Pflichtenheft – Fakturierungssystem

SE1 Team 2 – Hochschule Mannheim

Christopher Lampert

11.06.2026

Inhaltsverzeichnis

Pflichtenheft – Fakturierungssystem

Modul: Software Engineering 1 **Team:** SE1 Team 2 – Hochschule Mannheim **Version:** 1.0 **Stand:** 11.06.2026 **Autor:** Christopher Lampert **Bezug:** Lastenheft v1.0 (15.05.2026), Project Charter v1.1 (15.05.2026) – Fakturierungssystem

Dieses Pflichtenheft (System Requirements Specification, SRS) ist das Antwortdokument des Auftragnehmers (SE1 Team 2) auf das Lastenheft v1.0. Es beantwortet die Frage, **wie** die im Lastenheft geforderten Leistungen umgesetzt werden. Im V-Modell steht es auf der Ebene der Systemanforderungen und bildet den direkten Input für den Komponenten- und Integrationstest.

Freigabeübersicht

Funktion	Name	Rolle / Gruppe	Datum
Ersteller	Christopher Lampert	Autor, Gruppe H (Kundenverwaltung)	11.06.2026
Prüfer	Prof. Dr. Gerd Marmitt	Auftraggeber / Gutachter	*(offen)*
Freigebender	SE1 Team 2 (Gruppenleiter)	Projektleitung / Freigabe	*(offen)*

Änderungshistorie

Version	Datum	Autor	Änderung
1.0	11.06.2026	C. Lampert	Initiale Erstellung des Pflichtenhefts auf Basis Lastenheft v1.0 und Project Charter v1.1: funktionale Anforderungen F-PV/F-KV/F-DP/F-GUI mit Satzschablone; nicht-funktionale Anforderungen mit messbaren Kriterien; Datenobjekte mit Java-Datentypen; Schnittstellen IF-01 bis IF-03; Geschäftsregeln GR-01 bis GR-06 mit Java-Umsetzungshinweisen; UML-Diagramme (Use-Case, Kontext, Klassen, Komponenten, Sequenz); testbare Abnahmekriterien (Testfälle TC-*); vollständige Traceability-Matrix LH ↔ PH.
1.0	24.06.2026	C. Lampert	Redaktionelle Überarbeitung (Review-Einarbeitung) ohne Versionserhöhung: Freigabeübersicht um Rolle und Datum ergänzt; sechs UML-Diagramme als generierte PNG-Grafiken eingebunden, PlantUML-Quellen nach Anhang 9.5 verschoben; organisatorische Gruppenaufteilung aus Kap. 3.1 entfernt (gehört in den Project-Charter), Modul-Package-Zuordnung beibehalten; Abweichungshinweis zu BA-PV/KV-05…08 in Kap. 4 und Kap. 9.4 bereinigt; Begründung zur BigDecimal-Wahl in Kap. 6.1 ergänzt.

---

1. Einleitung

1.1 Zweck des Dokuments

Das Pflichtenheft beschreibt die technische Umsetzung der im Lastenheft v1.0 spezifizierten fachlichen Anforderungen an das **Fakturierungssystem**. Während das Lastenheft aus Auftraggebersicht festlegt, **was** das System

leisten soll, legt dieses Pflichtenheft aus Auftragnehmersicht fest, **wie** die Realisierung erfolgt (Architektur, Technologie, Datentypen, Schnittstellen, Detaildesign).

Aspekt	Lastenheft v1.0	Pflichtenheft (dieses Dokument)
Sichtweise	Auftraggeber (Fachseite)	Auftragnehmer (Entwicklungsteam)
Leitfrage	Was soll das System leisten?	Wie wird es umgesetzt?
Lösungsneutral	ja	nein
V-Modell-Rolle	Eingabe für Abnahmetests	Eingabe für Komponenten- und Integrationstest

1.2 Geltungsbereich

Das Pflichtenheft umfasst die technische Spezifikation aller vier Pflichtmodule:

- **Produktverwaltung** (Gruppe G)
- **Kundenverwaltung** (Gruppe H)
- **Dokumentenprozess** (Gruppe F)
- **Programmoberfläche / GUI** (Gruppe E)

sowie deren Zusammenspiel über die in Kapitel 6.2 definierten Schnittstellen.

