

Höhere Technische Bundeslehranstalt Eisenstadt

Bad Kissingen-Platz 3, 7000 Eisenstadt



DIPLOMARBEIT

Visualisierung eines Ersatzteillagers

Verfasst von

NUSSBAUMER Sven

5AHME

PICHLER Noah

5AHME

Betreut von

Dipl. –Ing. (FH) LANG Thomas

Schuljahr 2018/19

Eisenstadt, am 04.04.2019

Vorwort

Bei diesem Dokument handelt es sich um die Dokumentation der Diplomarbeit mit dem Titel „Visualisierung eines Ersatzteillagers“. Dabei wird, in Zusammenarbeit mit der Firma „MEWA Textil-Service GmbH“, ein Informationsdisplay entwickelt, welches eine visuelle Auskunft über den genauen Lagerort von Ersatzteilen bietet. Diese Visualisierung soll als Orientierungshilfe für Mitarbeiter und Angestellten dienen. Grundlage dieser Diplomarbeit ist der Wunsch der Firma „MEWA Textil-Service GmbH“, welche über einen definierten Bereich verfügt, in welchem die Positionen der jeweiligen Ersatzteile präzise hervorgehoben und dargestellt werden. Da diese Arbeit bereits von einem früheren Diplomarbeitsteam in Auftrag genommen wurde, jedoch den Anforderungen der Firma nicht gerecht wurde, kam diese Diplomarbeit zustande. Jedem Ersatzteil ist ein spezieller Lagerort über ein Wartungsprogramm zuwiesen. Damit eben diese Lagerorte nicht mühsam gesucht werden müssen, wurde eine Visualisierung auf einem Bildschirm mit entsprechend unterstützenden Ortungshilfen realisiert.

Dieses Werk beziehungsweise diese Dokumentation richtet sich in erster Linie an die teilhabenden Personen, an die Lehrerschaft, sowie an die nachfolgende Schülerschaft der HTBLA Eisenstadt. Es wird darauf verwiesen, dass die Inhalte dieses Werkes zusammenhängend nicht weiterverwendet werden dürfen.

Inhaltsverzeichnis

Vorwort	2
Inhaltsverzeichnis	3
1. Einleitung	6
2. Abstract	7
3. Allgemeines	8
3.1. Eidesstattliche Erklärung	8
3.2. Kontaktdaten	9
3.2.1. Diplomanden	9
3.2.2. Kooperationspartner	10
4. Danksagung	11
5. Kickoff	12
5.1. Entstehung	12
5.2. Aufgabenstellung	13
5.2.1. Anforderungen	13
5.2.2. Arbeitspakete	14
5.3. Geplantes Endergebnis	15
5.4. Verantwortlichkeiten	16
5.5. Ressourcenplanung	17
6. Grundlagen	18
6.1. Datenbanken	18
6.1.1. Das relationale Datenbankmodell	19
6.1.1.1. Primärschlüssel	19
6.1.1.2. Fremdschlüssel	20
6.1.2. ADO.NET	20
6.2. SQL	21
6.2.1. SQL SELECT Befehl	21

6.2.2.	SQL INSERT INTO Befehl	22
6.3.	C#	24
6.3.1.	Klassen und Objekte	24
6.3.2.	Events	25
6.3.3.	INI Datei	25
6.3.4.	App.config	26
7.	Dokumentation SQL Server 2016.....	27
8.	Realisierung der Datenbank	31
8.1.	ER-Diagramm	31
8.2.	Faktoren der Bildschirmgrößen	32
8.3.	Aufbau der Teilliste.....	32
8.4.	Aufbau der Gebäude-Tabelle	33
8.5.	Aufbau der Raum-Tabelle	33
8.6.	Aufbau der Regal-Tabelle	34
9.	Programmtechnische Ausführungen	35
9.1.	Realisierung: INI-File	35
9.2.	Realisierung: App.config	35
9.2.1.	Schreiben in die App.config	36
9.2.2.	Lesen aus der App.config	36
9.3.	Such-Funktion	36
9.4.	Lesen aus der Datenbank.....	37
9.5.	Schreiben in die Datenbank.....	37
9.6.	Klassenbeschreibung	38
9.6.1.	Regal	38
9.6.2.	Raum.....	39
9.6.3.	Gebäude	41
9.6.4.	Rahmen	41
10.	Handbuch	42

10.1.	Benutzer.....	42
10.2.	Admin	46
11.	Evaluation.....	49
11.1.	Gegenüberstellung	49
11.2.	Übergabe an die Firma	49
11.3.	Persönliche Erfahrungen.....	49
12.	Tätigkeitsnachweis	51
12.1.	Sven Nussbaumer	51
12.2.	Noah Pichler	53
13.	Literatur- & Quellennachweis	55
14.	Abbildungsverzeichnis	56
15.	Anhang	58
15.1.	Create-Scripts SQL Tables	58
15.1.1.	tbl_nb_Gebäude	58
15.1.2.	tbl_nb_Räume	58
15.1.3.	tbl_nb_Regale	59
15.1.4.	tbl_nb_teileliste.....	59
15.2.	CD-ROM	60

1. Einleitung

Es wurde eine Visualisierung für ein Ersatzteillager der Firma „MEWA Textil-Service GmbH“ entwickelt, programmiert und realisiert. Vor Umsetzung dieser Diplomarbeit mussten Ersatzteile mühsam und ohne jegliche Orientierungshilfen gesucht werden. Die neue Visualisierung ermöglicht eine einfachere Auffindung eines Ersatzteiles für alle Mitarbeiter.

Das gesuchte Ersatzteil wird vom Mitarbeiter mittels Barcodescanner oder durch händische Eingabe eingelesen. Durch Eingabe des Barcodes, beziehungsweise Name oder Artikelnummer des Ersatzteiles in die Suchleiste, erscheint eine Liste mit allen passenden Komponenten. Mithilfe eines Filters ist es möglich diese Teile nochmals einzuteilen, um eine genauere Suche zu ermöglichen. Durch einen Doppelklick auf das gesuchte Ersatzteil öffnet sich die Gebäudeansicht. Das Gebäude, in dem sich das ausgewählte Teil befindet, ist rot umrahmt. Ein weiterer Mausklick stellt die Rauman sicht des jeweiligen Gebäudes dar, ebenfalls rot umrahmt. Nachdem man oder frau sich versichert hat in welchem Raum sich das Ersatzteil befindet, wird durch weiteres Klicken die Regalansicht geöffnet. Klickt der Mitarbeiter nun auf das hervorgehobene Regal, erscheint eine Abbildung der Fächer, wobei das relevante Fach, indem sich das Ersatzteil befindet, aufleuchtet. In diesem Fach werden die darin befindlichen Ersatzteile abgebildet und das gewünschte Ersatzteil ist somit ersichtlich. Nachdem sich der Mitarbeiter der Position des gewünschten Ersatzteiles bewusst ist, kann durch einen weiteren Mausklick auf den dargestellten Pfad zum Startpunkt beziehungsweise zu vorherigen Schritten der Visualisierung zurückgesetzt werden.

Zu beachten ist, dass ein Setup über den Admin-Screen, vor dem erstmaligen Benutzen der Visualisierung, notwendig ist. Um Zugriff auf das Admin-Fenster zu erhalten, muss auf den Button „Admin Login“ geklickt werden. Der Admin dient als Bearbeitungsoberfläche. Werden neue Räume hinzugefügt, Regale oder Fächer gekauft, so können diese einfach in der Visualisierung angepasst werden. Durch Betätigen der beschrifteten Buttons und mithilfe eines Drop-Down Menüs ist es möglich alle Eigenschaften wie Anordnung, Größe, Name und Position von Räumen, Regalen und Fächern beliebig zu bearbeiten.

2. Abstract

The main purpose of this diploma thesis was to develop, program and implement a spare part warehouse for the “MEWA Textil-Service GmbH”. For a long time, spare parts had to be searched for tediously and without the help of any guides or support. The new visualization should make it easier for employees to find a desired spare part.

If an employee scans the requested spare part by means of a scanning device, or by entering name or item number into the displayed search bar, a list containing all desired pieces appears. With the help of a filter it is possible to divide these parts again, in order to provide a more precise search. By double-clicking on the spare part you are looking for, the building view opens. The building in which the selected part is located is framed in red. Another mouse click displays the room view of the respective building, also framed in red. After man or woman has insured oneself in which room the spare part is, the shelf view is opened by further clicking. If the employee now clicks on the highlighted shelf, a picture of the compartments appears, with the relevant compartment where the spare part is located lighting up. In this compartment the items inside are shown and the desired spare part is thus visible. After the employee is aware of the position of the desired spare part, a further mouse click on the displayed path can be used to reset to the starting point or to previous steps of the visualization.

Please note that a setup via the admin screen is necessary before using the visualization for the first time. To get access to the admin window, click on the button “Admin-Login”. The Admin serves as the editing interface. If new rooms are added, shelves or compartments purchased, these can be easily adapted in the visualization. By pressing the labeled buttons and using a drop-down menu it is possible to edit all properties such as arrangement, size, name and position of rooms, shelves and compartments as desired.

3. Allgemeines

3.1. Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche gekennzeichnet habe.

Eisenstadt, am 04.04.2019

Unterschrift: _____

NUSSBAUMER Sven Mario

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche gekennzeichnet habe.

Eisenstadt, am 04.04.2019

Unterschrift: _____

PICHLER Noah

3.2. Kontaktdaten

3.2.1. Diplomanden

NUSSBAUMER Sven Mario

Am Hauerschlüssel 24,
A-7203 Wiesen

E-Mail: nussbaumer.sven@htleisenstadt.at

PICHLER Noah

Kirschengasse 1,
A-7033 Pötsching

E-Mail: pichler.noah@htleisenstadt.at

3.2.2. Kooperationspartner

MEWA Textil-Service GmbH

Reinhartsdorfgasse 18,
A-2324 Schwechat-Rannersdorf

Tel. Nr.: 01 70 776 770

BLUEML Christoph

E-Mail: Christoph.Blueml@mewa.at

BANDAT Manuel

E-Mail: Manuel.Bandat@mewa.at

LANG Thomas

E-Mail: thomas.lang@htleisenstadt.at

4. Danksagung

Vorerst möchten wir, Nussbaumer Sven Mario und Pichler Noah, uns bei allen am Projekt beteiligten Parteien bedanken.

Dazu zählen seitens der Schule, Dipl.-Ing. (FH) Lang für seine Unterstützung bei jeglichen Fragen oder Problemen und der Ermöglichung dieser Diplomarbeit. Außerdem möchten wir uns bei unserem Abteilungsvorstand an der HTBLA Eisenstadt, DI Dr. Michael Türk und unserem Klassenvorstand Mag. Gert Rivalta, bedanken, welche uns sämtliche Unterlagen zur Verschriftlichung dieser Diplomarbeit zur Verfügung gestellt haben und uns in organisatorischen Fragen zur Seite gestanden sind.

Besonderer Dank gilt der Firma „MEWA Textil-Service GmbH“, für die Möglichkeit der Verwirklichung dieser Diplomarbeit und die tatkräftige Unterstützung. Alle benötigten Komponenten, wie Barcodescanner, Bildschirm und Lagertabellen wurden zeitgerecht zur Verfügung gestellt und bei Fragen beziehungsweise Unklarheiten oder Problemen war ein reibungsloser Austausch möglich. Daher möchten wir uns bei Herrn Blueml und Herrn Bandat im Speziellen für einen gelungenen Ablauf der gesamten Diplomarbeit bedanken.

5. Kickoff

5.1. Entstehung

Zum offiziellen Start der Diplomarbeit wurde eine Sitzung mit dem Betreuungslehrer Lang Thomas abgehalten, um die Rahmenbedingungen und diversen Aufgabenstellungen zu klären. Um unnötigen Komplikationen und Unklarheiten vorzubeugen, baten wir die Firma „MEWA Textil-Service GmbH“ um ein Treffen. Die darauffolgende Sitzung wurde daher im Firmengebäude des Auftraggebers abgehalten. Dieses Treffen sollte hauptsächlich zur ersten Kontaktaufnahme zwischen Auftraggeber und Auftragnehmer, Informationsaustausch und Absprache der Gegebenheiten und Anforderungen der Diplomarbeit dienen. Im Firmengebäude angekommen wurden wir in den firmeninternen Besprechungsraum gebeten. Unser Auftraggeber Christoph Blueml stellte sich und einige seiner Kollegen kurz vor. Nach einer anschließenden Doku über die Firma „MEWA Textil-Service GmbH“ konnten wir uns einen klaren Überblick der Firma verschaffen. Das Unternehmen bietet unter anderem Berufskleidung, Putztücher und Handtuchrollen an. Die Textilien werden zum Kunden gebracht, abgeholt, gewaschen, gepflegt und gegebenenfalls bei Verschleiß ersetzt. Zusätzlich werden Artikel für Arbeitsschutz, technischen Bedarf und Freizeit per Katalog vertrieben. Anschließend wurden Aufgabenstellung und Arbeitsablauf der Diplomarbeit besprochen. Nach Beendigung der Sitzung fand eine Führung durch das Firmengelände statt. Zusätzlich beider Lagerhallen wurden uns alle unterschiedlichen Regalräume, Regale und Fächer samt Ersatzteilen gezeigt. Abschließend wurde der nächste Besprechungstermin festgelegt. Außerdem wurde beschlossen, dass der Auftraggeber, Christoph Blueml, im Laufe der Diplomarbeit des Öfteren die HTBLA Eisenstadt aufsuchen wird, um den Fortschritt der Arbeit zu begutachten beziehungsweise um aufkommende Fragen oder Problemstellungen zu beseitigen. In Folge dessen wurde vom Auftraggeber ein Pflichtenheft erstellt, welches unter anderem die Aufgabenstellung, sowie Zusammenfassung der Sitzung beinhaltet. Dieses Pflichtenheft diente schlussendlich als Grundlage der Verschriftlichung dieser Diplomarbeit.

5.2. Aufgabenstellung

Es ist eine Visualisierung und Orientierungshilfe für die Firma „MEWA Textil-Service GmbH“ zu planen und umzusetzen. Das Unternehmen verfügt über einen definierten Bereich in welchem Ersatzteile in diversen Schränken, Regalen oder ähnlichen Plätzen gelagert werden. Jedem Ersatzteil ist ein spezieller Lagerort über ein Wartungsprogramm (SAP) zugewiesen. Damit diese Lagerorte nicht mühsam gesucht werden müssen, soll eine Visualisierung auf einem Bildschirm und entsprechenden Orientierungshilfen unterstützend wirken. Das fertig gestellte Diplomarbeitsprojekt soll es dem Instandhaltungsteam ermöglichen, den Lagerplatz von Ersatzteilen möglichst einfach und schnell aufzufinden.

Die Visualisierung soll in der Programmiersprache C# erstellt werden. Außerdem ist eine Kommunikation zwischen dem Programm und dem firmeninternen SQL-Server zu realisieren. Mittels Bardcodescan oder manueller Eingabe soll nach den Ersatzteilen gesucht werden können. Filter für Namen und Kennzeichnung soll die benutzerdefinierte Suche noch weiter unterstützen. Zusätzlich soll ein Admin-Login mit Benutzername und Passwort dafür sorgen, dass Räume, Regale und Fächer beliebig angelegt werden können.

5.2.1. Anforderungen

Nr.	1	Titel	Hardware
<u>MUST HAVE:</u> <ul style="list-style-type: none"> Die Visualisierung soll auf einem Bildschirm dargestellt werden. - HTL Damit die Barcodes der relevanten Ersatzteile erfasst werden können soll ein von der Firma bereitgestellter Scanner verwendet werden. - MEWA Durch scannen eines Barcodes sollen alle passenden Komponenten angezeigt werden und durch Auswahl auf den Lagerort/-platz verwiesen werden. - HTL Zuständigkeit: MEWA und HTL Eisenstadt			
Nr.	2	Titel	Software
<u>MUST HAVE:</u> <ul style="list-style-type: none"> Zur Umsetzung der Datenbank soll MySQL verwendet werden. Die Visualisierung soll über C# programmiert werden. Zuständigkeit: HTL Eisenstadt			

5.2.2. Arbeitspakete

Nr.:	AP01	Bezeichnung	Konzepterstellung
<p>Das erste Arbeitspaket umfasst folgende Themen:</p> <ul style="list-style-type: none"> • Erstellung eines Konzepts mit besprochenen Punkten aus der Startbesprechung • Erstellung formal Projektierung <p>Zuständigkeit: HTL Eisenstadt</p>			
Nr.:	AP02	Titel	MySQL-Server
<p>Das zweite Arbeitspaket umfasst folgende Themen:</p> <ul style="list-style-type: none"> • Erstellung INI-File - HTL • Daten aus INI-File lesen - HTL • Erstellung Server - HTL • Start Programmierung Oberfläche zur Visualisierung - HTL • Kommunikation zwischen Programm und Server - HTL • Excel-Ersatzteilliste zur Verfügung stellen - MEWA <p>Zuständigkeit: MEWA und HTL Eisenstadt</p>			
Nr.:	AP03	Titel	Oberfläche
<p>Das dritte Arbeitspaket umfasst folgende Themen:</p> <ul style="list-style-type: none"> • Verknüpfen von Datenbank und Oberfläche - HTL • MA-Liste automatisch einlesen - HTL • Erstellung Suchfunktion und Filter - HTL • Raumpläne zur Verfügung stellen - MEWA <p>Zuständigkeit: MEWA und HTL Eisenstadt</p>			
Nr.:	AP04	Titel	Admin
<p>Das vierte Arbeitspaket umfasst folgende Themen:</p> <ul style="list-style-type: none"> • Start Programmierung Admin-Fenster • Erstellung Raster und Reiter für Regale • Erstellung Tabellen für Gebäude, Räume und Regale <p>Zuständigkeit: HTL Eisenstadt</p>			
Nr.:	AP05	Titel	Fächer
<p>Das fünfte Arbeitspaket umfasst folgende Themen:</p> <ul style="list-style-type: none"> • Räumlichkeiten und Regale grafisch darstellen • Oberfläche anpassen <p>Zuständigkeit: HTL Eisenstadt</p>			

5.3. Geplantes Endergebnis

Nach Abschluss der Diplomarbeit soll der Firma „MEWA Textil-Service GmbH“ eine funktionsfähige Visualisierung für das Ersatzteillager inklusive der notwendigen Dokumentation übergeben werden. Das Projekt ist erfolgreich, wenn alle relevanten Ersatzteile, Lagerorte und sonstige notwendige Informationen aus einer Datenbank entnommen und in einer Visualisierung grafisch dargestellt werden.

