### Systemdokumentationen

| **Feld** | **Beschreibung** |
| --- | --- |
| **Name des Artefakts** | Systemdokumentationen |
| **Verantwortlich** | IT-Systemverantwortlicher |
| **Ziel des Dokuments** | Detaillierte technische Beschreibung des Systems zur Unterstützung der Wartung und Weiterentwicklung. |
| **Wesentliche Inhalte** | - Architekturdiagramme  - Beschreibung der Systemkomponenten  - Schnittstellenbeschreibungen |
| **Erstellungszeitpunkt** | Während der Entwicklungs- und Testphase des Projekts |
| **Nutzung im Projektverlauf** | Grundlage für die Wartung und Weiterentwicklung der Anwendung |
| **Abnahme** | Durch die IT-Systemverantwortlichen und die Entwickler; Überprüfung der Vollständigkeit und Korrektheit der Systemdokumentationen |
| **Abhängigkeiten** | - Technische Spezifikationen  - Systemarchitektur-Diagramme |
| **Versionierung** | Versionskontrolle mittels Versionsmanagement-Tools |
| **Vorlagen und Tools** | Textverarbeitungssoftware (z.B. MS Word, Google Docs) |
| **Zusätzliche Anmerkungen** | Regelmäßige Überprüfung und Aktualisierung der Systemdokumentationen notwendig, um sicherzustellen, dass sie den aktuellen technischen Anforderungen und Implementierungen entsprechen |

# Systemdokumentation für das CRM-System

## 1. Einleitung

Diese Systemdokumentation beschreibt das CRM-System und seine Erweiterung zur Befüllung mit Scoring-Daten. Sie dient als Leitfaden für Entwickler, Administratoren und Benutzer, um die Struktur und Funktionsweise des Systems zu verstehen.

## 2. Systemübersicht

Das CRM-System verwaltet Kundendaten und ermöglicht Vertriebsmitarbeitern, Kundeninteraktionen zu verfolgen. Die Erweiterung ermöglicht das Hochladen von Scoring-Daten, die monatlich von einem Drittanbieter bereitgestellt werden. Diese Daten werden per Excel-Datei hochgeladen und automatisch den entsprechenden Kunden zugeordnet und gespeichert.

## 3. Architektur

### 3.1 High-Level Architekturdiagramm

plaintext

+---------------------+ +-------------------+ +-------------------+ +------------------+

| | | | | | | |

| Benutzer +----->| Web-Frontend +----->| Backend-Server +----->| Datenbank |

| | | | | | | |

+---------------------+ +-------------------+ +-------------------+ +------------------+

| | | |

| | | |

| v v v

| +-------------------+ +-------------------+ +------------------+

| | Upload-Interface | | Validierungsmodul | | Datenbank |

| +-------------------+ +-------------------+ | (verschlüsselt) |

| | |

| v v

+--------------------------------------------> Fehlerlog

## 4. Komponentenbeschreibung

### 4.1 Web-Frontend

* **Beschreibung**: Benutzeroberfläche, die Vertriebsmitarbeitern den Zugang zum CRM-System ermöglicht.
* **Technologie**: React.js
* **Funktionalität**:
  + Login und Authentifizierung
  + Kundenverwaltung
  + Upload von Scoring-Daten

### 4.2 Backend-Server

* **Beschreibung**: Serverseitige Anwendung zur Verarbeitung der Anfragen des Web-Frontends.
* **Technologie**: Node.js mit Express.js
* **Funktionalität**:
  + Datenverarbeitung
  + Geschäftslogik
  + API-Endpunkte

### 4.3 Datenbank

* **Beschreibung**: Speicherung aller Kundendaten und Scoring-Daten.
* **Technologie**: MySQL
* **Tabellen**:
  + Kunden
  + Scoring\_Daten
  + Fehlerlog

### 4.4 Validierungsmodul

* **Beschreibung**: Modul zur Validierung der hochgeladenen Scoring-Daten.
* **Technologie**: Node.js Modul
* **Funktionalität**:
  + Überprüfung des Dateiformats (Excel)
  + Validierung der Daten auf Konsistenz und Vollständigkeit

### 4.5 Fehlerlog

* **Beschreibung**: Protokollierung von Fehlern, die während des Upload- und Validierungsprozesses auftreten.
* **Technologie**: MySQL
* **Funktionalität**:
  + Speicherung von Fehlern mit Zeitstempel und Fehlerbeschreibung
  + Zugriff auf Fehlerberichte für Administratoren

