# outPort - Portierlösung

DIPLOMARBEIT

Höhere Abteilung für Informatik

01/10/2025 – 26/03/2026

**Projektmitglieder:** Buchner Herbert
Hinterdorfer Julian
Inspruckner Janick

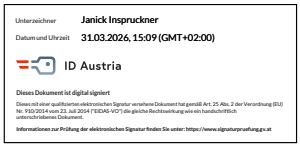
**Betreuer:** Prof. Dipl.-Ing. Dr. Michael Buchberger

# Eidesstattliche Erklärung

Hiermit versichern wir, die vorliegende Arbeit selbständig, ohne fremde Hilfe und ohne Benutzung anderer als der von uns angegebenen Quellen angefertigt zu haben. Alle Stellen, die wörtlich oder sinngemäß aus fremden Quellen direkt oder indirekt übernommen wurden, sind als solche gekennzeichnet.

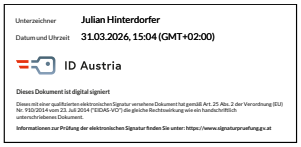
Bei der Erstellung der Arbeit haben wir die generativen KI-Tools Gemini, DeepL AI und ChatGPT zu folgendem Zweck verwendet: zur Rechtschreib- und Grammatikprüfung, zur sprachlichen und stilistischen Überarbeitung der selbst verfassten Rohtexte sowie zur Rechercheunterstützung.

Perg, 31. März 2026 Unterschrift



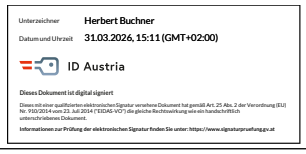

(Janick Inspruckner)

Perg, 31. März 2026 Unterschrift




(Julian Hinterdorfer)

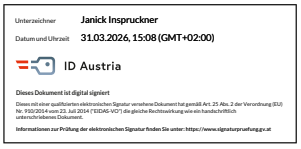
Perg, 31. März 2026 Unterschrift




(Herbert Buchner)

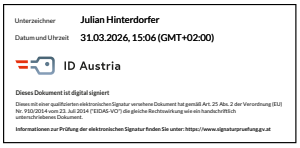
# Gendererklärung

Die in dieser Diplomarbeit verwendeten Personenbezeichnungen beziehen sich immer gleichermaßen auf weibliche und männliche Personen. Auf eine Doppelnennung und gegenderte Bezeichnungen wird zugunsten einer besseren Lesbarkeit verzichtet.



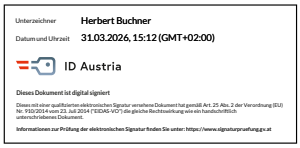

Perg, 31.März.2026 Unterschrift _____

(Janick Inspruckner)




Perg, 31.März.2026 Unterschrift _____

(Julian Hinterdorfer)




Perg, 31.März.2026 Unterschrift _____

(Herbert Buchner)

# Danksagung

Wir möchten uns bei allen Personen und Organisationen bedanken, die uns bei der Erstellung dieser Diplomarbeit begleitet und unterstützt haben.

Ein besonderer Dank gilt unserem Diplomarbeitsbetreuer an der HTL Perg, Prof. Dipl.-Ing. Dr. Michael Buchberger. Durch seine konstruktive Kritik und Unterstützung hat er maßgeblich zum Gelingen dieser Arbeit beigetragen.

Ein weiterer großer Dank gebührt der Firma SYSco EDV, die uns die großartige Möglichkeit geboten hat, dieses Projekt praxisnah zu entwickeln. Insbesondere möchten wir uns bei Ing. Dominik Bindreiter und Andreas Aichinger bedanken. Sie hatten stets ein offenes Ohr für unsere Fragen, haben uns bei technischen Herausforderungen weitergeholfen und uns während unseres Praktikums maßgeblich unterstützt.

Abschließend gilt unser großer Dank allen Bekannten, Verwandten und Freunden, die uns bei der Korrektur der Diplomarbeit unterstützt haben.

# Abstract

SYSco EDV GmbH develops customized IT and software solutions for external partners and companies. The company's main focus lies on business software such as ERP, CRM, and PPS systems, as well as on tailor-made software solutions designed to meet specific customer requirements. In addition, SYSco provides comprehensive IT services, including consulting, implementation, support, as well as infrastructure and security solutions. The company's headquarters are located in Schwertberg, and it employs over 50 people.

As part of this diploma thesis, the modern porter solution „outPort" was developed for SYSco EDV GmbH. This solution represents a comprehensive supplier and visitor management system that digitalizes and optimizes the entire reception process. The main objective of this project was to simplify the registration and handling of visitors, reduce manual effort at the reception desk, and at the same time provide a clear, secure, and user-friendly interface for administration and management.

The developed system enables suppliers and visitors to independently register via a multilingual self-service terminal. Key features include intuitive data entry, confirmation of mandatory legal documents (such as GDPR and safety instructions), printing of visitor badges including QR codes for check-out, and direct notification of the responsible host. The latter was implemented through seamless email and calendar integration using the Microsoft Graph API.

Reception staff and administrators are provided with a centralized, role-based dashboard that offers a real-time overview of all current, planned, and past visits. From a technical perspective, the project was implemented using a modern technology stack consisting of a C#.NET backend and a React frontend. By introducing standardized and partially automated processes, „outPort" significantly reduces workload at reception, and creates a professional first impression for company visitors.

# Zusammenfassung

SYSco EDV GmbH entwickelt maßgeschneiderte EDV- und Softwarelösungen für externe Partner und Unternehmen. Dabei liegt der Fokus insbesondere auf Unternehmenssoftware wie ERP-, CRM- und PPS-Systemen sowie auf individuellen Softwarelösungen, die exakt auf die Anforderungen der Kunden abgestimmt sind. Zusätzlich bietet das Unternehmen umfassende IT-Dienstleistungen, darunter Beratung, Implementierung, Support sowie Infrastruktur- und Sicherheitslösungen. Der Hauptsitz des Unternehmens befindet sich in Schwertberg, und beschäftigt über 50 Mitarbeiter.

Im Rahmen dieser Diplomarbeit wurde für die SYSco EDV GmbH die moderne Portierlösung „outPort“ entwickelt. Dabei handelt es sich um ein umfassendes Lieferanten- und Besuchermanagementsystem, das den gesamten Empfangsprozess digitalisiert und optimiert. Ziel der Arbeit war es, die Registrierung und Verwaltung von Gästen zu vereinfachen, den manuellen Aufwand am Empfang zu reduzieren und gleichzeitig eine klare, sichere sowie benutzerfreundliche Darstellung für die Verwaltung und Administration zu gewährleisten.

Das entwickelte System ermöglicht Lieferanten und Besuchern, sich über ein Self-Service-Terminal selbstständig und in mehreren Sprachen zu registrieren. Zu den umgesetzten Kernfunktionen zählen neben der intuitiven Datenerfassung auch die Bestätigung von rechtlichen Pflichtdokumenten (DSGVO, Sicherheitsunterweisungen), der Druck von Besucherausweisen inklusive QR-Code für den späteren Check-out sowie die direkte Benachrichtigung der zuständigen Gastgeber. Letzteres wurde durch eine nahtlose E-Mail- und Kalenderintegration über die Microsoft Graph API realisiert.

Das Empfangspersonal und die Administratoren erhalten über ein zentrales, rollenbasiertes Dashboard in Echtzeit einen übersichtlichen Status aller anwesenden, geplanten und historischen Besuche. Technisch wurde das Projekt auf Basis eines modernen Technologie-Stacks, bestehend aus einem C#.NET-Backend und einem React-Frontend, umgesetzt. Durch die standardisierten und teilautomatisierten Abläufe bietet „outPort“ eine erhebliche Arbeitserleichterung für die Rezeption und hinterlässt einen professionellen ersten Eindruck bei Gästen des Unternehmens.

# Inhaltsverzeichnis

# 1 Einleitung

In diesem Kapitel wird die Ausgangssituation des Projekts beschrieben sowie die Problemstellung und die daraus resultierenden Anforderungen dargestellt. Anschließend werden die zentralen Begriffe und Abläufe der Portierlösung erläutert, um ein grundlegendes Verständnis für das System zu schaffen.

Darauf aufbauend werden die Zielsetzungen definiert, die sowohl das technische Projektziel als auch die geschäftlichen Aspekte umfassen. Abschließend wird das Projektumfeld beschrieben und ein Überblick über das erzielte Projektergebnis gegeben.

## 1.1 Ausgangslage

Die Firma SYSco EDV verfügt bereits über eine bestehende, einfache, funktionsfähige Portierlösung. Diese ist nun auf Basis eines moderneren Technologie-Stacks neu zu entwickeln und funktional zu erweitern. Die bereits bestehende Lösung ermöglicht Besuchern und Lieferanten, sich am Empfang anzumelden. Der entsprechende Eintrag wird anschließend im Dashboard angezeigt, wo auch das Abmelden möglich ist. Das Projekt „outPort“ soll die Abläufe mit Besucher- und Lieferantendaten optimieren. Zusätzliche Funktionen wie Selbstregistrierung, Voranmeldung und der Druck eines Besucherausweises werden in der vorliegenden Arbeit implementiert.

## 1.2 Problemstellung

Die bestehende Lösung weist mehrere funktionale und organisatorische Defizite auf. Das Unternehmen verfügt bereits über eine einfache Portierlösung, die aktuell nur die Besucheranmeldung sowie eine Adminübersicht der anwesenden Besucher bietet. Um einen modernen und effizienten Besucherprozess zu gewährleisten, ist dieser Funktionsumfang nicht ausreichend.

Ein zentrales Problem ist die fehlende Selbstregistrierung (Besucher erfassen ihre Daten selbst über ein Kiosk-System oder eine mobile Web-Oberfläche). Dies führt zu einem höheren Aufwand an der Rezeption, längeren Wartezeiten und einer erhöhten Fehleranfälligkeit bei der manuellen Dateneingabe. Zusätzlich fehlen multimediale Unterweisungen, wie beispielsweise

Sicherheitsinformationen als Video oder Bildern, wodurch Sicherheits- und Verhaltensregeln nicht standardisiert und nachvollziehbar vermittelt werden können.

Des Weiteren besteht keine zentrale Lösung für die strukturierte Datenhaltung und -auswertung. Dies führt dazu, dass Einblicke in die Besucherhistorie, Filterungen und Reports nur eingeschränkt möglich sind. Ebenso fehlt eine rollenbasierte Rechteverwaltung (Zugriffssteuerung nach Benutzerrollen wie Portier oder Admin), was die sichere Administration erschwert.

Letztendlich ist die aktuelle Lösung kaum integriert und automatisiert. Funktionen wie Gastgeber-Benachrichtigungen, Besucherausweisdruck oder Kalenderanbindungen sind nicht vorhanden. Bei der Verarbeitung personenbezogener Daten sind die Anforderungen der Datenschutz-Grundverordnung (DSGVO) zu erfüllen.

Die bestehende Lösung muss daher zu einem umfassenden, benutzerfreundlichen, integrierbaren und DSGVO-konformen Besuchermanagementsystem ausgebaut werden.

## 1.3 Begriffsdefinition dieser Arbeit

Eine Portierlösung zielt darauf ab, den organisatorischen Aufwand am Empfang zu reduzieren und die Arbeit des Empfangspersonals zu unterstützen. Dabei werden zwei Abläufe unterschieden: der Besucherprozess und der Lieferantenprozess.

### 1.3.1 Besucherprozess

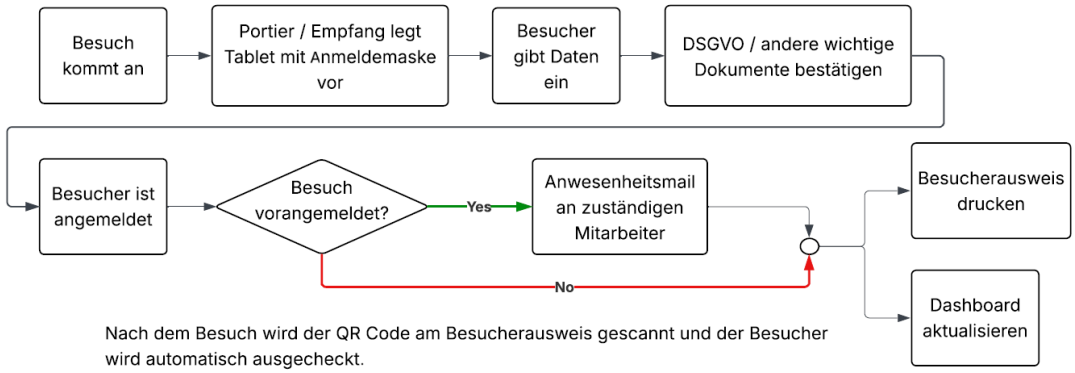



Abbildung 1: Ablauf Portierlösung - Besuch

Wie in Abbildung 1 dargestellt, wird dem Besucher am Empfang ein Tablet mit einer Anmeldemaske bereitgestellt. In dieser Maske werden die relevanten Besucherdaten erfasst. Darüber hinaus besteht die Möglichkeit, rechtlich erforderliche Dokumente wie die DSGVO-Hinweise oder

Sicherheitsunterweisungen anzuzeigen. Die Besucher nehmen diese Informationen zur Kenntnis und bestätigen dies im Rahmen des Anmeldeprozesses.

Nach Abschluss der Anmeldung werden im Hintergrund weitere Schritte automatisch ausgeführt. Wurde der Besucher zuvor von einem Mitarbeiter vorangemeldet und ist eine E-Mail-Adresse hinterlegt, wird der Gastgeber automatisch benachrichtigt, dass der Besucher eingetroffen ist. Parallel dazu wird ein Besucherausweis ausgedruckt und im Dashboard der Check-in-Zeitpunkt des Besuchers gespeichert. Das Dashboard ist die zentrale Übersichts- und Verwaltungsoberfläche der Portierlösung. Hier werden aktuelle und registrierte Besucher sowie aktuelle Lieferanten mit Status und Zeitstempeln angezeigt und verwaltet.

### 1.3.2 Lieferantenprozess

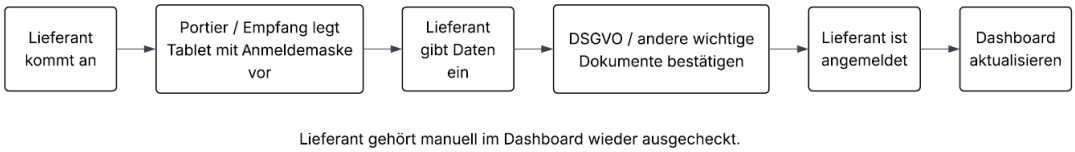



Abbildung 2: Ablauf Portierlösung - Lieferant.

Wie im Besucherprozess beschrieben (Abschnitt 1.3.1) wird dem Lieferanten am Empfang ebenfalls ein Tablet mit einer Anmeldemaske bereitgestellt. Die Datenerfassung sowie die Bestätigung relevanter rechtlicher Dokumente erfolgen nach demselben Ablauf.

Im Gegensatz zum Besucherprozess liegt der Fokus beim Lieferantenprozess auf Informationen wie dem vollständigen Namen, dem Kennzeichen und der Bestellnummer. Nach erfolgreicher Anmeldung wird der Lieferant im Dashboard als anwesend aufgeführt. Der Check-out erfolgt manuell, sobald der Lieferant das Gelände verlassen hat.

## 1.4 Zielsetzung

In diesem Abschnitt werden die Zielsetzungen des Projekts definiert. Dabei wird zwischen dem technischen Projektziel und dem daraus abgeleiteten Geschäftsziel unterschieden. Während das Projektziel die funktionale und technische Umsetzung der Portierlösung beschreibt, bezieht sich das Geschäftsziel auf die praktischen Vorteile und Auswirkungen im Unternehmensalltag.

### 1.4.1 Projektziel

Das Ziel des Projekts besteht darin, eine den Anforderungen entsprechende, voll funktionsfähige Portierlösung bereitzustellen. Dazu gehören die Voranmeldung durch einen Mitarbeiter, das Eintragen des Termins in den Outlook-Kalender des Gastgebers, eine Selbstanmeldung durch den Besucher oder Lieferanten am Empfang, der Druck eines Besucherausweises und die Benachrichtigung an den Gastgeber beim Eintreffen des Gastes. Das Dashboard, das über eine rollenbasierte Zugangskontrolle verfügt, zeigt Besucher und Lieferanten mit deren personenbezogenen Informationen sowie die Dauer ihres Aufenthalts an. Über das Dashboard können Statistiken eingesehen, Anmeldeformulare angepasst und rechtliche Dokumente verwaltet werden. Zudem können das Firmenlogo und das Erscheinungsbild des Dashboards konfiguriert werden. Die Anwendung muss klar strukturiert und intuitiv bedienbar sein, um Missverständnisse zu vermeiden. Um dies zu erreichen, muss die Anmeldemaske mehrsprachig verfügbar sein und das Dashboard übersichtlich und nutzerfreundlich gestaltet sein.

### 1.4.2 Geschäftsziel

Das aus dem Projektziel abgeleitete Geschäftsziel besteht aus weniger manuellen Arbeitsschritten, geringeren Wartezeiten und einer kürzeren durchschnittlichen Anmeldedauer. Durch die standardisierten und teilautomatisierten Abläufe wird der organisatorische Aufwand am Empfang reduziert, wodurch Ressourcen effizienter genutzt werden können. Gleichzeitig wird die Produktivität der Mitarbeitenden gesteigert, da Wartezeiten, Rückfragen sowie manuelle Tätigkeiten verringert werden und Prozesse konsistenter ablaufen.

Die Einführung der Portierlösung ermöglicht es, einen professionellen und strukturierten Empfangsprozess nach außen zu vermitteln, was einen positiven Eindruck hinterlässt und einen Wettbewerbsvorteil darstellt. Zusätzlich soll eine höhere Mitarbeiterzufriedenheit erreicht werden, da der Arbeitsaufwand am Empfang entlastet wird und die Abläufe nachvollziehbarer und verlässlicher sind. Der Verwaltungsaufwand wird insgesamt reduziert, da Besucher- und Lieferantendaten sowie Anwesenheiten konsistent erfasst werden. Dadurch werden Auswertungen und Nachweise einfacher und zuverlässiger generiert.

## 1.5 Projektumfeld

In diesem Abschnitt wird das Umfeld beschrieben, in dem das Projekt umgesetzt wurde. Dazu zählen der Auftraggeber, die schulische Betreuung sowie die organisatorischen Rahmenbedingun-

gen. Diese Faktoren haben sowohl die Anforderungen als auch die Durchführung des Projekts beeinflusst.

### 1.5.1 Auftraggeber

Der Auftraggeber SYSco EDV mit Sitz in Schwertberg ist ein Unternehmen, das auf IT-Dienstleistungen und Softwarelösungen spezialisiert ist. SYSco EDV definierte für das Projekt Anforderungen und unterstützte das Projektteam mit technischem Know-how. Die vorhandene Portierlösung bildet die Grundlage für die neue, erweiterte Anwendung. Während das Projektteam durch die Zusammenarbeit wertvolle Einblicke in reale Betriebsabläufe gewinnt, profitiert SYSco EDV im Gegenzug von einer technologisch modernisierten und funktional erweiterten Lösung, die künftig als Produkt eingesetzt werden soll.

### 1.5.2 Betreuungslehrer

Gemäß den geltenden Bestimmungen wird die Diplomarbeit fachlich und methodisch von Dir. Prof. Dipl.-Ing. Dr. Michael Buchberger betreut. Zu seinen Aufgaben zählten die regelmäßige Kontrolle des Projektfortschritts, die Unterstützung bei komplexen technischen Fragestellungen sowie die Sicherstellung der akademischen Qualitätsstandards. Durch kontinuierliche Feedbackgespräche wurde sichergestellt, dass das Projekt sowohl den schulischen Richtlinien als auch den Anforderungen aus der Praxis entspricht.

### 1.5.3 Schule

Das Projekt „outPort“ wird im Rahmen der Diplomarbeit an der Höheren Technischen Lehranstalt (HTL) Perg im Fachbereich Informatik umgesetzt. Die Schule stellt den organisatorischen, theoretischen und infrastrukturellen Rahmen für die Durchführung bereit. Das Ziel der Diplomarbeit besteht darin, die während der Ausbildung erworbenen Kompetenzen – insbesondere in den Bereichen Softwareentwicklung, Datenbankdesign und Projektmanagement – in einem praxisnahen Projekt anzuwenden und nachzuweisen.

## 1.6 Projektergebnis

Die Diplomarbeit „outPort“ ist eine Portierlösung. Das bedeutet, dass es möglich ist, sich über eine Benutzeroberfläche im Fall dieser Arbeit Besucher sowie Lieferanten zu registrieren und damit deren Verwaltung zu erleichtern. Zwei einzelne Registrierungsseiten ermöglichen die

Registrierung von zwei Arten von Gästen. Zusätzlich ist es möglich in den Registrierungsseiten die Sprache anzupassen, hinterlegte Dokumente zu lesen und diesen zuzustimmen sowie für den Registrierungsvorgang die eigenen Daten einzugeben.

In der Dashboard-Seite eines Portiers oder einer Person ohne Admin-Rechte wird eine Sammlung von verschiedenen Rastern angezeigt. Diese beinhalten eine Seite für aktive Besucher, eine Seite für aktive Trucker sowie eine tabellarische Ansicht der registrierten Besucher. Man kann Besucher registrieren, eine E-Mail-Adresse zur Benachrichtigung des Mitarbeiters hinterlegen, der den Gast erwartet, und einen Besucher vom Status „registriert“ auf „aktiv“ setzen, womit dieser aus der Ansicht „Registrierte Besucher“ gelöscht wird und in der Ansicht „Aktive Besucher“ erscheint.

In der Admin-Ansicht gibt es weitere Raster wie Statistiken, Berichte, rechtliche Dokumente und Einstellungen. In den Statistiken wird eine grafische Übersicht geliefert, über die man Erkenntnisse über Besuchszeiten und die Anteile von Besuchern zu Lieferanten erhält und auf Grundlage dieser gehandelt wird. Im Raster „Berichte“ wird ein PDF-Dokument der einzelnen Eintragungen der Besucher und Lieferanten generiert. In dem Raster „Rechtliche Dokumente“ können Dokumente hinterlegt werden, denen ein Lieferant und/oder Besucher zustimmen muss oder kann, um sich zu registrieren. Zuletzt gibt es dann noch eine Seite „Einstellungen“ in welcher man die Sprache, das Design und das angezeigte Logo einstellen kann.

# 2 Grundlagen und Methoden

In diesem Kapitel werden Begriffe und Konzepte erläutert, die für das Verständnis des Besuchermanagementsystems erforderlich sind. Dabei wird zunächst auf die beteiligten Personengruppen sowie zentrale Begriffsdefinitionen eingegangen.

Darüber hinaus werden die rechtlichen Rahmenbedingungen im Umgang mit personenbezogenen Daten dargestellt. Ein besonderer Fokus liegt dabei auf den Anforderungen des österreichischen Datenschutzgesetzes (DSG) sowie der Datenschutz-Grundverordnung (DSGVO).

Abschließend wird das Rollen- und Berechtigungskonzept beschrieben, das die Grundlage für den sicheren und kontrollierten Zugriff auf Funktionen und Daten im System bildet.

## 2.1 Begriffe und Einordnung

In diesem Kapitel werden grundlegende Begriffe definiert, um deren Verwendung in dieser Arbeit eindeutig zu klären.