Für die bessere Planung der Diplomarbeit und um dem Auftraggeber einen besseren Überblick über das Endergebnis liefern zu können, wurde ein Konzept erstellt. Damit soll das geplante Ergebnis veranschaulicht werden, auch wenn während der Realisierung Änderungen auftreten könnten. Das Konzept dient also dazu dem Auftraggeber, Firma „MEWA Textil-Service GmbH“, einen groben Überblick über das entstehende Endergebnis zu liefern.

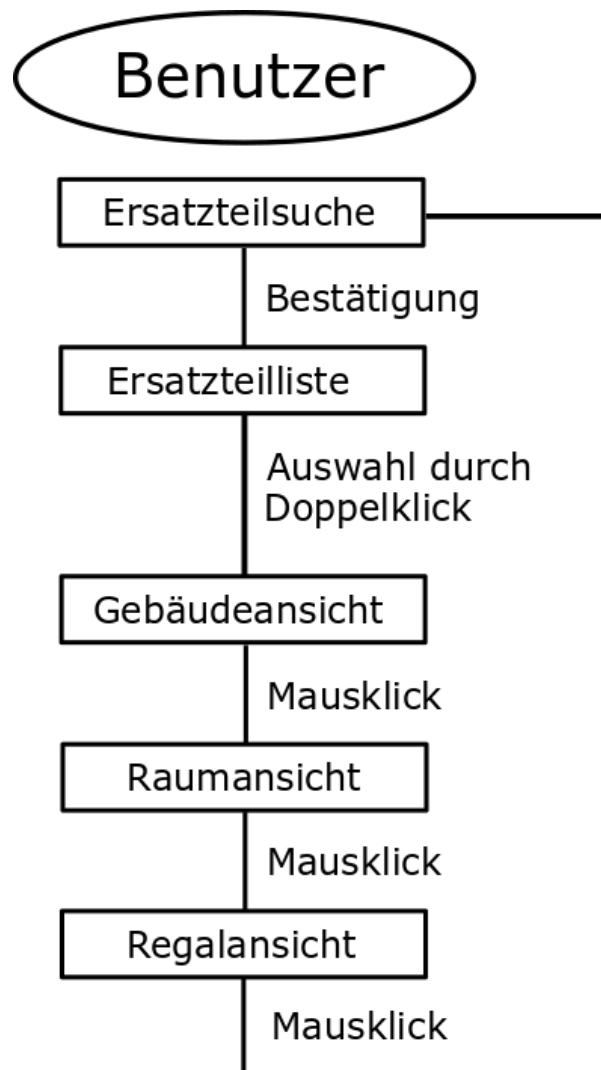


Abb. 5.3-1: Benutzerkonzept

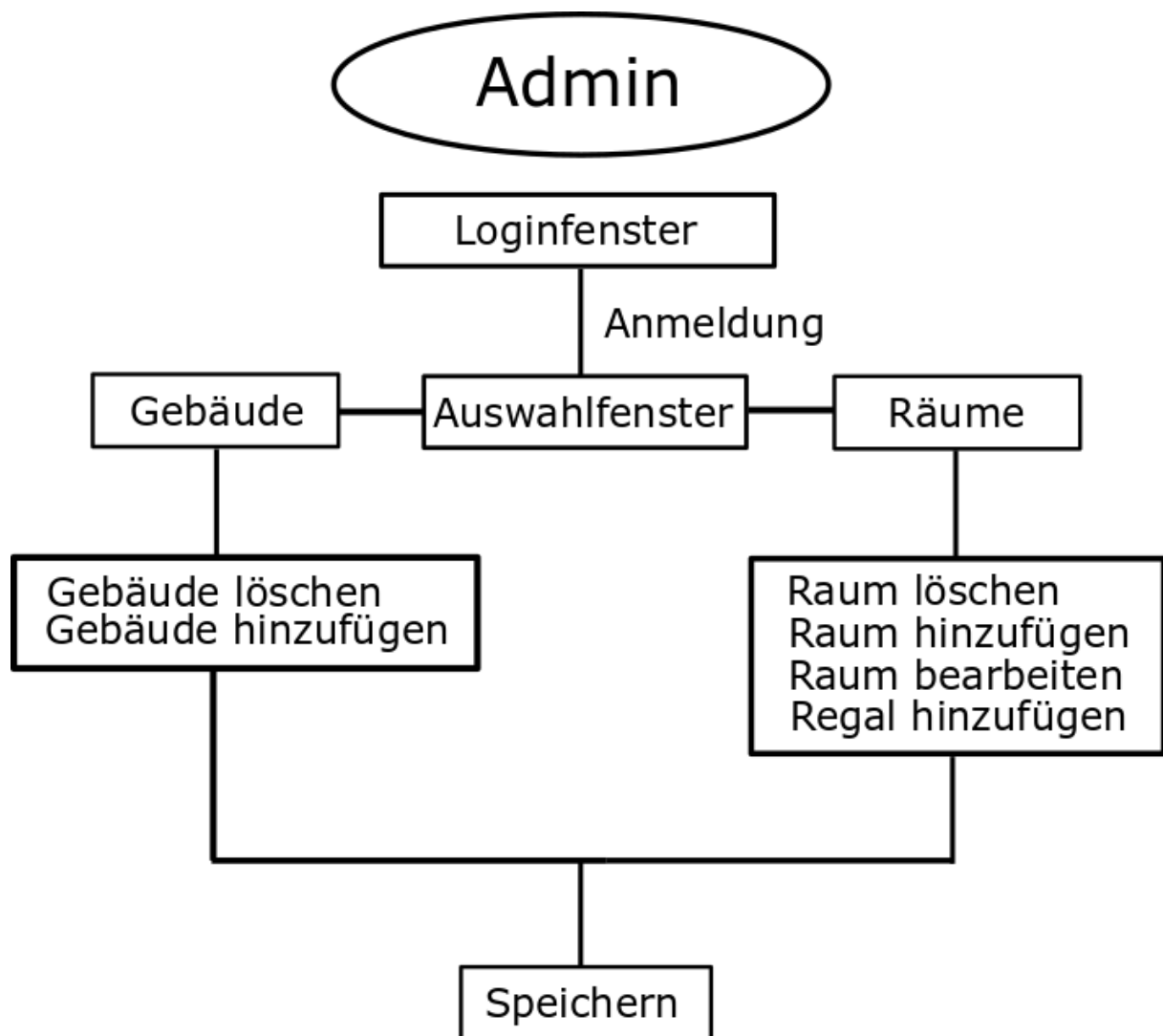


Abb. 5.3-2: Adminkonzept

5.4. Verantwortlichkeiten

Eigentlich wurde zu Beginn der Diplomarbeit festgelegt, welches Teammitglied welche Aufgaben übernimmt, um Unübersichtlichkeit und Wartezeiten auszugrenzen. Allerdings beschlossen wir nach zweiter Sitzung diese Regelung zu verwerfen, da gemeinsames Arbeiten an gleichen Teilgebieten problemlos und ohne Zeitverzögerung möglich war. Im Gegenteil, durch gemeinsamen Fokus auf ein Teilgebiet, konnten Arbeitszeit und -aufwand sogar reduziert werden.

Allerdings spiegeln sich die einzelnen Themengebiete in Arbeitsstunden wieder, wodurch eine grobe Einteilung möglich ist. Während Herr Pichler seinen Hauptfokus auf das Admin-Fenster und dessen Spezifikationen legte, fokussierte sich Herr Nussbaumer hauptsächlich mit der Verbindung zwischen Datenbank und Visualisierung.

5.5. Ressourcenplanung

Benötigte Komponenten, wie unter anderem Gebäude- & Raumpläne und Bildschirm, wurden von der Firma „MEWA Textil-Service GmbH“ kostenfrei zur Verfügung gestellt. Ein Barcodescanner, welcher in einer früheren Diplomarbeit entwickelt wurde, wurde samt Code ebenfalls zur Verfügung gestellt. Der Programmcode, welcher zur Inbetriebnahme des Scanners dient, wurde von einem HTL Absolventen und jetzigem Mitarbeiter der Firma bereitgestellt.

6. Grundlagen

6.1. Datenbanken

Eine Datenbank ist ein System zur elektronischen Verwaltung von Daten. Durch das effiziente Speichern von größeren Datenmengen und Datensätzen, können jene Daten in beliebiger Menge und Form benutzerdefiniert aufgerufen und bereitgestellt werden.

Ein Datenbanksystem, kurz DBS, besteht aus einem Datenbankmanagementsystem (DBMS) und die Menge der eigentlichen Daten, der Datenbank (DB). Diese beiden Komponenten ermöglichen es große Datenmengen strukturiert zu speichern und sie beliebig wiederzugeben. Grundsätzliche Aufgaben des Datenbankmanagementsystems sind Organisation, sowie Strukturierung der Daten. Zusätzlich kontrolliert das DBMS alle der lesenden und schreibenden Zugriffe auf die Datenbank.

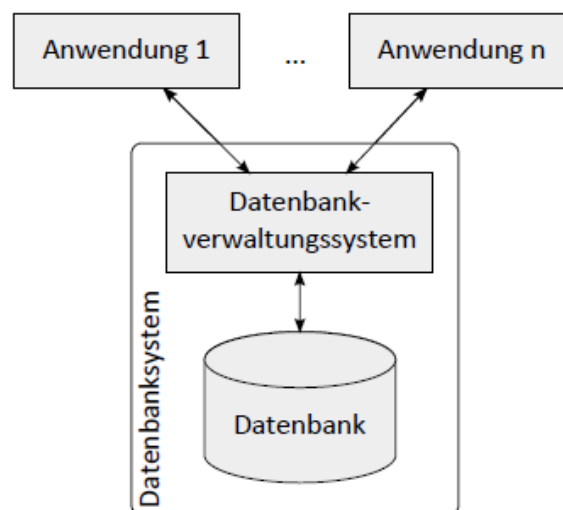


Abb. 6.1-1: Datenbanksystem¹

Wesentliche Aufgaben des DBMS sind die Organisation und Strukturierung der Daten sowie die Kontrolle der lesenden und schreibenden Zugriffe auf die Datenbank. Bei einem DBMS handelt es sich um eine Software, die auf einem Computer-System zu installieren ist. Je nach Anwendungsbereich befindet sich das DBMS auf einem Server oder auf einer Workstation wie einem PC. Um Daten der Datenbank abzufragen, zu speichern oder zu administrieren, bietet das Datenbank-Management-System eine spezielle Datenbanksprache. Ein Beispiel für eine solche Datenbanksprache ist SQL (Structured Query Language).

¹ <https://docplayer.org/docs-images/26/7417988/images/5-0.png>

6.1.1. Das relationale Datenbankmodell

Der Begriff der Relation selbst kommt aus der relationalen Algebra, einem Teilgebiet der Mathematik. Die relationale Algebra beschäftigt sich hauptsächlich mit der Mengentheorie und Operationen der Mengenlehre.

Ein relationales Datenbankmodell ist eine bestimmte und definierte Ansammlung von Tabellen, die jeweils miteinander verknüpft sind. Jede Zeile beziehungsweise jedes Tupel in einer Tabelle ist ein Datensatz. Ein Tupel besteht aus einer Reihe von Eigenschaften (Attributen), den Spalten der Tabelle. Ein Relationsschema legt dabei die Anzahl und den Typ der Attribute für eine Tabelle fest. Zusätzlich können Verknüpfungen, sogenannte Beziehungen, über festgelegte Schlüssel hergestellt werden. Relationen können als Tabellen dargestellt werden. Das bedeutet, relationale Datenbanken sind Datenbanken, die in Tabellen organisiert sind. Die Daten werden in Spalten und Zeilen strukturiert und mithilfe der relationalen Algebra lassen sich Operationen auf Tabellen durchführen.

Für die Beschreibung einer Relation/Tabelle und ihren Elementen werden unterschiedlichste Begriffe verwendet.

Relationales Modell	Darstellung als Tabelle	
Relation	Tabelle	Table
Tupel	Zeile (Datensatz, Entität, Objekt)	Row
Attribut	Spalte, Feld	Column
(Attribut) Wert	Inhalt einer Zelle (Datenfeld)	Value

Das relationale Modell ist das mit Abstand meistverbreitete Datenbankmodell. Das relationale Modell ist ein universales Modell, das sich für eine sehr breite Palette von Anwendungen einsetzen lässt. Im Gegenzug haben andere Modelle Vorteile, wenn die Anwendung genau zu diesem Datenbankmodell passt.

6.1.1.1. Primärschlüssel

Um Verwechslungen auszuschließen, ist es notwendig, für jeden Datensatz eine Kennung durch eindeutige Felder festzulegen. Diese Felder werden als Primärschlüssel (Primary Key) bezeichnet und sind wichtige Bestandteile des relationalen Datenbankmodells. Der Primärschlüssel dient zur eindeutigen Kennzeichnung und Identifizierung eines Datensatzes in einer Tabelle.

6.1.1.2. Fremdschlüssel

Der Fremdschlüssel (Foreign Key) ist ähnlich dem Primärschlüssel ein möglicher Bestandteil einer Tabelle in einer relationalen Datenbank. Hierbei handelt es sich um einen Primärschlüssel einer anderen oder aber derselben Tabelle verweist. Zu beachten dabei ist, dass der Fremdschlüssel nur jene Werte annehmen kann, die in der Referenztable vorhanden sind. Schlussendlich kann auch ein einzelner Fremdschlüssel von einer beliebigen Anzahl an Datensätzen verwendet werden.

6.1.2. ADO.NET

Bei ADO.NET handelt es sich um einen Teil der von Microsoft entwickelten .NET Plattform. Diese stellt eine Sammlung von Klassen dar, welche den Zugriff auf relationale Datenbanken gewährleisten. Es wird als direkter Nachfolger der ActiveX Data Objects (ADO) bezeichnet, wurde jedoch um eine Vielzahl von Funktionen erweitert. Jene Klassen mit der Bezeichnung „System.Data“ verwalten die Datenanbindung und Datenhaltung im Arbeitsspeicher. Dazu existieren unzählige verschiedene Klassen, wie beispielsweise jene, die Verbindungen zu einer externen Datenbank herstellen oder auch solche, die Tabellen im eigentlichen Arbeitsspeicher repräsentieren und darstellen und die Zusammenarbeit ermöglichen, sowie jene, die für die gesamte Datenbank im RAM stehen. Diese Klassen werden „DataSets“ genannt. ADO.NET soll also die Bereitstellung und Anzeige der Daten von der eigentlichen Datenbeschaffung zu trennen.²

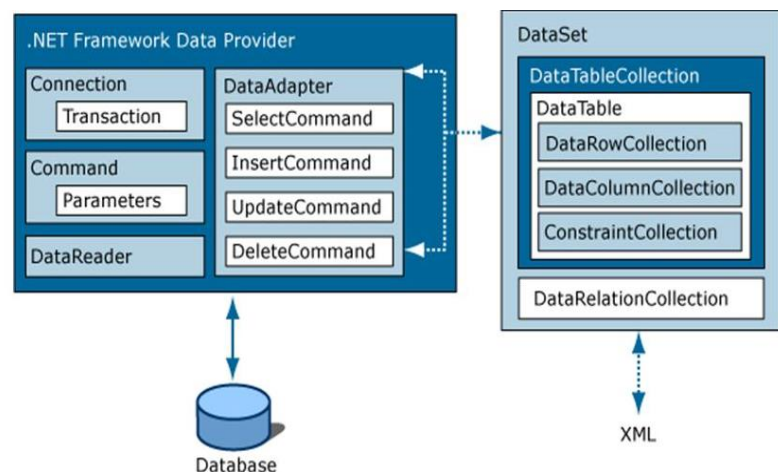


Abb. 6.1.2-1: ADO.NET Architektur

² Quelle Bild: https://images.slideplayer.com/25/7615782/slides/slide_6.jpg

6.2. SQL

Für die Verwendung von Daten aus einer Datenbank, wird SQL (Structured Query Language) verwendet. Diese Datenbanksprache basiert auf der relationalen Algebra, einer formalen Sprache für relationale Datenbanken, die es erlaubt, aus zwei oder mehreren Relationen eine neue bilden zu können. Die SQL Syntax basiert auf einer Reihe von Regelungen, durch die Elemente einer Sprache richtig kombiniert werden können.

