Desktop Purchasing System

**Hausarbeit**

des Studienganges TINF22

an der Dualen Hochschule Baden-Württemberg Ravensburg

von

Nicolas Koch

06.06.2024

Bearbeitungszeitraum 5 Wochen

Matrikelnummer, Kurs Matrikelnummer, Kurskürzel

Ausbildungsfirma Linde Engineering, Dresden

Gutachter der Dualen Hochschule Prof. Dr. Christoph Sandbrink

Erklärung

Ich versichere hiermit, dass ich meine Bachelorarbeit (bzw. Studien- und Projektarbeit) mit dem Thema: (…) selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.\*

\* falls beide Fassungen gefordert sind

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| Ort |  | Datum |  | Unterschrift |

Kurzzusammenfassung

In dieser Arbeit wird ein Thema im Rahmen des Kurses „Grundlagen des E-Business“ ausgearbeitet. Diese Arbeit besteht aus 2 Teilen. Der erste ist der Programmierteil in dem eine mögliche Implementation passend zum Thema erarbeitet wird. Der 2. Teil ist die schriftliche Ausarbeitung. Aufgrund des Praxisanteils wird dieser schriftliche Teil sich kürzer fassen und sich an dem stark an dem Praxisbeispiel orientieren. Deshalb stehen die Themen Business- und Marktanalyse, Anforderungen an die Software, IT-Architektur sowie ein Benutzerhandbuch im Vordergrund.

Inhaltsverzeichnis

[I. Abkürzungsverzeichnis 4](#_Toc167984376)

[II. Abbildungsverzeichnis 5](#_Toc167984377)

[1 Einleitung 6](#_Toc167984378)

[2 Hauptteil 7](#_Toc167984379)

[2.1 Vorbereitung 7](#_Toc167984380)

[2.2 Analyse 9](#_Toc167984381)

[2.3 Anforderungen 10](#_Toc167984382)

[2.4 IT-Architektur 10](#_Toc167984383)

[2.5 Benutzerhandbuch 10](#_Toc167984384)

[3 Fazit und Ausblick 10](#_Toc167984385)

[4 Literaturverzeichnis 10](#_Toc167984386)

1. Abkürzungsverzeichnis
2. Abbildungsverzeichnis

[Abbildung 1 - Austauschbeziehungen 7](#_Toc167984286)

# Einleitung

Das E-Business spielt seit der rasanten Entwicklung der Digitalisierung mehr und mehr eine größere Rolle bei Geschäftsprozessen. Dabei begleitet das E-Business jeden Bestandteil eines solchen Prozesses. Dazu zählen die Anbahnung, die Vereinbarung, sowie die Abwicklung. Nur Unternehmen, welche den Kurs der Digitalisierung und somit auch dem des E-Business verfolgt haben sind heute noch an der Spitze der Wirtschaft. Das E-Business ist als Hilfsmittel für die Vereinbarung eines Vertrages zweier Parteien zu sehen. Ein kleiner Bestandteil des E-Business sind die Desktop Purchasing Systeme, um die es in dieser Arbeit gehen soll. Genauer behandelt diese Arbeit im Folgenden einen selbstimplementierten Prototyp für ein solches DPS. DPS spielen eine zentrale Rolle um einige Abläufe innerhalb eines Unternehmens effizienter und kosteneffektiver zu gestalten. Verbessert werden können mit DPS jegliche Beschaffungsprozesse. Sie reduzieren den Aufwand von Bestellungen stark und minimieren Fehler indem sie es ermöglichen Bestellungen direkt vom Desktop aus zu verwalten. In Zukunft sollen nun nicht mehr manuelle Bestellungen getätigt werden, welche meist mündlich weitergegeben wurden, da diese sehr fehleranfällig und schlecht zurück verfolgbar sind. Dies führt auch dazu das die Compliance-Richtlinien besser eingehalten werden können. Compliance meint die Regeltreue zu dem jeweiligen Regelwerk auf welches sich ein Unternehmen festgelegt hat.