### 2.1.1 Besuchermanagement als Geschäftsprozess

Das Besuchermanagement umfasst einen organisierten Ablauf zur Anmeldung, Verwaltung und Dokumentation externer Personen wie Besucher oder Lieferanten, die ein Unternehmen betreten. Ein strukturiertes Besuchermanagement zielt darauf ab, den Empfangsprozess effizient und nachvollziehbar zu gestalten. Dies erhöht die Sicherheit im Unternehmen, da jederzeit bekannt ist, wer sich aktuell im Unternehmen befindet. Zudem wird ein professioneller und einheitlicher Ablauf sichergestellt, wodurch der organisatorische Aufwand für das Empfangspersonal reduziert und der erste Eindruck gegenüber externen Personen verbessert wird.

Der Prozess lässt sich in mehrere Schritte unterteilen. Idealerweise erfolgt vor dem Eintreffen eine Voranmeldung, bei der der Besucher bereits im System erfasst wird. Nach dem Eintreffen wird zuerst der Check-in durchgeführt. Dabei werden Besucher beziehungsweise Lieferanten registriert und der Status auf „anwesend“ gesetzt. Im Anschluss erfolgt der Aufenthalt im Unternehmen, bis der Check-out durchgeführt wird und die Person das Gelände wieder verlässt.

Abschließend werden die erfassten Informationen dokumentiert und für spätere Nachweise oder Auswertungen gespeichert. Während Besucher üblicherweise einem bestimmten Gastgeber zugeordnet sind, liegt der Fokus beim Lieferantenprozess stärker auf der Anlieferung, Abholung und der Anwesenheitsverwaltung.

Im Rahmen des Besuchermanagementprozesses werden insbesondere aktuelle Übersichten und nachvollziehbare Daten zu Besucher- und Lieferantenstatistiken bereitgestellt. Dazu zählt eine Echtzeitübersicht darüber, welche Personen sich gerade im Gebäude befinden, sowie eine Dokumentation von Ankunft und Abgang. Je nach Gestaltung können zusätzliche Schritte unterstützt werden, etwa das Ausdrucken eines Besucherausweises, die Benachrichtigung des zuständigen Mitarbeiters oder die Bestätigung rechtlicher Hinweise. Das Besuchermanagement bildet damit eine verlässliche Grundlage für die interne Organisation, Sicherheitsanforderungen und spätere Auswertungen. So können beispielsweise aktuelle Anwesenheitsberichte dazu beitragen, Sicherheits- und Notfallprozesse zu unterstützen. Darüber hinaus können Besucherzahlen analysiert werden. [1]

### 2.1.2 Prozessbeteiligte

Im Kontext eines Besuchermanagementsystems sind mehrere Personengruppen direkt am Prozess beteiligt. Diese Prozessbeteiligten verfolgen unterschiedliche Ziele und interagieren auf verschiedene Weise mit der Lösung. Dies führt zu unterschiedlichen Anforderungen in Bezug auf Bedienbarkeit, Informationsumfang und Berechtigungen.

- Die Bedürfnisse von **Besuchern** umfassen einen schnellen, verständlichen und möglichst fehlerfreien Anmeldeprozess. Dieser beinhaltet insbesondere die einfache Eingabe der erforderlichen Daten sowie die klare Vermittlung relevanter rechtlicher Hinweise, wie beispielsweise die DSGVO, Sicherheits- oder Datenschutzhinweise.
- Die Aufgabe des **Empfangs** ist die reibungslose Anmeldung des Besuchers. Der Empfang benötigt eine Lösung, um mehrere Anmeldungen parallel zu betreuen und Fehler bei der Datenerfassung zu vermeiden.
- **Gastgeber** sind Mitarbeitende, die Besucher empfangen. Für sie ist es von entscheidender Bedeutung, rechtzeitig über das Eintreffen informiert zu werden, damit der Besucher ohne zusätzliche Rückfragen am Empfang übernommen werden kann. Die automatische Benachrichtigung ist daher von wesentlicher Bedeutung.
- **Administratoren (Admin)** sind für die Konfiguration und Verwaltung des Systems zuständig. Dazu zählen das Anpassen von Formularen beziehungsweise rechtlichen Hin-

weisen sowie das Auswerten von Daten (z. B. Besucheraufkommen, Historie). Für die erfolgreiche Umsetzung ist es unerlässlich, konsistente Daten und klare Berechtigungen zu gewährleisten.

| **Prozessbeteiligte** | **Ziel** | **Anforderungen / Interaktion** |
|---|---|---|
| Besucher | schneller Check-in | Einfache Dateneingabe, klare Hinweise, Bestätigung DSGVO/Sicherheit |
| Empfang | reibungsloser Ablauf | Übersicht, parallele Anmeldungen, Fehlervermeidung, Check-in/Check-out Verwaltung |
| Gastgeber | Rechtzeitig informieren | Benachrichtigung bei Ankunft, eindeutige Zuordnung, wenig Rückfragen |
| Admin | Konfiguration | Formulare/Dokumente, Historie/Auswertungen, konsistente Daten |

### 2.1.3 Begriffsdefinitionen

Diese Begriffe werden in dieser Arbeit einheitlich wie folgt verwendet:

- **Besucher:** Externe Personen, die das Unternehmen zu einem bestimmten Zweck, z. B. für einen Termin oder Anlass, betreten und in der Regel einem Gastgeber bzw. einem Mitarbeiter zugeordnet sind. Ein Besucher wird im System als eigener Datensatz erfasst und wird während seines Aufenthalts als „anwesend“ geführt.
- **Lieferant/Trucker:** Eine externe Person, die das Unternehmen im Rahmen einer Lieferung oder Abholung von Waren betritt. Im Unterschied zum Besucher, steht beim Lieferanten nicht ein Termin mit einem Gastgeber im Vordergrund, sondern die Dokumentation der Anwesenheit im Zusammenhang mit dem Lieferprozess.
- **Voranmeldung:** Erfassung eines geplanten Besuchs vor dessen tatsächlichem Eintreffen. Dabei werden die relevanten personenbezogenen Daten im Voraus im System hinterlegt, und der zuständige Mitarbeiter wird dem Besuch als Gastgeber zugeordnet, um die Gastgeberbenachrichtigung zu ermöglichen.
- **Check-in:** Vorgang der Anmeldung beim Eintreffen in das Unternehmen. Im Zuge des Prozesses werden die erforderlichen Daten erfasst und der Status der Person im System auf „anwesend“ gesetzt. Dies erfolgt durch das Setzen eines Check-in-Zeitpunkts (aktuelle Zeit) im Datensatz der Person.
- **Check-out:** Vorgang der Abmeldung beim Verlassen des Unternehmens. Der Status der Person wird auf „abwesend“ gesetzt, indem dem Datensatz der Person ein Check-out-Zeitpunkt gesetzt wird. Damit wird der Besuch bzw. die Lieferung im System abgeschlossen.

- **Dashboard:** Zentrale Übersichts- und Verwaltungsoberfläche der Portierlösung. Im Dashboard werden aktuelle und registrierte Besucher sowie aktuelle Lieferanten mit Status- und Zeitinformationen angezeigt. Zusätzlich bietet das Dashboard administrative Funktionen wie das Suchen nach Besuchern, die Auswertung von Daten sowie die Konfiguration von Anmeldemasken und rechtlichen Hinweisen.

## 2.2 Datenverwaltung und Datenhaltung laut DSG

In diesem Abschnitt werden jene Gesetzesabschnitte des österreichischen Datenschutzgesetzes dargestellt, die für die Verarbeitung personenbezogener Daten im Rahmen der Diplomarbeit relevant sind. Maßgeblich ist insbesondere das in § 1 DSG verankerte Grundrecht auf Datenschutz beziehungsweise auf Geheimhaltung personenbezogener Daten. Darüber hinaus sind für „outPort“ vor allem § 4 DSG, § 6 DSG und § 24 DSG von Bedeutung.[2]

Das österreichische DSG schützt personenbezogene Daten dort, wo für die betroffene Person ein Interesse am Schutz und an der Geheimhaltung besteht. Daraus ergibt sich für „outPort“, dass jede Verarbeitung personenbezogener Daten einen klaren Grund braucht und nur für den jeweiligen Zweck den erforderlichen Umfang beinhalten darf, um rechtskonform zu sein.[2]

Zusätzlich ist zu beachten, dass das DSG in § 4 an die Datenschutzgrundverordnung, die für alle Unionsmitgliedstaaten gilt, anknüpft. In dieser Arbeit liegt der Fokus jedoch auf den nationalen Vorgaben des österreichischen DSG, insbesondere auf dem Geheimhaltungsanspruch, dem Datengeheimnis, dem Rechtsschutz der betroffenen Personen und den organisatorischen Anforderungen an einem regelkonformen Umgang mit personenbezogenen Daten.[2]

### 2.2.1 Personenbezogene Daten im System

Um die Einhaltung der DSG zu gewährleisten, muss zuerst definiert werden, welche personenbezogenen Daten im System verarbeitet werden und welche Personengruppen davon betroffen sind. Personenbezogene Daten entstehen durch die Verarbeitung von Daten natürlicher Personen. Im Rahmen dieser Diplomarbeit betrifft dies Daten von Besuchern, Lieferanten und in einem gewissen Ausmaß auch von Mitarbeitern, da Kontaktdaten des Arbeitsplatzes verwendet werden.[2]

Die im System benutzten Daten umfassen:

- **Identifikationsdaten:** Vorname, Nachname

- **Organisationsdaten:** zugehörige Organisation/Firma
- **Lieferungsdaten:** Kennzeichen des Lieferantenfahrzeugs und die Bestellnummer
- **Zeitpunkte:** der erwartete Check-in-Zeitpunkt, der tatsächliche Check-in-Zeitpunkt und der Check-out-Zeitpunkt
- **E-Mail-Adressen:** optionale E-Mail-Adresse, an die eine Benachrichtigung gesendet wird

Aus rechtlicher Sicht ist dabei nicht nur auf klassische Identifikationsdaten wie Namen zu achten, sondern auch Organisationsbezüge, Zeitangaben zu Anmeldungen, Kennzeichen, Bestellnummern oder E-Mail-Adressen können personenbezogene Daten darstellen, sofern diese einer Person zugeordnet werden können. Deshalb ist das Grundrecht auf Geheimhaltung nach § 1 DSG zu beachten.[2]

Dadurch wird die Frage relevant, wer innerhalb des Unternehmens auf diese Daten zugreifen darf. Nach § 6 DSG unterliegen personenbezogene Daten dem Datengeheimnis. Daraus folgt, dass Mitarbeiter personenbezogene Daten nur im Rahmen ihrer dienstlichen Aufgaben und unter Wahrung der Vertraulichkeit sehen und bearbeiten dürfen, sofern dies in einem Dienstvertrag oder einem zusätzlichen Dokument geregelt ist. Der Zugriff auf personenbezogene Daten ist daher durch rollenbasierten Zugriff auf jene Personen zu beschränken, die diese Informationen für ihre Arbeit benötigen.[2]

### 2.2.2 Zweckbindung und Datenminimierung

Durch das DSG ist definiert, dass bei jeder Verarbeitung personenbezogener Daten zu beachten ist, dass der Umgang mit diesen Daten auf einen klar bestimmbaren und sachlich rechtfertigbaren Zweck beschränkt bleibt. „outPort“ ist daher dazu gebunden nur Daten zu verarbeiten, die dem Zweck der Registrierung von Besuchern und Benachrichtigung von Mitarbeitern dienen. Daten, die für die Zutrittsverwaltung nicht erforderlich sind, müssen optional sein, dürfen den Vorgang der Registrierung nicht behindern und dürfen nur bei freiwilliger Ausfüllung gespeichert werden. Datenminimierung ist in diesem Zusammenhang essentiell, um die Einhaltung des § 1 DSG zu gewährleisten. [2]

Zu beachten ist die Verarbeitung personenbezogener Daten dann, wenn diese nicht nur der Zutrittsverwaltung dienen, sondern darüber hinaus zur Beobachtung und Bewertung von Verhaltensmustern. Statistiken sind daher nur insoweit zulässig, wenn sie einem legitimen, organisatorischen Zweck dienen und keine sachfremde Verhaltensüberwachung einzelner Personen

ermöglichen. Sofern im System statistische Übersichten vorgesehen sind, ist darauf zu achten, dass diese nur befugtem Personal, etwa Administratoren, zugänglich sind.[2]

### 2.2.3 Aufbewahrung und Löschung

Im Rahmen des Datenschutzgesetzes ist die Regelung der Speicherung, Aufbewahrung und Löschung personenbezogener Daten zu beachten, da sich aus Art. 5 Abs. 1 lit. e DSGVO [3] ergibt, dass personenbezogene Daten nicht länger gespeichert werden dürfen, als es für den ursprünglichen Zweck der Verarbeitung erforderlich ist.

Daher gibt es für die vorliegende Diplomarbeit einen Löschvorgang, der in einer relationalen Datenbank ausgeführt wird, um zusammen die Nachvollziehbarkeit und Konsistenz nach den datenschutzrechtlichen Vorgaben sicherstellt. Der Löschvorgang definiert eine Zeitspanne, in der die Daten gespeichert bleiben. Sobald dieser Zeitpunkt überschritten worden ist, werden die Einträge gelöscht. Damit wird sichergestellt, dass Datensätze nach dem Ende ihres vorgesehenen Zwecks nicht weiter benutzt werden.[3] Im Rahmen des Besucher- und Lieferantenmanagement-systems wird eine Speicherfrist von **30 Tagen** für personenbezogene Daten festgelegt. Diese Frist dient der klaren Zweckbindung gemäß Art. 5 Abs. 1 lit. e DSGVO [3] und stellt sicher, dass die Daten nur solange aufbewahrt werden, wie es erforderlich sind. Eine kürzere Frist würde die Möglichkeit einschränken, auf Nachfragen nachträglich reagieren zu können, während eine längere Speicherung das Risiko von Datenschutzverletzungen erhöht. Durch die Festlegung von 30 Tagen wird somit ein ausgewogenes Verhältnis zwischen notwendiger Verfügbarkeit der Daten und Minimierung der datenschutzrechtlichen Risiken gewährleistet.

Weiters ist zu beachten, dass auch andere, indirekt verbundene, Datenbanken bei der Speicherung der Daten zur Sicherung der Datenbestände die DSG einhalten müssen, welche der Verantwortung des Betreibers unterliegen.

## 2.3 Rollen- und Berechtigungskonzept

Im Projekt „outPort“ werden personenbezogene Besucherdaten verarbeitet. Aus diesem Grund ist ein klar definiertes Rollen- und Berechtigungskonzept erforderlich. Personenbezogene Daten sind Informationen, die eine Person direkt oder indirekt identifizierbar machen. Darunter fallen beispielsweise der Name, die Kontaktdaten oder der Besuchszeitpunkt. Das Ziel des Konzepts besteht darin, den Zugriff auf Funktionen und Daten so zu steuern, dass jeder Benutzer nur jene Möglichkeiten erhält, die für seine Aufgabe erforderlich sind. Dadurch

werden Fehlbedienungen reduziert, sensible Daten besser geschützt und die Anforderungen der Datenschutz-Grundverordnung (DSGVO) unterstützt. [4]

Für die Benutzer- und Rechteverwaltung wird Keycloak eingesetzt. Keycloak ist ein Identity- und Access-Management-System (IAM). Es ist eine zentrale Lösung für Anmeldung, Benutzerverwaltung und Zugriffskontrolle. In „outPort“ übernimmt Keycloak die Authentifizierung der Benutzer und stellt dem System die Informationen bereit, welche Rolle ein Benutzer besitzt. Auf Grundlage dieser Rolle trifft das Backend die Entscheidung über die freizugebenden Funktionen und Daten. Das Backend stellt den Serverteil der Anwendung dar, der für die Verarbeitung von Anfragen, die Speicherung von Daten und die Ausführung von Geschäftslogik zuständig ist.

### 2.3.1 Rollenbasierte Zugriffskontrolle – Grundprinzip (Least Privilege)

In „outPort“ wird eine rollenbasierte Zugriffskontrolle (RBAC) umgesetzt. RBAC steht für „Role-Based Access Control“, auf Deutsch „Zugriffssteuerung auf Basis von Rollen“. Das bedeutet, dass Berechtigungen nicht einzelnen Personen individuell zugewiesen werden, sondern über Rollen erfolgen. Eine Rolle bezeichnet ein vordefiniertes Profil, dem spezifische Rechte (erlaubte Aktionen) zugewiesen sind. Die Zuweisung einer Rolle an einen Benutzer ist die Grundlage für die Berechtigung zum Zugriff auf die entsprechenden Funktionen.

Dabei wird das Least-Privilege-Prinzip angewendet. Least-Privilege bedeutet, dass jeder Benutzer nur jene Berechtigungen erhält, die er für seine Aufgaben zwingend benötigt. Dieses Prinzip ist von entscheidender Bedeutung, da bei einem Besuchermanagementsystem eine Vielzahl von Daten als sicherheits- bzw. datenschutzrelevant einzustufen sind. Ein zu breiter Zugriff erhöht das Risiko von Datenmissbrauch oder unbeabsichtigten Änderungen. Das Vorgehen orientiert sich an etablierten Sicherheitskontrollen, insbesondere der Zugriffskontrolle und der Protokollierung [5].

Für die technische Umsetzung ist außerdem die Unterscheidung zwischen zwei Begriffen wichtig:

- Zu den erforderlichen Prozessen zählt die Authentifizierung, bei der der jeweilige Benutzer nachgewiesen wird (z. B. über die Anwendung Keycloak).
- Die Autorisierung bezeichnet die Entscheidung darüber, welche Berechtigungen dem Benutzer gewährt werden, wie beispielsweise der Zugriff auf bestimmte Funktionen und Daten.

In der aktuellen Implementierung sind zwei Rollen definiert. Die Rolle `admin` besitzt Vollzugriff auf alle vorgesehenen Funktionen. Die Rolle `reception` entspricht der Portier-Tätigkeit und ist auf operative Funktionen im Empfangsprozess beschränkt.

### 2.3.2 Rechte-Matrix

Die Rechte-Matrix dokumentiert, welche Rolle welche Funktionen ausführen darf. Die Basis bildet die aktuelle Umsetzung im Dashboard. Die Oberfläche enthält Navigationspunkte, die abhängig von der Rolle angezeigt werden. Darüber hinaus sind bestimmte Aktionen über API-Aufrufe realisiert. Eine API ist eine Programmierschnittstelle. In diesem Projekt sind damit HTTP-Endpunkte des Backends gemeint. Ein API-Endpunkt ist eine konkrete URL-Funktion, wie zum Beispiel /visitor/checkin.

Dabei ist zu beachten, dass eine Einschränkung im Frontend nur steuert, was in der Benutzeroberfläche sichtbar ist. Das Frontend stellt den Teil der Anwendung dar, der im Browser ausgeführt wird. Für eine sichere Umsetzung ist eine Autorisierung im Backend erforderlich, um das Umgehen von Funktionen durch direkte Aufrufe von URLs zu verhindern.

| Funktion (Bezug zur aktuellen Implementierung) | `reception` | `admin` |
|---|---|---|
| Aktive Trucker anzeigen | X | X |
| Trucker auschecken | X | X |
| Registrierte Besucher anzeigen | X | X |
| Besucher einchecken | X | X |
| Besucher manuell anlegen | X | X |
| Aktive Besucher anzeigen | X | X |
| Besucher auschecken | X | X |
| Besucherausweis als PDF herunterladen | X | X |
| Statistiken anzeigen | | X |
| Reports als PDF herunterladen | | X |
| Rechtliche Dokumente verwalten | | X |
| Einstellungen/Konfiguration | | X |

Tabelle 1: Rechte-Matrix auf Basis der Dashboard-Rollensteuerung (`admin` und `reception`)

Die Rolle „reception“ ist für operative Tätigkeiten ausgelegt. Zu den Aufgaben gehören die Bearbeitung aktueller Vorgänge (Check-in, Check-out, Anzeige aktiver bzw. registrierter Einträge) sowie das Anlegen neuer Besucher vor Ort. Funktionen, die administrative oder auswertende Aufgaben umfassen (Statistiken, Reports, Verwaltung rechtlicher Dokumente und Einstellungen), sind ausschließlich der Rolle „admin“ zugeordnet.

### 2.3.3 Zugriff auf sensible Daten und Logs

Neben der Freigabe der Funktionen ist der Umgang mit sensiblen Daten und Protokollen von wesentlicher Bedeutung. Besonders sensibel sind dabei personenbezogene Besucherdaten wie Name, Unternehmen, E-Mail-Adresse, erwarteter und tatsächlicher Check-in sowie Check-out. Zusätzlich können weitere, projektabhängige Zusatzfelder gespeichert werden (z. B. benutzerdefinierte Formularfelder). Um die Offenlegung dieser Daten zu minimieren, wird der Zugriff über Rollen eingeschränkt und die Anzeige auf den für die Aufgabe notwendigen Umfang begrenzt.

Logging bezeichnet die Protokollierung von Systemereignissen (z. B. Anmeldung, Check-in, Check-out, PDF-Erstellung). Audit-Logs sind Protokolle, die sicherheitsrelevante Aktionen nachvollziehbar dokumentieren. So kann später festgestellt werden, wer welche Aktion zu welchem Zeitpunkt ausgelöst hat. Da Protokolle sensible Informationen enthalten können (z. B. Benutzerkennung, Zeitstempel, betroffene Datensätze), wird der Zugriff darauf restriktiv geregelt.

In diesem Rollenmodell ist vorgesehen, dass der Zugriff auf systemweite Logs und Audit-Logs der Rolle `admin` vorbehalten ist. Die Rolle `reception` erhält keinen Zugriff auf detaillierte Protokolle, da dies für die operative Arbeit nicht erforderlich ist. Die Umsetzung folgt dem Prinzip, dass Zugriffskontrolle und Nachvollziehbarkeit sich ergänzen [5].

## 2.4 Anforderungen

Wie bereits in den vorangegangenen Kapiteln dargelegt, wurden Problemstellung, Zielsetzung und Projektergebnis in den Abschnitten 1.2, 1.4 und 1.6 ausführlich beschrieben. In diesem Kapitel werden die daraus abgeleiteten Ziele sowie die wichtigsten Anforderungen zusammengefasst. Die Anforderungen dienen als Orientierungshilfe für die Umsetzung und stellen sicher, dass der Funktionsumfang klar abgegrenzt ist.

### 2.4.1 Funktionale Anforderungen

Die funktionalen Anforderungen definieren die wesentlichen Funktionen, die „outPort“ bereitstellt. Im Fokus stehen dabei ein effizienter Ablauf am Empfang sowie die Unterstützung von wiederkehrenden Prozessen wie dem Check-in von Besuchern und Lieferanten.

**Effizienter Check-in:** Die Erfassung von Besucher- und Lieferantenvorgängen muss so gestaltet sein, dass sie im Empfangsalltag in kurzer Zeit durchgeführt wird. Zu den relevanten Aspekten zählen insbesondere die automatische Speicherung des Check-in-Zeitpunkts sowie die Reduzierung der erforderlichen Eingaben auf das notwendige Minimum.

**Unterstützung von Voranmeldungen:** Für geplante Besuche ist eine Voranmeldung möglich, um die Datenerfassung am Empfang zu reduzieren. Um in Fällen leicht abweichender Eingaben (z. B. Tippfehler oder unterschiedliche Schreibweisen) eine Voranmeldung schnell auffinden zu können, wird eine fehlertolerante Suche (Fuzzy Search) eingesetzt. Die Voranmeldung dient außerdem als Grundlage für die automatische Benachrichtigung des Gastgebers.