Durch sogenannte SQL Statements ist es möglich mit dem Server zu interagieren. Wie zum Beispiel durch das Erstellen, Löschen oder Verändern einer Datenbank, einer Tabelle oder eines Datensatzes.

Eines der am wichtigsten und am häufigsten verwendeten Befehle ist das SQL SELECT Statement. Dieser Befehl dient hauptsächlich für die Auswahl von Daten aus einer oder mehreren Tabellen.

6.2.1. SQL SELECT Befehl

Die einfachste Grundform des Befehls hat den folgenden Aufbau:

SELECT Liste FROM Tabelle

Unmittelbar nach dem Schlüsselwort *SELECT* folgt die Angabe, welche Felder der angesprochenen Tabellen im Ergebnis angezeigt werden sollen. Diesen Bereich einer Abfrage wird im Allgemeinen als Liste oder Klausel bezeichnet. Die angegebene Tabellenquelle *Tabelle* kann einer Basistabelle, einer verknüpften oder abgeleiteten Tabelle entsprechen.

In der allereinfachsten Form wie oben dargestellt, werden sowohl die *WHERE*-Klausel, die die Einschränkungen der Abfrage definiert, wie auch das *ORDER BY*, welche Werte ab- oder aufsteigend anzeigen lässt, weggelassen.

SELECT Liste FROM Tabelle WHERE Suchbedingung ORDER BY Bedingung [ASC/DESC]

Mithilfe des *SELECT INTO*-Befehls können neue Tabellen auf der Grundlage von vorhandenen Daten angelegt werden. Dieses Statement ist vor allem im Bereich der

Datenarchivierung, dem Datentransport oder auch für Backup-Aufgaben hilfreich, wenn für die umfangreichen Datenänderungen eine Sicherheitskopie angelegt werden soll.

Das *SELECT INTO*-Statement legt eine neue Tabelle auf der Basis einer vorhandenen Tabellenherkunft an. Dabei werden automatisch die Bezeichnungen und Datentypen der Spalten aus der Tabellenquelle übernommen, weitere Eigenschaften wie Schlüssel, Einschränkungen oder Indizes allerdings nicht.

SELECT Liste INTO NeueTabelle FROM Tabelle

Voraussetzung für die Ausführung allerdings ist, dass die Zieltabelle noch nicht existiert. Sollte dies jedoch der Fall sein, so kann die Tabelle vor dem *SELECT INTO*-Befehl mit dem Befehl *DROP TABLE Tabelle* gelöscht werden.³

6.2.2. SQL INSERT INTO Befehl

Mit dem *INSERT INTO*-Statement ist es möglich neue Datensätze in einer Tabelle anzufügen. Zusätzlich besteht die Möglichkeit größere Datenmengen aus einer Quell- in eine Zieltabelle zu verschieben.

Die allgemeine Form des *INSERT INTO*-Befehls ermöglicht das Einfügen eines einzelnen Datensatzes in eine Zieltabelle.

INSERT INTO Tabelle VALUES ({ DEFAULT / NULL / Ausdruck } [, ... n])

Mit dieser Schreibweise wird ein neuer Datensatz in die Tabelle *Tabelle* eingefügt. Dabei zu beachten ist, dass alle Tabellenspalten in der richtigen Reihenfolge eingegeben werden müssen. Soll ein Wert ausgelassen werden, so kann die Variable *NULL* verwendet werden, oder der SQL-Server generiert mit dem Ausdruck *DEFAULT* einen Standardwert. Es werden nur dann Felder dargestellt, insofern sie der Bestimmung *NOT NULL* entsprechen und kein Standardwert zugewiesen ist. Eine einfachere Version wäre, wenn im Datensatz nur Standardwerte eingetragen werden sollen, wodurch sich eine relativ einfache Syntax ergibt.

³ Entwicklerbuch: Microsoft SQL Server 2008 R2

INSERT INTO Tabelle DEFAULT VALUES

Es ist möglich mit einem einzelnen *SELEC*-Befehl gleich mehrere Datenzeilen gleichzeitig anzulegen. Dazu werden einfach mehrere Spaltenlisten, die jeweils durch ein Komma voneinander getrennt sind, aneinandergereiht. Jede einzelne Spaltenliste muss den oben genannten Anforderungen entsprechen. Da es aufgrund von einer großen Menge an Datensätzen zu schneller Unübersichtlichkeit führt, kann man die zu füllenden Spalten in einer Liste explizit benennen.

Das *INSERT INTO*-Statement ist allerdings nicht nur für die Übertragung von Datensätzen innerhalb einer Tabelle zuständig, sondern auch für die Übertragung von Datensätzen zwischen zwei unabhängigen Tabellen.

INSERT INTO Tabelle (Spaltenliste) derived-table / execute-statement

Diese Schreibweise ähnelt stark der zuvor beschriebenen Grundform, doch anstelle der Schlüsselwörter steht der *SELECT*-Befehl *derived-table* oder der Aufruf einer gespeicherten Prozedur *execute*. Natürlich müssen auch bei diesem Statement die Reihenfolge und Datentypen der Felder kompatibel sein.⁴

⁴ Entwicklerbuch: Microsoft SQL Server 2008 R2

6.3. C#

Für die Visualisierung des Ersatzteillagers wurde die Programmiersprache C# (gesprochen: C Sharp) aufgrund der Vielzahl von informationstechnischen und wirtschaftlichen Vorteilen gewählt. Die objektorientierte Programmiersprache ist international anerkannt und genormt und gehört neben C++ und Java zu den mächtigsten und heute am weitesten verbreiteten Hochsprachen. Mithilfe von C# ist eine Vielzahl an Anwendungen realisierbar, von einfachen Dienstprogrammen im Hintergrund, über Konsolen-, Desktop- und Webanwendungen, bis hin zur Spieleentwicklung. C# ist im Vergleich zu anderen Programmiersprachen sehr sicher gegen die falsche Verwendung von Speicherbehandlung und Sicherheitslücken. Doch nicht nur Verwendungszweck, sondern auch fortgeschrittene Fehlerbehandlung und Fehlervermeidung, sowie Plattformunabhängigkeit bildeten ebenfalls wichtige Kriterien bei der Auswahl der Programmiersprache.⁵

6.3.1. Klassen und Objekte

Klassen werden als die grundlegendsten der C#-Typen bezeichnet. Eine Klasse ist eine Datenstruktur, die einen Zustand (Felder) und Aktionen (Methoden und andere Funktionsmember) in einer einzigen Einheit kombiniert. Eine Klasse stellt eine Definition für dynamisch erstellte Instanzen der Klasse, auch bekannt als Objekte bereit. Klassen unterstützen Vererbung und Polymorphie. Dies sind Mechanismen, durch die abgeleitete Klassen erweitert und Basisklassen spezialisiert werden können.

Neue Klassen werden mithilfe von Klassendeklarationen erstellt. Diese beginnt mit einem Header, der die Attribute und Modifizierer der Klasse, den Namen der Klasse, die Basisklasse und die von der Klasse implementierten Schnittstellen angibt. Auf den Header folgt der Klassenkörper, welcher aus einer Liste der Memberdeklarationen, die zwischen den Trennzeichen { und } eingefügt werden, besteht.⁶

⁵ <https://www.der-wirtschaftsingenieur.de/index.php/programmiersprache-c-sharp/>

⁶ <https://docs.microsoft.com/de-de/dotnet/csharp/tour-of-csharp/classes-and-objects>



6.3.2. Events

Events sind Funktionen, welche einem bestimmten Objekt zugeordnet sind und auch nur von diesem aufgerufen werden können. Dabei repräsentieren Events häufig die Reaktion auf bestimmte Objektzustände oder Objektänderungen. Oft treten diese Events in Verbindung mit einer Eingabe des Benutzers auf. Wird z.B. das Drücken einer Taste betrachtet, so werden insgesamt vier Events mit unterschiedlichen Event-Handlern ausgelöst.

- MouseDown-Event: Repräsentiert das Hinunterdrücken einer Taste
- Click-Event: Repräsentiert die Reaktion eines Controls auf einen Klick
- MouseClick-Event: Repräsentiert die Reaktion eines Controls auf den Mausklick
- MouseUp-Event: Repräsentiert das Loslassen der Maus

Durch das Aneinanderreihen solcher Events, kann eine Eingabe sehr gut in verschiedene Programmabschnitte zerlegt werden. So kann beispielsweise bei einem Drag&Drop-Vorgang ein MouseDown-Event für die Auswahl der Datei zuständig sein.

6.3.3. INI Datei

Eine INI-Datei, auch als Initialisierungsdatei bezeichnet, ist eine Textdatei, wie in Abbildung 6.3.3-1 zu erkennen, deren Inhalt Schlüssel-Wert-Paare enthält, die durch Sektionen gegliedert werden. Initialisierungsdateien werden hauptsächlich von Microsoft-Windows-Anwendungen als Konfigurationsdatei verwendet und dienen als Dateiformat zur Speicherung von Programmkonfigurationen. Durch den einfachen Aufbau und die Benutzerfreundlichkeit wird das Format auch betriebssystemübergreifend verwendet, um eine dauerhafte Speicherung der Einstellungen von Programmen zu ermöglichen.⁷

```
[Server]
MS_SQL_Server_Name=DESKTOP-C52099I\ERSATZTEILLAGER
MS_SQL_Anzeigename=MS SQL Betrieb
MS_SQL_Treiber=SQL Server
MS_SQL_Description=MS SQL Server Express für Betrieb
MS_SQL_Database=Betrieb
MS_SQL_User=mewa45
```

Abb. 6.3.3-1: INI-File

⁷ <https://de.wikipedia.org/wiki/Initialisierungsdatei>

6.3.4.App.config

Im einfachsten Fall wird die App.config als eine XML-Datei mit einer Vielzahl von möglichen vordefinierten Konfigurationsabschnitten und dessen Unterstützung bezeichnet. Ein Konfigurationsabschnitt dabei ist ein bestimmter Ausschnitt aus XML mit einem Schema, das dazu dient, eine Art von Informationen zu speichern.

Die Einstellungen können über eingebaute Konfigurationsabschnitte wie „connectionStrings“ oder „appSettings“ konfiguriert werden. Es besteht ebenfalls die Möglichkeit eigene benutzerdefinierte Konfigurationsabschnitte hinzuzufügen. Dies ist ein eher fortgeschrittenes Thema, allerdings sehr leistungsfähig für die Erstellung von stark typisierten Konfigurationsdateien. Typischerweise werden für Webanwendungen eine web.config verwendet, während Windows/GUI/Dienstanwendungen eine App.config verwenden.

Konfigurationsdateien auf Anwendungsebene erben Einstellungen aus globalen Konfigurationsdateien, wie z.B. der machine.config.

7. Dokumentation SQL Server 2016

Nach Start der Installationsdatei des SQL Servers wird das SQL-Server Installationscenter vom Installationsassistenten ausgeführt. Unter dem Reiter Installationsregeln überprüft SQL-Server Setup auf mögliche Probleme, die beim Ausführen der Installation auftreten könnten. Da bei unserer Setup-Datei keine Unregelmäßigkeiten aufgetreten sind, wie auf Abbildung 7-1 erkenntlich ist, wurden die Komponenten für die Installation ausgewählt.

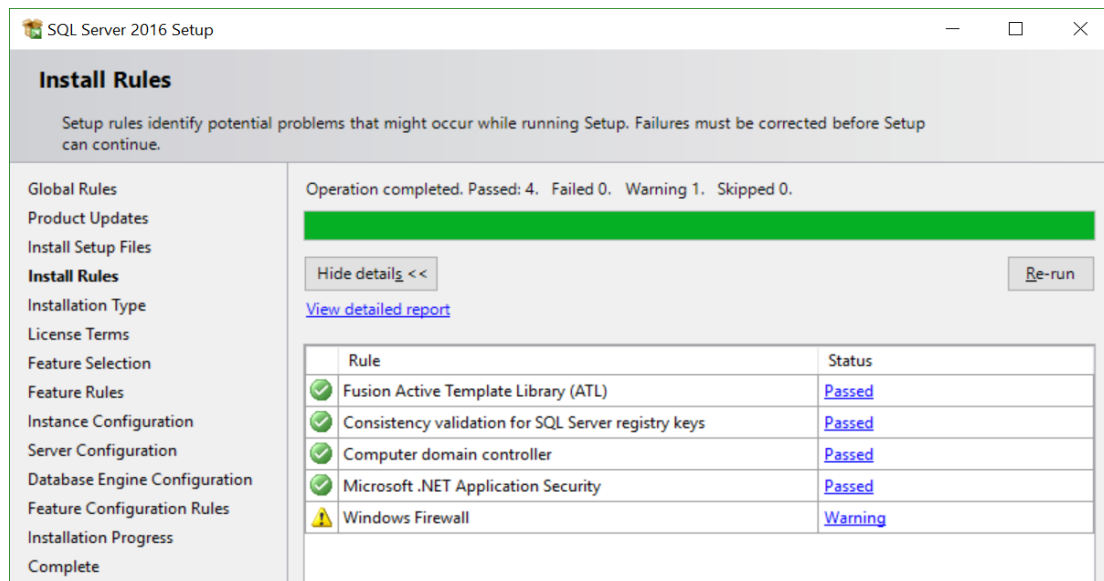


Abb. 7-1: Installationsregeln

Nach der Bestimmung der Komponenten und des spezifischen Funktionsnamens wurde eine Beschreibung der einzelnen Komponentengruppen angezeigt. Die jeweils erforderlichen Komponenten sind im Bereich „Prerequisites for selected features“ oder auch „Voraussetzungen für ausgewählte Funktionen“ ersichtlich.

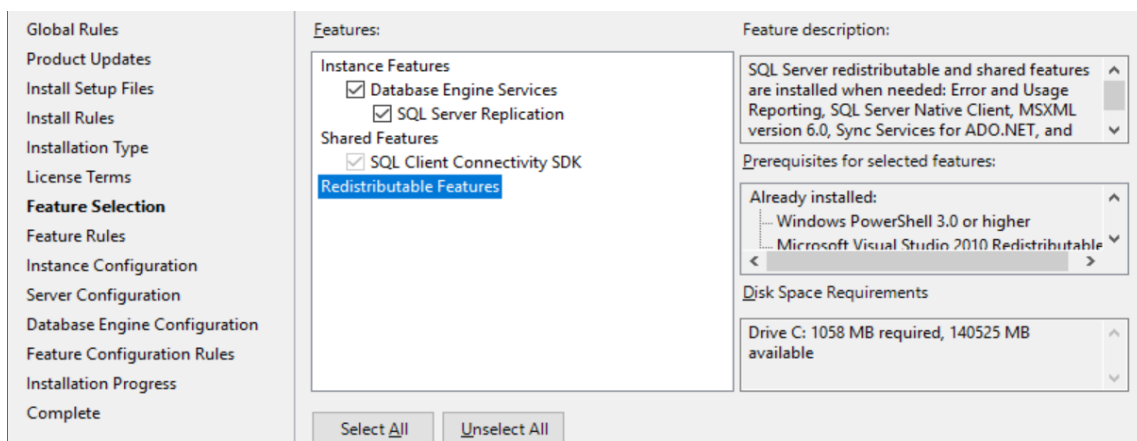
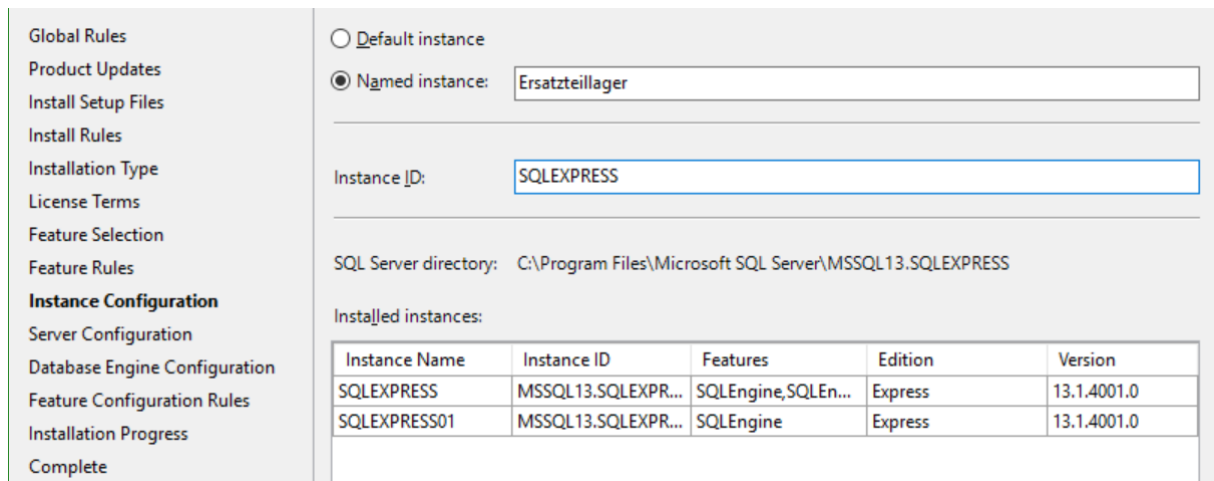


Abb. 7-2: Funktionsauswahl

Im Reiter der Instanzkonfiguration ist zwischen Standardinstanz oder benannter Instanz zu entscheiden. Wie in Abbildung 7-3 zu sehen ist wurde die Instanz „Ersatzteillager“ benannt.



