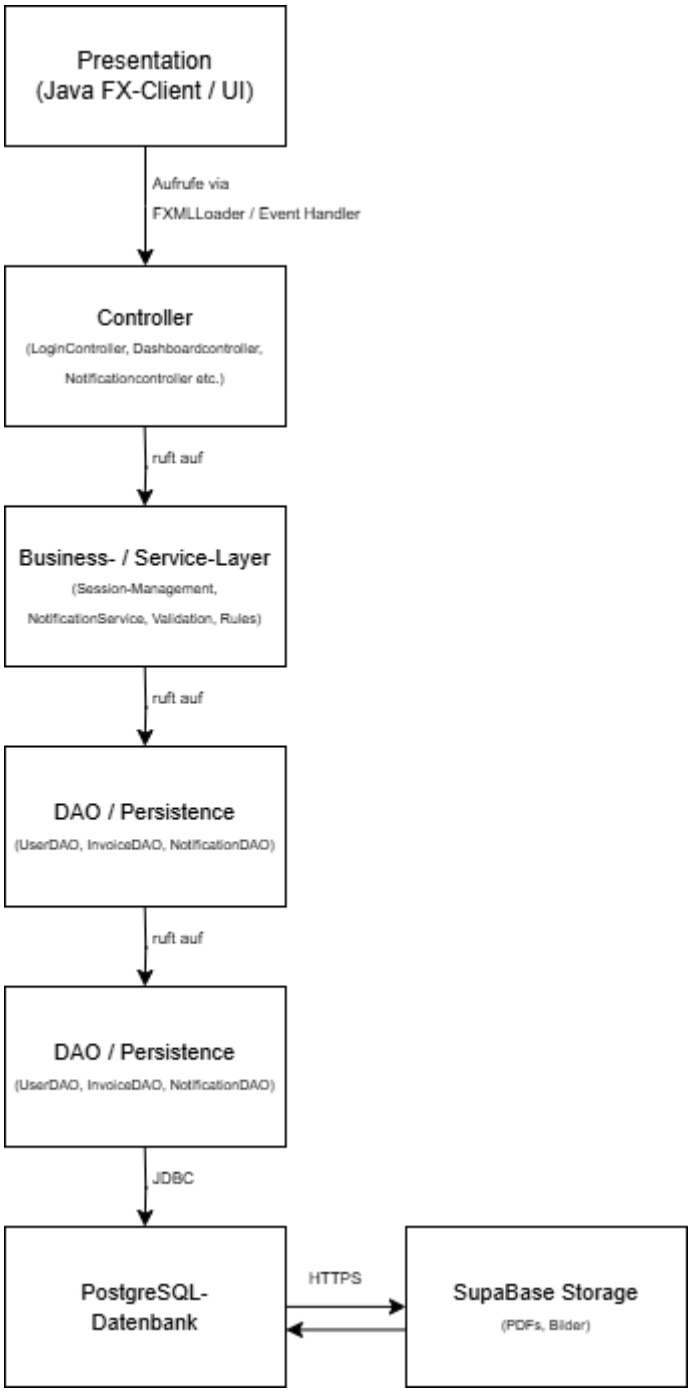
Systemdokumentation Lunchify

Team 3

Stand: Juni 2025
Version: 1.2

Contents

1 Architekturübersicht



Lunchify ist eine klassische **3-Schichten-Architektur**:

- **Präsentation**
    - JavaFX-Client (Module javafx.controls, javafx.fxml)
    - FXML-Views, Controller-Klassen, CSS-Styles
- **Geschäftslogik**
    - Controller- und DAO-Klassen (z. B. InvoiceDAO, UserDAO, NotificationDAO)

- Session-Management (Session-Singleton)
- Passwort-Hashing mit BCrypt

- **Datenhaltung**
  - Supabase PostgreSQL + Storage
  - JDBC-Kommunikation für DB, HTTPS für Storage
  - RLS-Policies auf Tabellenebene zur Zugriffskontrolle

2 Datenbankmodell

2.1 Tabelle benutzer

Spalte	Typ	Beschreibung
id	bigint	PK: Benutzer-ID
name	text	Anzeigename
email	text	UQ: Login-E-Mail
rolle	text	Enum: admin / user
passwort	text	BCrypt-Hash
must_change_password	boolean	Flag: Passwortänderung erforderlich

2.2 Tabelle rechnung

Spalte	Typ	Beschreibung
id	bigint	PK: Rechnungs-ID
user_id	bigint	FK → benutzer(id): Zugehöriger User
file_url	text	Pfad im Supabase-Bucket
type	text	Enum: RESTAURANT / SUPERMARKET
invoice_amount	numeric	Ursprünglicher Betrag (€)
reimbursement_amount	numeric	Berechneter Erstattungsbetrag (€)
status	text	**Enum**: SUBMITTED, REJECTED, EDITED
upload_date	date	Datum des Uploads