Zudem hat das Unternehmen einen besseren Blick auf ihre Einkaufskosten und kann damit die Bestellungen besser regulieren. Außerdem kann man im Nachgang aufgrund der besseren Übersicht Sammelbestellungen in Auftrag geben, welche wiederum zu Mengenrabatten führen können.

# Hauptteil

## Vorbereitung

### Architektur

Die Architektur von Desktop-Purchasing-Systemen ist eine typische auf Internet-Technologien basierende Client-Server-Architektur. Dabei besteht ein solches System aus einem Server, welcher mehrere Clients bedienen kann. Die Kommunikation zwischen den beiden Komponenten erfolgt dabei über die TCP/IP-Protokollfamilie. Das Internet Protocol (IP) übernimmt dabei die Aufgabe Datenpakete unternehmensintern und über Unternehmensgrenzen hinaus vom Sender zum Empfänger zu transportieren. Für diese Art der Kommunikation muss vorher keine Verbindung zwischen Empfänger und Sender bestehen.[[1]](#endnote-1)

#### Clients

Im Bereich der Desktop-Purchasing-Systeme sind die Clients sehr simpel gehalten. Man benötigt lediglich einfache internetfähige Arbeitsplatzrechner. Über die Webschnittstelle zum Server wird dann z.B. der MSCP durchsucht und/oder Bestellungen angelegt, gelöscht oder bearbeitet. MSCP (Master Product and Supplier Catalog) sind elektronische Produktkataloge mit einer Zusammenstellung von Artikelinforationen mehrerer Lieferanten. Auch die Administration sowie die Pflege des Systems erfolgt über einen solchen Client. Meist wird bei der Clientanwendung über die Nutzerrolle entschieden ob es sich um einen Mitarbeiter oder einen Administrator handelt. Je nach Rolle ist der Benutzer dann dazu befugt administrative Aufgaben über die Anwendung zu erledigen oder nicht. Die Kommunikation zwischen den beiden Komponenten erfolgt dabei stets über das auf TCP/IP basierte http-Protokoll.[[2]](#endnote-2)

#### Server

Die Clients sind also nur dazu da eine Schnittstelle zwischen dem Server und den Benutzern zu schaffen. Der Server jedoch stellt die Software dar, welche für die wirkliche Applikationslogik, sowie die korrekte Datenhaltung verantwortlich ist. In großen Anwendungsszenarien ist ein solches Server System über mehrere Rechner verteilt um echte Nebenläufigkeit (gleichzeitige Ausführung mehrerer Prozesse). Dies führt zu einer stark verbesserten Skalierbarkeit, da man weitere Rechner mit in das Server System einbinden kann um eine höhere Gesamtleistung zu erreichen. … -6 S. 157,158.

Dieses Server System besteht aus 3 Arten von Teilkomponenten. Die eine Art Prozesse und Workflows, besteht aus Komponenten, welche die Logik zur Ausführung von Bestellungen sowie zur Verwaltung von Bestellanforderungen enthalten. Diese Komponenten sind diejenigen, welche direkt mit der Datenbank, in welcher Warenkörbe Bestellanforderungen oder auch Bestellungen gespeichert werden, kommunizieren.

Komponenten für Produktkatalog und Content Management enthalten die Logik zur Suche, Sourcing und Produktkonfiguration durch den Verbraucher, sowie zur Pflege der Katalogdaten durch die Einkaufsabteilung, Dienstleister oder Lieferanten. Sie haben direkten Zugriff auf den MSPC.

Die Letzte Art sind die Komponenten zur Systemadministration. Diese werden in der Regel von der Einkaufsabteilung genutzt, um Benutzerprofile, Genehmigungsregeln und Lieferantenprofile zu verwalten und in einer entsprechenden Datenbank abzulegen.[[3]](#endnote-3)

Ein Bild, das Text, Screenshot, Display, parallel enthält.