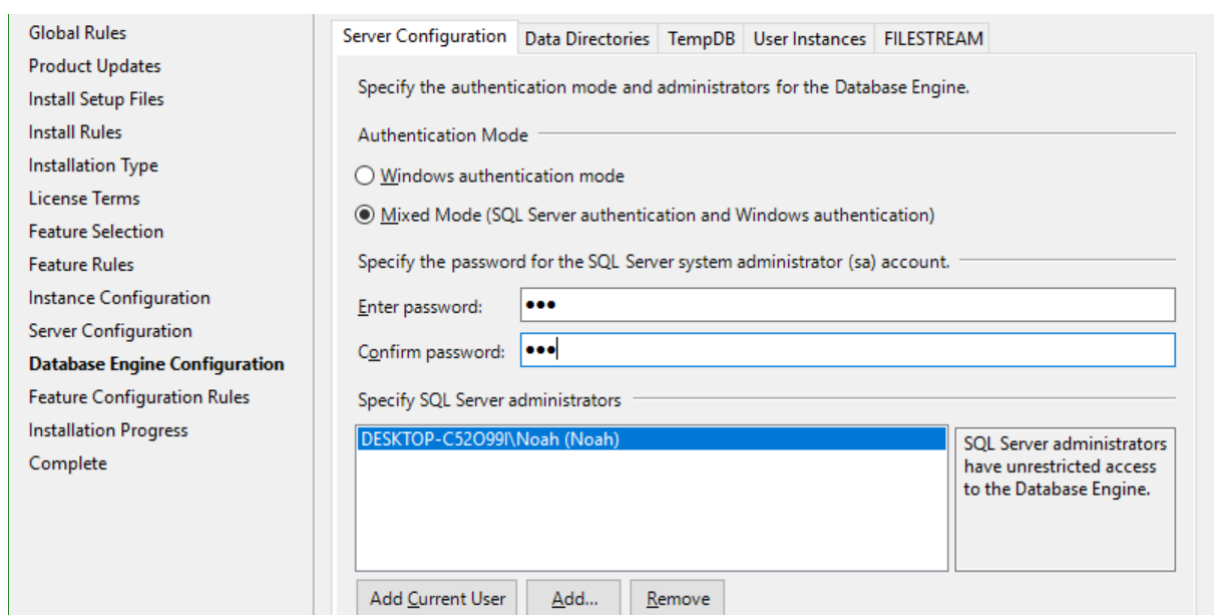
The screenshot shows the 'Instance Configuration' step of the SQL Server Enterprise Setup Wizard. The left sidebar lists various configuration steps, with 'Instance Configuration' currently selected. The main area shows options for 'Default instance' (unselected) and 'Named instance' (selected). The 'Named instance' is set to 'Ersatzteillager'. Below this, the 'Instance ID' is set to 'SQLEXPRESS'. The 'SQL Server directory' is 'C:\Program Files\Microsoft SQL Server\MSSQL13.SQLEXPRESS'. A table titled 'Installed instances' shows two existing instances: 'SQLEXPRESS' and 'SQLEXPRESS01', both with 'MSSQL13.SQLEXPRESS' as the Instance ID and 'SQLEngine, SQLEn...' as features.

Instance Name	Instance ID	Features	Edition	Version
SQLEXPRESS	MSSQL13.SQLEXPRESS	SQLEngine, SQLEn...	Express	13.1.4001.0
SQLEXPRESS01	MSSQL13.SQLEXPRESS	SQLEngine	Express	13.1.4001.0

Abb. 7-3: Instanzkonfiguration

Zusätzlich werden im Raster SQL-Server Instanzen dargestellt die sich auf dem lokalen Rechner befinden. Sollte bereits eine sogenannte „Default instance“ beziehungsweise Standardinstanz auf dem Rechner installiert sein, muss eine benannte Instanz vom SQL Server installiert werden.

Als nächstes wird der SQL-Server Sicherheitsmodus festgelegt und bestimmte Benutzer oder gar ganze Gruppen als Administratoren der SQL Server-Datenbank-Engine hinzugefügt. Für die Authentifizierung wie in Abbildung 7-4 dargestellt wurde ein vereinbartes Passwort für das vorhandene SQL-Server Systemadministratorkonto angegeben.



The screenshot shows the 'Database Engine Configuration' step of the SQL Server Enterprise Setup Wizard. The left sidebar lists various configuration steps, with 'Database Engine Configuration' currently selected. The main area shows the 'Server Configuration' tab. It prompts the user to 'Specify the authentication mode and administrators for the Database Engine.' The 'Authentication Mode' is set to 'Mixed Mode (SQL Server authentication and Windows authentication)'. Below this, the user is prompted to 'Specify the password for the SQL Server system administrator (sa) account.' The 'Enter password' and 'Confirm password' fields are both filled with '...'. Below these fields, the 'Specify SQL Server administrators' section shows a list of administrators, with 'DESKTOP-C52099\Noah (Noah)' selected. A note on the right states 'SQL Server administrators have unrestricted access to the Database Engine.' At the bottom, there are buttons for 'Add Current User', 'Add...', and 'Remove'.

Abb. 7-4: Datenbank Engine Konfiguration

Es muss mindestens ein Systemadministrator für die SQL-Server Instanz angegeben werden.

Nach Abschluss der Installation öffnet sich nach kurzer Wartezeit der SQL-Server Import/Export-Assistent für weitere Konfiguration.



Abb. 7-5: SQL-Server Import/Export-Assistent

Zum Import der Ersatzteilliste aus dem Lagersystem wurde ein Excel-Sheet von der Firma bereitgestellt, welches mithilfe des Import Assistenten des SQL-Servers importiert wurde.

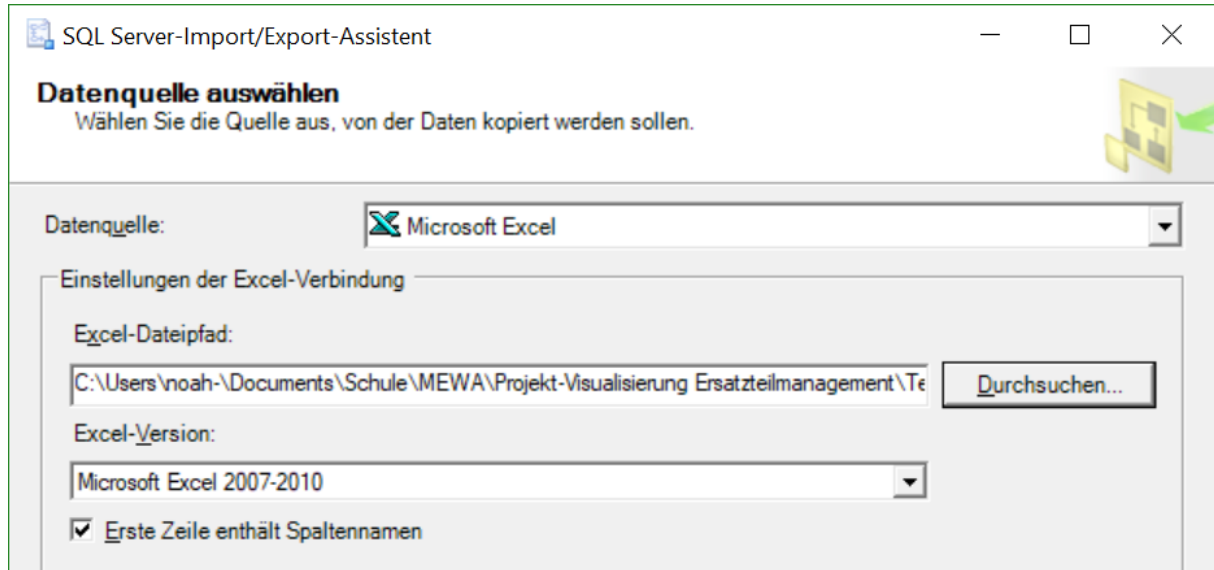


Abb. 7-6: Import Datenquelle

Nach Auswahl des richtigen Dateipfads, indem sich die Ersatzteilliste befindet, wird ein SQL Code generiert um die Tabelle auf der Datenbank zu erstellen und zu verwalten. In der folgenden Ansicht wird der benötigte Programmcode veranschaulicht. Zusätzlich ist der Inhalt des Excel-Sheets im Reiter „Ergebnisse“ dargestellt.

SQLQuery2.sql - DES...L Betrieb (sa (53))

```

/***** Skript für SelectTopNRows-Befehl aus SSMS *****/
SELECT TOP 1000 [Material]
, [Materialnummer]
, [Werk]
, [LOrt]
, [MArt]
, [Materialart]
, [Warengruppe]
, [Warengruppe1]
, [Platz]
, [Bestand]
, [BME]
FROM [MS_SQL_Betrieb].[dbo].[abf Teileliste$]

```

100 %

Ergebnisse Meldungen

	Material	Materialnummer	Werk	LOrt	MArt	Materialart	Warengruppe
1	000000110000000015	Wellendichtring Viton/FPM Ø30xØ62x7 3760	1045	9000	ZLMT	IH Lagemat. bewertet	M0808
2	000000110000000021	Rundbürste ZZB PA 6.10	1045	9000	ZLMT	IH Lagemat. bewertet	M0699
3	000000110000000025	Dichtungshalte	1045	9000	ZLMT	IH Lagemat. bewertet	M0603
4	000000110000000027	Drehgelenk VA IG/AG ¼" 2817-	1045	9000	ZLMT	IH Lagemat. bewertet	M0601
5	000000110000000044	Schwimmerschal 240V AC/0 IP 68	1045	9000	ZLMT	IH Lagemat. bewertet	M0814
6	000000110000000053	Turbokreisler-A TUA-2f mi Ø250	1045	9000	ZLMT	IH Lagemat. bewertet	M0601
7	000000110000000060	Zylinder CP3520 Ø37,6xØ55 Messi	1045	9000	ZLMT	IH Lagemat. bewertet	M0603
8	000000110000000061	Adapter weiblich CP3520 Ø33,3xØ46 Messi	1045	9000	ZLMT	IH Lagemat. bewertet	M0603
9	000000110000000062	Adapter männlich CP3520 Ø33,8xØ44 Messi	1045	9000	ZLMT	IH Lagemat. bewertet	M0603
10	000000110000000063	Distanzstück +F CP3520 5xØ46x20, VA	1045	9000	ZLMT	IH Lagemat. bewertet	M0603

Die Abfrage wurde erfolgreich ausgeführt. | DESKTOP-C52099\ERSATZTEILL... | sa (53) | MS SQL Betrieb | 00:00:00 | 1000 Zeilen

Abb. 7-7: Programmcode

Hier noch ein abschließender Einblick in die benutzerformatierte Tabelle (siehe Abb. 7-8).

Site	null	System.ComponentModelMode
Tables	(System.Data.DataTableCollection)	System.Data.DataTableCol
Count	1	int
IsReadOnly	false	bool
IsSynchronized	false	bool
List	Count = 1	System.Collections.ArrayLi
[0]	(Table)	object (System.Data.DataT
Rohdatenansicht		
SyncRoot	(System.Data.DataTableCollection)	object (System.Data.DataT
Statische Member		
Nicht öffentliche Member		
Columns	{System.Data.DataColumnCollection}	System.Data.DataColumn
Count	11	int
IsReadOnly	false	bool
IsSynchronized	false	bool
List	Count = 11	System.Collections.ArrayLi
[0]	{Material}	object (System.Data.DataC
[1]	{Materialnummer}	object (System.Data.DataC
[2]	{Werk}	object (System.Data.DataC
[3]	{LOrt}	object (System.Data.DataC
[4]	{MArt}	object (System.Data.DataC
[5]	{Materialart}	object (System.Data.DataC
[6]	{Warengruppe}	object (System.Data.DataC
[7]	{Warengruppe1}	object (System.Data.DataC
[8]	{Platz}	object (System.Data.DataC
[9]	{Bestand}	object (System.Data.DataC
[10]	{BME}	object (System.Data.DataC
Rohdatenansicht		

Abb. 7-8: Tabellenansicht

8. Realisierung der Datenbank

8.1. ER-Diagramm

Zur Realisierung der Datenbank wurde ein ER-Diagramm entworfen, welches die einzelnen Tabellen und ihre Beziehungen zueinander beschreibt. Dabei werden die Primärschlüssel mit gelben Schlüsselssymbolen und die Fremdschlüssel mit roten Quadraten markiert. Unter den Tabellen herrscht eine 1:n Beziehung, das bedeutet, wenn eine Tabelle einen Fremdschlüssel besitzt, definiert dieser, zu welchem Eintrag er in der korrespondierenden Tabelle mit dem passenden Primärschlüssel gehört. Daher gilt Primärschlüssel ist gleich Fremdschlüssel.

Beispiel: In der Tabelle „Gebäude“ befindet sich ein Eintrag mit der g_id 5 (Primärschlüssel). Nun können in der Tabelle „Räume“ 3 Einträge mit der ra_id 1, 2 und 3 (Primärschlüssel) stehen, welche alle die ra_gebäude_id 5 (Fremdschlüssel) besitzen. Da auch kein weiteres Gebäude mit der g_id 5 auftreten kann sind diese 3 Räume dem Gebäude eindeutig zugeordnet.

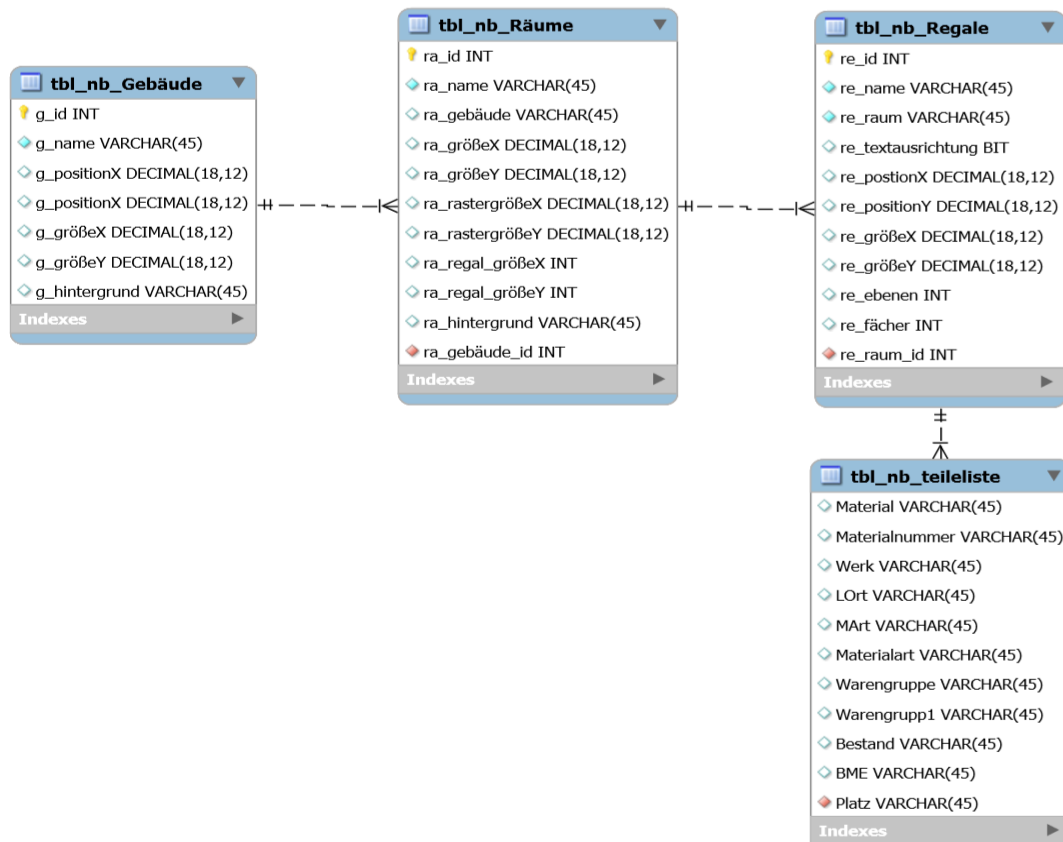


Abb. 8.1-1: ER-Diagramm

8.2. Faktoren der Bildschirmgrößen

Die Breiten und Höhen der Steuerelemente werden als Faktor der Bildschirmbreite abgespeichert, damit jene Elemente auch gleichzeitig bei verschiedenen Auflösungen dargestellt werden können. Die zuerst angedachte Ausführung, die Breiten und Höhen als Absolutwerte in Pixel zu speichern, ist unter diesen Verhältnissen nicht brauchbar. Beim Abspeichern der Werte im Programm wird somit der Absolutwert zuerst durch die aktuelle Bildschirmbreite oder Höhe in Pixel dividiert und beim Auslesen wieder mit dem aktuellen Bildschirmformat multipliziert.