**Dokumentenbestätigung:** Im Rahmen des Check-ins sind die relevanten Dokumente anzuzeigen. Diese umfassen sowohl rechtliche Informationen, wie beispielsweise die Datenschutz-Grundverordnung, als auch organisatorische Inhalte wie zum Beispiel Sicherheitsunterweisungen. Die Konfiguration, welche Dokumente in der Anmeldemaske angezeigt werden, ist im Dashboard möglich.

**Benachrichtigung des Gastgebers:** Sobald ein vorab angemeldeter Besucher eintrifft, wird der zuständige Gastgeber automatisch benachrichtigt. Dadurch wird der manuelle Aufwand am Empfang reduziert und die Wartezeiten für Besucher werden verkürzt. Die Benachrichtigung erfolgt per E-Mail.

**Automatisches Drucken eines Besucherausweises:** Nach dem erfolgreichen Check-in wird ein Besucherausweis erstellt und ausgedruckt. Dieser dient der eindeutigen Identifikation am Standort und erfüllt Sicherheitsanforderungen. Der Ausweis wird auch im Dashboard als PDF zur Verfügung gestellt.

**Zentrale Übersicht (Dashboard):** Das Dashboard ist für Administratoren sowie für Empfangspersonal zugänglich. Allerdings haben diese unterschiedliche Rechte und Ansichten. Beide dürfen aktuelle Lieferanten sowie aktuelle und geplante Besucher einsehen. Personen, die bereits ausgecheckt wurden, werden nicht mehr in der Übersicht angezeigt, bleiben jedoch in der Datenbank als historische Einträge erhalten. Darüber hinaus hat der Administrator zusätzlich folgende Möglichkeiten: Statistiken einsehen, Berichte erstellen lassen und Dokumente, die in der Anmeldemaske angezeigt werden, überarbeiten.

**Administration und Konfiguration:** Administrationsfunktionen dienen dazu, die Lösung an den Betrieb anzupassen. Dazu zählen konfigurierbare Formulare (zusätzliche Felder), die Verwaltung von Dokumenten sowie grundlegende Einstellungen zur Darstellung (Logo). Dadurch ist ein standortspezifischer Einsatz des Systems ohne Codeänderungen möglich.

**Rollen und Berechtigungen:** Da in der Anwendung personenbezogene Daten verarbeitet werden, müssen Zugriffe auf die Daten rollenbasiert eingeschränkt werden. Operative Tätigkeiten am Empfang und administrative Funktionen (Konfiguration, Auswertungen) sind getrennt. Dadurch sehen Benutzer nur jene Funktionen, die für ihre Aufgaben erforderlich sind. Dies entspricht dem Least-Privilege-Prinzip.

**Mehrsprachigkeit und Branding:** Die Benutzeroberfläche muss mehrere Sprachen unterstützen, um die Bedürfnisse unterschiedlicher Nutzergruppen zu erfüllen. Zusätzlich ist ein einheitliches Erscheinungsbild vorhanden, damit die Lösung an unterschiedlichen Standorten konsistent eingesetzt werden kann.

### 2.4.2 Nicht-funktionale Anforderungen

Nicht-funktionale Anforderungen beschreiben Qualitätsmerkmale, die für den Einsatz am Empfang von entscheidender Bedeutung sind. Eine Portierlösung wird unter Zeitdruck bedient, verarbeitet personenbezogene Daten und muss daher sowohl benutzerfreundlich als auch sicher und reaktionsschnell sein.

**Usability:** Die Eingabemaske ist so zu gestalten, dass sie effizient bedienbar ist. Durch Pflichtfelder und Validierung werden Fehleingaben reduziert und die Bedienung bei hoher Auslastung erleichtert. Mehrsprachige Texte unterstützen unterschiedliche Besuchergruppen und erhöhen die Akzeptanz der Lösung.

**Sicherheit und Datenschutz:** Die Erhebung und Anzeige personenbezogener Daten ist auf das erforderliche Maß zu beschränken. Es ist essenziell, dass Zugriffe rollenbasiert eingeschränkt werden, um operative und administrative Bereiche klar zu trennen.

**Performance und Zuverlässigkeit:** Häufig genutzte Ansichten reagieren ohne spürbare Verzögerung, um den Prozessfluss nicht zu unterbrechen. Darüber hinaus ist sicherzustellen, dass

Fehlerfälle verständlich behandelt und angezeigt werden, beispielsweise bei ungültigen Eingaben.

## 2.5 UI/UX-Methodik

Für das Besuchermanagement-Frontend und das Self-Service-Terminal ist eine durchdachte UI/UX-Methodik von entscheidender Bedeutung, da die Anwendung von unterschiedlichen Benutzergruppen genutzt wird. UI steht für „User Interface“, also Benutzeroberfläche. Darunter fallen alle sichtbaren Elemente wie Buttons, Eingabefelder und Navigationselemente. UX steht für „User Experience“ und beschreibt, wie einfach und intuitiv die Bedienung im gesamten Ablauf ist. Im Projekt „outPort“ liegt der Schwerpunkt auf einer intuitiven Bedienbarkeit, kurzen Durchlaufzeiten und einer möglichst fehlerfreien Datenerfassung.

Die Umsetzung erfolgt gemäß einem prozessorientierten Vorgehen. Zunächst werden die typischen Abläufe (z. B. Anmeldung eines LKW-Fahrers, operative Vorgänge im Dashboard) in einzelne Schritte zerlegt. Für jeden Schritt werden notwendige Eingaben, Pflichtbedingungen und mögliche Fehlerfälle festgelegt. Aus diesen Anforderungen ergeben sich Regeln für die terminaltaugliche Benutzerführung, Validierung sowie für Mehrsprachigkeit und Branding, die in den folgenden Abschnitten detailliert erläutert werden.

### 2.5.1 Terminalgeeignete Benutzerführung (Kiosk-Flow als Prinzip)

Der Begriff Terminalbetrieb beschreibt die Nutzung von Anwendungen an einem stationären Gerät, häufig per Touchscreen. Die Benutzerführung orientiert sich am Kiosk-Flow. Ein Kiosk-Flow beschreibt eine fest definierte, möglichst lineare Abfolge von Schritten, die den Benutzer ohne Umwege zum Ziel führt.

Die Anmeldemaske für LKW-Fahrer wurde als kompakte, zentrierte Benutzeroberfläche umgesetzt. Der Bildschirm fokussiert auf die wesentlichen Eingaben (vollständiger Name, Kennzeichen, Bestellnummer) und führt den Benutzer ohne zusätzliche Navigation direkt zur Anmeldung. Zusätzlich sind Pflichtdokumente in den Prozess integriert, wodurch der Ablauf ohne Wechsel in andere Systeme abgeschlossen wird. Die Oberfläche ist in mehreren Sprachen verfügbar, ohne dass sich der Ablauf oder das Layout verändert.

Auch im Dashboard wird die Benutzerführung über eine klar strukturierte Navigation umgesetzt. Die Navigation ist links positioniert und ermöglicht einen schnellen Wechsel zwischen operativen Bereichen. Zusätzlich ist die Navigation rollenabhängig reduziert. In der Empfangs-Ansicht sind

nur jene Bereiche sichtbar, die für den Empfangsprozess notwendig sind (siehe Abbildung 3). Diese Reduktion unterstützt die Orientierung, verkürzt Entscheidungswege und reduziert das Risiko, selten genutzte oder administrative Funktionen versehentlich aufzurufen.

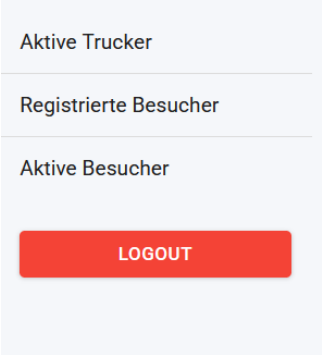



Abbildung 3: Reduzierte Navigation im Dashboard für reception.

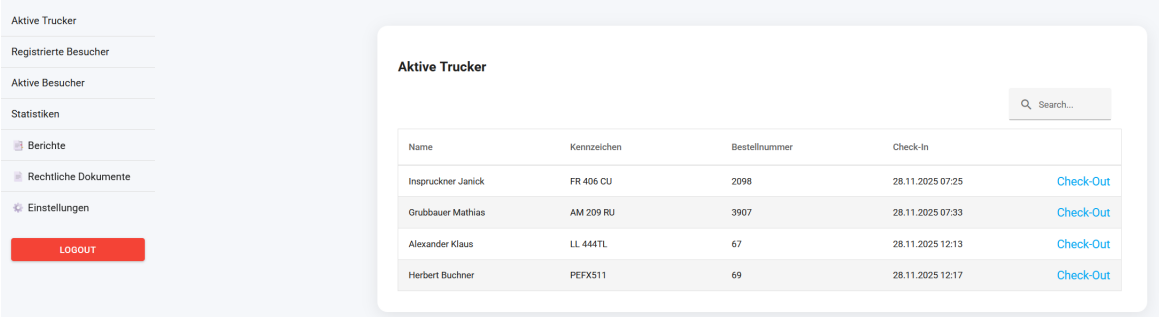



Abbildung 4: Dashboard-Ansicht Navigation und Tabelle aktive Trucker.

### 2.5.2 Validierungskonzept (Pflichtfelder, Plausibilität)

Der Begriff „Validierung“ bezeichnet den Prozess der Prüfung von Eingaben und Bedingungen, bevor eine Aktion ausgeführt oder Daten gespeichert werden. Plausibilität beschreibt die inhaltliche Sinnhaftigkeit einer Eingabe. Dies bedeutet, dass leere Namen oder unplausible Werte nicht zulässig sind. Das Ziel besteht darin, die Anzahl der Eingabefehler zu reduzieren und gleichzeitig die Datenqualität zu erhöhen.

In der Terminalmaske wird die Validierung durch eine klare Freischaltlogik sichtbar. Es ist zu beachten, dass eine Anmeldung erst möglich ist, nachdem die erforderlichen Angaben gemacht wurden und die Pflichtdokumente bestätigt sind (siehe Abbildung 8). Zusätzlich wird ein Pflichtdokument (AGB) eingebunden. AGB sind Allgemeine Geschäftsbedingungen, also Vertragsbedingungen, die vor der Anmeldung akzeptiert werden müssen. Durch die Option zum Öffnen des Dokuments wird sichergestellt, dass die Inhalte vor der Bestätigung eingesehen werden können.

Das Dashboard unterstützt die Fehlervermeidung durch eine strukturierte Darstellung und eine schnelle Auffindbarkeit. Wie in Abbildung 4 ersichtlich, steht eine Suchfunktion zur Verfügung, welche das Auffinden von Einträgen erleichtert. Dadurch wird die Wahrscheinlichkeit, dass Vorgänge auf falschen Datensätzen durchgeführt werden, signifikant gesenkt.

Das Validierungskonzept ist zweistufig aufgebaut:

- Frontend-Validierung: Prüfung direkt in der Oberfläche, um dem Benutzer sofort Rückmeldung zu geben.
- Backend-Validierung: Prüfung im Serverteil der Anwendung, um fehlerhafte oder manipulierte Anfragen zu verhindern.

### 2.5.3 Mehrsprachigkeit und Branding: Anforderungen und Regeln

Mehrsprachigkeit bezeichnet die Verfügbarkeit der Oberfläche in mehreren Sprachen. Branding beschreibt die Anpassung der Oberfläche an das Erscheinungsbild eines Unternehmens. Dies umfasst die Implementierung eines Logos sowie definierter Gestaltungselemente, um eine konsistente visuelle Identität zu schaffen. Das Erscheinungsbild eines Unternehmens wird als Corporate Design bezeichnet und beschreibt einheitliche Gestaltungsregeln.

Die Mehrsprachigkeit ist direkt am Einstieg der Terminalmaske umgesetzt. Dabei bleiben Layout, Reihenfolge und Bedienlogik unverändert, während alle sichtbaren Texte übersetzt werden. Die Konsistenz des Systems reduziert den Lernaufwand und erhöht die Verständlichkeit für internationale Nutzer.

Für Mehrsprachigkeit gelten folgende Anforderungen:

- Alle sichtbaren Texte müssen übersetzbar sein (Buttons, Überschriften, Feldbezeichnungen und Hinweise).
- Die Sprachwahl soll früh im Ablauf stattfinden und während des Prozesses beibehalten werden.
- Bei fehlenden Übersetzungen muss eine definierte Standardsprache verwendet werden, damit keine leeren Texte entstehen.

Branding wird in der Terminalmaske durch die Einbindung des Unternehmenslogos sichtbar. Dabei ist sicherzustellen, dass Branding die Bedienbarkeit nicht negativ beeinflusst. Insbesondere ist auf eine gute Lesbarkeit und einen hohen Kontrast zu achten.

# 3 Produkt

In diesem Kapitel wird die praktische Umsetzung des Systems „outPort“ beschrieben. Dabei werden die zentralen Funktionen und Abläufe erläutert sowie deren Realisierung im System dargestellt.

Der Fokus liegt auf der Funktionalität des Admin-Dashboards, der Registrierung und Anmeldung von Besuchern und Lieferanten sowie den Self-Service-Komponenten. Zusätzlich werden die angebundenen Integrationen, wie Kalender, Benachrichtigungen und das Ausweissystem, beschrieben. Ziel ist es, einen nachvollziehbaren Einblick in die technische Umsetzung der zuvor definierten Anforderungen zu geben.

## 3.1 Systemkonzept und Architektur

Als Grundlage des Aufbaus fungiert das Client-Server-Prinzip, welches in Client-Server-Prinzip näher erläutert wird, dass auf den zwei fundamentalen Säulen, dem Server und dem Client basiert. Dazu gibt es noch einen eigenen Authentifizierungs-Service, der den gesicherten Zugriff auf „outPort“ sichert und gleichzeitig für die Benutzerrollenverwaltung verantwortlich ist und unter „Rollenbasierte Zugriffskontrolle – Grundprinzip (Least Privilege)“ erklärt worden ist. Für das Testen der REST-Schnittstellen wurde die Swagger-API benutzt. Darüber hinaus werden die Daten in einer MSSQL-Datenbank persistiert, welche in den unteren Abschnitten als auch in Datenbank erklärt wird.

### 3.1.1 Client-Server-Prinzip

Es gibt bei dem Prinzip der Client-Server-Kommunikation zwei Ansätze, welche zum einen die 2-Tier-Architektur und zum anderen die 3-Tier-Architektur sind. Bei ersterem wird die Kommunikation zwischen dem Client-Layer und dem Data-Layer implementiert. Dieser Ansatz ist für die Diplomarbeit jedoch nicht geeignet, da keine klare Trennung zwischen Präsentationslogik, Anwendungslogik und Datenhaltung besteht. Dadurch wird eine Persistierung der Daten nur unzureichend unterstützt. Aus diesem Grund wird in dieser Arbeit die 3-Tier-Architektur verwendet. In der es die Presentation-Tier, Logic-Tier und die Data-Tier gibt, die verschiedene

Aufgabenbereiche der Applikation erfüllen. Dabei übernimmt die Logic-Tier die Business-Logik und die Kommunikation mit der Data-Tier. Die Data-Tier dient der Persistenz der Daten und liefert Daten an die Logic-Tier. Zuletzt bildet die Presentation-Tier die Benutzeroberfläche und gibt die Benutzerinteraktionen an das Logic-Tier weiter. In der unteren Abbildung 5 wird dargestellt, wie die einzelnen Ebenen miteinander arbeiten und wie diese zusammenhängen.

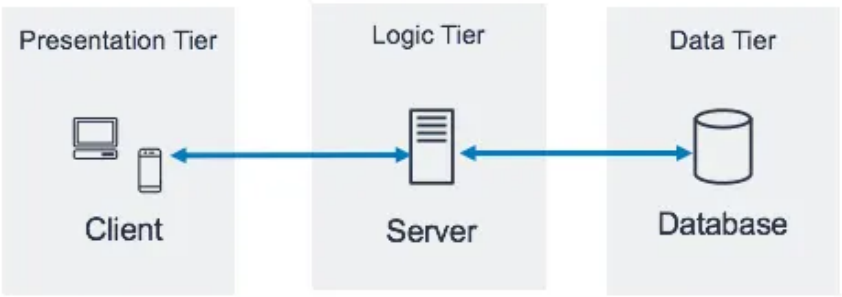



Abbildung 5: Darstellung eines 3-Schichten-Client-Server-Prinzips

Für den Zwischenschritt der Authentifizierung und Autorisierung wird der externe Identity und Access Management Service Keycloak verwendet. Dabei übernimmt die Presentation-Tier die automatische Weiterleitung zu Keycloak im Falle einer Erstanmeldung. Nach erfolgreicher Authentifizierung stellt Keycloak einen Bearer-Token bereit, der bei Requests ans Backend mitgeschickt wird und Informationen über den Zugriff auf verschiedene Funktionen und Daten beinhaltet, die bei dem Client angezeigt werden dürfen.

### 3.1.2 Komponentenübersicht (Frontend, Backend, Datenbank)

Das **Frontend** wurde mit React programmiert und ist die Benutzeroberfläche, die jeder, wenn auch nur zum Teil, sehen kann. Die Navigation innerhalb der Anwendung wird durch React Router umgesetzt, der definiert, welche Komponente bei welcher URL angezeigt wird. Die einzelnen UI-Elemente wie Datenraster, Diagramme und Formulare werden von der DevExtreme Bibliothek zur Verfügung gestellt. Zusätzlich ermöglicht die Integration von `i18next` eine mehrsprachige Benutzeroberfläche, die in der Einstellungsansicht angepasst werden kann. Zuletzt enthält das Frontend getrennte Raster für Besucher, Lieferanten, Statistiken, rechtliche Dokumente und Einstellungen.

Das **Backend** wurde mit ASP.NET Core implementiert und stellt die Logik des Systems bereit. Die Kommunikation zwischen Frontend und Backend erfolgt über REST-API-Endpunkte. Im Backend gibt es mehrere Controller, die die HTTP-Requests entgegennehmen und bestimmte

Aufgabenbereiche kapseln. Dazu gehört die Verarbeitung von Besucherdaten, die Verwaltung von Lieferanten, die Bereitstellung und Verwaltung rechtlicher Dokumente, die Logo-Verwaltung sowie E-Mail-Funktionen. Die eigentliche Geschäftslogik ist wiederum in Services implementiert, die dann die Anfrage umsetzen und den Status des Erfolgs an den Controller zurückgeben.

Die **Datenbank** dient der persistenten Speicherung aller Benutzerdaten. Die Anbindung erfolgt über Entity Framework Core mit einem `AppDbContext`. Auf dieser Grundlage werden Besucher, Lieferanten und rechtliche Dokumente als Entitäten verwaltet.

Zusammengefügt bilden diese drei Säulen dann die 3-Tier-Architektur mit Presentation-Tier im Browser, Logic-Tier im Webserver und Data-Tier in einer relationalen Datenbank.

### 3.1.3 Datenmodell-Konzept

Das Datenmodell besteht aus den Entitäten `Visitor`, `Trucker` und `LegalDocument`. Diese Entitäten beinhalten jene Attribute, die für Check-in, Check-out sowie Einstellungen und weitere Funktionen notwendig sind.

Die Entität `Visitor` enthält die Daten eines Besuchers. Diese bestehen aus dem Vornamen und Nachnamen, der Organisation, dem geplanten Check-in, dem tatsächlichen Check-in, dem Check-out, der optionalen E-Mail-Adresse und der vorgesehenen Besuchsdauer. Dabei wurde darauf geachtet, durch Werte wie den tatsächlichen Check-in Situationen aus der realen Welt abzubilden.

Die detaillierten Attribute der Visitor-Entität sind in der untenstehenden Tabelle 2 zusammengefasst.

Die Entität `Trucker` repräsentiert Lieferanten. Hier werden insbesondere Name, Kennzeichen, Bestellnummer sowie Check-in- und Check-out-Zeitpunkte gespeichert. Im Unterschied zur Entität `Visitor` wird ein Lieferant anders modelliert, da die jeweiligen Attribute aus datenhaltungstechnischen Gründen nur teilweise für die jeweils andere Entität relevant sind.

Die detaillierten Attribute der Trucker-Entität sind in der untenstehenden Tabelle 3 zusammengefasst.

Die Entität `LegalDocument` dient zur Verwaltung von Dokumenten, die Besuchern vor dem Check-in angezeigt oder administrativ verwaltet werden können. Dazu gehören Titel, Dateiname, Dokumentinhalt, Benutzergruppe, Pflichtstatus und Erstellungszeitpunkt. Dieses Modell ermöglicht es, für unterschiedliche Benutzergruppen unterschiedliche Dokumente bereitzustellen, etwa für Besucher oder Lieferanten.

| Attribut | Datentyp | Beschreibung |
|---|---|---|
| Id | int | Eindeutige Identifikationsnummer des Besuchers. |
| FirstName | string | Speichert den Vornamen des Besuchers. |
| LastName | string | Speichert den Nachnamen des Besuchers. |
| Organization | string | Gibt die Organisation, Firma oder Institution an, der der Besucher angehört. |
| CheckIn | DateTime? | Speichert den tatsächlichen Zeitpunkt des Check-ins. |
| CheckOut | DateTime? | Speichert den tatsächlichen Zeitpunkt des Check-outs. |
| CheckInExpected | DateTime? | Gibt den erwarteten beziehungsweise geplanten Check-in-Zeitpunkt an. |
| Mail | string? | Optional: E-Mail-Adresse des Besuchers für Benachrichtigungen. |
| VisitDuration | int? | Optional: Geplante Besuchsdauer in Minuten. |

Tabelle 2: Attribute der Visitor-Entität

| Attribut | Datentyp | Beschreibung |
|---|---|---|
| Id | int | Eindeutige Identifikationsnummer des Truckers. |
| FullName | string | Speichert den vollständigen Namen des Truckers. |
| LicensePlate | string | Enthält das Kennzeichen des zugehörigen Fahrzeugs. |
| OrderNumber | string | Speichert die zugehörige Auftragsnummer der Lieferung oder Dienstleistung. |
| CheckIn | DateTime? | Zeitpunkt, zu dem der Trucker eingecheckt wurde. |
| CheckOut | DateTime? | Zeitpunkt, zu dem der Trucker ausgecheckt wurde. |

Tabelle 3: Attribute der Trucker-Entität

Die detaillierten Attribute der LegalDocument-Entität sind in der untenstehenden Tabelle 4 zusammengefasst.

| Attribut | Datentyp | Beschreibung |
|---|---|---|
| Id | int | Eindeutige Identifikationsnummer des Dokuments. |
| Title | string | Enthält den Titel des rechtlichen Dokuments. |
| FileName | string | Speichert den Dateinamen des hinterlegten Dokuments. |
| FileData | byte[] | Enthält den binären Inhalt der Datei (digitales Dokument). |
| UserType | string | Gibt an, für welche Benutzergruppe das Dokument vorgesehen ist (`visitor` oder `trucker`). |
| Required | bool | Kennzeichnet, ob es sich um ein verpflichtendes Dokument handelt. |
| CreatedAt | DateTime | Speichert den Zeitpunkt der Erstellung bzw. Hinterlegung des Dokuments. |

Tabelle 4: Attribute der LegalDocument-Entität

### 3.1.4 Integrationskonzept (E-Mail, Kalender, Druck)

Für die Integration von **E-Mails** war die Idee, einem Gastgeber in dem Unternehmen eine Benachrichtigung per E-Mail zu senden, wenn die E-Mail-Adresse in einem optionalen Feld ausgefüllt wurde bei einer Vorabanmeldung des Besuchs, dieser Besuch erschienen ist und der Portier den Eintrag des Besuchers eincheckt. Nachdem der Check-in-Prozess durchgegangen ist, bekommt der Mitarbeiter mit der hinterlegten E-Mail-Adresse, eine Benachrichtigung per E-Mail. Dazu wird im Backend ein E-Mail-Service eingesetzt, der über SMTP Nachrichten versendet. Zusätzlich ist eine Anbindung an Microsoft Graph vorhanden, um Besucherbenachrichtigungen an die hinterlegten Mitarbeiter zu senden. **Microsoft Graph** ist eine Schnittstelle von Microsoft, über die auf verschiedene Dienste innerhalb von Microsoft 365, wie beispielsweise Outlook, zugegriffen wird.