Automatisch generierte Beschreibung

In Abb. 1 wird eine solche Serverstruktur grafisch dargestellt. In dem Zentrum befinden sich die 3 Arten der Sever Komponenten. Links neben den Arten sieht man die Schnittstellen (APIs) für die internen Informationssysteme. Rechts ist die Kommunikation zu externen Systemen z.B. zu Lieferantensystemen abgebildet. Am oberen Rand ist die Benutzerschnittstelle also der Client angedeutet. Nicht zu erkennen auf dieser Abbildung ist der Zugriff von der Benutzerschnittstelle auf die Daten des Servers. Dies erfolgt heutzutage über eine oder mehrere Web APIs. Innerhalb einer Art von Komponenten sieht man ein Datenbanksystem, welches alle Daten, welche die Komponenten benötigen beinhaltet. Also der generelle Fall für eine Komponente in einer Serverstruktur ist vereinfacht eine Datenbank mit Anbindung einer Web API. Dabei bestimmt die Komponenten jeweils welche Funktionen die Web API zum Bearbeiten des Datenbestandes anbietet. Die Web APIs sind in dem Fall eine Schnittstelle zum Client, zu internen Informationssystemen oder zu externen Lieferantensystemen.[[4]](#endnote-4)

### Ein Bild, das Text, Screenshot, Schrift, Zahl enthält. Automatisch generierte BeschreibungAustauschbeziehung

Abb. 1 - Austauschbeziehungen

In Abb. 1 Sieht man eine breite Übersicht über die elektronischen Geschäftsbeziehungen. Für den elektronischen Handel (Electronic Commerce) relevant sind die 2 Optionen Business-to-Business (B2B) und Business-to-Consumer (B2C). Wie der Name schon verrät Ist der Leistungsanbieter bei beiden Beziehung eine Firma. Also behandelt das eCommerce alle die Beziehungen bei denen ein Unternehmen eine Dienstleistung stellt. Ein Beispiel für B2C wäre zum Beispiel ein e-Shop über den ein Unternehmen Dienstleistungen an Kunden verkauft. Das Desktop Purchasing System jedoch ist ein Bestandteil des B2B. Die Benutzer eines solchen DPS sind Mitarbeiter eines Unternehmens und für die Bereitstellung von den benötigten Arbeitsmitteln ist das Unternehmen selbst verantwortlich. Also ist der Leistungsnachfrager der Beziehung von DPS ein Unternehmen. Und die benötigten Materialien werden stets von Lieferanten also anderen Unternehmen bereitgestellt. Deswegen ist auch der Leistungsanbieter ein Unternehmen. S.2 - 6

## Analyse

Um Zeit und Komplexität zu sparen ist es notwendig sich vor einem jeden Projekt zu überlegen, welche Technologien man benutzt. In wirtschaftlichem Kontext ist dies nochmal wichtiger um sowohl Geld als auch Zeit einzusparen. Da die Struktur eines DPS schon erläutert wurde kann man die Analyse nun gezielt durchführen. Man muss nun für die 3 Komponenten Datenbank, Web API und Client jeweils eine geeignete Technologie ausfindig machen.

### Datenbankmanagementsystem

Als DBMS (Datenbankmanagementsystem) kommen heutzutage sehr viele verschiedene Systeme in Frage. Zu den bekanntesten zählen dabei SQLite, MariaDB, MySQL, PostgreSQL, Oracle, und Microsoft SQL Server. Den wichtigsten Punkt für einen solches Testprojekt ist das die Technologie Open Source ist. Ein Open Source Programm muss den Quellcode enthalten und die Verbreitung in Quellcode als auch in kompilierter Form erlauben. Der Quellcode muss die bevorzugte Form sein, in der ein Programmierer das Programm ändern würde. Absichtlich verschleierter Quellcode ist bei Open Source Software nicht zulässig.[[5]](#endnote-5) Mit der Open Source Eigenschaft kommt zudem eine hohe Anzahl an Dokumentationen und Support über Online-Portale einher. Diese Eigenschaft erfüllen die MariaDB, SQLite und PostgreSQL DBMSs. Um es simpel zu halten werden im Folgenden auch nur diese 3 evaluiert. Zunächst einmal sind alle 3 DBMSs relational, das heißt sie verwenden Datentabellen als Datenablage, von denen 2 Tabellen mithilfe eines gemeinsamen Feldes verknüpft sein können.[[6]](#endnote-6) Zudem unterstützen alle 3 Systeme die grundlegenden SQL-Funktionalitäten.

