FT (09): Kundenverwaltung

Ziel / Zweck

Dieses Feature stellt die Verwaltung von Kundenstammdaten bereit, damit Termine nicht mehr mit frei erfassten Kundendaten arbeiten müssen. Termine referenzieren künftig ein Projekt und über dieses einen Kunden und übernehmen Adresse sowie Kontaktdaten daraus, um Konsistenz, Wiederverwendbarkeit und saubere Historien sicherzustellen. Einem Kunden können Notizen zugeordnet werden.

Fachliche Beschreibung

Die Kundenverwaltung ermöglicht das Anlegen, Bearbeiten und Anzeigen von Kunden. Pro Kunde werden Stammdaten gespeichert, insbesondere **Name/Firma**, **Kundennummer**, **Adresse** und **Telefonnummer**.
Ein Kunde kann beliebig viele Projekte und damit indirekt beliebig viele Termine besitzen. In der Kundendetailansicht wird eine **Projektliste** angezeigt, die alle dem Kunden zugeordneten Projekte umfasst (z. B. Aufbau, Service, Nachbesserung).
Kunden dürfen nicht physisch gelöscht werden, sobald sie in Projekten verwendet werden. Stattdessen können sie **deaktiviert/archiviert** werden. Archivierte Kunden bleiben in bestehenden Projekten und in der Historie sichtbar, stehen jedoch **nicht mehr für neue Projekte** zur Auswahl.
Kunden haben eine **Notizenliste** (0..n). Notizen werden in der Kundendetailansicht als vertikale Kärtchenliste dargestellt und über einen Richtext-Editor verwaltet. Die Verwaltungslogik für Notizen ist in FT (13): Notizverwaltung definiert. Notizen sind **kundenbezogen** und **projektunabhängig**.

Regeln & Randbedingungen

- Kundendaten (Name, Kundennummer, Adresse, Telefon) werden **zentral** am Kunden gepflegt.
- Kunden dürfen **nicht gelöscht** werden, wenn sie in Projekten verwendet werden.
- Jeder Kunde besitzt ein **is_active** Flag (Standardwert: TRUE).

- Deaktivierte/archivierte Kunden (is_active = FALSE):
  - bleiben in bestehenden Projekten sichtbar,
  - erscheinen in der Historie,
  - dürfen **nicht** mehr für neue Projekte ausgewählt werden.
- Pflichtfelder:
  - Kundenname bzw. Firma,
  - Telefonnummer,
  - Kundennummer (aus WAWI).
- Die Adresse ist Pflicht, sofern sie für Planung oder Druck benötigt wird.
- Notizen sind optional und werden über die Relationstabelle `customer_note` mit dem Kunden verknüpft.
- Ein Kunde kann 0..n Notizen haben.
- Notizen werden gemäß FT (13): Notizverwaltung verwaltet.
- Das Löschen eines Kunden löscht auch alle zugehörigen Notizen (CASCADE über customer_note).

▼ Datenmodell

Tabelle: customer

Feld	Typ	Beschreibung
id	BIGSERIAL PRIMARY KEY	Eindeutige ID
customer_number	TEXT NOT NULL UNIQUE	Kundennummer aus WAWI
name	TEXT NOT NULL	Kundenname / Firma
phone	TEXT NOT NULL	Telefonnummer
address_line1	TEXT NULL	Adresszeile 1
address_line2	TEXT NULL	Adresszeile 2
postal_code	TEXT NULL	Postleitzahl
city	TEXT NULL	Stadt
is_active	BOOLEAN NOT NULL DEFAULT TRUE	Aktiv-Status

Feld	Typ	Beschreibung
created_at	TIMESTAMPTZ NOT NULL DEFAULT NOW()	Erstellungszeitpunkt
updated_at	TIMESTAMPTZ NOT NULL DEFAULT NOW()	Letzter Änderungszeitpunkt

Relationstabelle: customer_note

Die Zuordnung von Notizen zu Kunden erfolgt über eine n:m-Beziehung.

Feld	Typ	Beschreibung
customer_id	BIGINT NOT NULL	Fremdschlüssel zu customer
note_id	BIGINT NOT NULL	Fremdschlüssel zu note
PRIMARY KEY	(customer_id, note_id)	Composite Key

Constraints:

- Foreign Key zu customer(id) ON DELETE CASCADE
- Foreign Key zu note(id) ON DELETE CASCADE

Hinweise:

- Die Tabelle `note` ist in FT (13): Notizverwaltung definiert.
- Die Tabelle `project` referenziert customer über customer_id (siehe FT (02): Projektverwaltung).

Aufgaben/Notizen/Fragen/Probleme

MuGPlan

Aa Name	# Feature	Quelle	Status	zugewiesen
Kundenverwaltung um Notizen ergänzen	9		Nicht begonnen	

Use Cases

▼ UC: Kunde anlegen

Akteur

RO (01): Disponent (z. Zt. Angela, vertretende Mitarbeiter)

Ziel

Einen neuen Kunden mit Adresse und Kontaktdaten im System anlegen.

Vorbedingungen

- Der Disponent ist angemeldet und berechtigt.

Ablauf

1. Der Disponent startet „Kunde anlegen“.
2. Das System zeigt ein Eingabeformular für Kundendaten an.
3. Der Disponent erfasst Name/Firma, Adresse und Telefonnummer und bestätigt.
4. Das System validiert Pflichtfelder und legt den Kunden an.
5. Das System zeigt die Kundendetailansicht an.