Das **Kalenderkonzept** basiert ebenfalls auf Microsoft Graph und funktioniert folgendermaßen: Wenn ein Besucher im Vorhinein angemeldet wird, wird ein Kalendereintrag erstellt mit den übergebenen Attributen des Tabelleneintrags, sodass zuständige Gastgeber geplante Besuche direkt in ihrem Kalender angezeigt bekommen.

Im Bereich **Druck** werden zwei Funktionen bereitgestellt. Zum einen können Besucherausweise als PDF-Dateien generiert werden, die den Namen des Besuchers, die Organisation, die Besuchszeit sowie einen QR-Code für den Check-out enthalten, dazu wird QuestPDF und QRCoder verwendet die näher in „QuestPDF“ erklärt werden. Wohingegen die Berichtserstellung mit DevExpress Reporting umgesetzt wurde und ermöglicht Berichte, im PDF-Format zu exportieren.

## 3.2 Überblick und Benutzerrollen

Das System wurde so konzipiert, dass unterschiedliche Benutzergruppen jeweils nur jene Funktionen sehen und verwenden können, die für ihren Aufgabenbereich relevant sind. Die Umsetzung der rollenbasierten Zugriffskontrolle erfolgt mit **Keycloak**. Dafür wird ein eigener Realm mit der Bezeichnung `sco_portier` verwendet.

Die Authentifizierung der internen Benutzer erfolgt über Keycloak. Nach erfolgreicher Anmeldung stellt Keycloak ein Token bereit, in dem unter anderem die zugewiesenen Benutzerrollen enthalten sind. Im Frontend werden diese Rollen über `realm_access.roles` ausgelesen und zur Steuerung der sichtbaren Menüpunkte und geschützten Ansichten verwendet.

Für die Umsetzung wurden zwei interne Rollen definiert:

- **admin**
- **reception**

Diese Rollen werden in Keycloak als **Realm-Rollen** angelegt und anschließend über das **Role Mapping** einzelnen Benutzern zugewiesen. Zusätzlich gibt es noch die Rollen der Besucher/Lieferanten und der Gastgeber, die nicht in Keycloak angelegt wurden, da diese keinen Zugriff auf die Anwendung haben beziehungsweise nur die Registrierungsseite sehen und deswegen keine spezifische Rolle benötigen.

### 3.2.1 Portier bzw. Empfang

Die Rolle **Portier/Empfang** ist für den Empfang vorgesehen. Benutzer mit dieser Rolle erhalten Zugriff auf jene Funktionen, die für die Verwaltung von Besuchern und Lieferanten notwendig sind. Im Dashboard sieht ein Benutzer mit der Rolle `reception` ausschließlich die Menüpunkte für:

- aktive Trucker
- registrierte Besucher
- aktive Besucher

Damit kann der Empfang bereits vorangemeldete Besucher einsehen, diese beim Eintreffen einchecken und nach Abschluss des Besuchs wieder auschecken. Ebenso können aktive Trucker verwaltet und ausgecheckt werden.

Technisch wird dies im Frontend dadurch umgesetzt, dass die Navigationspunkte des Dashboards mit einer Rollenliste versehen sind. Vor dem Rendern der Navigation wird geprüft, ob die dem Benutzer im Keycloak-Token zugewiesenen Rollen mit den erlaubten Rollen des jeweiligen Menüpunktes übereinstimmen. Für die Empfangsrolle werden daher nur jene Ansichten eingeblendet, die mit `reception` verknüpft sind. Funktionen wie Statistik, Berichte, Einstellungen oder die Verwaltung rechtlicher Dokumente sind für diese Rolle nicht sichtbar.

### 3.2.2 Admin

Die Rolle **Admin** besitzt den Zugriff auf alle Ansichten innerhalb des Systems. Ein Benutzer mit der Rolle `admin` sieht im Dashboard nicht nur die operativen Bereiche des Empfangs, sondern zusätzlich auch administrative und auswertungsbezogene Funktionen. Dazu zählen insbesondere:

- Statistik
- Berichte
- Verwaltung rechtlicher Dokumente
- Einstellungen

Der Administrator kann somit Berichte exportieren, Pflichtdokumente für Besucher oder Lieferanten hochladen und Systemeinstellungen wie Sprache, Theme oder Logo anpassen zusätzlich zu den Funktionen und Ansichten, die der Empfang einsehen darf.

Die Umsetzung erfolgt gleichfalls über Keycloak. Im Frontend wird geprüft, ob der Benutzer die Rolle `admin` besitzt. Ist dies der Fall, werden zusätzliche Ansichten freigeschaltet. Darüber hinaus ist in der Einstellungsseite selbst nochmals eine Rollenprüfung enthalten.

### 3.2.3 Gastgeber (Mitarbeiter)

Die Rolle des **Gastgebers** beziehungsweise Mitarbeiters ist nicht als eigenständige Benutzerrolle mit direktem Zugriff auf die Applikation umgesetzt. Mitarbeiter verwenden weder das Dashboard noch andere Ansichten der Anwendung. Ihre Einbindung erfolgt ausschließlich indirekt über Benachrichtigungen und Kalendereinträge.

Wird bei der Vorabanmeldung eines Besuchers die E-Mail-Adresse des zuständigen Mitarbeiters hinterlegt, wird dieser beim Eintreffen des Besuchers automatisch benachrichtigt. Zum Zeitpunkt der Vorabanmeldung wird mit Microsoft Graph ein entsprechender Kalendereintrag

für den Mitarbeiter erstellt. Der Gastgeber erhält somit alle Informationen außerhalb der Anwendung.

### 3.2.4 Besucher und Lieferanten

**Besucher** und **Trucker** beziehungsweise Lieferanten greifen nicht auf das interne Dashboard zu, sondern ausschließlich auf eine jeweils dafür vorgesehene Registrierungsseite. Dort geben sie die für ihre Anmeldung erforderlichen Daten ein, beispielsweise Name, Organisation.

Vor dem Absenden der Registrierung müssen gegebenenfalls erforderliche Dokumente bestätigt werden, die durch die Admin-Rolle hinterlegt wurden. Nach erfolgreicher Eingabe der Daten wird die Anmeldung abgeschlossen und eine Bestätigungsseite angezeigt. Weitere Funktionen innerhalb der Anwendung stehen diesen Benutzergruppen nicht zur Verfügung.

## 3.3 Besucherprozess

Der Besucherprozess in „outPort“ ist so gestaltet, dass er den gesamten Ablauf von der ersten Planung bis zum Verlassen des Standorts digital abbildet. Das Ziel ist es, die Kommunikation zwischen dem Empfang, dem Gastgebern und den Besuchern zu automatisieren und Fehlerquellen zu minimieren. Der Prozess unterteilt sich dabei in mehrere Phasen, die sicherstellen, dass alle Informationen vorhanden sind und rechtliche Vorgaben eingehalten werden. Im Folgenden werden die einzelnen Schritte beschrieben.

### 3.3.1 Voranmeldung

Die Voranmeldung eines Besuchers erfolgt in „outPort“ direkt über das Dashboard. Dafür gibt es eine eigene Eingabemaske, über welche die wichtigsten Daten für den geplanten Besuch erfasst werden. Wie in Abbildung 6 dargestellt, umfassen die erforderlichen Angaben Vorname, Nachname, Organisation, eine optionale E-Mail-Adresse (für die Benachrichtigung des Gastgebers), den erwarteten Zeitpunkt der Registrierung sowie die geplante Dauer des Besuchs. Zudem kann angegeben werden, ob es sich um einen ganztägigen Besuch handelt.

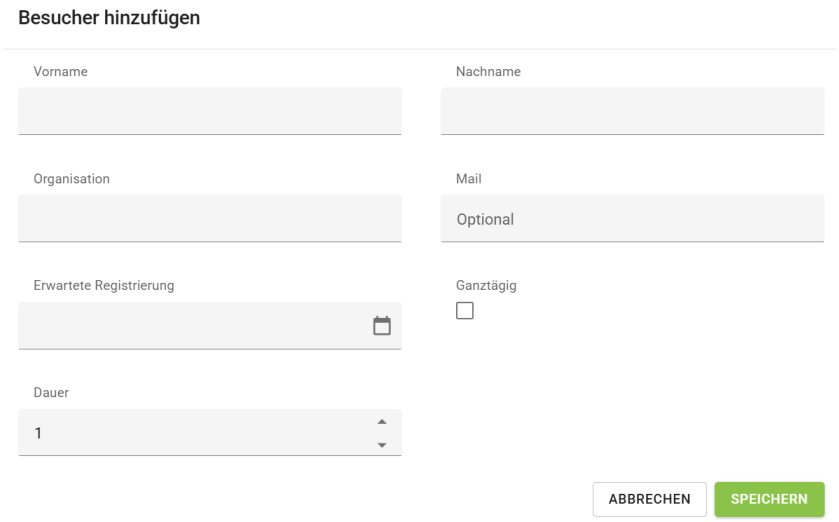



Abbildung 6: Maske im Dashboard Voranmeldung Besucher

Nach dem Speichern wird der Besucher als geplanter Eintrag im System angelegt. Die vorangemeldeten Besucher werden in einer eigenen Übersicht im Dashboard angezeigt. Eine solche Übersicht ist in Abbildung 7 dargestellt. Sie hilft dem Empfang dabei, erwartete Besucher schnell zu finden und den Überblick über anstehende Termine zu behalten.

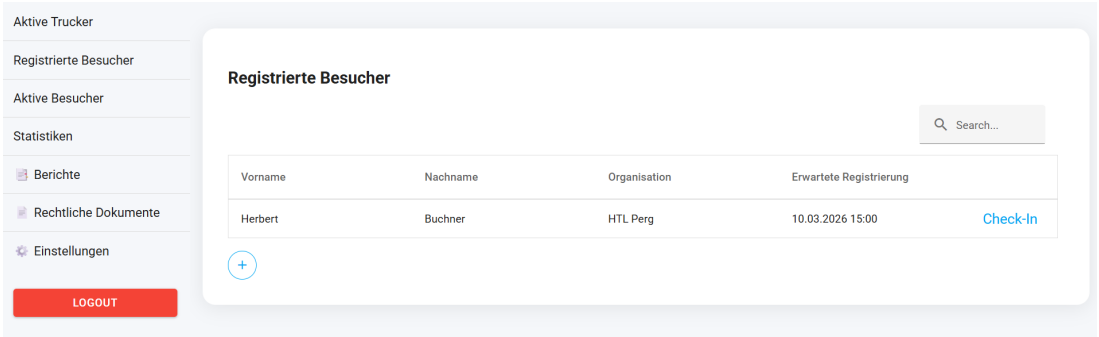



Abbildung 7: Dashboard-Übersicht vorangemeldete Besucher

Die Voranmeldung dient damit vor allem der organisatorischen Vorbereitung des Besuchs. Gleichzeitig bildet sie die Grundlage für die spätere Benachrichtigung des Gastgebers und eine zentrale Übersicht über geplante Besuche.

### 3.3.2 Check-in am Empfang

Beim Eintreffen eines Besuchers erfolgt die Anmeldung am Empfang über eine eigene Check-in-Maske. Diese Maske ist in Abbildung 8 dargestellt. Der Besucher gibt dort die erforderlichen Daten wie Vorname, Nachname und Organisation ein. Zusätzlich können in diesem Schritt verpflichtende Dokumente angezeigt werden, welche vor dem Abschluss der Anmeldung bestätigt werden müssen.

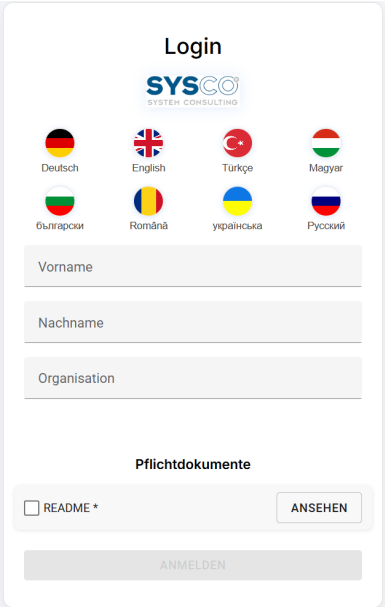



Abbildung 8: Anmeldemaske Check-in Besucher am Empfang.

Die Maske ist bewusst einfach aufgebaut, damit der Ablauf auch unter Zeitdruck schnell durchgeführt werden kann. Durch die mehrsprachige Darstellung eignet sich der Check-in außerdem für unterschiedliche Besuchergruppen. Der Empfang wird dadurch entlastet, da der Besucher die Eingaben selbst vornimmt und nicht alle Daten manuell durch das Personal erfasst werden müssen.

Nach erfolgreicher Anmeldung wird der Besucher im System als aktuell anwesend geführt. Im Dashboard scheint dieser Eintrag anschließend in der Übersicht der aktiven Besucher auf. Diese Ansicht ist in Abbildung 9 dargestellt. Dort sind die aktuell anwesenden Personen mit den wichtigsten Informationen sowie der Zeitpunkt des Check-ins ersichtlich.

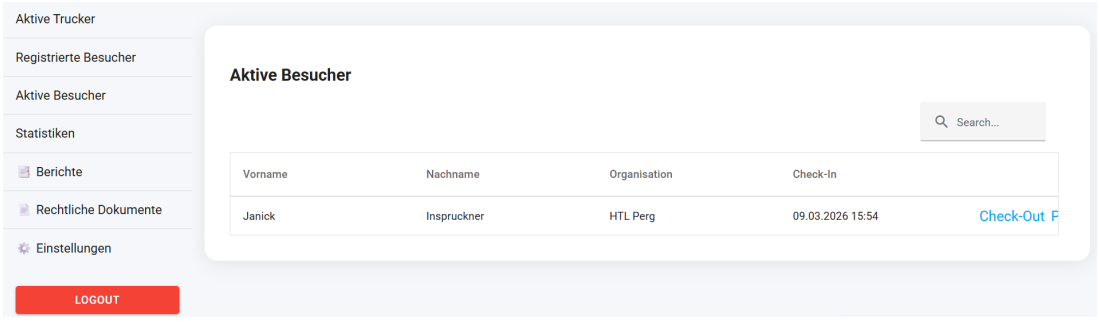



Abbildung 9: Dashboard-Übersicht aktuell anwesende Besucher

Die Übersicht der aktiven Besucher dient dem Empfang als zentrale Arbeitsgrundlage während des laufenden Betriebs. Sie ermöglicht es, anwesende Personen schnell zu erfassen, den aktuellen Stand im Blick zu behalten und bei Bedarf weitere Aktionen wie den späteren Check-out auszuführen, sofern sich Besucher nicht mit dem Besucherausweis ausgecheckt haben.

### 3.3.3 Unterweisung und Formularerfassung

Im Rahmen des Check-ins können dem Besucher zusätzlich Pflichtdokumente angezeigt werden. Dazu zählen je nach Einsatzbereich beispielsweise Datenschutzinformationen, Sicherheitsunterweisungen oder andere rechtlich bzw. organisatorisch relevante Hinweise. Diese Dokumente müssen vor Abschluss der Anmeldung zur Kenntnis genommen und bestätigt werden, wenn diese im Dashboard als Pflichtdokumente gekennzeichnet wurden.

Wie bereits in Abbildung 8 dargestellt, ist dieser Bereich direkt in die Anmeldemaske integriert. Dadurch bleibt der Ablauf für den Besucher übersichtlich, da die Erfassung der persönlichen Daten und die Bestätigung der erforderlichen Dokumente in einem gemeinsamen Prozess erfolgen.

Die Einbindung von Pflichtdokumenten unterstützt einen einheitlichen Ablauf am Empfang. Gleichzeitig wird sichergestellt, dass wichtige Informationen nicht außerhalb des Systems abgewickelt werden, sondern direkt im Registrierungsprozess berücksichtigt sind.

### 3.3.4 Ausweiserstellung und Druck

Nach dem erfolgreichen Check-in wird in „outPort“ automatisch ein Besucherausweis erzeugt. Dieser dient dazu, den Besucher während seines Aufenthalts eindeutig zu kennzeichnen und den Ablauf am Empfang weiter zu standardisieren. Der Ausweis wird direkt nach der Anmeldung erstellt und kann anschließend gedruckt werden.

Ein Beispiel für einen solchen Besucherausweis ist in Abbildung 10 dargestellt. Der Ausweis enthält die wichtigsten Informationen zum Besuch, darunter den Namen des Besuchers, die zugehörige Firma sowie die Besuchszeit. Zusätzlich wird ein QR-Code erzeugt, welcher für den späteren Check-out verwendet werden kann.

Durch die automatische Erstellung des Ausweises entfällt ein zusätzlicher manueller Schritt am Empfang. Gleichzeitig wird sichergestellt, dass jeder Besucher einen einheitlich aufgebauten Ausweis erhält. Der integrierte QR-Code erleichtert außerdem den späteren Check-out, da der Besucher beim Verlassen des Standorts direkt über den Ausweis identifiziert werden kann.

### 3.3.5 Benachrichtigung des Gastgebers

Wurde ein Besucher vorab angemeldet und ist im System eine E-Mail-Adresse hinterlegt, kann der zuständige Gastgeber nach dem erfolgreichen Check-in automatisch benachrichtigt werden.

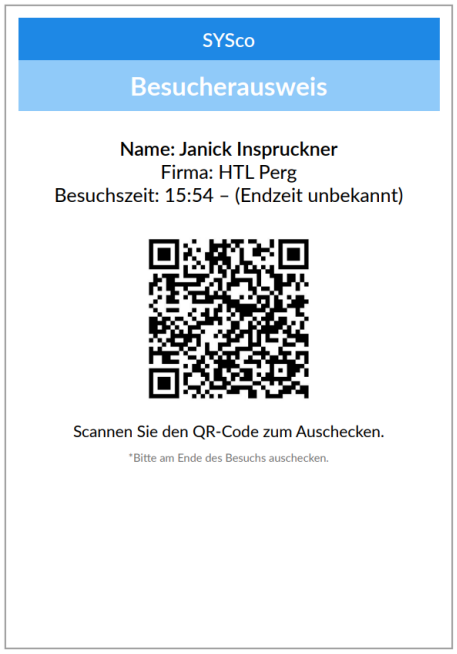



Abbildung 10: Automatisch erzeugter Besucherausweis mit QR-Code für den Check-out.

Dadurch entfällt die manuelle Verständigung durch den Empfang und der Informationsfluss innerhalb des Unternehmens wird verbessert.

Die Benachrichtigung erfolgt im Hintergrund direkt nach der Anmeldung des Besuchers. Der Gastgeber erhält dadurch die Information, dass der erwartete Besuch eingetroffen ist und am Empfang registriert wurde. Dies unterstützt einen reibungslosen Ablauf, da der Besucher nicht zusätzlich manuell angekündigt werden muss.

### 3.3.6 Check-out und Besuchshistorie

Nach Abschluss des Besuchs erfolgt der Check-out. Dieser kann entweder durch das Scannen des QR-Codes am Besucherausweis oder manuell über das Dashboard durchgeführt werden. Dadurch wird der Aufenthalt des Besuchers im System beendet.

Mit dem Check-out wird der Besucher nicht mehr in der Übersicht der aktuell anwesenden Personen angezeigt. Der zugehörige Datensatz bleibt jedoch in der Datenbank erhalten und kann dadurch auch nachträglich nachvollzogen werden.

Die Trennung zwischen aktiven und abgeschlossenen Besuchen unterstützt den operativen Ablauf am Empfang. Während in der aktiven Übersicht nur aktuell anwesende Personen angezeigt werden, bleiben frühere Besuche als Historie in der Datenbank gespeichert.

## 3.4 Funktionalität Admin-Dashboard

Das Admin-Dashboard stellt die zentrale Verwaltungs- und Übersichtsoberfläche von „outPort“ dar. Ein Dashboard ist eine Oberfläche, die wichtige Informationen und Funktionen gebündelt anzeigt, um typische Aufgaben effizient ausführen zu können. Im laufenden Betrieb dient das Dashboard insbesondere der Verwaltung von An- und Abreisen, der schnellen Auffindung von Einträgen sowie der Übermittlung des aktuellen Status am Standort.

Die Navigation ist links angeordnet und gliedert die Anwendung in übersichtliche Bereiche. Der Hauptbereich enthält tabellarische Listen. Eine tabellarische Liste ist eine Darstellung in Zeilen und Spalten, die sich für den Empfangsbetrieb besonders eignet, da viele Datensätze gleichzeitig sichtbar sind. Zusätzlich stehen pro Datensatz Aktionen zur Verfügung (z. B. Check-in oder Check-out), wodurch wiederkehrende Tätigkeiten effizient erledigt werden können.

Ein wesentliches Ziel des Dashboards ist die Reduktion von Bedienfehlern. Dafür werden Besucher und Trucker nach Status getrennt dargestellt (registriert, aktiv). Diese Trennung wirkt wie eine Filterung. Der Begriff „Filterung“ beschreibt das Verfahren, mittels dessen ausschließlich jene Datensätze angezeigt werden, die ein bestimmtes Kriterium erfüllen. In „outPort“ erfolgt diese Filterung primär über getrennte Reiter (z. B. „Registrierte Besucher“ und „Aktive Besucher“) statt über komplexe Filterdialoge. Dadurch bleibt der Ablauf am Empfang einfacher.

### 3.4.1 Besucherübersicht (Suche, Check-in/Check-out)

Die Besucherübersicht unterstützt den vollständigen Prozess vom Eintreffen bis zum Verlassen des Standorts. Dabei werden Besucher in zwei Kernzustände unterteilt:

- Registrierte Besucher: Besucher sind bereits erfasst, aber noch nicht eingecheckt.
- Aktive Besucher: Besucher sind eingecheckt und befinden sich aktuell am Standort.

Diese Aufteilung entspricht dem praktischen Ablauf am Empfang und ermöglicht es, Einträge schnell und eindeutig einzuordnen.

#### 3.4.1.1 Registrierte Besucher

In der Ansicht „Registrierte Besucher“ werden alle Besucher angezeigt, die noch nicht eingecheckt sind (siehe Abbildung 7). Die Tabelle enthält die wesentlichen Basisinformationen, um eine Person eindeutig zuzuordnen: Vorname, Nachname und Organisation sowie den Zeitpunkt der erwarteten Registrierung. Die erwartete Registrierung beschreibt den geplanten Ankunftszeitpunkt, sofern dieser bekannt ist.

Zur schnellen Auffindbarkeit steht eine Suchfunktion zur Verfügung. Eine Suchfunktion ist ein Eingabefeld, das Datensätze anhand eingegebener Zeichenketten durchsucht (z. B. Namen oder Organisation). Dadurch werden Wartezeiten reduziert, da der Portier nicht manuell durch lange Listen scrollen muss. In einem Empfangsszenario ist dies von entscheidender Bedeutung, insbesondere bei der gleichzeitigen Ankunft mehrerer Besucher.