8.3. Aufbau der Teilliste

Die Teilliste besteht aus diversen Daten im Stringformat, welche aus einer vorgefertigten Excel-Tabelle der Firma importiert wurden. Die Tabelle ist dabei mit firmeninternen Daten befüllt, die allerdings oft für die eigentliche Anwendung unbedeutend sind.

Bezeichnung	Datentyp	Beschreibung
Material	nvarchar(MAX)	Bezeichnung des Materials
Materialnummer	nvarchar(MAX)	Der Zahlencode des Material nach denen mit Hilfe des Barcodescanner gesucht werden kann
Werk	nvarchar(MAX)	-
LOrt	nvarchar(MAX)	-
MArt	nvarchar(MAX)	-
Materialart	nvarchar(MAX)	-
Warengruppe	nvarchar(MAX)	Bezeichnungscode für die Warengruppe
Warengruppe1	nvarchar(MAX)	Name der Warengruppe
Platz	nvarchar(MAX)	Ist die Bezeichnung des Lagerplatzes, die sich aus Regalbezeichnung, Nummer der Ebenen & Fachnummer zusammensetzt
Bestand	nvarchar(MAX)	-
BME	nvarchar(MAX)	-

8.4. Aufbau der Gebäude-Tabelle

Bezeichnung	Datentyp	Beschreibung
g_id	int	ID des Gebäudes - Primärschlüssel
g_name	nvarchar(MAX)	Bezeichnung des Gebäudes
g_positionX	decimal(18, 12)	Ist der X-Abstand zwischen dem linken Containerrand und der linken Rahmenseite als Faktor der gesamten Bildschirmbreite
g_positionY	decimal(18, 12)	Ist der Y-Abstand zwischen dem oberen Containerrand und der oberen Rahmenseite als Faktor der gesamten Bildschirmhöhe
g_größeX	decimal(18, 12)	Ist die Breite des Rahmens als Faktor der gesamten Bildschirmbreite
g_größeY	decimal(18, 12)	Ist die Höhe des Rahmens als Faktor der gesamten Bildschirmhöhe
g_hintergrund	nvarchar(MAX)	Ist der Verzeichnispfad der Bilddatei die als Hintergrundbild eingerichtet ist

8.5. Aufbau der Raum-Tabelle

Bezeichnung	Datentyp	Beschreibung
ra_id	int	ID des Raums - Primärschlüssel
ra_name	nvarchar(MAX)	Bezeichnung des Raums
ra_gebäude	nvarchar(MAX)	Bezeichnung des Gebäudes, welchem der Raum zugeordnet ist
ra_gebäude_id	int	ID des Gebäudes - Fremdschlüssel
ra_größeX	decimal(18, 12)	Die Breite des Raums als Faktor der gesamten Bildschirmbreite
ra_größeY	decimal(18, 12)	Die Höhe des Raums als Faktor der gesamten Bildschirmhöhe
ra_rastergrößeX	decimal(18, 12)	Die Breite einer Rasterfläche als Faktor der gesamten Bildschirmbreite
ra_rastergrößeY	decimal(18, 12)	Die Höhe einer Rasterfläche als Faktor der gesamten Bildschirmhöhe
ra_regal_größeX	int	Die Standardregalbreite als Anzahl von Rasterflächen
ra_regal_größeY	int	Die Standardregalhöhe als Anzahl von Rasterflächen
ra_hintergrund	nvarchar(MAX)	Ist der Verzeichnispfad der Bilddatei die als Hintergrundbild eingerichtet ist

8.6. Aufbau der Regal-Tabelle

Bezeichnung	Datentyp	Beschreibung
re_id	int	ID des Regals - Primärschlüssel
re_name	nvarchar(MAX)	Bezeichnung des Regales
re_raum	nvarchar(MAX)	Bezeichnung des Raums in dem sich das Regal befindet
re_raum_id	int	ID des Raums - Fremdschlüssel
re_textausrichtung	bit	Boolean welcher Beschreibt ob das Regal senkrecht oder waagrecht ausgerichtet ist. 0 (false) → waagrecht 1 (true) → senkrecht
re_positionX	decimal(18, 12)	Der X-Abstand zwischen dem linken Raumrand und der linken Regalseite als Faktor der Bildschirmbreite
re_positionY	decimal(18, 12)	Der Y-Abstand zwischen dem oberen Raumrand und der oberen Regalseite als Faktor der Bildschirmbreite
re_größeX	decimal(18, 12)	Breite des Regales als Faktor der Bildschirmbreite
re_größeY	decimal(18, 12)	Höhe des Regales als Faktor der Bildschirmhöhe
re_ebenen	int	Anzahl der Ebenen des Regales
re_fächer	int	Anzahl der horizontalen Fächer des Ragles

9. Programmtechnische Ausführungen

9.1. Realisierung: INI-File

In folgender Abbildung 9.1-1 ist ein Ausschnitt des Programmcodes zu erkennen, welches den Arbeitsschritt des Imports der INI-File, sowie der Datenzuweisung der Variablen zeigt.

```
/// <summary> Importier die ini file und weist die Daten den Variablen zu
private string Lade_ini_file()
{
    try
    {
        var MyIni = new IniFile(ini_file_path);
        string servername = MyIni.Read("MS_SQL_Server_Name", "Server");
        string database = MyIni.Read("MS_SQL_Database", "Server");
        string user = MyIni.Read("MS_SQL_User", "Server");
        string psw = MyIni.Read("MS_SQL_passwort", "Server");

        string connectionString = "Server=" + servername + "; Database= "
        return connectionString;
    }
    catch
    {
        return null;
    }
}
```

Abb. 9.1-1: INI-File

9.2. Realisierung: App.config

In folgendem Abschnitt ist ein Codeausschnitt der appSettings dieser Visualisierung zu sehen. Der Key ist immer die Bezeichnung der Einstellung, wie eine Variable, und Value bestimmt den Wert des Keys. Beispielweise ist der erste Key das Verzeichnis der INI-File, während der zweite Key einen String mit den Namen der sichtbaren Spalten enthält.

```
<appSettings>
  <add key="ini_file_verzeichnis" value="C:\Users\noah-\Documents\Schule\MEWA\einstellungen.ini" />
  <add key="sichtbare_spalten" value="Material, Materialnummer, Werk, Warengruppe, Warengruppe1, Platz" />
  <add key="watchdog_wartezeit" value="180"/>
</appSettings>
```

Abb. 9.2-1: appSettings

9.2.1. Schreiben in die App.config

Die folgende Abbildung 9.2.1-1 zeigt den Programmcode um das Schreiben auf die App.config Datei zu ermöglichen.

```
//Aktualisiert die App.Config Datei
var config = ConfigurationManager.OpenExeConfiguration(ConfigurationUserLevel.None);
var appSettingssection = config.GetSection("appSettings");
appSettingssection.CurrentConfiguration.AppSettings.Settings["ini_file_verzeichnis"].Value = ini_file_path;
appSettingssection.CurrentConfiguration.AppSettings.Settings["sichtbare_spalten"].Value = dargestellte_spalten;
appSettingssection.CurrentConfiguration.AppSettings.Settings["watchdog_wartezeit"].Value = watchdog_wartezeit.ToString();
config.Save();
ConfigurationManager.RefreshSection("appSettings");
```

Abb. 9.2.1-1: Schreiben in die App.config

9.2.2. Lesen aus der App.config

Wie das Lesen aus der App.config ermöglicht wird, zeigt folgender Ausschnitt aus dem Programmcode. Mithilfe des Konfigurationsmanagers „ConfigurationMannager“ kann auf die jeweiligen Einstellungen zugegriffen und anschließend abgerufen werden.

```
public string ini_file_path = ConfigurationManager.AppSettings["ini_file_verzeichnis"];
public string dargestellte_spalten = ConfigurationManager.AppSettings["sichtbare_spalten"];
public int watchdog_wartezeit = Int32.Parse(ConfigurationManager.AppSettings["watchdog_wartezeit"]);
```

Abb. 9.2.2-1: Lesen aus der App.config

9.3. Such-Funktion

```
StringBuilder suchstring = new StringBuilder();
//Thread.BeginThreadAffinity();
foreach (DataColumn column in Teilliste.Tables[0].Columns)
{
    if (dargestellte_spalten.Contains(column.ColumnName))
    {
        suchstring.AppendFormat("{0} Like '%{1}%' OR ", column.ColumnName, search_word);
    }
}
suchstring.Remove(suchstring.Length - 3, 3);

DataRow[] row_findings = Teilliste.Tables[0].Select(suchstring.ToString());
foreach (DataRow row in row_findings)
{
    Findings.ImportRow(row);
}
```

Abb. 9.3-1: Suchfunktion

Die selbst algorithmisch konstruierte Suchfunktion hat eine Vielzahl von Aufgaben. Zuerst wird der Suchstring aus den Spalten gebaut und die Rows (Reihen), in denen Zellen einen Teil des Strings oder gar das exakte Suchwort befinden, werden herausgefiltert. Die gefundenen Reihen werden anschließend in die Ergebnis-Table eingetragen.

9.4. Lesen aus der Datenbank

Um das Lesen aus der Datenbank zu ermöglichen wird der `connectionString` aus dem INI-File geladen. Anschließend wird die SQL Connection initialisiert, der `connectionString` dient dabei als Parameter. Nach Initialisierung der Verbindung wird der SQL-Befehl definiert und die `SqlDataAdapter`-Klasse mit den beiden Parametern initialisiert. Der `SqlDataAdapter` bereitet die Ergebnisse als fertigen `DataTable` und generiert die dazugehörigen SQL-Befehle (INSERT, DELETE, UPDATE) um mit den Daten arbeiten zu können.

```
DataSet lokation = new DataSet();
string connectionString = Lade_ini_file();
using (var conn = new SqlConnection(connectionString))
{
    //importiert den Namen des gesuchten Raumes
    string command = "SELECT re_raum_id FROM tbl_nb_Regale WHERE re_name = '"+ gesuchtes_regal

    using (var cmd = new SqlCommand(command, conn))
    {
        SqlDataAdapter adapt = new SqlDataAdapter(cmd);

        conn.Open();
        adapt.Fill(lokation, "gesuchter_raum");
        conn.Close();
    }
}
```

Abb. 9.4-1: Lesen aus der DB

9.5. Schreiben in die Datenbank

Das Schreiben in die Datenbank funktioniert ähnlich wie das Lesen. Der Unterschied besteht darin, dass dem `SqlDataAdapter` ein `SqlCommandBuilder` zugeordnet wird. Sollte der Adapter keinen passenden SQL-Befehl haben, sorgt die Klasse dafür, dass ein neuer SQL-Befehl (INSERT, DELETE, UPDATE) generiert wird. Der Block wird über `Update` aufgerufen. Dieser UPDATE-Befehl erhält als Vorlage einen Table (`serverdaten.Tables[0]`) und gleicht diesen mit dem Table in der Datenbank ab. Dabei hat jede Reihe einen spezifischen Status, „normal“ für keinen Befehl, „geändert“ für einen UPDATE-Befehl, „neu“ für einen INSERT-Befehl und „gelöscht“ für einen DELETE-Befehl. Je nach Statusbezeichnung wird für die Reihe ein eigener SQL-Befehl generiert, welcher anschließend an die Datenbank weitergesendet wird.

```
using (SqlConnection conn = new SqlConnection(connectionString))
{
    var cmd = "SELECT * from tbl_nb_Gebäude";
    SqlDataAdapter adapt = new SqlDataAdapter(cmd, conn);
    new SqlCommandBuilder(adapt);

    conn.Open();
    adapt.Update(serverdaten.Tables[0]);
    conn.Close();
}
```

Abb. 9.5-1: Schreiben in die Datenbank

9.6. Klassenbeschreibung

Zur Beschreibung und Abbildung der verschiedenen Ansichten und Objekten im Lager wurden eigene Klassen angelegt die das Erstellen der Ansichten erleichtern sollen indem bereits gegebene grafische Steuerelemente angepasst wurden.

9.6.1. Regal

Die Regal-Klasse hat ihr grafisches Aussehen von der Windows.Forms.Button-Klasse vererbt. Die Regale werden dynamisch der Teileliste generiert und einer Liste zugeordnet, welche die Regale für den Benutzer zur Verfügung stellt.

```
private void gen_regale()
{
    string[] Regale = new string[Suche.sTeilliste.Rows.Count];
    int a = 0;
    for (int i = 0; i < Suche.sTeilliste.Rows.Count; i++)
    {
        if (Suche.sTeilliste.Rows[i].ItemArray[8].ToString().Length >= 6)
        {
            if (!Regale.Contains(Suche.sTeilliste.Rows[i].ItemArray[8].ToString().Substring(0, 6)))
            {
                Regale[a] = Suche.sTeilliste.Rows[i].ItemArray[8].ToString().Substring(0, 6);
                a++;
            }
        }
    }
    Array.Resize(ref Regale, a);
}
```

Abb. 9.6.1-1: Regal-Klasse

Die Regale können dann durch den Drag&Drop Vorgang einem Raum zugeordnet werden. Innerhalb des Raums sind die Regale durch Benutzereingaben manipulierbar. Diese Manipulationen werden durch diverse Key- oder Mouse-Event-Handler abgearbeitet.

```
MouseDown += new System.Windows.Forms.MouseEventHandler(this.Regal_MouseDown);
MouseMove += new System.Windows.Forms.MouseEventHandler(this.Regal_MouseMove);
MouseUp += new System.Windows.Forms.MouseEventHandler(this.Regal_MouseUp);
MouseDoubleClick += new System.Windows.Forms.MouseEventHandler(this.Regal_DoubleClick);
KeyUp += new System.Windows.Forms.KeyEventHandler(this.Regal_KeyUp);
KeyDown += new System.Windows.Forms.KeyEventHandler(this.Regal_KeyDown);
```

Abb. 9.6.1-2: Event-Handler

Dabei startet das MouseDown-Event den Vorgang zum Verschieben des Regals, während das MouseMove-Event für den Verschiebevorgang zuständig ist. Das MouseUp-Event sorgt anschließend für die Beendigung des Vorgangs. Zusätzlich wird die Platzierung des Regales an das Raster des Raumes angepasst. Bei Auslösen des MouseDoubleClick-Events öffnet sich

das Bearbeitungsfenster für das fokussierte Regal. Das KeyUp-Event sorgt für die Löschung des aktiven Regals, sollte es durch die Entf-Taste aufgerufen werden. Während das KeyDown-Event für die Verschiebung des derzeitigen aktiven Regals verantwortlich ist, je nachdem, welche Pfeiltaste dieses Event auslöst.

Des Weiteren wurde die OnPaint-Methode der Klasse so modifiziert, dass der Text in der Methode um 90° gedreht wird, wenn die Textausrichtungsvariable des Regales dem Wahrheitswert TRUE entspricht.

```
if (textausrichtung)
{
    System.Drawing.StringFormat stringFormat = new System.Drawing.StringFormat();
    stringFormat.Alignment = System.Drawing.StringAlignment.Center;
    stringFormat.LineAlignment = System.Drawing.StringAlignment.Center;
    stringFormat.FormatFlags = System.Drawing.StringFormatFlags.FitBlackBox;
    Text = null;

    pevent.Graphics.TranslateTransform((Width - Height) / 2, Width - ((Width - Height) / 2));
    pevent.Graphics.RotateTransform(270);
    pevent.Graphics.DrawString(Name, Font, System.Drawing.Brushes.Black, new System.Drawing.Rectangle(-6, 0, Width+12, Height), stringFormat);
}
```

Abb. 9.6.1-3: Textausrichtung

9.6.2. Raum

Die Raum-Klasse hat ihr grafisches Layout von der tabPage-Klasse vererbt. Der Administrator legt über die Adminoberfläche die Räume mit ihren Eigenschaften eigenhändig an. Dabei wird nach den Eingaben des Benutzers ein Raster erstellt welches zur Unterteilung des Raumes dient. Die Visualisierung des Rasters erfolgt dabei durch folgende Methode.

```
private void raster_zeichnen(object sender, System.Windows.Forms.PaintEventArgs e)
{
    Raum raum_p = (Raum)sender;

    for (int x = 0; x <= raum_p.Width; x+=raum_p.rastergroßeX)
    {
        e.Graphics.DrawLine(Pens.Gray, x, 0, x, raum_p.Height);
    }
    for (int y = 0; y <= raum_p.Height; y+=raum_p.rastergroßeY)
    {
        e.Graphics.DrawLine(Pens.Black, 0, y, raum_p.Width, y);
    }
}
```

Abb. 9.6.2-1: Rastervisualisierung

Das Gedankenmodell hinter dem Raster besteht aus einem zweidimensionalen Array aus booleschen Werten, welches angibt, ob die Rasterflächen frei oder bereits mit einem Regal besetzt sind.

```
public void Platzraster_erstellen()
{
    raum.platz = new bool[(int)rastergroesseX_e.Value, (int)rastergroesseY_e.Value];
    for (int i = 0; i < rastergroesseX_e.Value; i++)
    {
        for (int e = 0; e < rastergroesseY_e.Value; e++)
        {
            raum.platz[i, e] = false;
        }
    }
}
```

Abb. 9.6.2-2: Rasterflächen

Sollte ein Regal im Raum platziert, gedreht oder verschoben werden, sollte immer eine Überprüfung erfolgen, ob dieses Feld laut dem Platzraster noch frei ist. Im folgenden Programmausschnitt 9.6.2-3 ist diese Methode dargestellt.

```
public System.Drawing.Point Platz_Raster_Kontrolle()
{
    for (int x = 0; x < this.Width / Raum.rastergroesseX; x++)
    {
        for (int y = 0; y < this.Height / Raum.rastergroesseY; y++)
        {
            if ((Left / Raum.rastergroesseX) + x >= Raum.Width / Raum.rastergroesseX || (Top / Raum.rastergroesseY) + y >= Raum.Height / Raum.rastergroesseY || Left < 0 || Top < 0)
            {
                return alte_location;
            }
            else if (Raum.platz[(Left / Raum.rastergroesseX) + x, (Top / Raum.rastergroesseY) + y] == true)
            {
                return alte_location;
            }
        }
    }
}
```

Abb. 9.6.2-3: Feldüberprüfung

Sollte der Platz nicht frei sein, so wird das Regal auf seine alte Position zurückgesetzt und der Vorgang wird als ungültig abgestempelt. Sollte der Vorgang allerdings erfolgreich sein, so wird das Raster mit den entsprechenden Werten aktualisiert.

```
public void Platz_Raster_Aktualisieren(bool wert)
{
    for (int x = 0; x < this.Width / Raum.rastergroesseX; x++)
    {
        for (int y = 0; y < this.Height / Raum.rastergroesseY; y++)
        {
            Raum.platz[(Left / Raum.rastergroesseX) + x, (Top / Raum.rastergroesseY) + y] = wert;
        }
    }
}
```

Abb. 9.6.2-4: Aktualisierung

9.6.3. Gebäude

Die Gebäude-Klasse hat ihr grafisches Aussehen ebenfalls von der TabPage-Klasse vererbt, mit dem Unterschied, dass sie nur um drei Mouse-Events erweitert wurde.

