobjects()
RETURNS int
WITH EXECUTE AS N'dbo'
AS
BEGIN
    declare @id_upgraddiagrams      int
    declare @id_sysdiagrams         int
    declare @id_helpdiagrams        int
    declare @id_helpdiagramdefinition int
    declare @id_creatediagram       int
    declare @id_renamediagram       int
    declare @id_alterdiagram        int
    declare @id_dropdiagram         int
    declare @InstalledObjects       int

    select @InstalledObjects = 0

    select @id_upgraddiagrams = object_id(N'dbo.sp_upgraddiagrams'),
           @id_sysdiagrams = object_id(N'dbo.sysdiagrams'),
           @id_helpdiagrams = object_id(N'dbo.sp_helpdiagrams'),
           @id_helpdiagramdefinition = object_id(N'dbo.sp_helpdiagramdefinition'),
           @id_creatediagram = object_id(N'dbo.sp_creatediagram'),
           @id_renamediagram = object_id(N'dbo.sp_renamediagram'),
           @id_alterdiagram = object_id(N'dbo.sp_alterdiagram'),
           @id_dropdiagram = object_id(N'dbo.sp_dropdiagram')

    if @id_upgraddiagrams is not null
        select @InstalledObjects = @InstalledObjects + 1
    if @id_sysdiagrams is not null
        select @InstalledObjects = @InstalledObjects + 2
    if @id_helpdiagrams is not null
        select @InstalledObjects = @InstalledObjects + 4
    if @id_helpdiagramdefinition is not null
        select @InstalledObjects = @InstalledObjects + 8
    if @id_creatediagram is not null
        select @InstalledObjects = @InstalledObjects + 16
    if @id_renamediagram is not null
        select @InstalledObjects = @InstalledObjects + 32
    if @id_alterdiagram is not null
        select @InstalledObjects = @InstalledObjects + 64
    if @id_dropdiagram is not null
        select @InstalledObjects = @InstalledObjects + 128