Der Hauptunterschied liegt darin, dass SQLite eine integrierte Datenbank ist. Das bedeutet, dass diese Datenbank nicht eigenständig als eigener Prozess läuft. Sondern es koexistiert mit der Anwendung in welcher sie implementiert ist. Das bedeutet auch das die Datenbank nicht über einer Netzwerkschnittstelle erreichbar ist. Die Vorteile liegen dabei das also keine Netzwerk Konfiguration oder eine Administration für die Datenbank eingerichtet werden muss. Das resultiert in einer enormen Zeitersparnis und weniger Komplexität für das System[[7]](#endnote-7). Bei kleinen bis mittleren Datenbeständen ist dieses DBMS auch sehr performant, da es serverlos ist.

PostgreSQL und MariaDB sind das genaue Gegenteil sie laufen in einem eigenständigen Prozess und sind somit besser geeignet für Anwendungen welche hoch skalieren können. Der ganze Aufwand der mit SQLLite vermieden wird, ist jedoch bei den beiden DBMSs wieder existent. Ein weiterer Vorteil der beiden ist, dass sie für große Datenbestände ausgerichtet sind und somit auch bei diesen noch sehr performant sind.

Auch wenn für MariaDB einige Vorkenntnisse sprechen fällt die endgültige Wahl eindeutig auf SQLite. SQLLite ist für eine Testanwendung im Kontext der DPS die bessere Wahl. Es spart Zeit sowie Komplexität und ist auch für die Verteilung des DPSs auf andere Maschinen gut geeignet. Außerdem, ist der Datenbestand dieses Testprojektes gering und die Benutzeranzahl eingegrenzt. Das bedeutet das dieses Projekt nicht hoch skaliert und somit SQLite alle benötigten Anforderungen erfüllt.

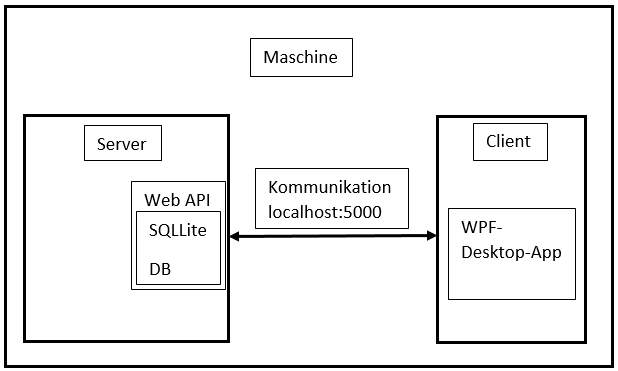
### Web API

Web APIs finden heutzutage mehr und mehr Anwendungen und sind mittlerweile der Standard für die Kommunikation zwischen Frontend und Backend. Aufgrund ihrer Popularität gibt es viele verschiedene Framworks zur Implementierung einer solchen API. Zu den bekanntesten zählen ASP.NET Core, Spring Boot, Django REST Framework und Express.js.

ASP.NET Core ist ein Framework von Microsoft und wird in C# implementiert. Die wichtigsten Vorteile sind das man es plattformunabhängig entwickeln kann, einfach hilfreiche Libraries über den integrierten Nuget-Package-Manager installieren kann, und es eine schon vorimplementierte „dependency injection“ unterstützt.[[8]](#endnote-8) Spring Boot ist ein Java Framework und unterstützt einen bei der Implementierung von Web APIs und hat aufgrund seiner großen Community eine umfangreiche Dokumentation und Support. Das Django Rest Framework hingegen nutzt die Stärken von Python und zeichnet sich durch die Einfachheit und auch der umfangreichen Dokumentation aus. Bei der Entscheidung, welches Framework nun das beste sei kann es aufgrund von persönlichen Erfahrungen und Erkenntnissen zu verschiedenen Meinungen kommen. Sie haben alle ihre Vorteile und bringen einen an sein Ziel. Für dieses Projekt jedoch wird ASP.NET aufgrund von persönlichen Erfahrungen und Kenntnissen Einsatz finden. Oft wird nach persönlichen Präferenzen entschieden und das ist auch von Vorteil, denn die Arbeit, welche benötigt wird, um sich in ein neues Framework einzuarbeiten kann sehr schnell sehr hoch werden. Wenn die Arbeit größer wir als der Nutzen kann man es in einem realen Projekt auch als unwirtschaftlich bezeichnen.