Das MouseDown-Event startet den Zeichenvorgang zum Zeichnen eines Rahmens. Der Start erfolgt nur, wenn die Strg-Taste gedrückt ist. Als Startposition des Rahmens wird die aktuelle Mausposition genommen.

```
private void Gebaeude_MouseDown(object sender, MouseEventArgs e)
{
    if (strg)
    {
        zeichnen = true;
        rahmen.zeichnen = false;
        achseX = e.X;
        achseY = e.Y;
        rahmen.Size = new Size(0, 0);
        rahmen.Location = new Point(achseX, achseY);
        this.Controls.Add(rahmen);
        rahmen.BringToFront();
    }
}
```

Abb. 9.6.3-1: Gebäude-MouseDownEvent

Beim anschließenden Bewegen der Maus wird das MouseMove-Event ausgelöst, welches die Differenz zwischen der Startposition und der aktuellen Mauszeigerposition als Größe des Rahmens festlegt.

```
private void Gebaeude_MouseMove(object sender, MouseEventArgs e)
{
    if (zeichnen)
    {
        rahmen.Width = e.X - rahmen.Left;
        rahmen.Height = e.Y - rahmen.Top;
    }
}
```

Abb. 9.6.3-3: Gebäude-MouseMoveEvent

Beim Loslassen der Maustaste wird der Zeichenvorgang beendet und der Rahmen wird zur Darstellung aktualisiert.

```
private void Gebaeude_MouseUp(object sender, MouseEventArgs e)
{
    if (zeichnen)
    {
        zeichnen = false;
        rahmen.zeichnen = true;
        rahmen.Refresh();
    }
}
```

Abb. 9.6.3-2: Gebäude-MouseUpEvent

9.6.4. Rahmen

Die Rahmen-Klasse ist eine vererbte PictureBox-Klasse, und dient zur Markierung des Gebäudes in einem Gebäudeobjekt. Die einzige Erweiterung ist eine Bearbeitung der OnPaint-Methode um aktiven Zeichnen, ein rotes Viereck am Rand des Rahmens zu versehen.

```
protected override void OnPaint(PaintEventArgs e)
{
    base.OnPaint(e); //Ist die standard Zeichenfunktion der PictureBox-Klasse
    if (zeichnen)
    {
        e.Graphics.DrawRectangle(new System.Drawing.Pen(System.Drawing.Color.Red, 5), 0, 0, this.Width - 5, this.Height - 5);
    }
}
```

Abb. 9.6.4-1: OnPaint-Methode

10. Handbuch

Dieses Handbuch dient zur Bedienung der Ersatzteillagervisualisierung für die Firma „MEWA Textil-Service GmbH“ und soll bei Fragen oder Unklarheiten Auskunft geben. Diese Dokumentation enthält essentielle Informationen von Benutzer- und Adminansicht. Von einfachem Bedienkonzept und Login, bis hin zur Anlegung und Bearbeitung von Räumen, Regalen und Fächern.

10.1. Benutzer

Bei Start des Programms erscheint die Ausgangsansicht der Visualisierung, welche in folgender Abbildung 10.1-1 sichtbar ist.

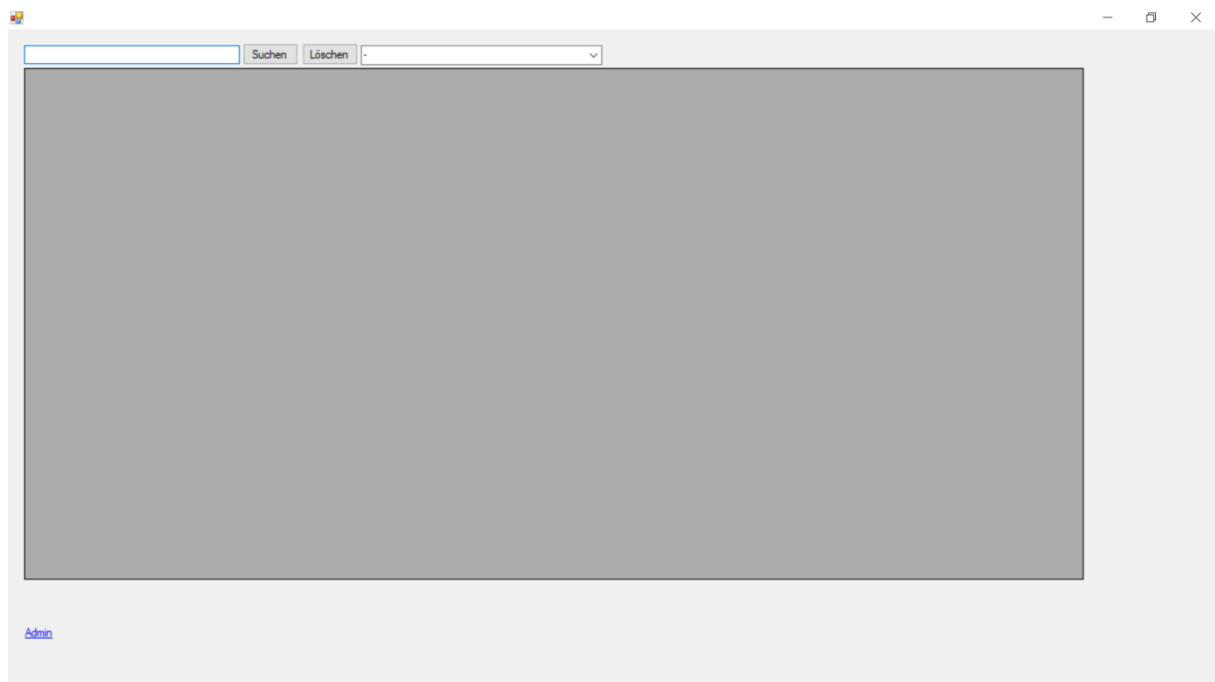


Abb. 10.1-1: Hauptansicht

Nun kann mithilfe eines Barcodescanners oder durch händische Eingabe in der Suchleiste nach dem erwünschten Ersatzteil gesucht werden. Durch Betätigung des „Suchen“ Buttons oder der Eingabetaste wird die Eingabe bestätigt und passende Suchergebnisse werden in Tabellenansicht, wie in Abbildung 10.1-2 ersichtlich, dargestellt.

Material	Materialnummer	Werk	LÖt	Warengruppe	Warengruppe1	Platz
000000110000000172	Elektromotor 830420150 1320/0.12	1045	9000	M0608	E-Motoren	FEISTMANTL
0000001100000002492	Elektromotor DU 1401/1 1415/2.5	1045	9000	M0608	E-Motoren	0902AF0302
0000001100000004741	Trommelmotor PT113X2 1260/0.3	1045	9000	M0608	E-Motoren	0902AF0202
0000001100000005695	Elektromotor 1LA7073-4 1370/0.37	1045	9000	M0608	E-Motoren	0902AA0202
0000001100000005748	Spindelantrieb 24VDC-2A	1045	9000	M0608	E-Motoren	0202AC03
0000001100000005782	Elektromotor FOL635/4 1340/0.12	1045	9000	M0608	E-Motoren	0902AF0201
0000001100000005790	Motor RF80L/2 2820/1.1	1045	9000	M0608	E-Motoren	0902AE0201
0000001100000005806	Elektromotor K21R132S2 2880/5.5	1045	9000	M0608	E-Motoren	
0000001100000005807	Elektromotor K21R132SX 2880/7.5	1045	9000	M0608	E-Motoren	0902AA0102
0000001100000005808	Elektromotor AMEE 90S 2850/1.5	1045	9000	M0608	E-Motoren	0902AA0102
0000001100000005819	Elektromotor K21R112M2 2900/4	1045	9000	M0608	E-Motoren	0902AA0102
0000001100000005820	Elektromotor K11R132S2 2860/5.5	1045	9000	M0608	E-Motoren	0902AF0303
0000001100000005870	Elektromotor LG4183AAA 1465/18.5	1045	9000	M0608	E-Motoren	
0000001100000005874	Motor FCMP160L 1470/15	1045	9000	M0608	E-Motoren	0902AE0301
0000001100000005883	Motor HGL132L 2600/34.3	1045	9000	M0608	E-Motoren	0902AD0303
0000001100000005985	Elektromotor 37F91229 1425/4	1045	9000	M0608	E-Motoren	0902AE0201
0000001100000006066	Elektromotor SCP160L-4	1045	9000	M0608	E-Motoren	
0000001100000006124	Elektromotor KDG2132S 2920/5.5	1045	9000	M0608	E-Motoren	0902AF0303
0000001100000006125	Elektromotor KTE1180M2 2920/22	1045	9000	M0608	E-Motoren	0902AF0301
0000001100000006138	Trommelmotor TM113825 0.37kW	1045	9000	M0608	E-Motoren	0902AF0202
0000001100000006167	Elektromotor 1ETEEF-71 2810/0.37	1045	9000	M0608	E-Motoren	0902AF0302
0000001100000006189	Elektromotor AY 132M4 1445/7.5	1045	9000	M0608	E-Motoren	0902AE0202
0000001100000006196	Elektromotor Q4 OED 11	1045	9000	M0608	E-Motoren	
0000001100000006218	Elektromotor Y2-180L6 960/15	1045	9000	M0608	E-Motoren	0903AA0302
0000001100000006229	Elektromotor KPER 90L2 1.85	1045	9000	M0608	E-Motoren	0902AF0302
0000001100000006252	Elektromotor 1LG4183-2 2945/22	1045	9000	M0608	E-Motoren	0902AE0302

Abb. 10.1-2: Suchergebnisse

Da bei großer Anzahl von Ersatzteilen ein unübersichtliches Suchergebnis sehr wahrscheinlich ist, kann mithilfe eines Filters die Tabellenansicht weiter eingegrenzt werden.

Material	Materialnummer
000000110000000015	Wellendichtring Viton/FPM
000000110000000021	Rundbürste ZZB PA 6.10
000000110000000025	Dichtungshalter
000000110000000027	Drehgelenk VA IG/AG 3/4"
000000110000000044	Schwimmerschal 240V AC/
000000110000000053	Turbokreislauf-A TUA-2f mi
000000110000000060	Zylinder CP3520 Ø37,6xØ
000000110000000061	Adapter weiblich CP3520 Ø3
000000110000000062	Adapter männlich CP3520 Ø
000000110000000063	Distanzstück +F CP3520 5
000000110000000064	Distanzstück D CP3520 Ø
000000110000000065	Ventilsatz CP3520
000000110000000066	Ölfalz CP3520 30x50x4 Fil
000000110000000067	Scheibe ND-Dic CP3520 Ø
000000110000000068	Adapter ND-Dic CP3520 Ø
000000110000000087	Druckwächter LGW 50 A2
000000110000000096	Führungsschienen Nr. 3 Eck
000000110000000102	Kettenschloss 12B-1 VA 8
000000110000000111	...

search

Antriebstechnik elektrisch: Frequenzumrichter.
 Baueinheit von Pumpe und Motor
 Dampftechnik (Ventile, Kondensatableiter, Schieber)
 Dichtungen: O-Ringe, Puffer, Wellendichtringe, et
 Elektr.Bauteile: Netzgeräte, Elektronik, Schaltkrei
 Elektr.Bauteile: Sensorik, Regeltechnik, Thyristors
 E-Motoren
 Ersatzteile Lüfter (Keine Normteile)
 Getriebe
 Getriebe- Motoren
 Heizleiter, Schaltschränke, Transform., Schienen,
 Hydraulik/Fluidtechnik: (Zylinder, Ventile, Kugelhä
 HZ Nichteisenmetalle, Pressteile, Dreh- und Fräste
 HZ Stahl/Eisen/Legierungen, Pressteile, Dreh- und
 Lüfter
 Norm-/Standardteile : Führungen
 Norm-/Standardteile : Schrauben, Nieten, S-Ring,
 Norm-/Standardteile : Sonst.
 Norm-/Standardteile : Wälzlager
 Pneum. Antriebe, Hydraulische Antriebe
 Pneumatik:(Zylinder, Ventile, Messeinrichtungen)
 Pumpen
 Pumpen (Keine Normteile)
 Schläuche : (Wasser-, Well-, Plastik-, Gummischla
 Schmier- und Betriebsstoffe
 Sicherheitskennz.: Schilder, Etiketten, Hautschutz
 Sicherungen, Relais, Schalter, Schütze, Leuchtm
 Sonstige Antriebstechnik
 Sonstige Elektroartikel

Abb. 10.1-3: Filter

Durch Doppelklick auf das gewünschte Ersatzteil wird der Benutzer automatisch zur Gebäudeansicht weitergeleitet. Das Gebäude in dem sich das gewünschte Objekt befindet ist rot markiert. Im linken unteren Teil der Ansicht wird ein Pfad dargestellt, dieser dient zur weiteren Navigation durch die Visualisierung, um beispielsweise einen Schritt zurückzusetzen beziehungsweise nochmals aufzurufen, wie in Abbildung 10.1-4 dargestellt.



Abb. 10.1-4: Gebäudeansicht

Nachdem sich der Benutzer versichert hat, in welchem Gebäude sich das Ersatzteil befindet, öffnet sich, nach einem Mausklick auf das rot markierte Feld, die Rauman sicht.

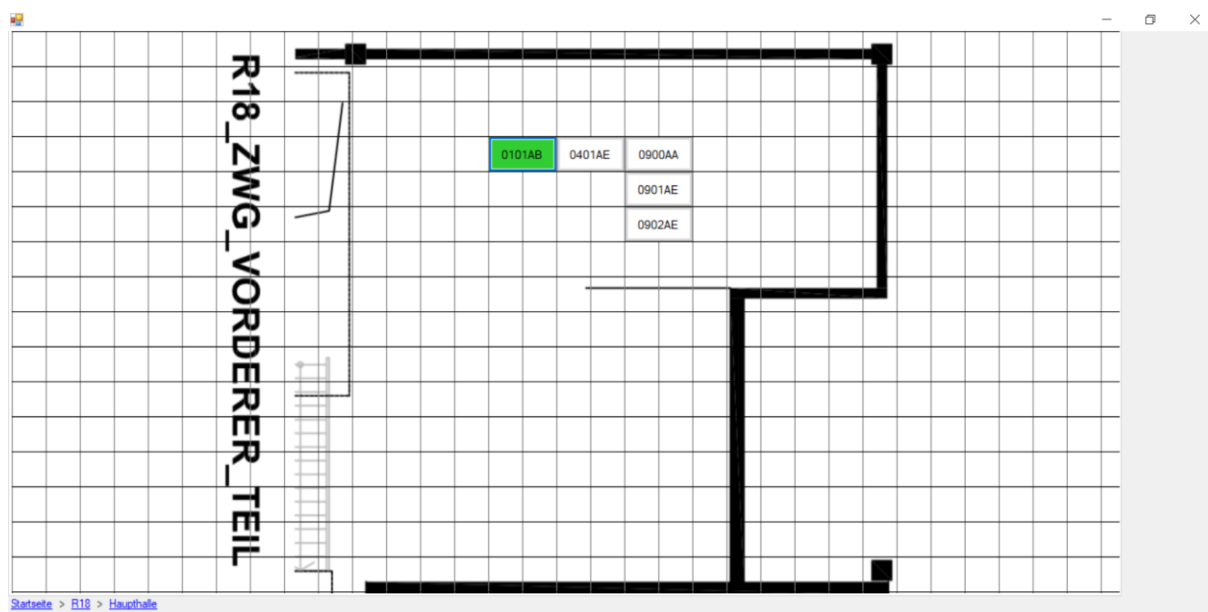


Abb. 10.1-5: Rauman sicht

Wie in Abbildung 10.1-5 zu erkennen ist, zeigt diese Ansicht einen genauen Plan des Raumes mit den allen darin aufgestellten Regalen an. Das Regal, in dem sich das gesuchte Ersatzteil befindet ist hervorgehoben und grün markiert dargestellt. Zusätzlich ist zu erwähnen, dass sich der Navigationspfad um einen neuen Schritt erweitert hat, um eine gezielte und dauerhafte Navigation durch die gesamte Visualisierung gewährleisten zu können.