2.3 Tabelle benutzer_rechnung_starred

Spalte	Typ	Beschreibung

id	bigint	PK
benutzer_id	bigint	FK → benutzer(id): wer gefavorit hat
rechnung_id	bigint	FK → rechnung(id): welche Rechnung
starred_date	timestamp	Zeitpunkt des Favorisierens

2.4 Tabelle notification

Spalte	Typ	Beschreibung
id	bigint	PK
user_id	bigint	FK → benutzer(id): Empfänger
message	text	Benachrichtigungstext (z. B. “Rechnung editiert”)
timestamp	timestamp	Erstellungs-/Versandzeit
erstatteter_betrag	numeric	Optional: Betrag der bearbeiteten Rechnung

2.5 Tabelle notification_preferences

Spalte	Typ	Beschreibung
user_id	bigint	FK → benutzer(id)
type	text	Enum: INVOICE_EDITED, INVOICE_REJECTED
enabled	boolean	Ein/Aus-Flag

2.6 Tabelle refund_config

Spalte	Typ	Beschreibung
category	text	PK: RESTAURANT / SUPERMARKET
amount	numeric	Max. Rückerstattung in €

3 Supabase Storage

- **Bucket**: rechnung
- **Dateinamensschema**:

  `<userId>_<timestamp>_<originalFilename>.<ext>`
- **Öffentliche URL**:

  https://<PROJECT>.supabase.co/storage/v1/object/public/rechnung/<trimmed-filename>
- **RLS**:

- Jeder User darf nur seine eigenen Dateien hochladen
- Öffentliche Leserechte (nur im Lesemodus) für Images

4 Sicherheitskonzepte

4.1 Passwort-Hashing

- BCrypt (org.mindrot.jbcrypt.BCrypt)
- Default-Passwort default123 → Flag must_change_password = TRUE

4.2 Datenbank-RLS

- **rechnung**:

```
ALTER TABLE rechnung ENABLE ROW LEVEL SECURITY;
CREATE POLICY select_own_invoices
 ON rechnung FOR SELECT
 USING (user_id = auth.uid());
```

- **benutzer_rechnung_starred**: analog

4.3 DB-Verbindung

- SSL-verbindung via JDBC-URL-Parameter.

4.4 Benutzerrollen

- Postgres-Enum benutzer_rolle (admin, user).

5 Applikationsschichten & Komponenten

5.1 Präsentation (GUI)

- **FXML-Views**
  - UserDashboardView.fxml
    - Linke Sidebar mit Logo (skaliert auf 80 px), Navigationstasten, Logout.
    - Oben-rechts: Notification-Button (orange, rund), “ADD NEW +”-Button, Username.
    - Zwei TilePane-Bereiche: Starred / Recently Viewed.
  - SettingsView.fxml
    - Sektionen: Account Settings (Change Password, Update Profile, Notification Preferences), App Settings (Language).
  - NotificationPreferencesView.fxml
    - Modal mit CheckBox-Toggles für jede Notification-Art.

- NotificationsView.fxml
    - Modal mit ListView der letzten Notifications (Datum + Text).
- Weitere: LoginView.fxml, AdminDashboardView.fxml, RequestHistoryView.fxml, AddInvoiceView.fxml, u. a.

- **Controller-Klassen**
    - UserDashboardController
        - Füllt TilePanes über InvoiceDAO.
        - Öffnet modale Dialoge für Upload, Settings, Notifications.
    - SettingsController
        - Öffnet Change-Password, Update-Profile, Notification-Preferences Modals.
    - NotificationPreferencesController
        - Lädt/speichert Präferenzen via UserDAO.updateNotificationPref(...).
        - Feedback durch Fade-Transition des Labels.
    - NotificationsController
        - Initialisiert ListView mit NotificationDAO.getNotificationsForUser(...).