## Anforderungen

## IT-Architektur

Die endgültige IT-Architektur besteht nach der umfassenden Analyse aus einem DBMS einer Web API und einer Desktop Anwendung. Für das DBMS kommt SQLite zum Einsatz. Dieses wird in die Web API integriert. Die Web API wird durch die Vorerfahrung mithilfe von ASP .NET Core implementiert. Das Frontend also die Desktop App wird mit dem, für Windows bekannten Desktop-Framework, WPF realisiert. Am Ende werden die beiden Anwendungen zusammen über eine .bat gestartet. Beide Applikationen werden also lokal laufen, um die Verwendung eines Servers zu vermeiden. Dennoch stellt die API kombiniert mit der Datenbank die Serverseite dar und die WPF-App übernimmt den Client. Beim Start der API wird sie auf dem Port localhost:5000 laufen und für die WPF-App lokal erreichbar sein.

## Benutzerhandbuch

## Vorteile

Für jede Anwendung ist ein Benutzerhandbuch ein unverzichtbares Dokument. Zum einen erleichtert es logischer Weise den Anwendern das Leben und spart ihnen Zeit. Vor allem im unternehmerischem Kontext ist es essentiell denn wenn die Anwender die eigenen Mitarbeiter sind möchte man natürlich, dass Sie keine Zeit bei dem Anlegen von Bestellungen verschwenden. Der 2. Große Vorteil ist es, dass die Support-Anfragen sich erheblich verringern auch das resultiert wiederum in einer Einsparung von Mitarbeitern, also von Geld. Zudem sind auch Benutzer allgemein zufriedener und wenn sie zufriedener sind arbeiten sie effizienter und effektiver.