    return @InstalledObjects
END

```

## 2.4 Storage structure

The WHM Software is for a specific size of warehouse. On the basic installation it is possible to have 10 shelves with each 10 different levels. It is possible to have 99 shelves with each 99 different levels, but this must be configured in the database afterwards and is only done by one of our technicians. The scannable code which is located at every storage place has the same structure using the example: 10407. The first number ("1") indicates the warehouse. The second and third numbers ("04") indicates the shelf and the fourth and fifth number ("07") shows the floor. This structure was chosen because the whole number can be used to read where the wanted storage place is located.

	123 LagerplatzNr	123 Regal	123 Etage
28	10.308	3	8
29	10.309	3	9
30	10.310	3	10
31	10.401	4	1
32	10.402	4	2
33	10.403	4	3
34	10.404	4	4
35	10.405	4	5
36	10.406	4	6
37	10.407	4	7
38	10.408	4	8
39	10.409	4	9
40	10.410	4	10
41	10.501	5	1

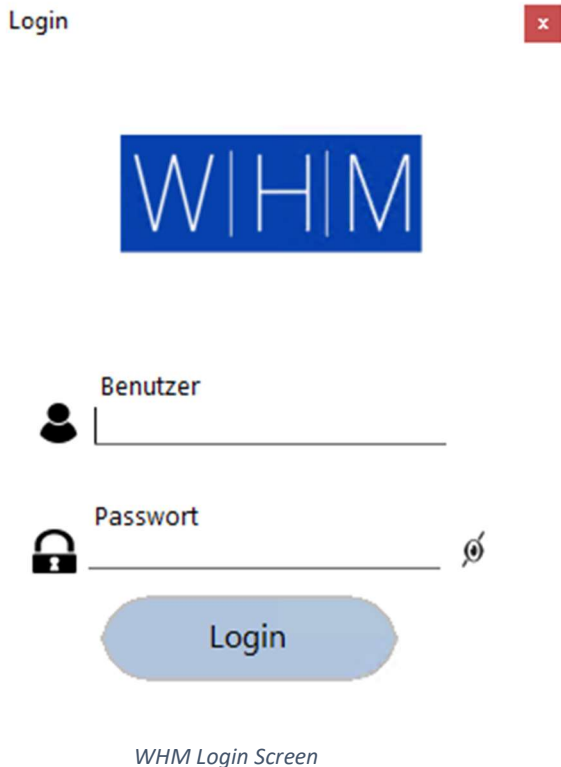
A new booked product gets a unique ongoing number to clearly identify a specific product.

### 3. User Manual

#### 3.1 How to Log On

To get started using the WHM Software, you need to know how to log on. The instructions given below are for the Version 1.0 of the WHM Software. Your login credentials are like your Windows credentials you log in to.

1. Click the WHM.exe on your desktop to start the program.



**The software only works on computers that are inside the domain network!**

2. Enter your Windows credentials and press "Login".

If the credentials are correct it goes to the Menu. If any problems occur contact your responsible system administrator.

### 3.2 Menu

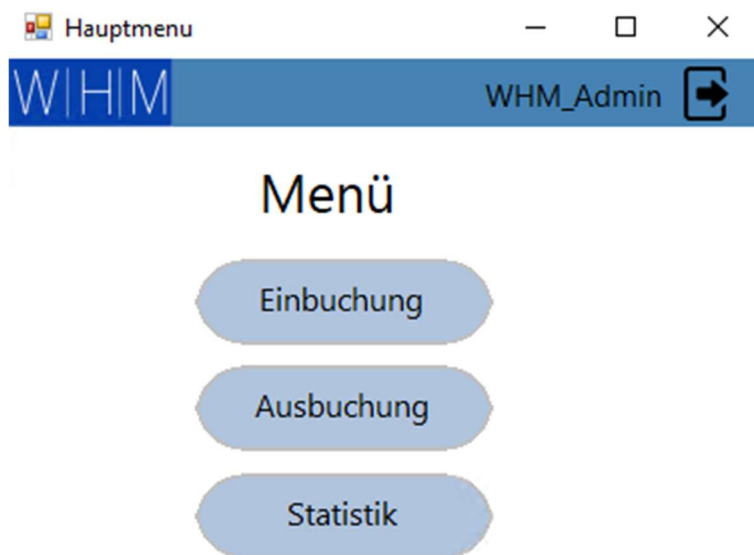
The main menu screen appears. If the login is successful but “Einbuchung”, “Ausbuchung” and “Statistik” does not show up you have missing permissions.

There are 3 different types of permission groups:

Administrator: Einbuchen, Ausbuchen, Statistik

Warehouse worker: Einbuchen, Ausbuchen

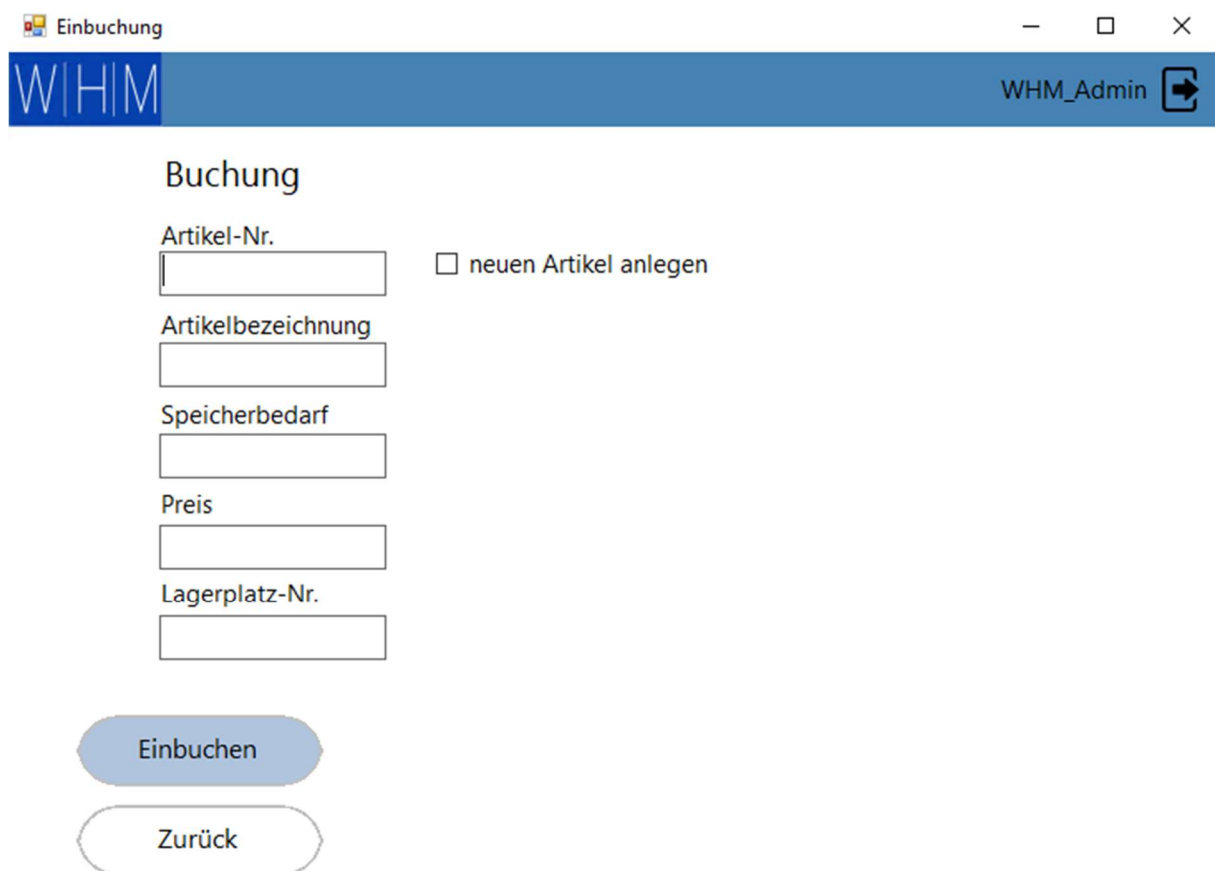
Purchasing: Statistik



### 3.3 Booking

#### 3.3.1 Register

To book an article into the warehouse you scan the article number into the “Artikel-Nr.” field. The article description and price are filled automatically based on the existing data in the warehouse database. The second step is to scan the storage place number into the “Lagerplatz-Nr.” to place the scanned article in the scanned storage place. A standard article has the storage size of 1 what means that It uses 1 regal in the warehouse. This must be written down in the “Speicherbedarf” field. If the storage size of a product is larger than 1 you need to add a 2 which means that the article is two shelves wide. If you press “Einbuchen” the article will be registered into the warehouse at the mentioned storage place number.



Einbuchung

WHM

WHM\_Admin

### Buchung

Artikel-Nr.  ☐ neuen Artikel anlegen

Artikelbezeichnung

Speicherbedarf

Preis

Lagerplatz-Nr.

Einbuchen

Zurück

#### 3.3.2 Create a new article

To add a new article into the warehouse database and register it into the warehouse you can click on the “neuen Artikel anlegen” checkbox. The “Artikel-Nr.” will turn grey because it has no article number at the time, and you can fill the rest. If everything is filled, you can press “Einbuchen” to register the article initial into the database and assign it at the same time to the mentioned storage place number.

### 3.4 Booking out

If an article is going to leave the warehouse and needed to be booking out, you use the “Ausbuchen” user interface. You scan the article that is placed in one shelf and afterwards you scan the assigned storage place number. If the fields are filled with the two numbers, you can press “Ausbuchen” to book the article out. Afterwards the free storage place can be used again for other articles.