Um die Suche des Ersatzteiles weiter zu vereinfachen, wird durch einen erneuten Mausklick auf das hervorgehobene Regal eine Regalansicht aufgerufen. Wie in der folgenden Abbildung 10.1-6 dargestellt, zeigt diese Ansicht den Innenraum des Regales gegliedert dar. Der Bereich, in dem sich das gesuchte Objekt laut Ersatzteilliste befinden soll, wird grün hinterlegt angezeigt.

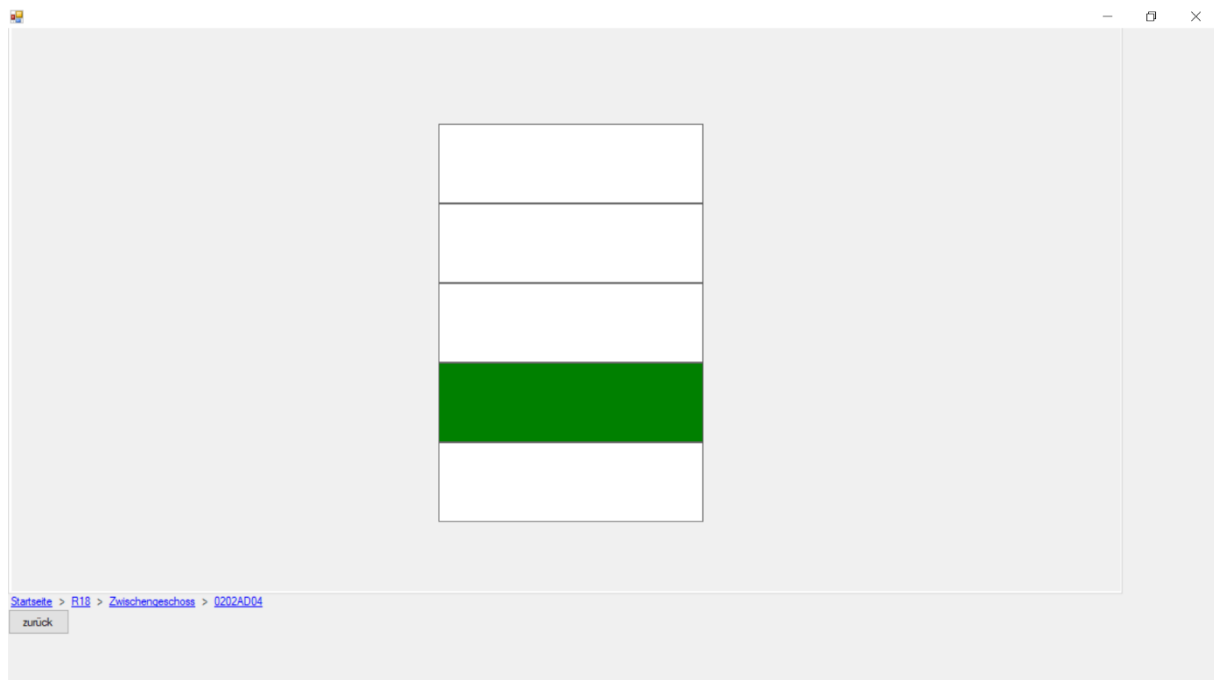


Abb. 10.1-6: Regalansicht

Abschließend kann mithilfe des Navigationspfades, wie eine vergrößerte Ansicht bei Abbildung 10.1-7 zeigt, jeder einzelne Arbeitsschritt durch einfaches Klicken nochmals abgerufen werden, insofern Gebäudeort, Raumnummer oder Regalplatzierung nicht mehr klar ist oder eine zusätzliche Überprüfung notwendig beziehungsweise gewünscht ist.



Abb. 10.1-7: Navigationspfad

10.2. Admin

Um Änderungen an Gebäuden, Räumen, Regalen, Fächern oder Lagern vorzunehmen, wurde ein Administrationsfenster entwickelt. Durch die Betätigung des „Admin“ Buttons, der in Abbildung 10.1-1 am linken unteren Rand erkennbar ist, wird ein Anmeldefenster aufgerufen (siehe Abbildung 10.2-1).

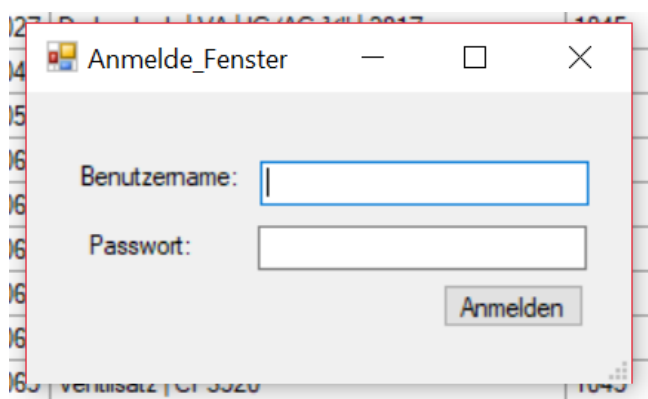


Abb. 10.2-1: Anmeldefenster

Mittels Eingabe des Benutzernamen und des Passwortes, die in einer externen angelegten INI-File definiert sind, öffnet sich die Adminansicht der Visualisierung. In der folgenden Ansicht werden alle Einstellungen des Administrators getroffen. Beispielsweise welche Tabellenspalten dargestellt werden sollen, Angabe des Dateipfades der INI-File oder die Zeiteinstellung des Watchdogs.

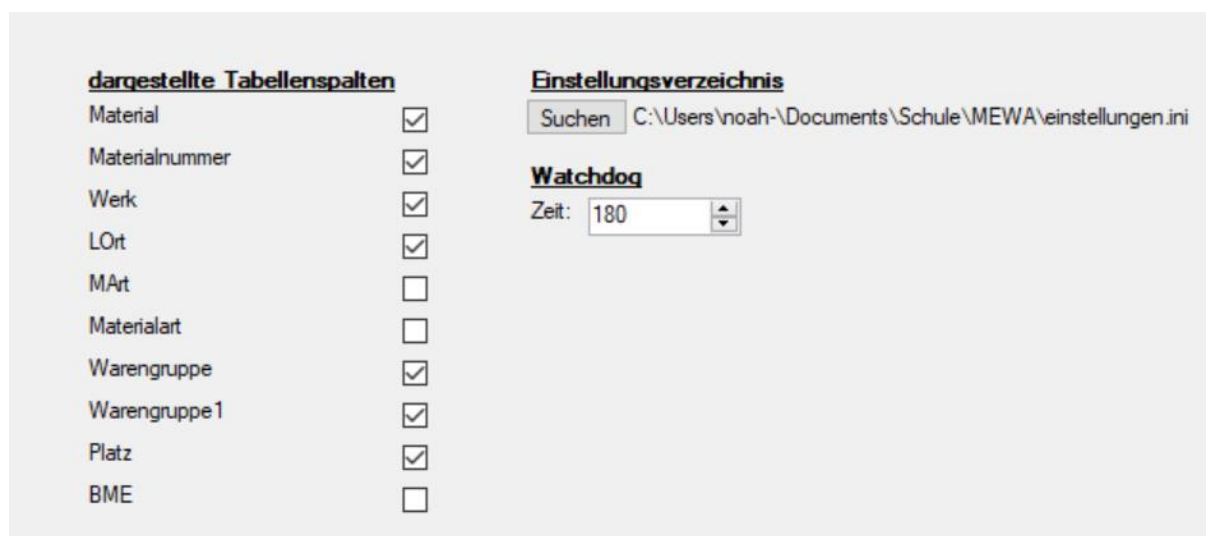


Abb. 10.2-2: AdminEinstellungen

Durch Bestätigung der gewählten Einstellungen wird automatisch das nächste Fenster geöffnet. Hier kann entschieden werden, ob Gebäude oder Räume bearbeitet, hinzugefügt oder gar gelöscht werden.



Abb. 10.2-3: Admin - Gebäude

Mit dem Prinzip „Drag & Drop“ können Regale aus der Liste in die Raumansicht gezogen werden. So ist es dem Administrator möglich, jedes Regal individuell zu verschieben, zu verdrehen oder zu entfernen, wie in folgender Abbildung 10.2-4 dargestellt.

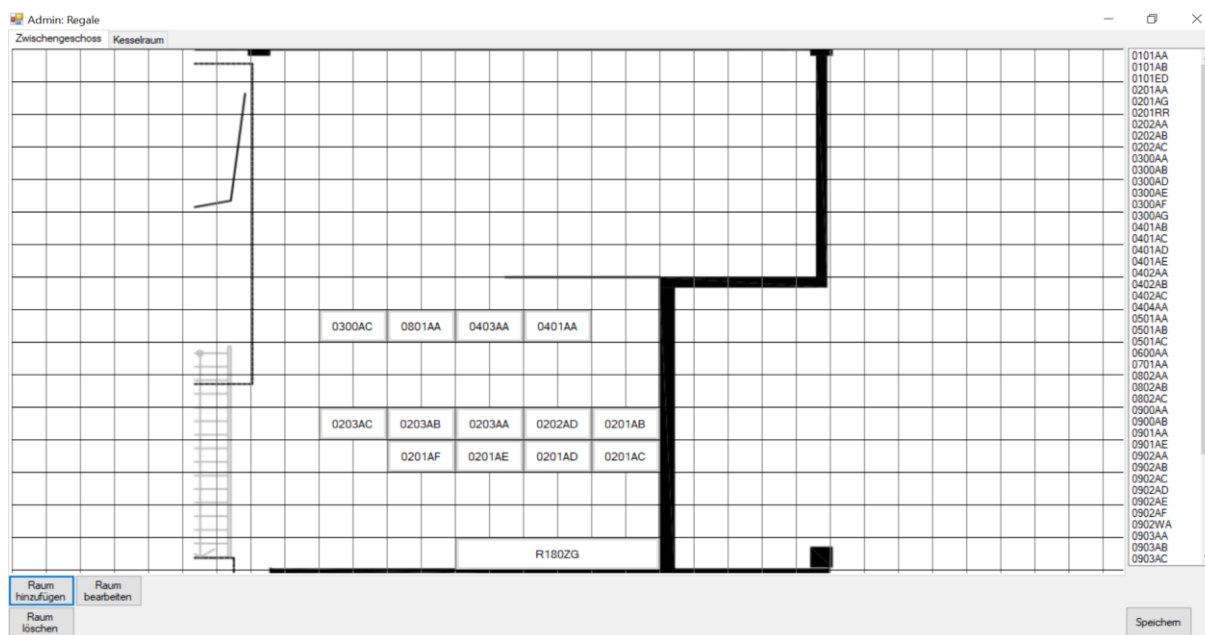


Abb. 10.2-4: Admin-Räume

Durch Buttons in Verbindung mit genau definierten Klickevents wird eine einfache Administration der spezifischen Raum- und Gebäudeanpassung ermöglicht. Wird zum Beispiel durch Betätigung des „Raum hinzufügen“ Buttons ein weiteres Fenster für spezifischere Angaben geöffnet (siehe Abbildung 10.2-5).

Um ungeplanten Aktionen vorbeugen zu können wird nach jeder Bearbeitung eine Sicherheitsmeldung ausgegeben. Erst nach Bestätigung dieser Abfrage (siehe Abbildung 10.2-6) wird die Änderung durchgeführt.

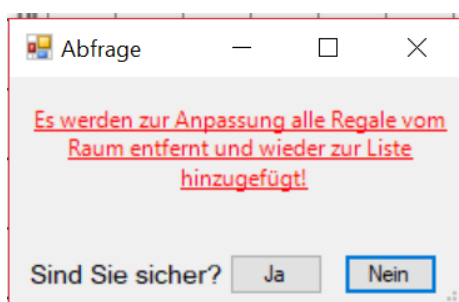


Abb. 10.2-6: Abfrage

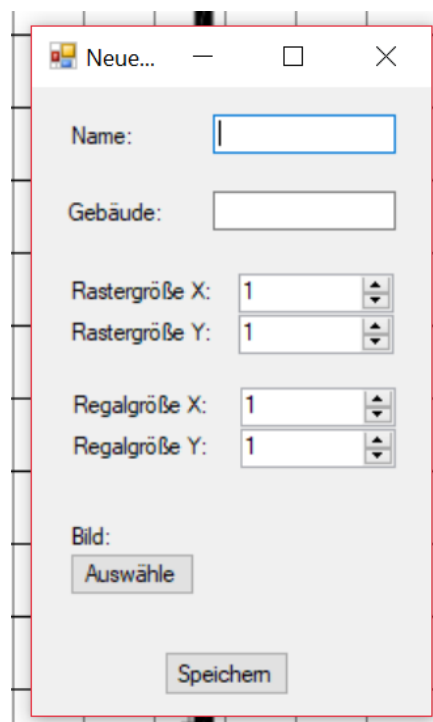


Abb. 10.2-5: Neuer Raum

11. Evaluation

Zum Abschluss erfolgt die Gegenüberstellung der ursprünglich geplanten Diplomarbeit und der schlussendlich realisierten Version, sowie Ablauf der Übergabe an die Firma „MEWA Textil-Service GmbH“.

11.1. Gegenüberstellung

Bei der Gegenüberstellung des Endprojekts mit dem anfänglichen Konzept, wie in Abbildung 5.3-1 und 5.3-2 ersichtlich, fällt auf, dass alle anfänglich angesprochenen Arbeitspakete und Forderungen erfüllt wurden. Zusätzlich wurden einige Funktionen wie zum Beispiel der Suchfilter realisiert um eine bessere Übersicht zu gewährleisten.

11.2. Übergabe an die Firma

Die Übergabe des Projektes an die Firma kann zum Zeitpunkt der Abgabe dieser Diplomarbeit noch nicht beurteilt werden, da diese erst aus Zeitmangel im Anschluss erfolgt. Grundsätzlich sind der Auftraggeber „MEWA Textil-Service GmbH“, sowie unsere Projektleiter Herr Bandat und Herr Blüml mit der Realisierung der Visualisierung äußerst zufrieden. Dank Testläufen ist sicher, dass alle Funktionen reibungslos ablaufen und einem reibungslosen Einsatz der Visualisierung des Ersatzteillagers nichts im Wege steht.

11.3. Persönliche Erfahrungen

Abschließend fehlen nur noch unsere persönlichen Erfahrungen und Eindrücke die sich im Laufe dieser Diplomarbeit gebildet haben. Wir, Sven Nussbaumer und Noah Pichler, möchten uns abschließend nochmals für die Kooperation mit der Firma „MEWA Textil-Service GmbH“ bedanken. Dank vielen Recherchen konnten wir zahlreiche Informationen und Erfahrungen in verschiedensten Themengebieten sammeln und unser technisches Wissen weiter steigern. Doch auch unsere Grundkenntnissen in C# und SQL, bis hin zum relationalen Datenbankmodell, die wir uns in unserer Lehrzeit in der HTL Eisenstadt angeeignet haben, waren eine große Hilfe bei der Fertigstellung dieser Diplomarbeit.

Die Arbeit an dieser Diplomarbeit und vor allem die Zusammenarbeit mit der Firma haben uns eine große Freude bereitet. Auch gemeinsame Sitzungen und Besprechungen liefen

reibungslos ab, wodurch wir ohne größeren Zeitdruck und ohne jegliche Auseinandersetzung die Diplomarbeit im geforderten Rahmen fertigstellen konnten. Allerdings kann nicht gesagt werden, dass die Diplomarbeit keine sehr zeitaufwändige und beanspruchungsvolle Aufgabe war. Im Gegenteil, der Großteil des Projekts entstand hauptsächlich in unserer Freizeit.

Abschließend möchten wir uns nochmals bei allen an dieser Diplomarbeit beschäftigten Personen für ihre Unterstützung bedanken. Ganz besonderer Dank gilt vor allem Herrn Bandat und Herrn Blüml für die problemlose Zusammenarbeit und unserem Betreuungslehrer Herrn Lang, der bei auftretenden Fragen immer Rat und Lösungsvorschläge parat hatte. Ohne diese drei Herrschaften wäre diese Diplomarbeit niemals zustande gekommen und wir hätten keine Gelegenheit bekommen unser Wissen unter Beweis zu stellen.