Der operative Kernvorgang in dieser Ansicht ist der Check-in. Der Begriff Check-in beschreibt den Vorgang, bei dem ein Besucher beim Eintreffen im System als anwesend markiert wird. Technisch gesehen wird in der Regel ein Zeitstempel verwendet. Ein Zeitstempel ist die gespeicherte Uhrzeit bzw. das Datum einer Aktion, die zu einem späteren Zeitpunkt nachvollzogen werden kann. Nach dem Check-in wird der Datensatz in die Liste der aktiven Besucher aufgenommen.

#### 3.4.1.2 Manuelles Hinzufügen eines Besuchers

Sollte ein Besucher nicht vorregistriert sein oder spontan eintreffen, besteht die Möglichkeit, dass der Portier den Datensatz direkt im Dashboard anlegt (siehe Abbildung 6). Durch das manuelle Hinzufügen wird der Aufwand für Medienbrüche reduziert und es wird verhindert, dass Besucher abgewiesen oder auf spätere Nachtragungen verwiesen werden müssen.

Die Eingabemaske umfasst Felder für Vorname, Nachname und Organisation. Zusätzlich besteht die Möglichkeit, eine E-Mail-Adresse zu hinterlegen, die als optional gekennzeichnet ist. Optional bedeutet, dass das Feld nicht verpflichtend ausgefüllt werden muss. Für die zeitliche Planung kann eine erwartete Registrierung gesetzt werden. Darüber hinaus besteht die Möglichkeit, die Ganztägigkeit und die Dauer zu erfassen. Ein Ganztagsbesuch ist ein Besuch, der für den gesamten Arbeitstag vorgesehen ist. Die Dauer beschreibt den geplanten Zeitraum in Stunden. Diese Angaben unterstützen eine bessere Planung und helfen, spätere Auswertungen über typische Aufenthaltszeiten zu erstellen.

#### 3.4.1.3 Aktive Besucher

In der Ansicht „Aktive Besucher“ werden alle aktuell anwesenden Besucher angezeigt (siehe Abbildung 9). Im Unterschied zur Registrierungsliste liegt der Fokus hier auf dem tatsächlichen Check-in-Zeitpunkt. Dadurch ist nachvollziehbar, seit wann sich eine Person am Standort aufhält. Wie auch bei den registrierten Besuchern steht eine Suchfunktion zur Verfügung, um Einträge schnell zu finden.

Der zentrale Vorgang in dieser Ansicht ist der Check-out. Der Check-out-Prozess bezeichnet den Zustand, in dem ein Besucher beim Verlassen des Standorts aus der aktiven Liste entfernt

und der Besuch als abgeschlossen markiert wird. Auch hier wird typischerweise ein Zeitstempel gesetzt, der die Endzeit dokumentiert.

Neben der Check-out-Funktion steht zusätzlich eine Funktion zur Ausgabe eines Besucherausweises zur Verfügung. Ein Besucherausweis dient dazu, die Anwesenheit einer Person zu dokumentieren und unterstützt organisatorische Abläufe am Standort, beispielsweise bei Zutrittskontrollen oder Rückfragen. In „outPort“ enthält der Besucherausweis einen QR-Code (siehe Abbildung 10). Ein QR-Code ist ein zweidimensionaler Code, der mit einem Scanner oder einer Kamera ausgelesen werden kann. Der QR-Code kann dazu verwendet werden, den Check-out-Prozess zu vereinfachen, indem der Code am Ende des Besuchs gescannt wird.

Die Besucherübersicht deckt drei Kernaufgaben ab: Die schnelle Auffindbarkeit wird durch eine Suchfunktion sichergestellt, die Statusdarstellung ist klar und übersichtlich in getrennten Reitern dargestellt und die direkten operativen Aktionen (Check-in, Check-out, Ausweisausgabe) erfolgen ohne zusätzliche Zwischenschritte.

### 3.4.2 Trucker-Übersicht

Neben Besuchern werden im Dashboard auch Trucker verwaltet. In diesem Kontext wird der Begriff „Trucker“ verwendet, um einen LKW-Fahrer zu bezeichnen, der für Anlieferungen oder Abholungen am Standort zuständig ist. Die Übersicht der Trucker zeigt alle aktuell anwesenden Trucker in tabellarischer Form (siehe Abbildung 4).

Die Darstellung enthält jene Informationen, die für die Zuordnung im Logistikprozess wesentlich sind:

- Name: Identifikation der Person
- Kennzeichen: eindeutige Zuordnung des Fahrzeugs
- Bestellnummer: Bezug zur Lieferung/Abholung
- Check-in: Zeitpunkt des Eintreffens

Auch in dieser Übersicht ist eine Suchfunktion integriert. Dadurch können Lieferanten anhand weniger Zeichen schnell gefunden werden, was insbesondere bei hoher Frequenz (mehrere Fahrzeuge) relevant ist.

Als operative Aktion steht der Check-out zur Verfügung. Der Check-out beendet den Aufenthalt des Truckers am Standort und sorgt dafür, dass die Liste der aktiven Trucker stets aktuell ist. Der sichtbare Check-in-Zeitpunkt ermöglicht zudem eine Überprüfung, ob Fahrzeuge

ungewöhnlich lange am Standort verbleiben. Dies ermöglicht eine schnelle Rücksprache oder Prozessprüfung.

Die Trucker Übersicht ergänzt die Besucherübersicht um einen zweiten, im Alltag häufigen Empfangsprozess. Dadurch wird sichergestellt, dass sowohl Personenbesuche als auch Lieferverkehre über eine einheitliche Oberfläche gesteuert werden können.

### 3.4.3 Statistiken und Berichte

Neben den operativen Übersichten bietet „outPort“ Funktionen zur schnellen Auswertung und Dokumentation. Dafür werden zwei Bereiche unterschieden:

- Statistiken: schnelle Übersicht über aktuelle Kennzahlen
- Berichte: exportierbare Zusammenfassung als Dokument

#### 3.4.3.1 Statistiken

Der Statistikbereich präsentiert zentrale Kennzahlen in grafischer Form (siehe Abbildung 11). Eine Kennzahl ist ein messbarer Wert, der einen Zustand oder eine Entwicklung zusammenfasst. Das gezeigte Diagramm stellt die Anzahlen aktiver Besucher, registrierter Besucher und aktiver Trucker gegenüber. Diese Darstellung unterstützt den Portier bzw. Administrator dabei, den aktuellen Zustand auf einen Blick zu erfassen, ohne einzelne Listen manuell auswerten zu müssen.

Die Statistik ist ein wertvolles Instrument, um rasch Entscheidungen zu treffen, beispielsweise um zu ermitteln, ob sich aktuell viele Personen am Standort aufhalten (z. B. für Sicherheits- oder Organisationszwecke) oder ob noch viele Registrierungen offen sind.

#### 3.4.3.2 Berichte

Im Bereich „Berichte“ steht eine tägliche Berichtsgenerierung zur Verfügung (siehe Abbildung 12). Ein Bericht ist eine zusammenfassende Darstellung in Dokumentform, die beispielsweise für die interne Ablage, den Nachweis oder die Weitergabe genutzt wird. Der Download erfolgt über einen einzigen Button, wodurch der Zugriff bewusst einfach gehalten ist. Als Format wird PDF verwendet. PDF ist ein Dateiformat, welches für Druck und Archivierung geeignet ist und ein stabiles Layout bietet.

Das Dashboard kombiniert Statistik und Bericht und deckt somit sowohl die kurzfristige Übersicht (Kennzahlen) als auch die dokumentierbaren Ergebnisse (PDF-Bericht) ab.

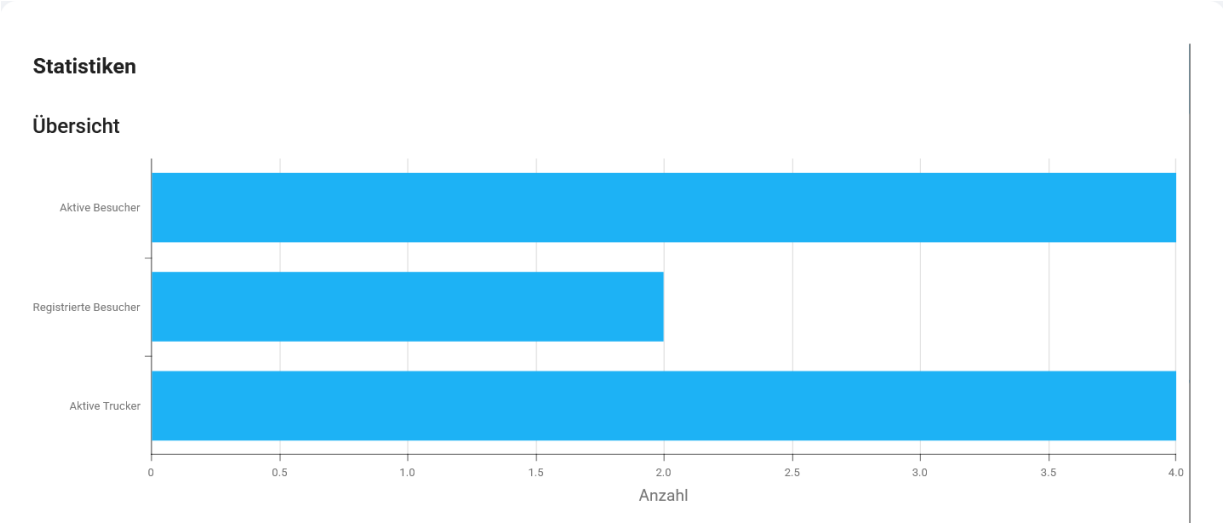



Abbildung 11: Statistikübersicht aktive Besucher, registrierte Besucher, aktive Trucker.

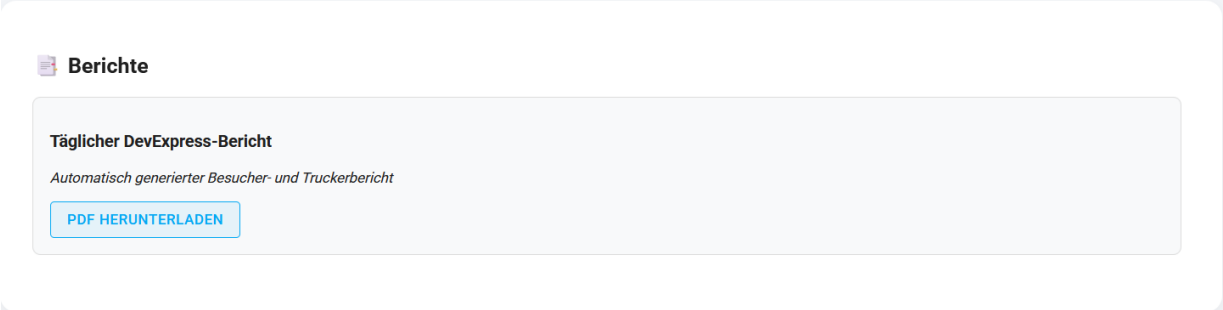



Abbildung 12: Berichtsbereich Download täglich generierter Besucher- und Lieferantenbericht.

## 3.5 Funktionalität Registrierung und Login

Die Registrierung und Anmeldung sind ein zentraler Bestandteil von „outPort“, da sie den Einstieg in die jeweiligen Prozesse ermöglichen. Je nach Benutzergruppe unterscheiden sich dabei die abgefragten Informationen. Aus diesem Grund wird in der Anwendung zwischen einem Besucher-Login und einem Lieferanten-Login unterschieden.

### 3.5.1 Registrierung für Besucher

Der Besucher-Login ist der Einstieg für den Besucher in den Registrierungsprozess. Die Anmeldung erfolgt über eine einfach gehaltene Eingabemaske, die sich leicht bedienen lässt. Wie in Abbildung 8 dargestellt, kann der Besucher zunächst eine Sprache auswählen. Anschließend werden Besucherinformationen wie Vorname, Nachname und Organisation erfasst. Zusätzlich können im unteren Bereich der Maske Pflichtdokumente angezeigt werden, die vor dem Fortset-

zen des Anmeldevorgangs bestätigt werden müssen. Dadurch werden sowohl die Datenerfassung als auch die notwendige Bestätigung in einer gemeinsamen Oberfläche zusammengeführt.

### 3.5.2 Registrierung für Lieferanten

Der Lieferanten-Login ist gleich aufgebaut wie der Besucher-Login. Auch hier kann zuerst eine Sprache ausgewählt werden, bevor die eigentlichen Daten eingegeben werden. Der wesentliche Unterschied liegt in den abgefragten Informationen. Es werden neben personenbezogenen auch lieferungsbezogene Daten erfasst. Dazu gehören der vollständige Name, das Kennzeichen und die Bestellnummer. Im unteren Bereich der Maske sind auch hier Pflichtdokumente aufgeführt, ohne deren Bestätigung die Anmeldung nicht fortgesetzt werden kann.

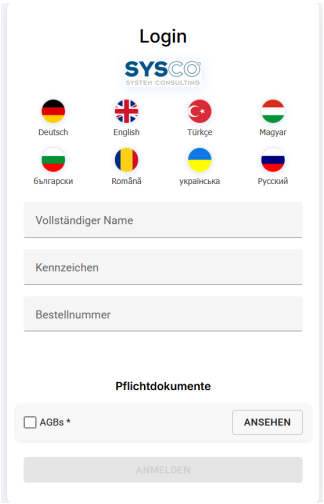



Abbildung 13: Anmeldemaske Check-in Lieferant am Empfang.

## 3.6 Self-Service-Terminal und Mobile UI

Das Self-Service-Terminal und die Mobile UI bilden in „outPort“ jene Oberflächen ab, die von Besuchern bzw. Lieferanten selbst bedient werden. Ziel ist es, den Aufwand am Empfang zu reduzieren und gleichzeitig einen standardisierten, nachvollziehbaren Anmeldeprozess bereitzustellen. Technisch wird dafür dieselbe Webanwendung in zwei Nutzungsszenarien eingesetzt: Als Terminalansicht am Empfang (Kiosk) und als mobile Weboberfläche am Smartphone.

Die Oberfläche ist als responsive Webanwendung umgesetzt. Responsive bedeutet, dass sich Layout und Anordnung der Elemente automatisch an unterschiedliche Bildschirmgrößen anpassen. Dadurch kann derselbe Ablauf auf einem großen Terminaldisplay sowie auf mobilen Geräten genutzt werden, ohne dass separate Anwendungen entwickelt werden müssen. Im Mittelpunkt

steht ein klarer Prozess: Sprache wählen, erforderliche Daten eingeben, Pflichtdokumente einsehen und bestätigen sowie die Anmeldung abschließen.

### 3.6.1 Kiosk-Modus und Usability

Der Kiosk-Modus beschreibt den Betrieb der Anwendung auf einem öffentlich zugänglichen Gerät (z. B. Tablet am Empfang), bei dem die Oberfläche möglichst fokussiert und ohne Ablenkungen verwendet wird. Charakteristisch ist dabei, dass der Nutzer nicht durch eine komplexe Navigation geführt wird, sondern einen klaren Ablauf in wenigen Schritten durchläuft. Für den Betrieb am Empfang ist dies wichtig, da sich Nutzergruppen stark unterscheiden können und die Bedienung oft unter Zeitdruck funktionieren muss.

In „outPort“ ist die Terminalansicht deshalb bewusst reduziert aufgebaut. Der Einstieg erfolgt über eine einzelne Anmeldemaske, die nur jene Eingaben abfragt, die für den jeweiligen Prozess notwendig sind. Für Besucher umfasst dies typischerweise Basisdaten wie Vorname, Nachname und Organisation. Für Lieferanten liegt der Fokus stärker auf lieferungsbezogenen Angaben wie dem vollständigen Namen, dem Kennzeichen und der Bestellnummer. Das Kennzeichen bezeichnet dabei die amtliche Fahrzeugkennung, die eine eindeutige Zuordnung des Fahrzeugs ermöglicht. Die Bestellnummer stellt den Bezug zu einer Lieferung oder Abholung her.

Um Fehleingaben zu reduzieren, wird der Ablauf durch Zustände der Oberfläche gesteuert. Ein Beispiel ist die Freischaltung der Abschlussaktion erst dann, wenn alle Pflichtbedingungen erfüllt sind. Pflichtbedingungen sind Bedingungen, die vor dem Fortsetzen zwingend erfüllt sein müssen. Dazu zählt insbesondere die Bestätigung erforderlicher Dokumente. Zusätzlich ist die Oberfläche so gestaltet, dass wichtige Aktionen klar erkennbar sind und Interaktionselemente ausreichend groß dimensioniert wurden, um eine sichere Bedienung per Touch zu ermöglichen.

Für den Kiosk-Betrieb ergeben sich folgende Anforderungen, die in „outPort“ umgesetzt wurden:

- Fokussierte Startansicht ohne unnötige Navigation, damit der Prozess direkt gestartet werden kann.
- Reduzierte Eingaben je Prozess (Besucher vs. Lieferanten), um die Anmeldedauer kurz zu halten.
- Klare Prozesszustände (z. B. Anmeldung erst möglich nach bestätigten Pflichtdokumenten).
- Einheitliche Darstellung der wesentlichen Informationen, damit sich der Ablauf auch für wechselnde Nutzergruppen konsistent anfühlt.

### 3.6.2 Mehrsprachigkeit

Mehrsprachigkeit bezeichnet die Verfügbarkeit derselben Oberfläche in mehreren Sprachen. Der Zweck ist, dass Besucher und Lieferanten unabhängig von ihrer Muttersprache den Anmeldeprozess korrekt durchführen können. In „outPort“ ist die Sprachwahl direkt am Einstieg platziert, sodass die Sprache gewählt werden kann, bevor personenbezogene oder lieferungsbezogene Daten eingegeben werden.

Wesentlich ist dabei, dass sich bei einem Sprachwechsel nur die Texte ändern, nicht jedoch die Struktur der Oberfläche. Struktur bedeutet hier: Reihenfolge der Schritte, Position der Felder und grundlegende Bedienlogik. Diese Konsistenz reduziert Missverständnisse, weil Nutzer nach dem Sprachwechsel nicht erneut „lernen“ müssen, wo Eingaben oder Bestätigungen zu finden sind.

Für die Mehrsprachigkeit werden folgende Regeln eingehalten:

- Übersetzt werden alle sichtbaren Texte, insbesondere Überschriften, Feldbezeichnungen, Hinweise und Buttons.
- Die Sprache wird zu Beginn festgelegt und während des Prozesses beibehalten, um Mischsprachen zu vermeiden.
- Für den Fall, dass eine Übersetzung fehlt, wird eine Standardsprache verwendet. Dies bezeichnet eine definierte Sprache, auf die automatisch zurückgegriffen wird, damit keine leeren oder unverständlichen Texte entstehen.

Die Umsetzung ist so gestaltet, dass sowohl die Terminalansicht als auch die mobile Oberfläche denselben Sprachumfang nutzen können. Damit bleibt die Bedienung in beiden Szenarien identisch, während die Verständlichkeit für internationale Nutzergruppen steigt.

### 3.6.3 Branding pro Standort

Branding bezeichnet die Anpassung der Oberfläche an das Erscheinungsbild eines Unternehmens. Dazu zählen insbesondere Logo, Farben und grundlegende Gestaltungselemente. „Pro Standort“ bedeutet, dass diese Gestaltung nicht nur global einmal definiert ist, sondern abhängig davon variieren kann, an welchem Standort das System eingesetzt wird. Ein Standort ist dabei eine organisatorische oder geografische Einheit (z. B. Werk, Niederlassung oder Empfangsbereich) mit potenziell eigenen Anforderungen an Darstellung und Dokumente.

Im aktuellen Stand ist das Branding vor allem über sichtbare Markenelemente wie das Logo abbildbar. Für einen standortabhängigen Einsatz ergibt sich daraus ein klares Zielbild: Bei

mehreren Standorten wird das Terminal automatisch das passende Erscheinungsbild laden, ohne dass dafür eine Codeänderung erforderlich ist. Eine Codeänderung würde bedeuten, dass die Anwendung neu gebaut oder manuell angepasst werden müsste, um ein anderes Logo oder Farbschema zu verwenden.

Ein mögliches Konzept für Branding pro Standort ist daher:

- Speicherung von Branding-Informationen je Standort (Logo, Farbwerte, ggf. Schriften) in einer zentralen Konfiguration.
- Auswahl des Standorts über eine feste Terminal-Konfiguration (z. B. Standort-ID) oder über eine Auswahl im Adminbereich.
- Automatisches Laden der Branding-Konfiguration beim Start der Oberfläche.

Damit wird erreicht, dass ein Self-Service-Terminal an verschiedenen Orten betrieben werden kann und dennoch ein konsistenter, standortspezifischer Auftritt gewährleistet wird. Gleichzeitig bleibt die technische Wartung vereinfacht, da das Frontend nicht für jeden Standort separat verteilt werden muss.

## 3.7 Integrationen

In „outPort“ werden mehrere Funktionen durch angebundene Systeme erweitert. Dazu zählen die Benachrichtigung des Gastgebers per E-Mail, die Kalenderintegration über Outlook sowie die Erstellung und Ausgabe von Besucherausweisen.

### 3.7.1 Kalenderintegration (Outlook)

Die Kalenderintegration erweitert „outPort“ um die Möglichkeit, vorangemeldete Besucher auch direkt mit einem Kalendereintrag in Outlook zu verknüpfen. Dadurch wird ein geplanter Besuch nicht nur im System selbst, sondern zusätzlich auch im Kalender des zuständigen Gastgebers oder in einem verwendeten Kalenderkonto abgebildet.

Wird ein Besuch angelegt, können die zugehörigen Informationen wie Name, Zeitpunkt, Ort und weitere ergänzende Angaben in einen Outlook-Termin übernommen werden. Ein Beispiel für einen solchen Kalendereintrag ist in Abbildung 14 dargestellt. Dadurch wird der geplante Besuch auch außerhalb des Dashboards sichtbar und kann einfacher in bestehende Terminabläufe eingebunden werden.


Abbildung 14: Kalendereintrag in Outlook Besucher

Da der Termin dadurch zusätzlich im Kalender auftaucht und nicht nur im Dashboard verwaltet wird, unterstützt dies die organisatorische Vorbereitung und erleichtert die Übersicht über anstehende Besuche.

### 3.7.2 Benachrichtigung (E-Mail)

Die E-Mail-Benachrichtigung dient dazu, den zuständigen Gastgeber über das Eintreffen des Besuchers zu informieren. Dadurch entfällt die manuelle Verständigung durch den Empfang und der Informationsfluss innerhalb des Unternehmens wird vereinfacht.

Sobald ein Besucher erfolgreich eingecheckt hat, wird im Hintergrund automatisch eine Benachrichtigung per E-Mail versendet. Diese enthält die wichtigsten Informationen zum eingetroffenen Besucher, wie den Namen und den Zeitpunkt des Check-ins. Ein Beispiel für eine solche Nachricht ist in Abbildung 15 dargestellt.