Einbuchung

WHM WHM\_Admin

### Buchung

Artikel-Nr.

Artikelbezeichnung

Speicherbedarf

Preis

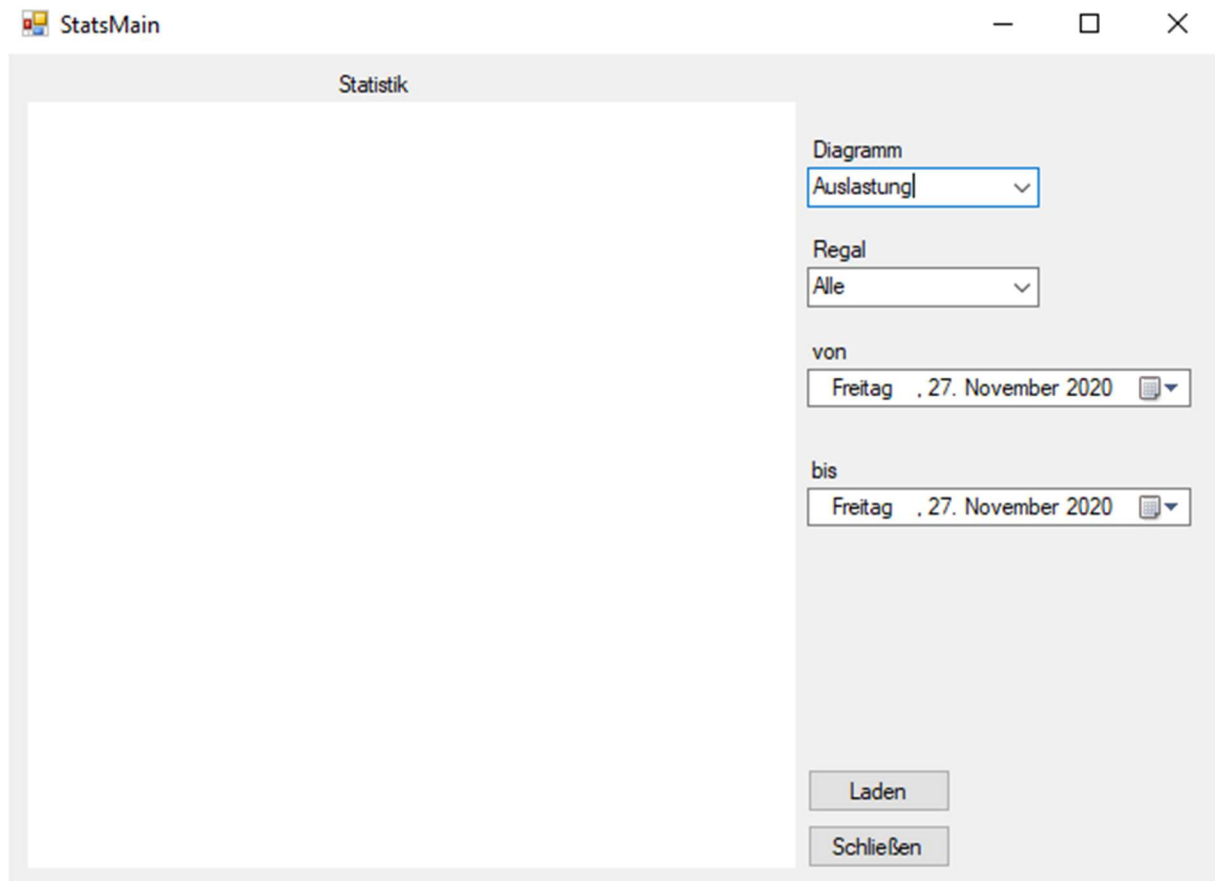
Lagerplatz-Nr.

Ausbuchen

Zurück

### 3.5 Statistic

It's possible to show a statistic under the "Statistik" user interface. There you can show yourself a statistic of the recent used warehouse. A distinction is made between utilization and article. When more detailed information's are required you can specify the shelves and the date to get a more accurate diagram.





## 4. Project Structure

You can see the project structure in the picture below. First, there is a folder named design. There you can find the RoundedButton- class. This class makes only a new design for the button. So you can add a button to the form, which has a new design with rounded edges and other color as the default.

Then there is a folder GUI. Inside of this, there three different classes: Einbuchung, Login and Menu. The class Login is our user interface for the program login. Class Menu has the user interface for the program menu, which follows the login form. The user can his menu, which based on his user group. Einbuchung has the user interface for both types of booking, in and out. Here you enter data like the number of the good or the storage place number.

The next folder are the resources. There you can find pictures, which are used in the program GUI. For example, the logo, which is on every form. After this, there is the folder statistic. Inside of this folder is only the StatsMain class, which is the GUI for the statistic.

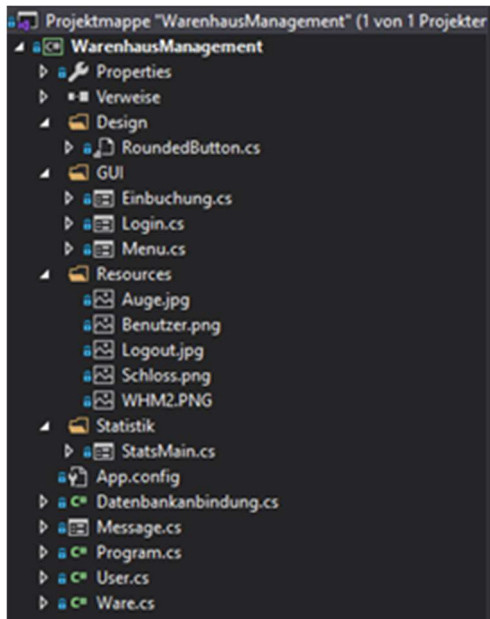
Then there is the App.Config file, where the user can do some configurations, like the db-server.

The next class is the “Datenbankanbindung” class. In this class takes the full communication with the database place. There are different methods, which have input and output parameters. For example, there is a method, which book a good into the warehouse.

The message class is just a form to show error messages. It is mainly used for error handling.

Next one is the user class. This class has fields for the data of a user like username, password and role. In the class are only getter and setters.

At least the class “Ware”. This class has all the same fields, like a good in the db. You can create a “Ware” in the code and can save all the attributes of it. In this class are only getter and setters.



Andre: folder statistic, class message

Lucas: class Datenbankbindung

Nico: some code in login.cs

Timon: folder gui completely with all classes, class user and class ware

## 5. Program Structure

In the following picture you can see the UML-class diagram of the program.

The class Message, StatsMain, Login, Menu und Buchung inherit of the class form.

In the class login is the login user interface. The method LDAPConnection checks the connection to the AD, checks the inserted username and password and when the username and password are correct, the group of the user is set in the user object.

Method pictureBoxPW\_Click is the event handler for the click event on the picturebox. The picturebox is next to the password textbox and when you click on it the password changes the password char. At the first click the password changes to normal letters and at the second click it changes to "\*" -symbols.

The method txtbx\_Password\_KeyDown is also an event handler for the keydown event in the password textbox. The method performs a click at the login button when the pressed key is return.

Next method is the btnlogin\_Click method. This is also an event handler for the click event on the login button. In this method the method LDAPConnection is called and the login process takes place. When the login process is successful, an object of the menu class is created and shows up.