## 5. Datenmodell

### 5.1 Tabellenstruktur

#### Tabelle: Kunden

| **Feldname** | **Datentyp** | **Beschreibung** |
| --- | --- | --- |
| Kunden\_ID | INT | Eindeutige Identifikation des Kunden |
| Name | VARCHAR(255) | Name des Kunden |
| Adresse | VARCHAR(255) | Adresse des Kunden |
| Scoring\_Wert | DECIMAL(5,2) | Scoring-Wert des Kunden |

#### Tabelle: Scoring\_Daten

| **Feldname** | **Datentyp** | **Beschreibung** |
| --- | --- | --- |
| Kunden\_ID | INT | Eindeutige Identifikation des Kunden |
| Scoring\_Wert | DECIMAL(5,2) | Scoring-Wert des Kunden |
| Upload\_Datum | DATE | Datum des Uploads der Scoring-Daten |

#### Tabelle: Fehlerlog

| **Feldname** | **Datentyp** | **Beschreibung** |
| --- | --- | --- |
| Fehler\_ID | INT | Eindeutige Identifikation des Fehlers |
| Kunden\_ID | INT | Eindeutige Identifikation des Kunden |
| Fehler\_Typ | VARCHAR(255) | Beschreibung des Fehlertyps |
| Datum | DATE | Datum des Fehlers |

## 6. Schnittstellen

### 6.1 API-Endpunkte

#### POST /upload

* **Beschreibung**: Hochladen einer Excel-Datei mit Scoring-Daten.
* **Header**: Authorization: Bearer <token>
* **Body**: Multipart-Form-Data mit der Datei.

#### GET /customers/:id/score

* **Beschreibung**: Abrufen des Scoring-Werts eines bestimmten Kunden.
* **Header**: Authorization: Bearer <token>

#### PUT /customers/:id/score

* **Beschreibung**: Manuelles Aktualisieren des Scoring-Werts eines bestimmten Kunden.
* **Header**: Authorization: Bearer <token>
* **Body**: JSON mit dem neuen Scoring-Wert.

### 6.2 Datenvalidierung

Die Datenvalidierung erfolgt durch das Validierungsmodul, das sicherstellt, dass alle erforderlichen Felder vorhanden sind und die Daten konsistent sind.

## 7. Sicherheitsmaßnahmen

### 7.1 Authentifizierung

* **Technologie**: JWT (JSON Web Tokens)
* **Beschreibung**: Jeder API-Aufruf muss ein gültiges Token im Authorization-Header mitführen.

### 7.2 Datenverschlüsselung

* **Übertragung**: Alle Datenübertragungen erfolgen über HTTPS.
* **Speicherung**: Scoring-Daten werden verschlüsselt in der Datenbank gespeichert.

### 7.3 Zugriffskontrolle

* **Beschreibung**: Rollenbasierte Zugriffskontrolle stellt sicher, dass nur berechtigte Benutzer Zugriff auf bestimmte Funktionen haben.

## 8. Fehlerbehandlung und Protokollierung

### 8.1 Fehlerbehandlung

* **Beschreibung**: Fehlerhafte Daten oder Upload-Fehler werden im Fehlerlog protokolliert.
* **API-Fehlerbehandlung**: Standardisierte Fehlermeldungen werden zurückgegeben.

### 8.2 Protokollierung

* **Technologie**: MySQL
* **Beschreibung**: Alle Fehler werden mit Zeitstempel und Fehlerbeschreibung im Fehlerlog gespeichert.

## 9. Wartung und Support

### 9.1 Wartung

* **Datenbankwartung**: Regelmäßige Backups und Überprüfung der Datenbankintegrität.
* **Systemwartung**: Regelmäßige Überprüfung der Serverleistung und -sicherheit.

### 9.2 Support

Für Fragen und Unterstützung kontaktieren Sie bitte das Entwicklungsteam:

* **Max Mustermann** - Projektleiter - max.mustermann@example.com
* **Maria Musterfrau** - IT-Systemverantwortliche - maria.musterfrau@example.com

## 10. Anhang

### 10.1 Glossar

* **JWT**: JSON Web Token, ein Standard für die sichere Übertragung von Informationen zwischen Parteien.
* **HTTPS**: Hypertext Transfer Protocol Secure, ein Protokoll zur sicheren Kommunikation über ein Computernetzwerk.
* **ORM**: Object-Relational Mapping, eine Programmiertechnik zur Umwandlung von Daten zwischen inkompatiblen Typensystemen in objektorientierten Programmiersprachen.