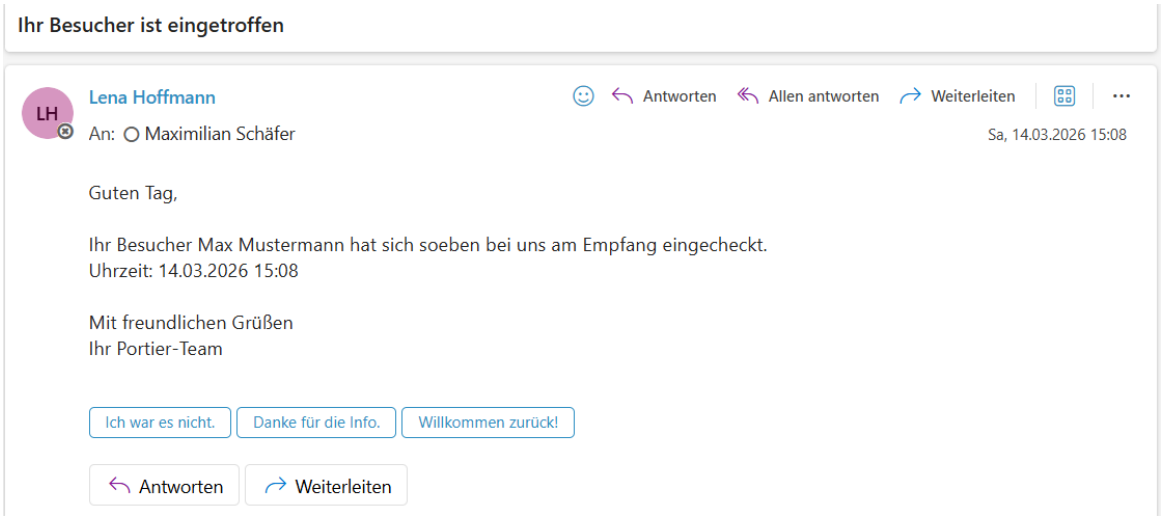



Abbildung 15: Automatische E-Mail-Benachrichtigung Ankunft Besucher

Mit dieser Benachrichtigung ist kein zusätzlicher Schritt des Empfangs nötig, um den Gastgeber zu informieren. Der weitere Ablauf wird dadurch unterstützt, da der Besucher schneller abgeholt und empfangen werden kann.

Durch die automatische Benachrichtigung wird die Kommunikation rund um den Besuchsprozess vereinheitlicht und direkt in das System eingebunden.

### 3.7.3 Drucker / Ausweissystem

Das Drucker- bzw. Ausweissystem ist in „outPort“ dafür zuständig, nach dem erfolgreichen Check-in einen Besucherausweis bereitzustellen. Dadurch erhält der Besucher eine eindeutige Kennzeichnung für die Dauer seines Aufenthalts.

Nach Anmeldung des Besuchers wird der Besucherausweis automatisch generiert und kann gedruckt oder als PDF genutzt werden. Ein Beispiel eines solchen Besucherausweises wird in Abbildung 10 dargestellt. Er enthält die wichtigsten Informationen, wie den Namen, die Firma und die Besuchszeit des Besuchers. Zusätzlich wird ein QR-Code eingebunden, der später für den Check-out gescannt werden kann.

Durch den Besucherausweis kann der Besucher während der gesamten Dauer seines Aufenthalts eindeutig identifiziert werden und der Abschluss des Besuchs wird mit dem QR-Code deutlich vereinfacht. Dadurch ist das Ausweissystem direkt in den Besuchsprozess eingebunden.

Die automatische Erstellung und der Druck des Ausweises sorgen dafür, dass dieser nicht manuell außerhalb des Systems erfolgen muss. Stattdessen wird der Ausweis direkt nach Anmeldung des Besuchs erzeugt und im weiteren Ablauf verwendet.

# 4 Technologien

## 4.1 Backend

Das Backend ist der serverseitige Teil der Anwendung und übernimmt in „outPort“ die Verarbeitung der fachlichen Logik sowie die Bereitstellung von Schnittstellen für das Frontend. Es verarbeitet Besucher- und Lieferantendaten, steuert zentrale Abläufe im System und stellt sicher, dass Eingaben aus dem Frontend korrekt weiterverarbeitet werden.

Das Backend ist zuständig für die Verarbeitung von Check-in- und Check-out-Vorgängen, die Speicherung und Abfrage von Daten sowie die Erstellung von Besucherausweisen. Zusätzlich bildet das Backend die Verbindung zu weiteren Funktionen wie der Datenbank, der Authentifizierung über Keycloak und externen Diensten.

Durch die serverseitige Verarbeitung von Daten im Backend bleibt die Geschäftslogik zentral an einer Stelle, was die Wartbarkeit und Erweiterbarkeit der Anwendung verbessert.

### 4.1.1 C# / .NET

Für die Umsetzung des Backends werden in „outPort“ C# und .NET verwendet. C# ist eine moderne, allgemeine und objektorientierte Programmiersprache, die für die Entwicklung vieler Anwendungsarten eingesetzt wird. .NET bildet dazu die technische Grundlage und stellt Laufzeitumgebung, Bibliotheken und Werkzeuge für die Ausführung und Entwicklung der Anwendung bereit. [6], [7]

Im Projekt werden C# und .NET vor allem für folgende technische Aufgaben eingesetzt:

- **Verarbeitung der serverseitigen Logik:** Check-in- und Check-out-Vorgänge sowie die Verarbeitung von Besucher- und Lieferantendaten.
- **Bereitstellung von API-Endpunkten:** Über diese werden Daten abgerufen, gespeichert oder verändert.
- **Kommunikation mit weiteren Komponenten:** C# und .NET werden zusätzlich für den Zugriff auf die Datenbank sowie für externe Anbindungen verwendet.

Durch den objektorientierten Aufbau lassen sich Controller, Services und Modelle klar voneinander trennen, wodurch der Code übersichtlich und einfach wartbar bleibt.

### 4.1.2 ASP.NET Core Web API

Mit dem ASP.NET Core Web API Framework werden serverseitig Schnittstellen bereitgestellt. ASP.NET Core ist ein plattformübergreifendes und quelloffenes Framework für Webanwendungen und Webdienste. Mit einer Web API werden Funktionen über HTTP-Endpunkte bereitgestellt, damit andere Teile der Anwendung Daten senden, abrufen oder verändern können. [8], [9]

In „outPort“ wird ASP.NET Core Web API vor allem für folgende technische Aufgaben eingesetzt:

- **Kommunikation zwischen Frontend und Backend:** Das Frontend sendet über definierte Endpunkte Daten an das Backend und ruft diese bei Bedarf wieder ab. Dazu zählen Registrierung, Check-in- und Check-out-Vorgänge sowie das Laden von Dashboard-Daten.
- **Trennung in Funktionsbereiche:** Die API ist in mehrere Controller aufgeteilt, damit verschiedene Bereiche getrennt behandelt werden können, wie beispielsweise Besucher, Lieferanten, rechtliche Dokumente, Reporting und E-Mail-Funktionen.
- **Serverseitige Verarbeitung von Anfragen:** Die eigentliche Logik wird im Backend verarbeitet und anschließend über die Endpunkte an das Frontend gegeben. Dadurch bleibt die Verarbeitung zentral im Backend.

ASP.NET Core Web API bildet damit die Verbindung zwischen UI, fachlicher Logik und Datenzugriff. Änderungen im Frontend können über Schnittstellen an das Backend weitergegeben werden, ohne dass die interne Verarbeitung direkt im Frontend umgesetzt werden muss.

### 4.1.3 Entity Framework Core

Entity Framework Core ist ein ORM-Framework (Object Relational Mapper) für .NET. Ein ORM ermöglicht es, mit Datenbanktabellen über C#-Objekte zu arbeiten, anstatt für jede Operation eine eigene SQL-Abfrage zu schreiben. Dadurch wird der Datenzugriff im Code übersichtlicher und leichter wartbar. [10], [11]

In „outPort“ wird das Entity Framework vor allem für folgende technische Aufgaben eingesetzt:

- **Zugriff auf Anwendungsdaten:** Über den zentralen Datenkontext AppDbContext können Besucher, Lieferanten und weitere Anwendungsdaten abgefragt, gespeichert und verändert werden.
- **Verbindung zwischen Modellen und Datenbank:** Die im Backend definierten Modelle werden über das Entity Framework mit den entsprechenden Datenbanktabellen verknüpft. Dadurch können Datensätze direkt über C#-Klassen verarbeitet werden.
- **Einbindung in das Backend:** Der Datenzugriff erfolgt nicht direkt im Frontend, sondern wird serverseitig im Backend verarbeitet. Dadurch bleibt die Datenverarbeitung zentral gebündelt und kann in Services und Controllern strukturiert verwendet werden.

In der Anwendung wird Entity Framework Core speziell dort eingesetzt, wo Besucher- und Lieferantendaten gespeichert oder abgerufen werden. Die Technologie erleichtert damit die Verbindung zwischen Datenbank und fachlicher Logik im Backend. Die eigentliche Datenbank wird in Abschnitt 4.3 näher beschrieben.

### 4.1.4 QuestPDF

QuestPDF ist eine Bibliothek zur Erstellung von PDF-Dokumenten in C#. Die Bibliothek ermöglicht es, Inhalte und Layout direkt im Code zu definieren, anstatt PDF-Dateien manuell zu erstellen. Dadurch können Dokumente strukturiert aufgebaut und automatisch erzeugt werden. [12], [13]

In „outPort“ wird QuestPDF vor allem für folgende Aufgaben eingesetzt:

- **Erstellung des Besucherausweises:** Nach dem erfolgreichen Check-in wird serverseitig ein PDF-Dokument erzeugt, welches als Besucherausweis verwendet werden kann.
- **Einheitliches Layout der Ausweise:** Durch die PDF-Erstellung im Backend wird sichergestellt, dass alle erzeugten Besucherausweise in derselben Struktur und im selben Layout generiert werden.
- **Bereitstellung für weitere Verarbeitung:** Das generierte PDF wird im Anschluss an das Frontend übergeben, wo es heruntergeladen oder gedruckt werden kann.

QuestPDF wird damit im Projekt als serverseitige Lösung zur automatischen Dokumenterstellung eingesetzt. Besonders für den Besucherausweis ist diese Technologie geeignet, da sich Inhalte wie Name, Firma, Besuchszeit und QR-Code direkt aus den erfassten Daten in das PDF übernehmen lassen.

### 4.1.5 Microsoft Graph API

Microsoft Graph ist eine einheitliche REST-API, die von Microsoft bereitgestellt wird. Sie dient als zentraler Zugangspunkt, um auf Daten und Dienste innerhalb von Microsoft 365 zuzugreifen. Anstatt für jeden Microsoft-Dienst eine separate Schnittstelle zu nutzen, bündelt Microsoft Graph diese unter einem gemeinsamen Endpunkt. [14]

In „outPort“ wird Microsoft Graph API vor allem für folgende Aufgaben eingesetzt:

- **Automatisierter E-Mail-Versand:** Wenn ein Besucher am Empfang eincheckt, nutzt das Backend die Graph API, um eine Benachrichtigungs-E-Mail über Exchange Online an den zuständigen Gastgeber zu senden.
- **Kalenderverwaltung:** Die Schnittstelle wird bei der Voranmeldung eines Besuchers verwendet, um einen entsprechenden Termin direkt im Outlook-Kalender des Mitarbeiters zu erstellen und diesen bei Bedarf wieder zu löschen.

## 4.2 Frontend

Das Frontend stellt den Teil der Anwendung dar, der im Browser ausgeführt wird und die Benutzeroberfläche darstellt. In „outPort“ umfasst das Frontend sowohl das Self-Service-Terminal (z. B. Lieferanten-Anmeldung) als auch das Dashboard für den laufenden Betrieb. Auf technischer Ebene werden Eingaben verarbeitet, Daten über Schnittstellen abgefragt und UI-Zustände verwaltet (z. B. welche Ansicht aktiv ist oder welche Tabelle geladen wird).

Für die Umsetzung werden Vite als Build-Tool, React als UI-Bibliothek und TypeScript als Programmiersprache verwendet. Ein Build bezeichnet den Prozess, bei dem aus dem Quellcode optimierte Dateien für die Auslieferung erzeugt werden. [15]

### 4.2.1 Vite

Vite ist ein Build-Tool mit integriertem Entwicklungsserver. Ein Entwicklungsserver ist ein lokaler Server, der die Anwendung während der Entwicklung bereitstellt und Änderungen sofort sichtbar macht. Ein zentrales Feature ist das Hot Module Replacement (HMR). HMR bedeutet, dass bei einer Codeänderung nur das betroffene Modul aktualisiert wird, ohne dass die gesamte Seite neu geladen werden muss. Dadurch verkürzt sich die Entwicklungszeit deutlich. [15]

In „outPort“ wird Vite hauptsächlich für drei technische Aufgaben eingesetzt:

- Lokale Entwicklung mit HTTPS: In der Vite-Konfiguration können ein Zertifikat und ein Schlüssel hinterlegt werden, wodurch der Dev-Server per TLS (Transport Layer Security, Standard für Transportverschlüsselung) betrieben wird. [16]
- Proxy zum Backend: Ein Proxy leitet bestimmte Pfade vom Frontend an das Backend weiter. Dies wird genutzt, damit Frontend und Backend während der Entwicklung wie ein System wirken und keine Konflikte durch getrennte Ports entstehen. [16]
- Projektkonfiguration: Vite wird über eine zentrale Konfigurationsdatei gesteuert (z. B. Definition von Alias-Pfaden und Server-Optionen). [16]

### 4.2.2 React

React ist eine UI-Bibliothek, die Oberflächen aus Komponenten aufbaut. Eine Komponente ist ein wiederverwendbarer UI-Baustein (z. B. Tabelle, Formular, Seite). Dies ermöglicht eine strukturierte Zerlegung des Dashboards in klar abgegrenzte Einheiten, was die Wartbarkeit verbessert. [17]

Für dynamische Oberflächenzustände verwendet React sogenannte Hooks. Ein Hook ist eine React-Funktion, die zusätzliche Features in Funktionskomponenten ermöglicht, insbesondere die Zustandsverwaltung (State) und Seiteneffekte (Effects). State beschreibt dabei den aktuellen Zustand, wie beispielsweise den ausgewählten Menüpunkt, geladene Daten oder den Status eines Popups. [18]

Neben dem State werden Effects verwendet, um die Anwendung mit externen Vorgängen zu synchronisieren. Ein Effect (umgesetzt über `useEffect`) ist eine Funktion, die nach dem Rendern ausgeführt wird, um mit externen Systemen zu interagieren (z. B. Netzwerkzugriffe für Datenaktualisierungen). [19]

In „outPort“ wird dies technisch im Dashboard angewendet für:

- die zustandsbasierte Navigation zur Steuerung der aktiven Ansicht.
- das Laden von Tabelleninhalten aus dem Backend.
- das Ein- und Ausblenden von Formular-Popups sowie die Verarbeitung von Benutzereingaben.

### 4.2.3 TypeScript

TypeScript ist eine Erweiterung von JavaScript mit statischer Typprüfung. Statische Typprüfung bedeutet, dass Typfehler bereits vor der Ausführung des Codes erkannt werden (z. B. wenn ein Wert mit einem falschen Datentyp übergeben wird). Dadurch werden potenzielle Laufzeitfehler reduziert und die Codequalität gesteigert. [20]

Im Projekt wird TypeScript insbesondere genutzt, um Datenstrukturen klar zu definieren. Dazu werden Objekttypen bzw. Interfaces verwendet. Ein Interface beschreibt die Struktur eines Objekts, also welche Felder vorhanden sind und welche Datentypen diese besitzen. [21]

Die wesentlichen Vorteile für „outPort“ sind:

- klar typisierte Datensätze (z. B. für Besucher und Lieferanten).
- eine verbesserte Unterstützung durch Entwicklungswerkzeuge wie Autocomplete und frühzeitige Fehlererkennung.
- ein geringeres Risiko von Feld- oder Namensfehlern bei der Verarbeitung von API-Antworten.

## 4.3 Datenbank

Für die Datenhaltung wird ein relationales Datenbankmanagementsystem verwendet, da die im System verarbeiteten Informationen klar strukturiert sind und in einem fachlichen Zusammenhang stehen. Insbesondere für Verwaltungsprozesse bietet ein relationales Modell den Vorteil, dass Daten konsistent, nachvollziehbar und gezielt abfragbar gehalten werden können [22].

Als Datenbank wird ein **Microsoft SQL Server** benutzt. Die Wahl fiel auf dieses System, da es sich besonders für strukturierte Geschäftsdaten eignet und das Unternehmen SYSco bereits mit einer Vielzahl solcher Datenbanken arbeitet. [23].

Ein wesentlicher Vorteil von dem Microsoft SQL Server liegt in seiner Stabilität und in den umfangreichen Möglichkeiten zur Verwaltung relationaler Daten. Darüber hinaus bietet das System eine gute Grundlage für spätere Erweiterungen, etwa für komplexere Auswertungen, zusätzliche Benutzergruppen oder weiterführende Berichtsfunktionen. Auch im Hinblick auf Wartbarkeit und Integration in eine .NET-basierte Anwendungslandschaft stellt der SQL Server eine passende Lösung dar [23], [24].

## 4.4 Keycloak

Authentifizierung und Autorisierung sind zentrale Sicherheitsfunktionen in „outPort“. Authentifizierung bedeutet, dass ein Benutzer nachweist, wer er ist (z. B. durch einen Login-Vorgang). Autorisierung beschreibt die Entscheidung des Systems, auf welche Funktionen und Daten dieser Benutzer zugreifen darf. In „outPort“ wird hierfür Keycloak eingesetzt, um Benutzeridentitäten zentral zu verwalten und eine rollenbasierte Zugriffskontrolle (RBAC) umzusetzen.

Keycloak ist ein Identity- und Access-Management-System (IAM). Ein IAM stellt eine zentrale Komponente dar, die Anmeldung, Benutzerverwaltung und Zugriffskontrolle für Anwendungen bereitstellt. In „outPort“ erfolgt die Authentifizierung der Benutzer durch Keycloak. Nach erfolgreicher Anmeldung wird ein Zugriffstoken bereitgestellt. Ein Zugriffstoken ist ein digitales Nachweisdokument, das bei Anfragen an das Backend mitgesendet wird und Informationen zur Identität sowie zu den Berechtigungen enthält.

Nach der Anmeldung werden die Benutzerrollen im Frontend analysiert, um eine optimale Anpassung der Benutzeroberfläche zu gewährleisten. Eine Rolle bezeichnet ein vordefiniertes Profil, das spezifische Rollen – wie beispielsweise `admin` oder `reception` – zusammenfasst. Dadurch werden im Dashboard ausschließlich jene Navigationspunkte und Funktionen angezeigt, die zur jeweiligen Rolle passen. Diese Reduktion verbessert die Benutzerfreundlichkeit und minimiert das Risiko von Fehlbedienungen.

Für die Sicherheit ist entscheidend, dass die Autorisierung nicht allein im Frontend erfolgt. Da das Frontend im Browser ausgeführt wird, ist es prinzipiell manipulierbar. Daher ist es unerlässlich, dass das Backend bei jeder Anfrage prüft, ob ein Benutzer über die erforderliche Berechtigung für den aufgerufenen API-Endpunkt verfügt. Ein API-Endpunkt ist eine konkrete URL-Funktion des Backends, wie zum Beispiel `/visitor/checkin`. Nur durch diese serverseitige Prüfung kann sichergestellt werden, dass Funktionen nicht durch direkte URL-Aufrufe umgangen werden.

Zusammenfassend ermöglicht Keycloak in „outPort“ eine zentrale Anmeldung und stellt die notwendigen Rolleninformationen bereit, auf deren Basis sowohl die Benutzeroberfläche als auch der Zugriff auf Backend-Funktionen kontrolliert werden. [25]

## 4.5 UI-Bibliotheken

Für die Gestaltung der Benutzeroberfläche wurden UI-Bibliotheken eingesetzt, um Oberflächenelemente nicht manuell implementieren zu müssen und diese wiederverwendbar zu machen.

Der Einsatz etablierter Bibliotheken reduziert den Implementierungsaufwand und unterstützt zugleich eine konsistente und wartbare Benutzeroberfläche. Dafür werden **DevExtreme** sowie **i18next** in Verbindung mit **react-i18next** benutzt [26], [27], [28].

### 4.5.1 DevExtreme

Die Oberflächenelemente wurden mit der Bibliothek **DevExtreme** erstellt. Dabei handelt es sich um eine UI-Komponentenbibliothek für Webanwendungen, die Komponenten für Frameworks, wie React, bereitstellt [26]. Diese Bibliothek eignet sich insbesondere für datenorientierte Geschäftsanwendungen, da sie sowohl klassische Formularelemente als auch komplexere Komponenten wie Tabellen, Diagramme oder Navigationscontainer umfasst [26].

Der Vorteil von DevExtreme ist die Vereinheitlichung des Erscheinungsbildes. Da die Komponenten aus derselben Bibliothek stammen, entsteht eine konsistente Bedienoberfläche mit einheitlicher Interaktion und wiedererkennbarem Design. Darüber hinaus unterstützt DevExtreme vordefinierte Themes, wodurch sich das visuelle Erscheinungsbild der Anwendung anpassen lässt, ohne einzelne Komponenten separat gestalten zu müssen [29].

### 4.5.2 i18next + react-i18next

Für die Mehrsprachigkeit der Anwendung wurden **i18next** und **react-i18next** eingesetzt. i18next ist ein Internationalisierungs-Framework für JavaScript, während react-i18next die Anbindung dieses Frameworks an React übernimmt [27], [30]. Dadurch werden sprachabhängige Inhalte zentral verwaltet und in React-Komponenten direkt eingebunden.

Im Projekt dient diese Kombination dazu, die Benutzeroberfläche in mehreren Sprachen bereitzustellen. Texte werden dabei nicht direkt in die Komponenten geschrieben, sondern über Übersetzungsschlüssel geladen. In den React-Komponenten erfolgt der Zugriff auf diese Übersetzungen über die von react-i18next bereitgestellten Funktionen, zum Beispiel `useTranslation`.[28], [30].

Insgesamt schaffen i18next und react-i18next eine geeignete Basis für die Internationalisierung der Anwendung. Die Bibliotheken ermöglichen eine saubere Trennung von Logik, Sprache und verbessern die Wartbarkeit der Oberfläche. [27], [28].

# 5 Implementierung

In diesem Kapitel werden die wichtigsten Funktionen und Mechanismen von „outPort“ und wie diese implementiert wurden aufgezeigt und erklärt.

## 5.1 Aufbau des Backends

Das Backend von „outPort“ ist als *ASP.NET Core Web API* auf Basis von *.NET 8* realisiert. Die Architektur ist darauf ausgelegt, die in Abschnitt 2.2.1 definierten personenbezogenen Daten sicher zu verarbeiten.

### 5.1.1 Controller und Services

Die Zusammenarbeit zwischen Controllern und Services folgt dem Prinzip der Schichtentrennung. Dieses Konzept stellt sicher, dass der Zugriff auf sensible Daten gemäß dem Datengeheimnis nach § 6 DSG (siehe Abschnitt 2.2.1) kontrolliert wird.

- **Controller:** Sie fungieren als API-Endpunkte und nehmen Anfragen entgegen. Sie stellen die erste Barriere dar, um sicherzustellen, dass Daten nur durch autorisierte Prozesse verarbeitet werden.
- **Services (`VisitorService`, `TruckerService`):** Hier liegt die Geschäftslogik. Die Services verwalten den Zugriff auf den `AppDbContext` und führen Operationen wie Check-in oder Check-out durch.