The method txtbx\_username\_MouseClick and the method txtbx\_Password\_MouseClick are both event handler for the click event on the textbox username and password. The methods delete the text in the textboxes.

Next on is the `btnlogin_MouseHover` and `btnlogin_MouseLeave` method. These are also event handlers for the mouse hover and mouse leave event at the login button. In this method the font size, font weight and the back color of the button change.

The class menu is the user interface for the user menu. There is a method called `pictureBox_logout_Click`. This is an event handler for the click event on the picture box. At this method, the user gets logged out. The menu form closes and the login shows up.

The method `btnEinbuchen_Click` is also an event handler for the click event on the "Einbuchen" button. In this method there will be a new object of the "Buchung" class created. Then the user interface for booking articles into the storage will show up.

The method `btnAusbuchen_Click` is also an event handler for the click event on the "Ausbuchen" button. In this method there will be a new object of the "Buchung" class created. Then the user interface for booking articles out of the storage shows up.

Next method is the `btnStatistik_Click` and it is also an event handler for the click event at the "Statistik" button. In this method there will be a new object of the "Statistik" class created. Then the user interface for the statistic shows up.

The `btnEinbuchen_MouseHover`, `btnEinbuchen_MouseLeave`, `btnAusbuchen_MouseHover`, `btnAusbuchen_MouseLeave`, `btnStatistik_MouseHover` and `btnStatistik_MouseLeave` methods are all event handlers for the mouse hover and mouse leave event at the "Einbuchen", "Ausbuchen" and "Statistik" buttons. The methods change the font size, font weight and the back color of the buttons.

Next class is "Buchung". This class is the user interface for the two kinds of booking, in and out. The first method is the `txtbx_ArtikelNr_KeyDown`. It is an event handler for the key down event at the textbox of the article number. If the pressed key is return, the `WarePruefen` method is called.

The method `txtbx_Lagerplatz_KeyDown` is also an event handler for the key down event at the textbox storage place number.

Next one is the method `checkBoxNeuerArtikel_CheckedChanged`. It is also an event handler for the event, which occurs when the checkbox check state changes. It sets the read only into true for the textbox article number, when the checkbox is checked. At all the other textboxes the read only value is false. If the checkbox is not checked, the read only value is at all text boxes true, excepted article number and storage place number.

The method `btnEinbuchen_Click` is also an event handler for the click event on the "Einbuchen" button. In this method the article will be booked into the storage to the storage place number, which the user also inserts. At first the method checks if it is a new article or an existing. If it is a new one, at first the article is created. Then the storage place number will be checked whether the place is free. If the place is free the article will be booked into the storage, else there will be shown up an error message.

The next method is the `btn_Ausbuchen_Click` event. It's also an event handler for the click event on the button "Ausbuchen". In this method an article will be booked out at the inserted storage place number. If the booking goes wrong, there will show up an error message at the user interface.

The method `btnZurueck_Click` is the event handler for the click event on the backwards button. This method closes this current form and shows up the menu again.

Next method is the `txtbx_ArtikelNr_Leave` and it is also an event handler for the event, which occurs when the focus leaves the textbox article number. The method calls the `WarePruefen` method.