Alternativabläufe

- Pflichtfelder fehlen: Das System speichert nicht und fordert zur Korrektur auf.
- Dublettenhinweis: Das System warnt bei ähnlichen Namen/Adressen und lässt das Speichern nach Bestätigung zu oder verweigert es gemäß Regel.

Ergebnis

Ein neuer Kunde ist gespeichert und kann bei der Terminvergabe ausgewählt werden.

▼ UC: Kunde bearbeiten

Akteur

RO (01): Disponent (z. Zt. Angela, vertretende Mitarbeiter)

Ziel

Kundendaten (Adresse, Telefon, Kundennummer) aktualisieren.

Vorbedingungen

- Ein Kunde existiert.
- Der Disponent ist berechtigt.

Ablauf

1. Der Disponent öffnet einen Kunden.
2. Das System zeigt die Kundendaten, Projektliste und Notizenliste an.
3. Der Disponent startet „Bearbeiten", ändert Daten und bestätigt.
4. Das System validiert und speichert die Änderungen.
5. Das System zeigt die aktualisierte Kundendetailansicht.

Alternativabläufe

- Pflichtfelder ungültig: Das System speichert nicht und fordert zur Korrektur auf.
- Abbruch: Änderungen werden verworfen.

Ergebnis

Die Kundendaten sind aktualisiert; die Verknüpfungen zu Projekten bleiben erhalten.

▼ UC: Kunde anzeigen (inkl. Terminliste)

Akteur

RO (01): Disponent (z. Zt. Angela, vertretende Mitarbeiter)

RO (02): Leser (Monteur, berechtigte Mitarbeiter)

Ziel

Kundendaten einsehen und die zugehörigen Projekte überblicken.

Vorbedingungen

- Der Kunde existiert.
- Der Nutzer besitzt Leserechte.

Ablauf

1. Der Nutzer wählt einen Kunden aus der Kundenliste.
2. Das System zeigt Stammdaten, eine Projektliste und eine Notizliste.
3. Der Nutzer kann ein Projekt aus der Liste öffnen (Anzeigen oder Bearbeiten gemäß Rolle).

Alternativabläufe

- Keine Projekte vorhanden: Das System zeigt eine leere Projektliste an.
- Keine Notizen vorhanden: Das System zeigt eine leere Notizliste an.

Ergebnis

Kundendaten, Projektübersicht und Notizen sind sichtbar.

▼ UC: Kunde deaktivieren / archivieren

Akteur

RO (03): Admin

Ziel

Einen Kunden aus dem aktiven Bestand entfernen, ohne die Historie zu verlieren.

Vorbedingungen

- Ein Kunde existiert und ist aktiv (is_active = TRUE).
- Der Nutzer ist berechtigt.

Ablauf

1. Der Nutzer öffnet den Kunden.
2. Der Nutzer wählt „Deaktivieren/Archivieren".
3. Das System zeigt eine Bestätigung an.
4. Der Nutzer bestätigt.
5. Das System setzt is_active auf FALSE.
6. Das System speichert die Änderung.

Alternativabläufe

- Abbruch: Keine Änderung.

Ergebnis

Der Kunde ist deaktiviert (is_active = FALSE), bleibt aber historisch und in Projekten sichtbar.

Alle Projekte, Notizen und zugehörigen Daten bleiben vollständig erhalten.

▼ UC: Kundennotizen verwalten

Akteur

RO (01): Disponent (z. Zt. Angela, vertretende Mitarbeiter)

Ziel

Kundenbezogene Notizen hinzufügen, bearbeiten, anzeigen oder löschen.

Vorbedingungen

Kunde existiert.

Ablauf

1. Disponent öffnet die Kundendetails.
2. System zeigt die Notizenliste als vertikale Kärtchenliste an.
3. Disponent führt gewünschte Notizaktionen aus (Hinzufügen, Bearbeiten, Löschen).
4. Detaillierte Abläufe siehe FT (13): Notizverwaltung.

Ergebnis

Notizen sind dem Kunden zugeordnet und in der Kundendetailansicht sichtbar.

Hinweis

Die fachliche Logik für Notizoperationen ist in FT (13): Notizverwaltung definiert.

▼ UC: Kunde reaktivieren

Akteur

RO (03): Admin

Ziel

Einen deaktivierten Kunden wieder in die aktive Nutzung aufnehmen.

Vorbedingungen

- Ein Kunde existiert und ist deaktiviert (is_active = FALSE).
- Der Nutzer ist berechtigt.

Ablauf

1. Der Nutzer öffnet einen deaktivierten Kunden (z.B. über Suche oder historische Ansicht).
2. Der Nutzer wählt „Kunde reaktivieren".
3. Das System setzt is_active auf TRUE.
4. Das System speichert die Änderung.

Ergebnis

Der Kunde ist wieder aktiv und steht für neue Projekte zur Verfügung.

Zusammenfassung (fachlich klar)

- FT (09) stellt **zentrale Kundenstammdaten** bereit.
- Projekte (und damit indirekt Termine) referenzieren Kunden, statt Kundendaten zu duplizieren.
- Historie bleibt erhalten durch **Deaktivieren/Archivieren** (is_active = FALSE) statt Löschen.
- Notizen werden über die Relationstabelle `customer_note` verknüpft (0..n).
- Notizlogik ist in FT (13): Notizverwaltung definiert.
- Das Löschen eines Kunden löscht auch dessen Notizen (CASCADE).
- Das Feature ist klar abgegrenzt und unterstützt Planung, Anzeige und Druck.