5.2 Geschäftslogik

- **DAO-Klassen**
    - InvoiceDAO
        - CRUD-Operationen (findAllInvoices, getAllInvoicesByUser, toggleStar, addInvoice, updateInvoice, deleteInvoice, updateInvoiceStatus, findInvoicesByMonth).
        - Status-Logik: String-Enums in DB, Upper-/Lowercase Konvertierungen
    - UserDAO
        - validateLogin, emailExists, addUser, updatePasswordAndClearFlag, getAllUsers, findEmailByBenutzerId
        - Methoden getNotificationPref, updateNotificationPref
    - NotificationDAO
        - getNotificationsForUser(int userId):

```
SELECT id, message, timestamp, erstatteter_betrag
  FROM notification
 WHERE user_id = ?
 ORDER BY timestamp DESC;
```

- **Services**
    - Session-Management: Session.setCurrentUser(...)
    - Business-Rules: Erstattungsbetrag = min(invoice_amount, refund_config.amount)

5.3 Datenhaltung

- **Kommunikation**
    - JDBC für alle DAOs.
    - HTTPS-Client für Datei-Upload/Download (Supabase) drehs.
- **Konfiguration**
    - DB-URL, Credentials in application.properties oder Umgebungsvariablen.

6 Notification System

6.1 Auslösung

- Bei Statuswechsel einer Rechnung (REJECTED oder nach Admin-Edit → EDITED)
- DAO-Methode updateInvoiceStatus(...) bzw. updateInvoice(...) erzeugt Eintrag in notification.

1.1 Persistenz

```
INSERT INTO notification(user_id, message, timestamp, erstatteter_betrag)
VALUES (?, ?, NOW(), ?);
```

6.2 Anzeige

- Glocken Button im Dashboard
- Modal (NotificationsView) zeigt ListView mit Datum + Text
- “Close” schließt den Dialog

7 Notification Preferences

- **UI**:
    - Im Settings-Modal unter “Notification Preferences”
    - Drei Toggle-Switches (jetzt: INVOICE_EDITED, INVOICE_REJECTED)
- **Tabelle**: notification_preferences (s. Abschnitt 2.5)
- **DAO**:
    - UserDAO.getNotificationPref(userId, type) → Boolean
    - UserDAO.updateNotificationPref(userId, type, enabled) → Boolean
- **Controller**:
    - Lädt Werte in initialize()

- Speichert bei jedem Toggle und zeigt kurzes Feedback

8 Tests & Qualitätssicherung

- **Unit Tests (JUnit 5)**
  - InvoiceDAO CRUD gegen H2 In-Memory
  - Erstattungslogik (min(invoiceAmount, cap))
  - UserDAO.validateLogin (Mocked JDBC)
- **Integrationstests**
  - HTTP-Upload/Download mit lokalem Supabase-Emulator
- **UI-Smoke Tests**
  - TestFX: Login → Dashboard → Navigation → Modals öffnen
- **Code Coverage**: Ziel ≥ 80 %
- **Linting & Analyse**
  - SonarQube Reports, Checkstyle, PMD

9 Build & Deployment

- **Build Tool**: Maven
- **JDK**: Java 17
- **Abhängigkeiten**:
  - javafx-controls, javafx-fxml v21
  - postgresql JDBC-Driver
  - jbcrypt für Hashing
  - controlsfx
  - junit-jupiter für Tests
- **Deployment**
  - mvn clean package
  - Setze Umgebungsvariablen: DB_URL, DB_USER, DB_PASS, SUPABASE_URL, SUPABASE_KEY
  - Starte via java -jar target/Lunchify.jar

10 Prozesse & Abläufe

10.1 User-Registration (Admin)

- UserDAO.addUser(...) → Default-Passwort + Flag

10.2 Erstlogin

- LoginController prüft must_change_password
- Modaler Dialog ChangePasswordView
- UserDAO.updatePasswordAndClearFlag(...)

10.3 Invoice Upload (User)

- AddInvoiceView + FileChooser → Supabase ↑
- InvoiceDAO.addInvoice(...) → Status = SUBMITTED

10.4 Favorisieren

- Klick auf Karte → InvoiceDAO.toggleStar(...)

10.5 Notification-Flow

- Statuswechsel auf REJECTED oder EDITED → NotificationDAO.insert(...)

10.6 Notification Preferences

- Settings → Toggle → Persist in notification_preferences

10.7 Invoice Edit (Admin)

- RequestHistory → Edit-Modal → InvoiceDAO.updateInvoice(...) → Status = EDITED
- Notification für betroffenen User

11 Appendix: Enums

```
-- Benutzer-Rolle
CREATE TYPE benutzer_rolle AS ENUM ('admin', 'user');
-- Rechnung-Kategorie
CREATE TYPE rechnung_kategorie AS ENUM ('RESTAURANT', 'SUPERMARKET');
-- Rechnung-Status
CREATE TYPE rechnung_status AS ENUM (
  'SUBMITTED',
  'REJECTED',
  'EDITED'
);
-- Notification-Typen (für Preferences)
CREATE TYPE notification_type AS ENUM (
  'INVOICE_EDITED',
  'INVOICE_REJECTED'
);
```