*Vielen **D**ank!*

12. Tätigkeitsnachweis

12.1. Sven Nussbaumer

Datum	Tätigkeit	Dauer in min	Dauer in h
21.06.2018	Besprechung - Aufteilung der Arbeitsgebiete	60	1
28.06.2018	Kickoff mit der Firma; Besprechung d. Anforderungen & Ziele; Besichtigung der Firma	300	5
29.06.2018	Brainstorming und Teambesprechung für möglichen Programmaufbau	120	2
30.06.2018	Konzeptentwicklung; Arbeitspaket 1	180	3
02.08.2018	Recherche: SQL Server 2016 Express	30	0,5
03.08.2018	Installation & Konfiguration SQL Server 2016 Express	90	1,5
06.08.2018	Besprechung mit MEWA; Konzeptbesprechung	120	2
07.08.2018	Einrichten des Servers	120	2
08.08.2018	Importieren der Teileliste in den Server	30	0,5
09.08.2018	Erstellung & Import der Ini-File; Arbeitspaket 2	60	1
10.08.2018	Start Programmierung Oberfläche zur Visualisierung	120	2
13.08.2018	Recherche: ConnectionStrings; Server/Anwendung	60	1
13.08.2018	Erstellen des ConnectionStrings; Verbindungsaufbau zwischen Server und Anwendung	30	0,5
15.08.2018	Besprechung mit MEWA	60	1
16.08.2018	Recherche: Datenbankaufbau, SQL und Datenbanken	240	4
17.08.2018	Verknüpfung von Datenbank und Oberfläche; Arbeitspaket 3	360	6
18.08.2018	Verknüpfung von Datenbank und Oberfläche; Arbeitspaket 3	240	4
12.09.2018	MA-Liste automatisch einlesen	120	2
17.09.2018	Besprechung mit MEWA		
13.09.2018	MA-Liste automatisch einlesen	180	3
26.09.2018	Recherche: Datenbanken & Tabellen	120	2
27.09.2018	Erstellung Tabellen für Gebäude, Räume & Regale	300	5
04.10.2018	Generieren der Regale aus der Teileliste, Debugging	240	4
05.10.2018	Erstellen der Regal-Tabelle im Server	180	3

06.10.2018	Erstellen des Raums, Platzieren der Regale im Raum	300	5
06.10.2018	Erstellen der Raumtabelle, Programmierung Datenbank	300	5
11.10.2018	Beprechung mit MEWA	60	1
11.10.2018	Anpassung der Regale	420	7
18.10.2018	Grafische Überarbeitung	180	3
25.10.2018	Fehler beheben, Debugging	300	5
02.11.2018	Kommunikation mit Datenbank anpassen	240	4
10.11.2018	Gebäudeasicht erstellen, Informieren über	300	5
24.11.2018	Programmoberfläche anpassen	120	2
29.11.2018	Besprechung mit MEWA	60	1
05.12.2018	Erstellen der restlichen Tabellen in der Datenbank	180	3
06.12.2018	Programmieren der Datenbankkommunikation	300	5
13.12.2018	Start der formellen Ausarbeitung und Dokumentation	360	6
26.12.2018	Ausarbeitung Dokumentation Diplomarbeit	180	3
27.12.2018	Ausarbeitung Dokumentation Diplomarbeit	120	2
16.01.2019	Vorbereiten auf die Besprechung	30	0,5
17.01.2019	Besprechung Mewa	60	1
17.01.2019	Anpassung des Programms, Dokumentieren	180	3
07.02.2019	Fehlerbehebung und Debugging	240	4
20.02.2019	Recherche: SQL Scripts	120	2
28.02.2019	Besprechung MEWA	60	1
02.03.2019	Anpassung der Hauptansicht	120	2
06.03.2019	Oberfläche anpassen	60	1
13.03.2019	Implementieren der App.Config-Datei	120	2
14.03.2019	Anpassung der Adminübersicht	30	0,5
14.03.2019	Anpassung der Einstellungsübersicht	60	1
14.03.2019	Anpassung der Gebäude- und Raumübersicht	240	4
15.03.2019	Dokumentieren und Ausarbeitung Diplomarbeit	360	6
16.03.2019	Dokumentieren und Ausarbeitung Diplomarbeit	540	9
23.03.2019	Finalisierung des Programmes	120	2
30.03.2019	Dokumentieren und Ausarbeitung Diplomarbeit	480	8
31.03.2019	Finalisierung des Diplomarbeitsdokumentation	600	10
Gesamt:	Anzahl der Daten: 56	10200	169,0

12.2. Noah Pichler

Datum	Tätigkeit	Dauer in min	Dauer in h
21.06.2018	Besprechung, Aufteilung Arbeitsgebiete	60	1
28.06.2018	Kickoff mit der Firma, Besprechung der Anforderungen und Ziele, Besichtigung der Firma	300	5
29.06.2018	Brainstorming und Ideen sammeln für möglichen Programmaufbau, Recherchieren	120	2
30.06.2018	Entwerfen des Aufbau des Programms	180	3
02.08.2018	Datenbankaufbau überlegen, allgemeines zu SQL und Datenbanken recherchieren	120	2
03.08.2018	SQL Server 2016 Express installieren und einrichten	90	1,5
06.08.2018	Besprechung mit MEWA, Besprechung des Konzeptes	120	2
07.08.2018	Einrichten des Server	120	2
08.08.2018	Importieren der Teileliste in den Server	30	0,5
09.08.2018	Erstellen der Ini-file und importieren der Ini-file in die Anwendung	60	1
13.08.2018	Erstellen der ConnectionString und testen des Verbindungsaufbau zwischen Server und Anwendung	30	0,5
14.08.2018	Ausprogrammieren der Suchfunktion	240	4
15.08.2018	Besprechung mit MEWA	60	1
16.08.2018	Ausprogrammieren der Filterfunktion, Einrichten des DataGridView	180	3
17.09.2018	Erstellen des Einstellungsfenster	270	4,5
20.09.2018	Besprechung mit MEWA	60	1
23.09.2018	Programmieren der Admin-Ansicht	300	5
24.09.2018	Programmieren der Admin-Ansicht	120	2
30.09.2018	Erstellen der Regal-Klassen, Informieren über Klassen in c#	240	4
04.10.2018	Generieren der Regale aus der Teileliste, Debugging	180	3
05.10.2018	Erstellen der Regal-Tabelle im Server	60	1
06.10.2018	Erstellen des Raums, Platzieren der Regale im Raum	300	5
06.10.2018	Erstellen der Raumtabelle, Programmierung Datenbank	300	5
11.10.2018	Besprechung mit MEWA	60	1
11.10.2018	Anpassung der Regale	300	5
14.10.2018	Einrichten eines Anmeldefensters	120	2
20.10.2018	Anpassung Anmeldefenster	120	2

18.10.2018	Grafische Überarbeitung	300	5
19.10.2018	Regale anpassbar machen, Dokumentieren	240	4
25.10.2018	Fehler beheben, Debugging	120	2
02.11.2018	Kommunikation mit Datenbank anpassen	120	2
10.11.2018	Gebäudeasicht erstellen, Informieren über	300	5
11.11.2018	Gebäude-Klasse anpassen	120	2
24.11.2018	Programmoberfläche anpassen	120	2
29.11.2018	Besprechung mit MEWA	60	1
05.12.2018	Erstellen der restlichen Tabellen in der Datenbank	60	1
06.12.2018	Programmieren der Datenbankkommunikation	120	2
08.12.2018	Erstellen der Fächer und Ebenen für die Regale	150	2,5
14.12.2018	Gebäudeansicht anpassen	120	2
05.01.2019	Adminansicht anpassen	180	3
16.01.2019	Vorbereiten auf die Besprechung	30	0,5
17.01.2019	Besprechung Mewa	60	1
19.01.2019	Anpassen des Programms, Dokumentieren	300	5
20.01.2019	Anpassung der Einstellungsübersicht	240	4
24.01.2019	Adminoberfläche anpassen und grafisch überarbeiten	120	2
27.01.2019	Erstellen der Benutzeransicht	300	5
30.01.2019	Anpassen der Benutzeransicht	200	3,3
02.02.2019	Anpassen der Benutzeransicht	180	3
09.02.2019	Adminansicht mit der Benutzeransicht verknüpfen	60	1
07.02.2019	Fehlerbehebung und Debugging	240	4
17.02.2019	Adminansicht anpassen, Dokumentieren	60	1
28.02.2019	Besprechung MEWA	60	1
01.03.2019	Austesten des Barcodescanner	30	0,5
02.03.2019	Anpassen der Hauptansicht	120	2
06.03.2019	Oberfläche anpassen	60	1
13.03.2019	Einstellungsübersicht anpassen, implementieren der App.Config-Datei	300	5
14.03.2019	Anpassen der Gebäude- und Raumübersicht	240	4
14.03.2019	Anpassen der Einstellungsübersicht	60	1
14.03.2019	Anpassen der Adminübersicht	30	0,5
17.03.2019	Dokumentieren und Ausarbeitung Diplomarbeit	120	2
20.03.2019	Dokumentieren und Ausarbeitung Diplomarbeit	120	2
23.03.2019	Finalisierung des Programmes	120	2
24.03.2019	Dokumentieren und Ausarbeitung Diplomarbeit	300	5
26.03.2019	Dokumentieren und Ausarbeitung Diplomarbeit	60	1
30.03.2019	Dokumentieren und Ausarbeitung Diplomarbeit	120	2
31.03.2019	Finalisierung des Diplomarbeitdokumentation	150	2,5
Gesamt:	Anzahl der Daten: 66	9830	162,8

13. Literatur- & Quellennachweis

- Entwicklerbuch: Microsoft SQL Server 2008 R2
- <https://www.der-wirtschaftsingenieur.de/index.php/programmiersprache-c-sharp/>
 - Stand: 30.03.2019
- <https://docs.microsoft.com/de-de/dotnet/csharp/tour-of-csharp/classes-and-objects>
 - Stand: 30.03.2019
- <https://de.wikipedia.org/wiki/Initialisierungsdatei>
 - Stand: 30.03.2019
- https://images.slideplayer.com/25/7615782/slides/slide_6.jpg
 - Stand: 30.03.2019
- <https://docplayer.org/docs-images/26/7417988/images/5-0.png>
 - Stand: 30.03.2019
- <https://stackoverflow.com/>
 - Stand: 30.03.2019

14. Abbildungsverzeichnis

Abb. 5.3-1: Benutzerkonzept	15
Abb. 5.3-2: Adminkonzept.....	16
Abb. 6.1-1: Datenbanksystem	18
Abb. 6.1.2-1: ADO.NET Architektur.....	20
Abb. 6.3.3-1: INI-File	25
Abb. 7-1: Installationsregeln	27
Abb. 7-2: Funktionsauswahl	27
Abb. 7-3: Instanzkonfiguration	28
Abb. 7-4: Datenbank Engine Konfiguration	28
Abb. 7-5: SQL-Server Import/Export-Assistent	29
Abb. 7-6: Import Datenquelle	29
Abb. 7-7: Programmcode	30
Abb. 7-8: Tabellenansicht	30
Abb. 8.1-1: ER-Diagramm	31
Abb. 9.1-1: INI-File	35
Abb. 9.2-1: appSettings	35
Abb. 9.2.1-1: Schreiben in die App.config	36
Abb. 9.2.2-1: Lesen aus der App.config.....	36
Abb. 9.3-1: Suchfunktion	36
Abb. 9.4-1: Lesen aus der DB	37
Abb. 9.5-1: Schreiben in die Datenbank	37
Abb. 9.6.1-1: Regal-Klasse	38
Abb. 9.6.1-2: Event-Handler	38
Abb. 9.6.1-3: Textausrichtung.....	39
Abb. 9.6.2-1: Rastervisualisierung.....	39
Abb. 9.6.2-2: Rasterflächen.....	40
Abb. 9.6.2-3: Feldüberprüfung.....	40
Abb. 9.6.2-4: Aktualisierung.....	40
Abb. 9.6.3-1: Gebäude-MouseDownEvent	41
Abb. 9.6.3-2: Gebäude-MouseUpEvent	41
Abb. 9.6.3-3: Gebäude-MouseMoveEvent	41
Abb. 9.6.4-1: OnPaint-Methode	41

Abb. 10.1-1: Hauptansicht	42
Abb. 10.1-2: Suchergebnisse.....	43
Abb. 10.1-3: Filter	43
Abb. 10.1-4: Gebäudeansicht	44
Abb. 10.1-5: Raumansicht.....	44
Abb. 10.1-6: Regalansicht	45
Abb. 10.1-7: Navigationspfad	45
Abb. 10.2-1: Anmeldefenster	46
Abb. 10.2-2: AdminEinstellungen	46
Abb. 10.2-3: Admin - Gebäude	47
Abb. 10.2-4: Admin-Räume.....	47
Abb. 10.2-5: Neuer Raum	48
Abb. 10.2-6: Abfrage	48

15. Anhang

15.1. Create-Scripts SQL Tables

15.1.1. tbl_nb_Gebäude

```
CREATE TABLE [dbo].[tbl_nb_Gebäude](
    [g_id] [int] IDENTITY(1,1) NOT NULL,
    [g_name] [nvarchar](max) NOT NULL,
    [g_positionX] [decimal](18, 12) NULL,
    [g_positionY] [decimal](18, 12) NULL,
    [g_größeX] [decimal](18, 12) NULL,
    [g_größeY] [decimal](18, 12) NULL,
    [g_hintergrund] [nvarchar](max) NULL,
    CONSTRAINT [PK_tbl_nb_Gebäude_1] PRIMARY KEY CLUSTERED
(
    [g_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

15.1.2. tbl_nb_Räume

```
CREATE TABLE [dbo].[tbl_nb_Räume](
    [ra_id] [int] IDENTITY(1,1) NOT NULL,
    [ra_name] [nvarchar](max) NOT NULL,
    [ra_gebäude] [nvarchar](max) NOT NULL,
    [ra_gebäude_id] [int] NULL,
    [ra_größeX] [decimal](18, 12) NULL,
    [ra_größeY] [decimal](18, 12) NULL,
    [ra_rastergrößeX] [decimal](18, 12) NULL,
    [ra_rastergrößeY] [decimal](18, 12) NULL,
    [ra_regal_größeX] [int] NULL,
    [ra_regal_größeY] [int] NULL,
    [ra_hintergrund] [nvarchar](max) NULL,
    CONSTRAINT [PK_tbl_nb_Räume_1] PRIMARY KEY CLUSTERED
(
    [ra_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

GO

```
ALTER TABLE [dbo].[tbl_nb_Räume] WITH CHECK ADD CONSTRAINT
[FK_tbl_nb_Räume_tbl_nb_Gebäude] FOREIGN KEY([ra_gebäude_id])
REFERENCES [dbo].[tbl_nb_Gebäude] ([g_id])
GO
```

```
ALTER TABLE [dbo].[tbl_nb_Räume] CHECK CONSTRAINT [FK_tbl_nb_Räume_tbl_nb_Gebäude]
GO
```

15.1.3. tbl_nb_Regale

```
CREATE TABLE [dbo].[tbl_nb_Regale](
    [re_id] [int] IDENTITY(1,1) NOT NULL,
    [re_name] [nvarchar](max) NOT NULL,
    [re_raum] [nvarchar](max) NOT NULL,
    [re_raum_id] [int] NULL,
    [re_textausrichtung] [bit] NULL,
    [re_positionX] [decimal](18, 12) NULL,
    [re_positionY] [decimal](18, 12) NULL,
    [re_größeX] [decimal](18, 12) NULL,
    [re_größeY] [decimal](18, 12) NULL,
    [re_ebenen] [int] NULL,
    [re_fächer] [int] NULL,
    CONSTRAINT [PK_tbl_nb_Regale_1] PRIMARY KEY CLUSTERED
(
    [re_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]

GO

ALTER TABLE [dbo].[tbl_nb_Regale] WITH CHECK ADD CONSTRAINT
[FK_tbl_nb_Regale_tbl_nb_Räume] FOREIGN KEY([re_raum_id])
REFERENCES [dbo].[tbl_nb_Räume] ([ra_id])
GO

ALTER TABLE [dbo].[tbl_nb_Regale] CHECK CONSTRAINT [FK_tbl_nb_Regale_tbl_nb_Räume]
GO
```

15.1.4. tbl_nb_teileliste

```
CREATE TABLE [dbo].[tbl_nb_teileliste](
    [Material] [nvarchar](max) NULL,
    [Materialnummer] [nvarchar](max) NULL,
    [Werk] [nvarchar](max) NULL,
    [LOrt] [nvarchar](max) NULL,
    [MArt] [nvarchar](max) NULL,
    [Materialart] [nvarchar](max) NULL,
    [Warengruppe] [nvarchar](max) NULL,
    [Warengruppe1] [nvarchar](max) NULL,
    [Platz] [nvarchar](max) NULL,
    [Bestand] [nvarchar](max) NULL,
    [BME] [nvarchar](max) NULL
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

15.2. CD-ROM

Die beigelegte CD enthält die gesamte digitale Ausarbeitung der Diplomarbeit. Programm, Scripts und digitale Dokumentation der Diplomarbeit sind darauf zu finden.