Die Bereitstellung der Funktionen erfolgt mittels *Dependency Injection*:

Listing 1: Zentralisierte Datenverarbeitung im TruckerService

```
1 public class TruckerService
2 {
3     private readonly AppDbContext _context;
4
5     public TruckerService(AppDbContext context)
6     {
7         _context = context; // Zugriffskontrolle auf die DB
8     }
9 }
```

### 5.1.2 Verarbeitung von Besucher- und Lieferantendaten

Bei der Verarbeitung der Daten wird konsequent das Prinzip der **Datenminimierung und Zweckbindung** verfolgt.

Im System werden nur die Datenfelder verarbeitet, die für den organisatorischen Zweck – die Zutrittsverwaltung und Sicherheit – notwendig sind. Dies entspricht den in Abschnitt 2.2.1 aufgelisteten Identifikations- und Lieferungsdaten.

Listing 2: Struktur der verarbeiteten Personen- und Zeitdaten

```
1  public class Visitor
2  {
3      public string FirstName { get; set; }
4      public string LastName { get; set; }
5      public string Organization { get; set; }
6      public DateTime? CheckIn { get; set; } // Zweck: Dokumentation Ankunft
7      public DateTime? CheckOut { get; set; } // Zweck: Dokumentation Verlassen
8  }
```

Der Prozess des Check-outs markiert das Ende des unmittelbaren Verarbeitungszwecks für den Aufenthalt am Gelände. Der `VisitorService` stellt durch die Validierung sicher, dass Daten nur korrekt abgeschlossen werden können. Das ist Voraussetzung für das **Aufbewahrungs- und Löschkonzept**.

Listing 3: Check-out Logik zur Einhaltung der Zweckbindung

```
1  public Visitor CheckOutVisitor(int id)
2  {
3      // Suche nach dem aktiven Datensatz (CheckOut == null)
4      var visitor = _context.Visitors.FirstOrDefault(v => v.Id == id && v.CheckOut == null);
5      if (visitor == null) throw new Exception("Besucher bereits ausgecheckt oder nicht
           vorhanden");
6
7      visitor.CheckOut = DateTime.Now; // Zeitstempel f r Nachvollziehbarkeit
8      _context.SaveChanges();
9      return visitor;
10 }
```

Sobald der `CheckOut`-Timestamp gesetzt ist, wird in den Zeilen 26 bis 43 des unteren Codes überprüft, ob die Daten schon älter sind als das `cutoffDate` und der Löschmechanismus löscht gegebenenfalls diese, wie es die Vorgaben des § 1 DSG verlangen [2].

Daten müssen nach Ablauf ihrer Aufbewahrungsfrist physisch aus der Datenbank entfernt werden. Hierfür wurde in der Applikation ein eigener Hintergrunddienst, der `DatabaseCleanupService`, implementiert.

Listing 4: Automatisierter Löschmechanismus als BackgroundService

```
1  using Microsoft.EntityFrameworkCore;
2  using outPort_React.Server.Data;
3
4  namespace outPort_React.Server.Services
5  {
6      public class DatabaseCleanupService : BackgroundService
7      {
8          private readonly IServiceProvider _serviceProvider;
```

```
9           private readonly ILogger<DatabaseCleanupService> _logger;
10          private readonly int _daysToKeep = 30; // Aufbewahrungsfrist laut L schkonzept
11
12          public DatabaseCleanupService(IServiceProvider serviceProvider,
                ILogger<DatabaseCleanupService> logger)
13          {
14              _serviceProvider = serviceProvider;
15              _logger = logger;
16          }
17
18          protected override async Task ExecuteAsync(CancellationToken stoppingToken)
19          {
20              while (!stoppingToken.IsCancellationRequested)
21              {
22                  _logger.LogInformation("Cleanup-Task startet: Pr fung auf abgelaufene
                        Datens tze...");
23
24                  using (var scope = _serviceProvider.CreateScope())
25                  {
26                      var context = scope.ServiceProvider.GetRequiredService<AppDbContext>();
27                      var cutoffDate = DateTime.Now.AddDays(-_daysToKeep);
28
29                      // 1. Besucher l schen, die l nger als X Tage ausgecheckt sind
30                      var expiredVisitors = context.Visitors
31                          .Where(v => v.CheckOut != null && v.CheckOut < cutoffDate);
32
33                      // 2. Trucker l schen, die l nger als X Tage ausgecheckt sind
34                      var expiredTruckers = context.Truckers
35                          .Where(t => t.CheckOut != null && t.CheckOut < cutoffDate);
36
37                      context.Visitors.RemoveRange(expiredVisitors);
38                      context.Truckers.RemoveRange(expiredTruckers);
39
40                      await context.SaveChangesAsync(stoppingToken);
41
42                      _logger.LogInformation("Bereinigung abgeschlossen.");
43                  }
44
45                  // Warte 24 Stunden bis zum n chsten Durchlauf
46                  await Task.Delay(TimeSpan.FromDays(1), stoppingToken);
47              }
48          }
49      }
50  }
```

Dieser Hintergrunddienst erbt von der abstrakten Klasse `BackgroundService` und läuft dauerhaft im Hintergrund der Applikation. In der Methode `ExecuteAsync` wird eine asynchrone Schleife gestartet, die in regelmäßigen Abständen eine Überprüfung der Datensätze und falls notwendig die Löschung von Datenbanksätzen vornimmt.

Zur Wahrung der Datenbankintegrität und Thread-Sicherheit wird bei jedem Durchlauf über den `IServiceProvider` ein neuer *Scope* (Gültigkeitsbereich) erstellt, um sicher auf den `AppDbContext` zuzugreifen. Anschließend berechnet das Programm ein Ablaufdatum (`cutoffDate`), welches sich aus dem aktuellen Datum abzüglich der definierten Aufbewahrungsfrist von 30 Tagen (`_daysToKeep`) ergibt.

Die entscheidende datenschutzrechtliche Filterung geschieht in den LINQ-Abfragen: Es werden ausschließlich jene Datensätze in die Löschliste aufgenommen, bei denen der `CheckOut`-Timestamp gesetzt ist und die älter als das `cutoffDate` sind. Durch die Methode `RemoveRange` werden diese Einträge aus den jeweiligen Tabellen (`Visitors` und `Truckers`) markiert und mit `SaveChangesAsync` unwiderruflich aus der Datenbank gelöscht.

## 5.2 Umsetzung des Besucherprozesses

Beim Besucherprozess durchläuft die Anwendung mehrere Ebenen. Die Voranmeldung, wie der Check-in und Check-out gehandhabt werden, wie mit Schreibfehlern im Check-in umgegangen wird und wie aktive und geplante Besucher verwaltet werden, wird in diesem Kapitel erklärt.

### 5.2.1 Voranmeldung und Fuzzy Search

Die Voranmeldung eines Besuchers wird im Backend über eine Methode verarbeitet. Dabei wird ein neuer Besucherdatensatz angelegt und zunächst ohne tatsächlichen Check-in bzw. Check-out gespeichert. Zusätzlich kann im Zuge der Voranmeldung automatisch ein Kalendereintrag erstellt werden, sofern eine erwartete Besuchszeit und eine E-Mail-Adresse hinterlegt sind.

Der folgende Codeausschnitt zeigt vereinfacht die Speicherung eines vorangemeldeten Besuchers sowie die optionale Erstellung eines Kalendereintrags:

Listing 5: Vereinfachte Verarbeitung einer Voranmeldung

```
public Visitor RegisterVisitor(Visitor visitor)
{
    visitor.CheckIn = null;
    visitor.CheckOut = null;

    _context.Visitors.Add(visitor);
    _context.SaveChanges();

    if (visitor.CheckInExpected.HasValue && !string.IsNullOrWhiteSpace(visitor.Mail))
    {
        var start = visitor.CheckInExpected.Value;
        DateTime end;

        if (visitor.VisitDuration == 480 || visitor.VisitDuration == null ||
            visitor.VisitDuration <= 0)
        {
            end = new DateTime(start.Year, start.Month, start.Day, 17, 0, 0);
        }
        else
        {
            end = start.AddMinutes(visitor.VisitDuration.Value);
        }

        _calendarService.CreateVisitorAppointment(
            recipientUpn: visitor.Mail,
            firstName: visitor.FirstName,
            lastName: visitor.LastName,
            organization: visitor.Organization,
            checkIn: start,
            checkOut: end
        );
    }

    return visitor;
}
```

Im gezeigten Ausschnitt wird der Besucher zunächst als geplanter Eintrag gespeichert. Dabei bleiben die Felder „Check-in“ und „Check-out“ leer, da der eigentliche Besuch noch nicht begonnen hat. Ist zusätzlich eine erwartete Besuchszeit vorhanden und wurde eine E-Mail-Adresse hinterlegt, wird ein Kalendereintrag über den Kalender-Service erstellt. Dadurch wird

die Voranmeldung nicht nur in der Datenbank gespeichert, sondern kann auch organisatorisch weiterverwendet werden.

Beim späteren Check-in wird geprüft, ob für den Besucher bereits ein geplanter Eintrag vorhanden ist. Da Eingaben in der Praxis leicht von der ursprünglichen Schreibweise abweichen können, wird dafür „Fuzzy Search“ (eine fehlertolerante Suche) verwendet.

Listing 6: "Fuzzy Search"nach vorangemeldeten Besuchern

```
1       var openVisitors = _context.Visitors
2           .Where(v => v.CheckOut == null)
3           .ToList();
4
5       var existing = openVisitors.FirstOrDefault(v =>
6       Fuzz.Ratio(v.FirstName.ToLower(), firstName.ToLower()) >= 90 &&
7       Fuzz.Ratio(v.LastName.ToLower(), lastName.ToLower()) >= 90 &&
8       Fuzz.Ratio(v.Organization.ToLower(), organization.ToLower()) >= 90
9       );
```

Hier werden zunächst alle noch offenen Besuchereinträge geladen. Dann wird geprüft, ob bereits ein vorhandener Datensatz zu den eingegebenen Werten passt. Dabei werden Vorname, Nachname und Organisation jeweils mit einem Ähnlichkeitswert verglichen. Erst wenn alle drei Vergleiche einen ausreichend hohen Wert erreichen, wird der bestehende Besucher als Treffer übernommen. Dadurch werden auch leicht abweichende Eingaben, etwa durch Tippfehler oder unterschiedliche Schreibweisen, trotzdem korrekt zugeordnet.

### 5.2.2 Check-in und Check-out

Der Check-in eines Besuchers wird im Backend über die Methode „CheckInOrCreateVisitor“ verarbeitet. In dieser Funktion kommt auch die „Fuzzy Search“ aus Abschnitt 5.2.1 vor. Dabei wird zunächst geprüft, ob bereits ein offener Besuchereintrag vorhanden ist. Wird ein passender Datensatz gefunden, wird der Besucher nicht neu angelegt, sondern im schon vorhandenen Datensatz der Check-in-Zeitpunkt gesetzt.

Der folgende Codeausschnitt zeigt den Teil der Methode, in dem ein vorhandener Besucher eingecheckt hat und anschließend weiterverarbeitet wird.

Listing 7: Vereinfachte Verarbeitung des Check-ins

```
1       if (existing != null)
2       {
3           existing.CheckIn = DateTime.Now;
4           _context.SaveChanges();
5
6           if (existing?.CheckInExpected.HasValue == true &&
                existing.CheckInExpected.Value.Date > DateTime.Now.Date)
7           {
8               await _calendarService.DeleteVisitorAppointmentAsync(existing.Mail,
                    existing.CheckInExpected.Value);
9               existing.CheckInExpected = null;
10          }
11
```

```
12          if (!string.IsNullOrWhiteSpace(existing.Mail))
13          {
14              _calendarService.SendVisitorMailAsync(
15                  mail: existing.Mail,
16                  firstName: existing.FirstName,
17                  lastName: existing.LastName,
18                  visitTime: existing.CheckIn.Value
19              );
20          }
21
22          return existing;
23      }
```

Im gezeigten Ausschnitt wird nach dem Auffinden eines bestehenden Besuchers der Check-in-Zeitpunkt gesetzt und in der Datenbank gespeichert. Zusätzlich kann, falls vorhanden, der Termin beim Gastgeber gelöscht und diesem eine Ankunftsbenachrichtigung gesendet werden.

Der Check-out wird in einer eigenen Methode verarbeitet. Dabei wird geprüft, ob der betreffende Besucher ausgecheckt wurde. Falls das nicht der Fall ist, wird der Check-out-Zeitpunkt gesetzt und der Datensatz aktualisiert.

Listing 8: Verarbeitung des Check-outs

```
1  public Visitor CheckOutVisitor(int id)
2  {
3      var visitor = _context.Visitors.FirstOrDefault(v => v.Id == id && v.CheckOut == null);
4      if (visitor == null) throw new Exception("Visitor not found or already checked out");
5
6      visitor.CheckOut = DateTime.Now;
7      _context.SaveChanges();
8      return visitor;
9  }
```

Dadurch wird sichergestellt, dass der Besucher nur einmal ausgecheckt werden kann. Mit dem gesetzten Check-out-Zeitpunkt erscheint der Datensatz auch nicht mehr in der Übersicht der aktuell anwesenden Besucher.

### 5.2.3 Verwaltung aktiver und geplanter Besucher

Um geplante und aktive Besucher zu verwalten, wird im Projekt der Datenkontext „AppDbContext“ verwendet, über den auf die Besuchereinträge zugegriffen werden kann. Jeder Besucher wird dabei als eigener Datensatz mit dem Typ „Visitor“ gespeichert.

Der folgende Codeausschnitt zeigt vereinfacht den zentralen Datenkontext sowie die für die Verwaltung relevanten Felder eines Besuchers.

Listing 9: Zentrale Speicherung von Besucherdaten

```
1  public class AppDbContext : DbContext
2  {
3      public DbSet<Visitor> Visitors => Set<Visitor>();
4  }
5
6  public class Visitor
7  {
8      public int Id { get; set; }
```

```
9       public string FirstName { get; set; }
10      public string LastName { get; set; }
11      public string Organization { get; set; }
12      public DateTime? CheckIn { get; set; }
13      public DateTime? CheckOut { get; set; }
14      public DateTime? CheckInExpected { get; set; }
15  }
```

Die Unterscheidung zwischen geplanten, aktiven und abgeschlossenen Besuchen erfolgt dabei über die gespeicherten Zeitinformationen. Ein vorangemeldeter Besucher besitzt noch keinen Check-in-Zeitpunkt. Ein aktiver Besucher hat bereits einen Check-in-Zeitpunkt, aber noch keinen Check-out-Zeitpunkt. Sobald ein Check-out vorhanden ist, gilt der Besuch als abgeschlossen und wird nicht mehr als aktiv geführt.

Die eigentliche Zuordnung zu den jeweiligen Ansichten erfolgt im Frontend innerhalb des Dashboards. Dort werden die vom Backend geladenen Besucherdaten anhand der Felder „Check-in“ und „Check-out“ gefiltert.

Listing 10: Unterscheidung zwischen registrierten und aktiven Besuchern im Dashboard

```
1  const registeredVisitors = visitors.filter(v => !v.checkIn).length;
2  const activeVisitors = visitors.filter(v => v.checkIn && !v.checkOut).length;
```

Im Dashboard werden dadurch Besucher ohne gesetzten Check-in als registrierte Besucher angezeigt. Besucher mit vorhandenem Check-in, aber ohne Check-out, erscheinen in der Übersicht der aktiven Besucher. Abgeschlossene Besuche bleiben zwar in der Datenbank erhalten, werden jedoch nicht mehr in der aktiven Ansicht angezeigt.

Zusätzlich stellt das Backend über einen Endpunkt alle Besucherdaten für das Dashboard bereit. Die eigentliche Darstellung und Trennung der Ansichten erfolgt anschließend clientseitig im Frontend.

## 5.3 Dokumentenerzeugung

Im Projekt „outPort“ wird ein Besucherausweis als PDF erstellt, um Besuchern eine kompakte und eindeutig zuordenbare Identifikation für den Aufenthalt am Standort bereitzustellen. Die Erstellung erfolgt serverseitig. Dies bedeutet, dass die Generierung nicht im Browser, sondern auf dem Server durchgeführt wird, wodurch Layout und Ausgabe zentral kontrolliert werden können.

### 5.3.1 Erstellung des Besucherausweises

Der Besucherausweis ist im aktuellen Stand final umgesetzt und enthält Name, Firma sowie die Besuchszeit. Letzteres wird aus dem Check-in-Zeitpunkt und der gespeicherten Besuchsdauer berechnet. Der Begriff Check-in beschreibt den Vorgang, bei dem ein Besucher beim Eintreffen als anwesend markiert wird. Die Besuchsdauer definiert dabei die geplante Aufenthaltszeit in Minuten.

Die Generierung wird im Backend über eine Service-Methode ausgelöst. Dabei wird zunächst der entsprechende Besucher aus der Datenbank geladen und anschließend die PDF-Erzeugung über eine Generator-Klasse gestartet. Das Ergebnis wird als Byte-Array zurückgegeben, welches anschließend über einen HTTP-Endpunkt als PDF-Datei an das Frontend geliefert wird.

Listing 11: Auslösen der PDF-Erzeugung im VisitorService

```
public byte[] GenerateVisitorPassPdf(int visitorId)
{
    var visitor = _context.Visitors.FirstOrDefault(v => v.Id == visitorId);
    if (visitor == null) return null;

    var generator = new VisitorPass();
    return generator.Generate(visitor);
}
```

Die eigentliche Formatierung des Ausweises erfolgt in der Klasse `VisitorPass`. Für die PDF-Erstellung wird QuestPDF verwendet. QuestPDF ist eine Bibliothek, mit der PDF-Dokumente über ein deklaratives Layoutmodell direkt im Code beschrieben werden können [31]. Der Ausweis wird als A6-Seite umgesetzt, da dieses Format optimal für Besucherausweise geeignet ist. Das Layout umfasst einen Kopfbereich mit Titel, einen Informationsblock (Name, Firma, Besuchszeit) sowie einen Bereich für den QR-Code.

Listing 12: Berechnung der Besuchszeit für den Ausweis (Auszug)

```
string visitTime;
if (visitor.CheckIn.HasValue && visitor.VisitDuration.HasValue)
{
    var endTime = visitor.CheckIn.Value.AddMinutes(visitor.VisitDuration.Value);
    visitTime = $"{visitor.CheckIn.Value:HH:mm} - {endTime:HH:mm}";
}
else if (visitor.CheckIn.HasValue)
{
    visitTime = $"{visitor.CheckIn.Value:HH:mm} - (Endzeit unbekannt)";
}
else
{
    visitTime = "Besuchszeit unbekannt";
}
```

### 5.3.2 QR-Code und PDF-Ausgabe

Zusätzlich zu den Textinformationen enthält der Ausweis einen QR-Code. In „outPort“ wird dieser verwendet, um den Check-out-Prozess am Standort zu vereinfachen.

Der QR-Code enthält eine URL, die den automatischen Check-out eines konkreten Besuchers auslöst. Die URL beinhaltet die `visitor.Id`, wodurch der Besucher eindeutig referenziert wird. Für die QR-Code-Erzeugung wird die Bibliothek QRCoder verwendet, welche aus einem Textinhalt eine Grafik generiert [32].

Listing 13: Erzeugung des QR-Codes aus einer Check-out-URL (Auszug)

```
var checkOutUrl = $"https://localhost:54250/visitor/auto-checkout/{visitor.Id}";

using var qrGenerator = new QRCodeGenerator();
var qrData = qrGenerator.CreateQrCode(checkOutUrl, QRCodeGenerator.ECCLevel.Q);
var qrCode = new BitmapByteQRCode(qrData);
byte[] qrBytes = qrCode.GetGraphic(20);
```

Die erzeugten QR-Code-Bytes werden anschließend in das PDF eingebettet und im Dokument zentriert dargestellt. Zusätzlich wird ein Hinweistext ausgegeben, der den Zweck des Codes erläutert.

Listing 14: Einbetten des QR-Codes in das PDF (Auszug)

```
qrBlock.Item()
    .AlignCenter()
    .Height(120)
    .Width(120)
    .Image(qrBytes, ImageScaling.FitArea);

qrBlock.Item()
    .PaddingTop(4)
    .Text("Scannen Sie den QR-Code zum Auschecken.")
    .FontSize(10)
    .AlignCenter();
```

Im Frontend wird der Download des PDFs über eine Schaltfläche ausgelöst. Dabei wird der PDF-Endpunkt aufgerufen, die Antwort als Binärdaten geladen und anschließend als Datei gespeichert. Binärdaten sind Rohdaten, die als Bytes übertragen werden und nicht als Text vorliegen. Der Browser erhält dadurch eine valide PDF-Datei, die gedruckt werden kann.

Listing 15: PDF-Download im Frontend (Auszug)

```
const response = await fetch(`https://localhost:7061/visitor/pdf/${visitorId}`);
const blob = await response.blob();
const url = window.URL.createObjectURL(blob);

const link = document.createElement('a');
link.href = url;
link.setAttribute('download', `VisitorPass_${visitorId}.pdf`);
document.body.appendChild(link);
link.click();
link.remove();
```

Damit ergibt sich ein durchgängiger Ablauf: Der Benutzer fordert den Ausweis im Dashboard an, das Backend generiert das Dokument inklusive QR-Code und das Frontend stellt den Download bereit. Der QR-Code erweitert den Ausweis um eine maschinenlesbare Komponente, die den Prozess signifikant beschleunigt.

## 5.4 Externe Anbindungen

Im Projekt „outPort“ wird bei der Voranmeldung eines Besuchers ein Termin im Outlook-Kalender des Gastgebers angelegt. Wenn dieser Besucher dann am Empfang eincheckt, wird dem Gastgeber eine Ankunfts-E-Mail gesendet. Die Implementierung wird in diesem Kapitel erläutert.

### 5.4.1 E-Mail-Benachrichtigung

Die automatische E-Mail-Benachrichtigung wird ausgelöst, sobald ein Besucher eincheckt, dem ein Gastgeber zugewiesen ist. Dadurch muss der Empfang den Gastgeber nicht mehr manuell verständigen. Das System benachrichtigt den Gastgeber und reduziert somit den Aufwand am Empfang.

Der folgende Codeausschnitt zeigt die Methode, welche die E-Mail über die Ankunft des Besuchers erstellt und versendet:

Listing 16: Automatische Benachrichtigung des Gastgebers

```
1  public async Task SendVisitorMailAsync(string mail, string firstName, string lastName,
       DateTime visitTime)
2  {
3      var message = new Message
4      {
5          Subject = "Ihr Besucher ist eingetroffen",
6          Body = new ItemBody
7          {
8              ContentType = BodyType.Text,
9              Content = $"Guten Tag,\n\n" +
10                       $"Ihr Besucher {firstName} {lastName} hat sich soeben bei uns am
                             Empfang eingecheckt.\n" +
11                       $"Uhrzeit: {visitTime:dd.MM.yyyy HH:mm}\n\n" +
12                       $"Mit freundlichen Gr\"u\ssen\nIhr Portier-Team"
13         },
14         ToRecipients = new List<Recipient>
15         {
16             new Recipient
17             {
18                 EmailAddress = new EmailAddress
19                 {
20                     Address = mail
21                 }
22             }
23         }
24     };
25
26     await _graphClient.Users["lena.hoffmann@devsysco.onmicrosoft.com"].SendMail.PostAsync(
27         new Microsoft.Graph.Users.Item.SendMail.SendMailPostRequestBody
28         {
29             Message = message,
30             SaveToSentItems = true
31         });
32 }
```

Es wird zuerst der Text der Nachricht aufgebaut. Dabei werden Vorname, Nachname und die Uhrzeit in einen vorgefertigten Text eingefügt. Anschließend wird die E-Mail über den Graph-Client versendet. Somit ist die Benachrichtigung direkt in den Check-in-Prozess eingebaut und muss nicht über ein separates System manuell ausgelöst werden.