## Ausführung

### Titelblatt

Titel: Benutzerhandbuch e-Business für DPSs

Name: Desktop Purchasing System

Version: 1.0

Autor: Nicolas Stefan Koch

Datum: 31.05.2024

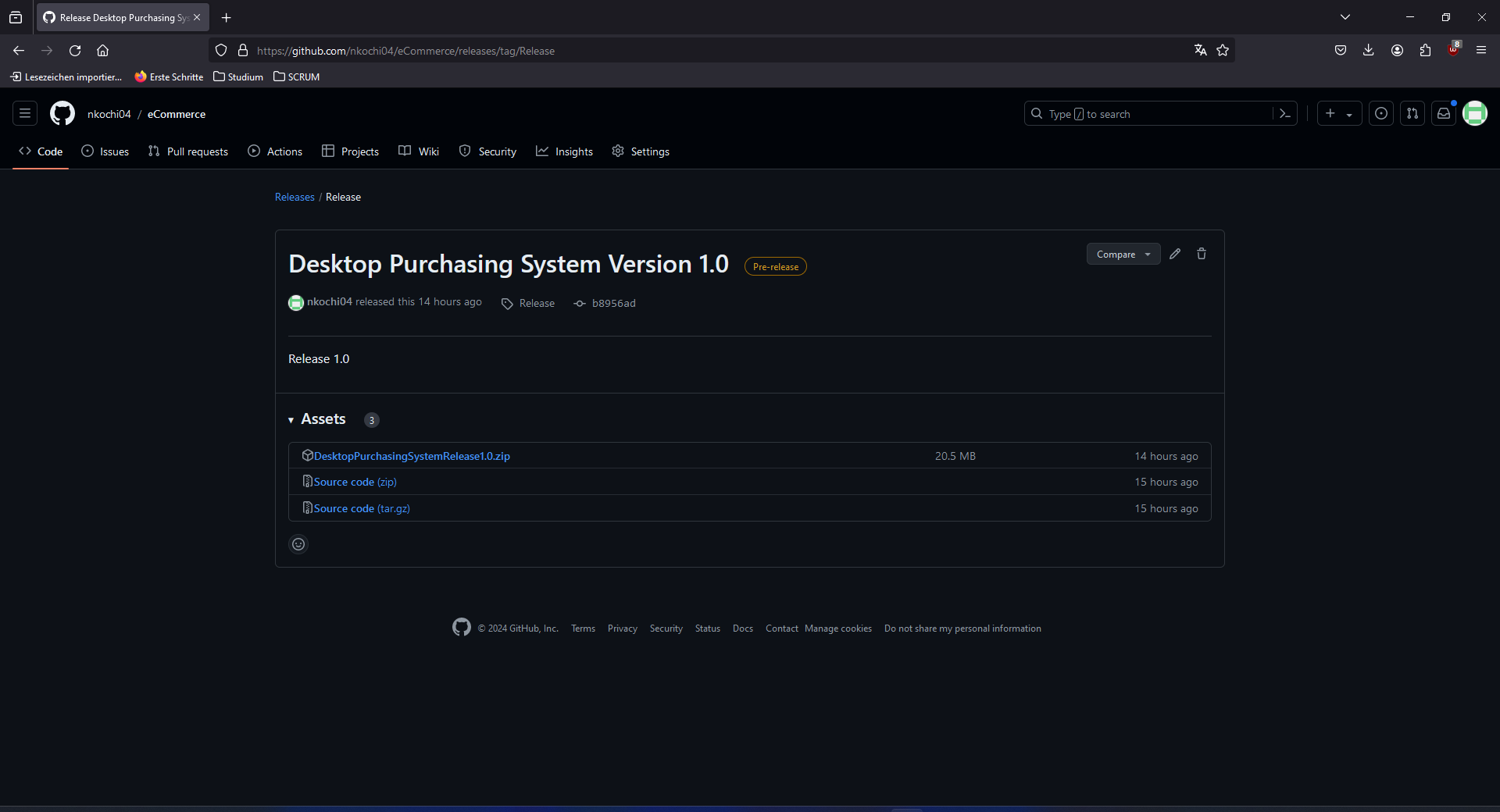
### Inhaltsverzeichnis

### Einführung

Diese Software ist eine Client-Anwendung zum vereinfachten elektronischen anlegen von Bestellungen. Sie ist ausschließlich für die Mitarbeiter von Firma XY zugänglich.

### Installation

1. Besuche diese Seite <https://github.com/nkochi04/eCommerce/releases/tag/Release> und installiere die Datei „DesktopAppPublishRelease1.0“ mit einem Klick auf sie.



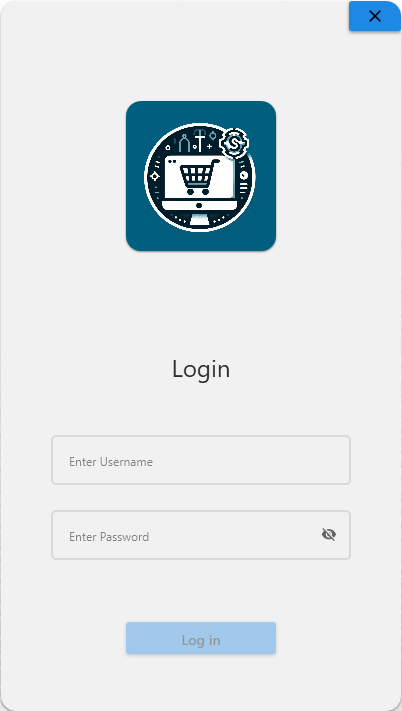
1. Entpacke diese Datei im Anschluss und starte das Programm mit einem Doppelklick auf die Datei „Start.bat“ im Order App.