The `WarePruefen` method checks the text of the article number textbox and if the text has the length 12 it cuts off the first 8 digits. This has the reason that a barcode has 12 digits, and the barcode scanner enters a string with 12 digits. At the next step, the string is converted to an int. Then a method of the “Datenbankanbindung” class is called to get the article name, size and price.

The method `LagerPruefen` checks the form of the text in the textbox storage place number. If the text has a length of 12 the first 8 digits will be cut off. This is also because of the format of the barcode. Then will be add a 1 at the first digit of the string and then it will be converted to an int. The 1 as the first digit is required, because all storage places in the database have the 1 as the first digit.

The `btnEinbuchen_MouseHover`, `btnEinbuchen_MouseLeave`, `btnAusbuchen_MouseHover`, `btnAusbuchen_MouseLeave`, `btnZuruck_MouseHover` and `btnZuruck_MouseLeave` methods are all event handlers for the mouse hover and mouse leave event at the “Einbuchen”, “Ausbuchen” and “Statistik” buttons. The methods change the font size, font weight and the back color of the buttons.

The last method in this class is the `txtx_Speicher_Leave` method and it is an event handler for the event, which occurs when the focus leaves the textbox storage size. The method checks the text in the textbox and if it is not 1 or 2 there shows up an error message, because the article size must be and 1 or 2.

The next class is the “Ware” class. It is the blueprint for an article object. All the methods in this class are getters and setters for the attributes like article ID, name, price or size.

The class user is the blueprint for an user object. It also has only getter and setter methods for the attributes like username, password and role.

The next class is the “Datenbankanbindung” class. This class has methods to communicate with the database.

The first method is `convertDate` method. This method converts a date into a date format which is the same as the database date format.

The next next method is the `CheckSlot` method. This method calls a database function, which returns whether the entered space is free. When the return value is 1 than the place is free.

The next method is the `CheckSlotUebergrosse`. In this method is a database function called, too. This database function checks, if the place next to the entered storage place number is free. When this function returns a 1 it is free. This function is required for articles with a size more than 1.

The method `CheckAusbuchen` calls a database function, which checks if the process for booking an article out has worked correctly. When everything is all right it returns 1.

Next one is the `EinbuchenMethode`. This method calls a database function, which returns the name, price and size from an article. The method returns this information in a list.

The next methode is `EinbuchenProzedur`. In this method the article and the storage place number are given to a database procedure an books this combination of article number and storage place number into the warehouse.

The method `AusbuchenProzedur` calls a database procedure, which books an article out of the warehouse. The input parameter for the procedure are the article number and the storage place number.

Another method is `NeuerArtikel`. This method calls a database procedure, which creates a new article with the entered name, size and price.

The next method `WareID` calls a function, which returns an article ID. You just must enter the name of the article. This function is used after creating a new article for getting the article ID of the new article.

Furthermore, there is a function `GetRegale`. It creates and execute a SQL command, which return the maximum of the shelves.

The method `GetAuslastung` calls a database function, which returns workload of the shelves. It returns an int.