### 5.4.2 Outlook-Kalenderintegration

Die Kalenderintegration ist dazu da, den geplanten Besuch nicht nur im System zu visualisieren, sondern auch als Termin im Outlook-Kalender des zuständigen Gastgebers abzubilden.

Der folgende Codeausschnitt zeigt die Erstellung eines Kalendereintrags für einen Besucher:

Listing 17: Erstellung eines Outlook-Kalendereintrags

```
public async Task<bool> CreateVisitorAppointment(string recipientUpn, string firstName,
    string lastName,
    string organization, DateTime checkIn, DateTime checkOut)
{
    try
    {
        var @event = new Event
        {
            Subject = $"Neuer Besucher: {firstName} {lastName}",
            Body = new ItemBody
            {
                ContentType = BodyType.Text,
                Content = $"Organisation: {organization}\nCheck-In: {checkIn}"
            },
            Start = new DateTimeTimeZone
            {
                DateTime = checkIn.ToString("s"),
                TimeZone = "Europe/Berlin"
            },
            End = new DateTimeTimeZone
            {
                DateTime = checkOut.ToString("s"),
                TimeZone = "Europe/Berlin"
            },
            Location = new Location
            {
                DisplayName = "Empfang"
            }
        };

        await _graphClient.Users[recipientUpn].Events.PostAsync(@event);
        return true;
    }
    catch
    {
        return false;
    }
}
```

Beim Erstellen des Termins werden Name, Organisation sowie die Start- und Endzeit des Besuchs in ein Outlook-Ereignis übernommen. Anschließend wird der Kalendereintrag über den Graph-Client im Kalender des Gastgebers gespeichert.

Zusätzlich ist im Projekt integriert, den Termin wieder zu löschen, sobald der Besucher eingecheckt wurde. Der nachfolgende Code zeigt die Methode zum Löschen des Termins:

Listing 18: Löschen eines vorhandenen Kalendereintrags

```
public async Task<bool> DeleteVisitorAppointmentAsync(string recipientUpn, DateTime
    appointmentDate)
{
    try
    {
        var events = await _graphClient.Users[recipientUpn].Events.GetAsync();

        var match = events?.Value?.FirstOrDefault(e =>
            e.Subject != null &&
            e.Subject.StartsWith("Neuer Besucher:") &&
            e.Start?.DateTime != null &&
```

```
11                  DateTime.Parse(e.Start.DateTime).Date == appointmentDate.Date
12              );
13
14              if (match == null)
15                  return false;
16
17              await _graphClient.Users[recipientUpn].Events[match.Id].DeleteAsync();
18              return true;
19          }
20          catch
21          {
22              return false;
23          }
24      }
```

# 6 Organisation

## 6.1 Risiken und Herausforderungen

Die Risiken, die sich während der Entwicklung des Projekts ergeben haben beziehungsweise die im Vorhinein schon identifiziert wurden, sind auf drei Stakeholder aufgeteilt.

Die Stakeholder in dieser Arbeit bestehen hauptsächlich aus der SYSco EDV GmbH, den einzelnen Mitgliedern der Diplomarbeit und der HTL Perg. Die SYSco EDV gilt als primärer Stakeholder und hat sowohl fachliche als auch organisatorische Bedingungen vorgegeben, die ebenfalls als Risiko gelten, da Ansprüche an das Design sowie an den Code den Projektverlauf verändern konnten. Um diesen Risikofaktor zu minimieren, wurden während des Praktikums zwischen dem 7.7.2025 und dem 1.8.2025 wöchentliche Meetings abgehalten, um sich auf die Fortführung der Entwicklung zu einigen.

Das „outPort“-Team als Stakeholder hatten die Herausforderung, mit einem noch teilweise unbekannten Technologie-Stack zu arbeiten, was zu Verzögerungen in der Entwicklung führen konnte, sowie dazu, dass es in den Meetings mit der SYSco EDV GmbH zu Missverständnissen kommen und sich die Qualität der Arbeit verändern konnte. Eine weitere Herausforderung stellte die Implementierung von Keycloak dar, da dieser Dienst separat von der vorliegenden Applikation betrieben wird und dadurch die Applikation von diesem Dienst abhängig ist. Um diese Risiken zu minimieren, wurden uns von der SYSco GmbH Ansprechpersonen zur Verfügung gestellt, um Fragen zu beantworten und Probleme wie diese gemeinsam zu lösen.

Ein weiteres Risiko ergibt sich daraus, dass das „outPort“-Team mit der Durchführung der Diplomarbeit zugleich die HTL Perg nach außen repräsentiert. Dadurch ist auch die Schule als Stakeholder zu betrachten, da zusätzlich zu den Anforderungen des Unternehmens und des Teams auch Erwartungen, Vorgaben und Bewertungskriterien der HTL-Perg zu berücksichtigen sind. Um dieses Risiko zu verringern, werden Vorlagen sowie weitere Hilfestellungen bereitgestellt, die den Verlauf der Arbeit unterstützen und dabei helfen, flexibler auf Fehler zu reagieren.

## 6.2 Projektplanung und Meilensteine

Im Rahmen des Projekts „outPort“ wurde die Umsetzung in mehrere aufeinander aufbauende Meilensteine gegliedert. Durch diese strukturierte Planung wird sichergestellt, dass die Entwicklung schrittweise vollzogen wird und einzelne Funktionen unabhängig voneinander getestet und erweitert werden können.

Die Meilensteine orientieren sich an den zentralen Anforderungen eines digitalen Besuchermanagementsystems. Dabei wurde mit grundlegenden Funktionen begonnen (z. B. Registrierung und Datenverwaltung), um das System anschließend sukzessive um erweiterte Features wie Reporting, die Integration externer Systeme sowie die Benutzerverwaltung zu erweitern.

### 6.2.1 Übersicht der Meilensteine

Die folgende Auflistung zeigt die definierten Meilensteine inklusive Zielsetzung und geplantem Fertigstellungsdatum.

- **Selbstregistrierung mit Multimedia-Funktionen (bis 09.08.2025)**
  Entwicklung einer responsiven Selbstregistrierungslösung für Kiosksysteme und mobile Geräte. Die Lösung umfasst interaktive Elemente wie Bilder, Videos und Anleitungen zur Besucherunterweisung.
- **Voranmeldung durch Mitarbeiter (bis 16.08.2025)**
  Implementierung eines Moduls zur Voranmeldung von Besuchern bis zu 30 Tage im Voraus. Die erfassten Daten werden beim späteren Check-in automatisch übernommen.
- **Erstellung von Besucherausweisen (bis 23.08.2025)**
  Umsetzung einer Funktion zur automatischen Generierung eines Besucherausweises beim Check-in, inklusive Name, Firma und berechneter Besuchszeit.
- **Benachrichtigung an den Gastgeber (bis 30.08.2025)**
  Entwicklung eines Systems zur automatischen Benachrichtigung des Gastgebers per E-Mail unmittelbar nach dem Eintreffen eines Besuchers.
- **Zentrale Besucherdatenbank und Reporting (bis 06.09.2025)**
  Aufbau einer zentralen Datenbank zur Speicherung aller Besucherdaten sowie Implementierung eines Reportingmoduls für statistische Auswertungen.

- **Integration von Terminplanungstools (bis 13.09.2025)**
  Anbindung an externe Kalendersysteme (z. B. Microsoft Outlook), um Besuchstermine automatisch zu synchronisieren.
- **Benutzerfreundliches Admin-Dashboard (bis 20.09.2025)**
  Entwicklung einer zentralen Oberfläche zur Verwaltung von Besuchern, zur Live-Übersicht des Standorts sowie zur Durchführung administrativer Aufgaben.
- **Zugangskontrolle und Rollenkonzept (bis 27.09.2025)**
  Umsetzung eines Rollen- und Berechtigungskonzepts mit differenzierten Benutzerrechten sowie der Protokollierung sicherheitsrelevanter Aktionen.
- **Anpassbare Anmeldeformulare (bis 04.10.2025)**
  Erweiterung des Systems zur Definition individueller Formularfelder durch Administratoren, um standortspezifische Daten abzufragen.
- **Corporate-Branding-Integration (bis 11.10.2025)**
  Implementierung der Möglichkeit zur Anpassung von Logos, Farben und Designelementen entsprechend der Corporate Identity des Unternehmens.
- **Mehrsprachigkeit (bis 18.10.2025)**
  Bereitstellung der Benutzeroberfläche in mehreren Sprachen sowie Sicherstellung der Erweiterbarkeit über zentrale Konfigurationsdateien.

# 7 Aufgabenverteilung

Um die Projektziele effizient umzusetzen, wurde das Gesamtprojekt in drei fachliche Teilbereiche gegliedert und den Projektmitgliedern wie folgt zugeteilt:

## 7.1 Buchner Herbert

Der Schwerpunkt von Herbert Buchner lag auf der Integration externer Systeme sowie der Automatisierung zentraler Arbeitsabläufe. Er realisierte den automatisierten Versand von Benachrichtigungen und implementierte ein System zum Druck von Besucherausweisen direkt vor Ort. Darüber hinaus analysierte und band er technische Schnittstellen (APIs) an bestehende Systeme wie Outlook an. Durch diese Maßnahmen gestaltete er den Besuchermanagementprozess effizienter und stellte gleichzeitig die Einhaltung der rechtlichen Anforderungen, insbesondere im Hinblick auf den Datenschutz (DSGVO), sicher.

## 7.2 Hinterdorfer Julian

Julian Hinterdorfer hat ein modernes Frontend für das Besuchermanagement gebaut. Besucher können sich jetzt ganz einfach selbst über ein Kiosk-Terminal oder das Smartphone anmelden. Bei der Entwicklung war es ihm besonders wichtig, dass die Bedienung so intuitiv ist, dass jeder damit klarkommt – egal wie technikaffin man ist. Er hat multimediale Inhalte wie Sicherheitsunterweisungen eingebunden und dafür gesorgt, dass das System mehrsprachig ist und die Formulare flexibel angepasst werden können. Außerdem hat er ein Branding-Konzept umgesetzt, mit dem die App optisch genau an das Design der jeweiligen Firma angepasst werden kann.

## 7.3 Inspruckner Janick

Janick Inspruckner hat ein skalierbares und datenschutzkonformes Backend für die Besucherdaten aufgebaut. Er hat dafür gesorgt, dass alle Daten strukturiert gespeichert, effizient ausgewertet und übersichtlich in Berichten dargestellt werden können. Neben der passenden Datenbankstruktur

hat er Funktionen wie das Filtern von Daten, die Generierung von Statistiken und verschiedene Exportmöglichkeiten umgesetzt. Zusätzlich hat er eine rollenbasierte Rechteverwaltung eingebaut, damit unterschiedliche Gruppen wie Portiere oder Admins genau auf die Funktionen zugreifen können, die sie für ihre Arbeit brauchen.

# 8 Resümee

## 8.1 Resümee Buchner Herbert

Die Zusammenarbeit mit SYSco EDV war für mich ein sehr guter Einblick in die spätere Arbeitswelt. Wir haben während des Praktikums den Großteil der Diplomarbeit mit der Hilfe von Dominik Bindreiter und Andreas Aichinger implementiert und konnten Woche für Woche Fortschritte sehen, was im Vergleich zu den Projekten in der Schule viel schneller ging und auch wesentlich effizienter erschien, da man sich auf eine Aufgabe fokussieren konnte. Nach zwei Meetings zur Besprechung des Fortschritts bekamen wir verständliche Kritik, die wir dann umsetzen konnten, sodass beide Seiten zufrieden waren.

Das Schreiben der eigentlichen Diplomarbeit war für mich eine Aufgabe, bei der ich mich zu Beginn sehr überwältigt gefühlt habe, da wir noch nie in so einem Ausmaß gemeinsam mit Mitschülern einen solchen technischen Bericht geschrieben hatten. Mit der Zeit und durch die Kritik von Dir. Prof. Dipl.-Ing. Dr. Michael Buchberger konnte ich ein Gefühl für die Konzipierung entwickeln und habe immer besser meine Arbeitsbereiche konstruktiv beschreiben können, was auch in der Zukunft einen Mehrwert für mich erbringt.

## 8.2 Resümee Hinterdorfer Julian

Im Rahmen dieser Diplomarbeit konnte ich sowohl fachlich als auch persönlich sehr positive Erfahrungen sammeln. Besonders hervorheben möchte ich die Zusammenarbeit mit unserem Auftraggeber SYSco EDV. Da es teilweise sehr herausfordernd war, sich in neue Technologien wie Vite und React einzuarbeiten, war ich sehr froh, dass wir von Dominik Bindreiter und Andreas Aichinger sehr schnell Antworten auf unsere Fragen bekommen haben.

Die Implementierung der Diplomarbeit konnten wir hauptsächlich im Zuge des Praktikums im Sommer bei SYSco EDV durchführen. Dies war ebenfalls eine sehr positive Erfahrung und dafür möchte ich nochmal Danke sagen.

Das Verfassen der Diplomarbeit selbst war durchaus eine Herausforderung. Eine technische Arbeit zu schreiben ist doch nochmal etwas ganz anderes als einen "normalen"Text im Deutschunterricht zu schreiben. Hierbei möchte ich mich sehr herzlichst bei Dir. Prof. Dipl.-Ing. Dr. Michael Buchberger bedanken, der uns zu Beginn die grundlegenden Regeln und Formen der Diplomarbeit erklärt hat.

Schlussendlich blicke ich sehr positiv auf die Diplomarbeitszeit zurück und kann sagen, dass ich sehr viele wertvolle Erfahrungen gesammelt habe.

Ein besonderer Dank gilt abschließend nochmals der Firma SYSco EDV für die Möglichkeit, dieses Projekt im Rahmen des Praktikums umzusetzen, sowie für die kontinuierliche Unterstützung während der gesamten Projektlaufzeit.

## 8.3 Resümee Inspruckner Janick

Die Diplomarbeit war für mich ein gutes Erlebnis, da ich mich in neue Themengebiete einarbeiten und mit Herausforderungen auseinandersetzen musste. Das selbständige Einarbeiten in neue Inhalte anhand großer Dokumentationen hat mir gezeigt, wie eigenverantwortliches Arbeiten im Arbeitsalltag aussehen kann, weil kein Lehrer zur Verfügung steht, der einem kurz mal eine Frage beantwortet. Diese Erfahrung hat mir mehr Selbstvertrauen im Umgang mit neuen Technologien und komplexen Aufgabenstellungen gegeben.

Das Praktikum in der Firma SYSco EDV war auch eine positive Erfahrung. Durch die Unterstützung von Dominik Bindreiter und Andreas Aichinger konnten viele offene Fragen geklärt werden, was uns die Arbeit erheblich erleichterte. Besonders geholfen hat uns das regelmäßige Feedback, wodurch wir unsere Anwendung stetig verbessert haben.

Das Verfassen der Diplomarbeit war jedoch eine große Herausforderung. Wirklich keine stilistischen Mittel zu verwenden wie wir es im Deutschunterricht gelernt haben war sehr schwierig. Jedoch möchte ich mich herzlich bei Dir. Prof. Dipl.-Ing. Dr. Michael Buchberger bedanken, der uns gleich zu Beginn die Grundregeln der Diplomarbeit erklärt hat.

Insgesamt hat mir das Entwickeln und Schreiben der Diplomarbeit sehr viel Spaß gemacht, und ich habe persönlich sehr viel mitgenommen.

# 9 Individuelle Inhaltsverzeichnisse

Im folgenden Punkt ist festgehalten, wer welche Kapitel der Diplomarbeit verfasst hat.

## Inhaltsverzeichnis Janick Inspruckner

# Inhaltsverzeichnis Julian Hinterdorfer

# Inhaltsverzeichnis Herbert Buchner

# Literaturverzeichnis


[1] Amy Bampton, *What is a visitor management system? A comprehensive guide*, letzter Zugriff am 18.02.2026 15:34. Adresse: https://signinapp.com/blog/what-is-a-visitor-management-system.

[2] Bundeskanzleramt der Republik Österreich, *Gesamte Rechtsvorschrift für Datenschutzgesetz, Fassung vom 11.03.2026*, Aktuelle konsolidierte Fassung, letzter Zugriff am 11.03.2026. Adresse: https://www.ris.bka.gv.at/GeltendeFassung.wxe?Abfrage=Bundesnormen&Gesetzesnummer=10001597.

[3] Europäisches Parlament und Rat der Europäischen Union. „Verordnung (EU) 2016/679 des Europäischen Parlaments und des Rates vom 27. April 2016 über den Schutz natürlicher Personen bei der Verarbeitung personenbezogener Daten (DSGVO), Art. 5 Abs. 1 lit. e.“ letzter Zugriff am 25.03.2026 17:42. Adresse: https://gdpr-info.eu/art-5-gdpr/.

[4] European Union, *Regulation (EU) 2016/679 (General Data Protection Regulation)*, letzter Zugriff am 18.02.2026. Adresse: https://eur-lex.europa.eu/eli/reg/2016/679/oj/eng.

[5] National Institute of Standards and Technology, *SP 800-53 Rev. 5: Security and Privacy Controls for Information Systems and Organizations*, letzter Zugriff am 18.02.2026. Adresse: https://csrc.nist.gov/pubs/sp/800/53/r5/final.

[6] Microsoft, *Introduction*, letzter Zugriff am 14.03.2026 12:49. Adresse: https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/language-specification/introduction.

[7] Microsoft, *Introduction to .NET*, letzter Zugriff am 14.03.2026 12:51. Adresse: https://learn.microsoft.com/en-us/dotnet/core/introduction.

[8] Microsoft, *Overview of ASP.NET Core*, letzter Zugriff am 14.03.2026 13:30. Adresse: https://learn.microsoft.com/en-us/aspnet/core/overview?view=aspnetcore-10.0.

[9] Microsoft, *Create web APIs with ASP.NET Core*, letzter Zugriff am 14.03.2026 13:37. Adresse: https://learn.microsoft.com/en-us/aspnet/core/web-api/?view=aspnetcore-10.0.

[10] Microsoft, *Entity Framework Core*, letzter Zugriff am 14.03.2026 14:16. Adresse: https://learn.microsoft.com/en-us/ef/core/.

[11] Microsoft, *DbContext Lifetime, Configuration, and Initialization*, letzter Zugriff am 14.03.2026 14:18. Adresse: https://learn.microsoft.com/en-us/ef/core/dbcontext-configuration/.

[12] QuestPDF, *QuestPDF*, letzter Zugriff am 14.03.2026 14:46. Adresse: https://www.questpdf.com/.

[13] QuestPDF, *Integration with ASP.NET*, letzter Zugriff am 14.03.2026 14:50. Adresse: https://www.questpdf.com/examples/aspnet-integration.html.

[14] Microsoft. „Übersicht über Microsoft Graph.“ letzter Zugriff am 20.03.2026 15:23. Adresse: https://learn.microsoft.com/de-de/graph/overview.

[15] Vite, *Getting Started (Guide)*, letzter Zugriff am 13.03.2026. Adresse: https://vitejs.dev/guide/.

[16] Vite, *Server Options (Vite Configuration)*, letzter Zugriff am 13.03.2026. Adresse: https://vitejs.dev/config/server-options.html.

[17] React, *Describing the UI*, letzter Zugriff am 13.03.2026. Adresse: https://react.dev/learn/describing-the-ui.

[18] React, *useState*, letzter Zugriff am 13.03.2026. Adresse: https://react.dev/reference/react/useState.

[19] React, *useEffect*, letzter Zugriff am 13.03.2026. Adresse: https://react.dev/reference/react/useEffect.

[20] TypeScript, *The TypeScript Handbook: Introduction*, letzter Zugriff am 13.03.2026. Adresse: https://www.typescriptlang.org/docs/handbook/intro.html.

[21] TypeScript, *Handbook: Object Types*, letzter Zugriff am 13.03.2026. Adresse: https://www.typescriptlang.org/docs/handbook/2/objects.html.

[22] Oracle. „What Is a Relational Database?“ letzter Zugriff am 18.03.2026 15:32. Adresse: https://www.oracle.com/database/what-is-a-relational-database/.

[23] Microsoft. „What is SQL Server?“ letzter Zugriff am 18.03.2026 15:55. Adresse: https://learn.microsoft.com/en-us/sql/sql-server/what-is-sql-server?view=sql-server-ver17.

[24] Microsoft. „Microsoft SQL Server EF Core Database Provider.“ letzter Zugriff am 18.03.2026 15:14. Adresse: https://learn.microsoft.com/en-us/ef/core/providers/sql-server/.

[25] Keycloak, *Keycloak Documentation*, letzter Zugriff am 16.03.2026. Adresse: https://www.keycloak.org/documentation.

[26] DevExpress. „Introduction to DevExtreme.“ letzter Zugriff am 18.03.2026 15:55. Adresse: https://js.devexpress.com/React/Documentation/Guide/Common/First_Steps/.

[27] i18next. „i18next Documentation: Introduction.“ letzter Zugriff am 18.03.2026 15:55. Adresse: https://www.i18next.com/.

[28] react-i18next. „Getting started | react-i18next documentation.“ letzter Zugriff am 18.03.2026 15:55. Adresse: https://react.i18next.com/getting-started.

[29] DevExpress. „Predefined Themes | React Documentation - DevExtreme.“ letzter Zugriff am 18.03.2026 15:55. Adresse: https://js.devexpress.com/React/Documentation/Guide/Themes_and_Styles/Predefined_Themes/.

[30] react-i18next. „Step by step guide / Using with hooks.“ letzter Zugriff am 18.03.2026 15:55. Adresse: https://react.i18next.com/latest/using-with-hooks.

[31] QuestPDF, *QuestPDF Documentation*, letzter Zugriff am 17.03.2026. Adresse: https://www.questpdf.com/documentation.html.

[32] QRCoder, *QRCoder (GitHub Repository)*, letzter Zugriff am 17.03.2026. Adresse: https://github.com/codebude/QRCoder.

# Abbildungsverzeichnis

# Tabellenverzeichnis

# Quellcodeverzeichnis

# Anhang

## A KI-Tools

Bei der Erstellung dieser Diplomarbeit wurden ChatGPT 5, sowohl als auch andere KI-Tools eingesetzt. Diese Tools sind unter den unten angegebenen Adressen online verfügbar. In der untenliegenden Tabelle werden alle benutzten KI-Hilfsmittel angeführt und für welchen Zweck diese verwendet worden sind.

- https://chatgpt.com/chat
- https://gemini.google.com/app?hl=de
- https://www.deepl.com/de/write

| **Tool** | **Verwendung** | **Bereich** | **Anmerkung** |
|---|---|---|---|
| ChatGPT (OpenAI) | ChatGPT wurde in dieser Arbeit genutzt, um die Formulierung mithilfe von eingegebenem Rohtext zu verbessern und um die Grammatik zu überprüfen. | Gesamte Arbeit | |
| Gemini (Google) | Gemini wurde zusätzlich zu ChatGPT verwendet, um Fehler im Ausdruck und in der Grammatik zu verbessern sowie die Recherche von Quellen zu erleichtern. | Gesamte Arbeit | |
| DeepL (DeepL SE) | DeepL wurde zusammen mit den anderen KI-Hilfstools verwendet, um den Satzaufbau zu verbessern und zur Übersetzung von deutschen Texten ins Englische. | Gesamte Arbeit | Speziell wurde DeepL in der Zusammenfassung zur Übersetzung des deutschen Textes verwendet. |

Tabelle 5: Einsatz von KI-basierten Hilfsmitteln