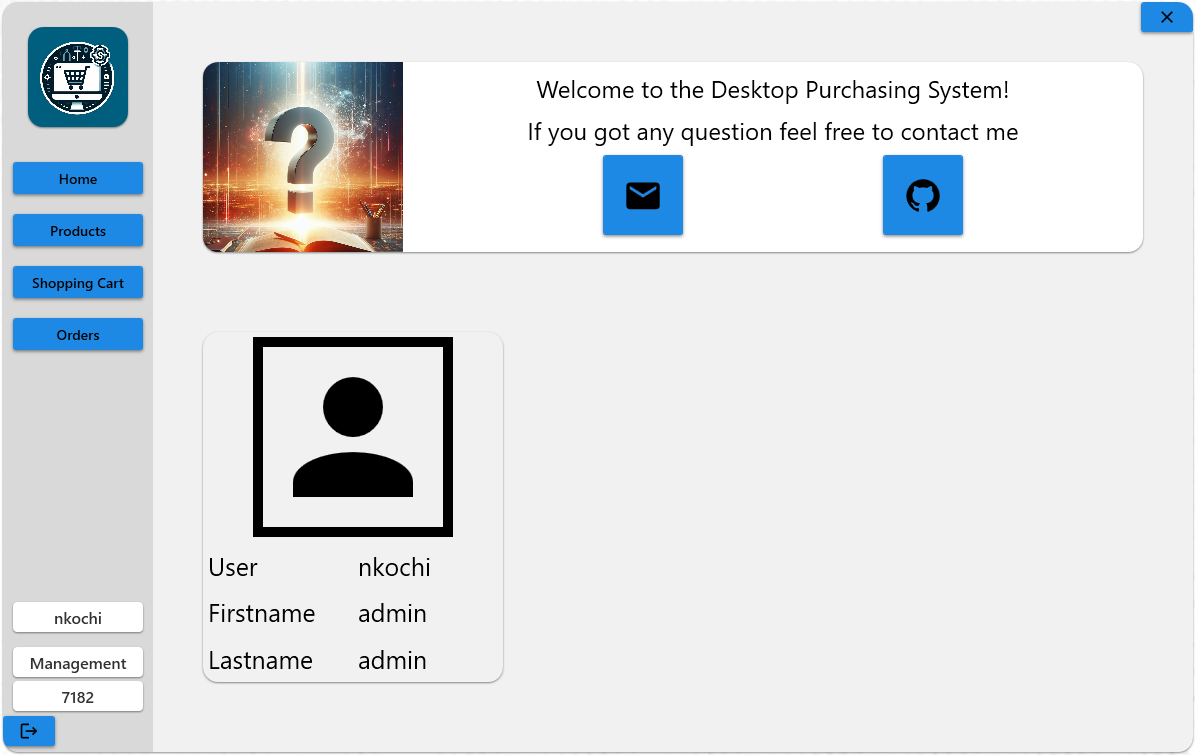
### Login

Nach dem Doppelklick auf die Start.bat sollte eine Kommandozeilen Fenster sowie ein Login Fenster aufgehen. Die Kommandozeile ist für die API da und sollte nicht geschlossen werden solange das Programm läuft. In dem Login Fenster kannst du dich nun mit deinem Arbeits-Account anmelden. Testdaten mit Usern aus de verschiedenen Abteilungen:

|  |  |
| --- | --- |
| **Test-User Login Daten** | |
| **Benutzername** | **Passwort** |
| nkochi | admin |
| admin1 | admin |
| … | |
| admin10 | admin |



### Benutzeroberfläche



# Fazit und Ausblick

# Literaturverzeichnis

1. Kollmann, *E-Business*, 157. [↑](#endnote-ref-1)
2. Kollmann, 157,158. [↑](#endnote-ref-2)
3. Kollmann, 158. [↑](#endnote-ref-3)
4. Kollmann, 159. [↑](#endnote-ref-4)
5. „The Open Source Definition“. [↑](#endnote-ref-5)
6. „Introduction to Relational Databases“. [↑](#endnote-ref-6)
7. „978-1-4302-3226-1.pdf“. [↑](#endnote-ref-7)
8. Pattankar, *Mastering ASP.NET Web API*. [↑](#endnote-ref-8)