The last method in this class is the method `GetWarenmenge`. This method also calls a database function which the quantity of the articles. The method returns a list of articles and their quantity.

The next class is the class rounded button which inherit by the class button. This button makes a new design for the buttons. The buttons are rounded now. This class is copied from the internet. The link from the website is in the sources.

The class message is a simple Form that is used to display error messages, mainly in the `StatsMain`.

The last class is `StatsMain`, it's the main class behind the statistics module.

The constructor of the class accepts an object of type "User" and saves it internally in order to connect to the database. It then proceeds to execute the method "Fill\_cb\_regal".

"Fill\_cb\_regal" executes "Get\_Regale" from the class "Datenbankanbindung" in order to fill the "Regale" combobox.

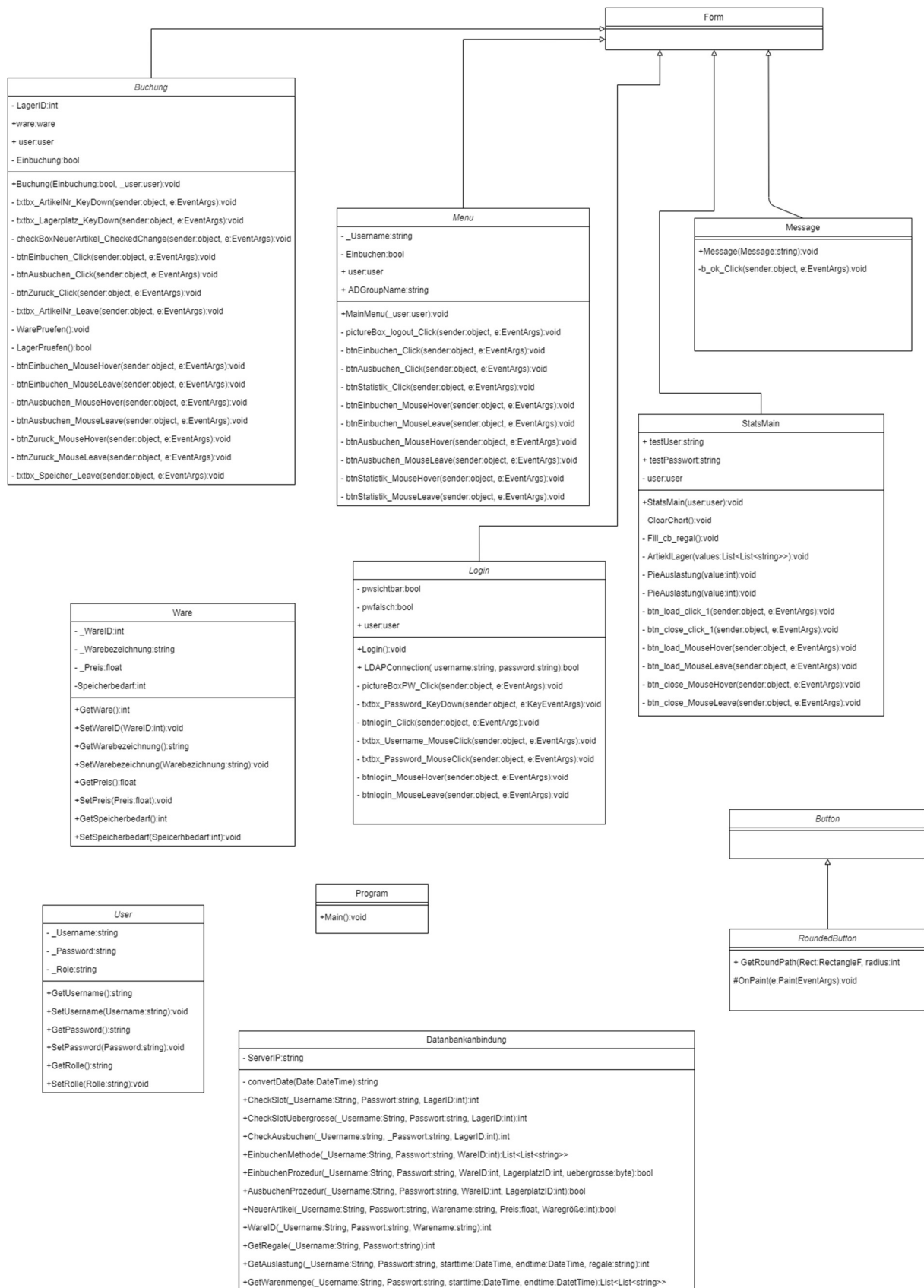
The method "ClearChart" clears the current chart and is used before a new chart is generated.

"ArtikelLager" creates a chart which shows how much of each product is/was available during the timespan set in the `DateTimePickers`. It accepts as parameter a two-dimensional list with the values from the database. It receives the values in the click event.

"PieAuslastung" generates a chart on the overall quantity of items in a specific shelf or overall during the set timespan. It accepts an integer as parameter which is the amount of space used and uses this to calculate the total amount of space occupied and empty.

The method "btn\_load\_Click\_1" loads the selected chart based on the selection in the "Diagramm" combobox. It executes the appropriate method for that chart with the necessary method from "Datenbankanbindung" to fill the chart with data.

The last few function consists of the method "btn\_close\_Click\_1" which closes the statistics window and a `MouseHover` and a `MouseLeave` event for both buttons, load and close, respectively which change the design and font of the buttons whilst hovering over them.

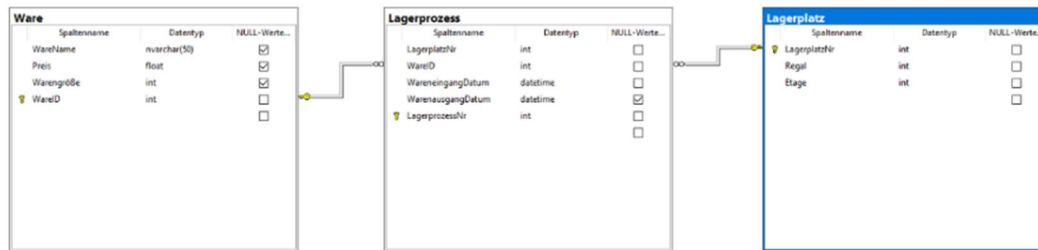


In the picture below you can see the Entity-Relationship-Model of the program. There are three tables. The first table is table “Ware”, where you can find all articles in our database. Each article has a name, a price, a size and an ID. The ID is the primary key of this table. This table has a relationship

to the next table “Lagerprozess”, here are all storage processes. It is a 1:N cardinality. One article is in one or more storage processes, but one storage process has only one article.

The table “Lagerprozess” has the following attributes: storage process number, which is the primary key, article number, which is the foreign key of the table “Ware”, storage place number, which is the foreign key for the table “Lagerplatz”, goods receipt date and goods issue date. This table has a 1:N relationship to table “Lagerplatz”. One storage process can only have one storage place number, but one storage place number can have one or more storage processes.

In the table “Lagerplatz” you can find all storage places. The table has the primary key storage place number. Furthermore, there are the attributes shelf and floor.



## 6. Hardware peripherals

In the program it is possible to use a barcode scanner. You can connect the barcode scanner to the PC via USB and start scanning straight away. You can use the barcode scanner for the article number and for the storage place number, when you create barcodes for them.

## 7. Description of program code

### 7.1 Login with AD

Methods from the “User” program code are being used when the user is starting the program, enter his credentials and press “Login”. The code sets the entered username and password to go to the next step.

```

7 Verweise | Timon Krüger, vor 82 Tagen | 1 Autor, 1 Änderung
public class User
{
    private string _Username;
    private string _Password;
    private string _Rolle;

    11 Verweise | Timon Krüger, vor 82 Tagen | 1 Autor, 1 Änderung
    public string GetUsername()
    {
        return _Username;
    }

    1 Verweise | Timon Krüger, vor 82 Tagen | 1 Autor, 1 Änderung
    public void SetUsername(string Username)
    {
        _Username = Username;
    }

    9 Verweise | Timon Krüger, vor 82 Tagen | 1 Autor, 1 Änderung
    public string GetPassword()
    {
        return _Password;
    }

    1 Verweise | Timon Krüger, vor 82 Tagen | 1 Autor, 1 Änderung
    public void SetPassword( string Password)
    {
        _Password = Password;
    }

    3 Verweise | Timon Krüger, vor 82 Tagen | 1 Autor, 1 Änderung
    public string GetRolle()
    {
        return _Rolle;
    }

    3 Verweise | Timon Krüger, vor 82 Tagen | 1 Autor, 1 Änderung
    public void SetRolle(string Rolle)
    {
        _Rolle = Rolle;
    }
}

```

In the following the “Login” code takes the entered credentials and tries to connect to the domain. The user needs to be a user of the domain. If the connection is working, the code is looking for the assigned groups of the user. The relevant AD groups for the program are: “WHM\_DB\_Admin”, “WHM\_DB\_Lager” and “WHM\_DB\_Einkauf. If the User is in multiple of these groups, it will prefer the ones in the bottom and will give this property to the user.

```

1 Verweise | Timon Krüger, vor 5 Tagen | 2 Autoren, 6 Änderungen
private void btnlogin_Click(object sender, EventArgs e)
{
    lbl_Status.Text = "";
    user.SetUsername(txtbx_Username.Text);
    user.SetPassword(txtbx_Password.Text);

    if (user.GetUsername() != "" && user.GetPassword() != null)
    {
        //Login mit AD wenn ausführender Rechner in Domäne

        string AnmeldeName = txtbx_Username.Text;
        string AnmeldePw = txtbx_Password.Text;
        try
        {
            using (PrincipalContext ctx = new PrincipalContext(ContextType.Domain))
            {
                // find a user
                UserPrincipal nutzer = UserPrincipal.FindByIdentity(ctx, AnmeldeName);

                if (nutzer != null)
                {
                    // get the user's groups
                    var groups = nutzer.GetAuthorizationGroups();

                    foreach (GroupPrincipal group in groups)
                    {
                        if (group.Name == "WHM_DB_Lager")
                        {
                            user.SetRolle("WHM_DB_Lager");
                        }
                        if (group.Name == "WHM_DB_Einkauf")
                        {
                            user.SetRolle("WHM_DB_Einkauf");
                        }
                        if (group.Name == "WHM_DB_Admin")
                        {
                            user.SetRolle("WHM_DB_Admin");
                        }
                    }
                }
            }
        }
    }
}

```



## 7.2 Booking code

In the “Buchung” code it creates a connection to the SQL server ip that is mentioned in the App.config. The authentication on the SQL server is with the “sa” user because this user is created with the SQL server creation and in addition the rights are managed with the Active Directory groups. Afterwards the program uses a SQL command to write the “WareID” and “LagerplatzID” into a new storage process. If its working the command will be executed, otherwise it will return the error message to the user.

```
public static bool EinbuchenProzedur(string _Username, string _Passwort, int WareID, int LagerplatzID)
{
    SqlConnection NewConnection = new SqlConnection("Server = " + ServerIP + "; Database = WHM; User id=sa; Password=Ers1234Ers1234;");
    SqlCommand NewCommand = new SqlCommand("exec p_insert_lagerprozess " + WareID + ", " + LagerplatzID + ";", NewConnection);
    try
    {
        NewCommand.Connection.Open();
        NewCommand.ExecuteNonQuery();
        NewCommand.Connection.Close();
        return true;
    }
    catch (Exception e)
    {
        Console.WriteLine(e.StackTrace);

        NewConnection.Close();

        NewCommand.Connection.Close();
        return false;
    }
}
```

## 7.3 Statistics “PieAuslastung”

This code example is from the class “StatsMain”. This method creates a pie chart based on the number of items in the warehouse. First, we declare a string array which is later used to fill the legend of the chart. Then we clear the chart in case another chart was opened before. After that we set the options for the chart like the title, font and the type of chart, in this case a pie chart. In the if-else branch we check the selected shelves. If all shelves have been selected it will proceed to take the total amount of all shelves, multiply it by the number of slots in the shelve and then subtract from it, the number of slots used. Else if only a specific shelve has been selected it will only take that specific shelve into account. That way we get the total amount of space available. After the branch the method adds the number of slots used to the chart. In case of an exception this function will create and show an object of the message class which shows the exception to the user.

```
private void PieAuslastung(int value)
{
    try
    {
        string[] types = { "Leer", "Voll" };
        ClearChart();

        this.c_chart.Titles.Add("Auslastung von " + cb_regal.Text + " vom " + dt_startDate.Value.ToShortDateString() + " bis " + dt_endDate.Value.ToShortDateString());
        this.c_chart.Titles[0].Font = new Font("Verdana", 12);
        this.c_chart.ChartAreas.Add(new ChartArea());
        this.c_chart.Series.Add(new Series());
        this.c_chart.Series[0].ChartType = SeriesChartType.Pie;

        if (cb_regal.SelectedItem.ToString() == "Alle")
        {
            this.c_chart.Legends.Add(types[0]);
            this.c_chart.Series[0].Points.AddY((cb_regal.Items.Count - 1) * 10 - value);
            this.c_chart.Series[0].Points[0].LabelText = types[0];
            this.c_chart.Series[0].Points[0].AxisLabel = ((cb_regal.Items.Count - 1) * 10 - value).ToString();
        }
        else
        {
            this.c_chart.Legends.Add(types[0]);
            this.c_chart.Series[0].Points.AddY(10 - value);
            this.c_chart.Series[0].Points[0].LabelText = types[0];
            this.c_chart.Series[0].Points[0].AxisLabel = (10 - value).ToString();
        }

        this.c_chart.Legends.Add(types[1]);
        this.c_chart.Series[0].Points.AddY(value);
        this.c_chart.Series[0].Points[1].LabelText = types[1];
        this.c_chart.Series[0].Points[1].AxisLabel = (value).ToString();
    }
    catch (Exception e)
    {
        new Message(e.Message);
    }
}
```

## 8. List of shortcuts

WHM = Warehouse Management Software

AD = Active Directory

OU = Organizational unit

DB = Database

MSSQL = Microsoft SQL

PK = Primary Key

FK = Foreign Key

GUI = Graphical User Interface

## 9. Sources

RoundedButton <https://stackoverflow.com/questions/28486521/rounded-edges-in-button-c-sharp-winforms> (access 24.02.2021 14:50)