1.3 Referenzen

- [**1**] `Lastenheft_v1_0.md` – Lastenheft Fakturierungssystem, Version 1.0, 15.05.2026
- [**2**] `project-charter_v1_1.md` – Project Charter Fakturierungssystem, Version 1.1, 15.05.2026
- [**3**] `week7slides.pdf` – Vorlesung Software Engineering 1, Woche 7: Pflichtenheft, Anforderungen und Traceability
- [**4**] § 14 UStG – Umsatzsteuergesetz, Pflichtangaben in Rechnungen
- [**5**] DSGVO – Verordnung (EU) 2016/679

1.4 Abgrenzung (Nicht-Bestandteil)

Übernommen aus den Nicht-Zielen des Lastenhefts (Kap. 1.3). Folgende Funktionen sind **nicht** Bestandteil der Realisierung:

- Mobile Anwendung sowie Cloud- bzw. Multi-Tenant-Betrieb
- Gleichzeitiger Mehrbenutzerbetrieb über Netzwerk
- Vollwertiges Buchhaltungssystem (Bilanz, Kontenrahmen, FiBu-Export)
- Strukturierte E-Rechnung (XRechnung, ZUGFeRD) – die einfache PDF-Ausgabe der Belege ist hingegen Bestandteil (F-DP-07, F-GUI-05)
- Anbindung an externe Buchhaltungs- oder ERP-Systeme
- Mehrsprachigkeit (Realisierung ausschließlich in deutscher Sprache)
- Authentifizierung bzw. Benutzeranmeldung – der Zugriffsschutz erfolgt über das Betriebssystem-Login (vgl. NF-SEC-01)

---

2. Systemüberblick

2.1 Kurzbeschreibung

Das Fakturierungssystem wird als **lokal installierte Java-Desktop-Anwendung** für einen **Einbenutzer-Arbeitsplatz** realisiert. Es bildet den durchgängigen Geschäftsprozess von der Angebotserstellung über Auftragsbestätigung und Lieferschein bis zur Rechnung ab und erlaubt den Export jedes Belegs als PDF.

2.2 Grobe Systemfunktionen

- CRUD-Operationen für Produkte (anlegen, bearbeiten, suchen, löschen)
- CRUD-Operationen für Kunden (anlegen, bearbeiten, suchen, löschen)

Abbildung 1: Use-Case-Diagramm des Fakturierungssystems mit Akteur „Anwender".

Abbildung 1: Abbildung 1: Use-Case-Diagramm des Fakturierungssystems mit Akteur „Anwender".

- Dokumentenworkflow Angebot → Auftragsbestätigung → Lieferschein → Rechnung mit automatischen Statusübergängen
- Automatische Summen- und Steuerberechnung gemäß Geschäftsregeln
- Beleg-Export als PDF (inkl. § 14 UStG-Pflichtangaben bei Rechnungen)
- Echtzeit-Suche und tabellarische Übersichten

2.3 Technologiestack

Bereich	Festlegung	Begründung
Programmiersprache	Java (LTS, ≥ 17)	Modulvorgabe; plattformunabhängige Desktop-Ausführung; objektorientiertes Schichtenmodell
GUI-Technologie	JavaFX	Moderne, deklarative Oberflächen (FXML), saubere Trennung von View und Logik; Swing als Alternative
Datenpersistenz	lokale JSON-Datei (z. B. Jackson)	Einbenutzer-Betrieb, keine Server-DB nötig; menschenlesbar und versionierbar; hinter Repository gekapselt (austauschbar gegen SQLite, vgl. NF-ARCH-02)
Tests	JUnit 5 (Jupiter)	Standard für Java-Komponententests; GUI-unabhängige Logiktests (NF-TEST-02)
PDF-Erzeugung	PDF-Bibliothek (z. B. Apache PDFBox)	Erzeugung druckbarer Belege über die Schnittstelle IF-03
Versionierung	Git / Gitty	Vorgabe Charter v1.1; Code-Reviews je Merge (NF-VER-02)

Hinweis: Die konkrete Wahl der Persistenz- und PDF-Bibliothek ist eine Realisierungsentscheidung des Auftragnehmers. Sie wird hinter den Schnittstellen IF-02 und IF-03 gekapselt und ist damit austauschbar, ohne die Logikschicht zu verändern.

2.4 Use-Case-Diagramm

Abbildung 1 zeigt den einzigen Akteur **Anwender** und die Hauptanwendungsfälle des Systems. Die Belegerzeugung folgt der festen Dokumentenkette (GR-01): Eine Auftragsbestätigung setzt ein angenommenes Angebot voraus, ein Lieferschein eine bestätigte Auftragsbestätigung und eine Rechnung einen gelieferten Lieferschein.

---

3. Stakeholder und Kontext

3.1 Stakeholder und Modulzuordnung

Die Stakeholder-Tabelle aus Lastenheft Kap. 3.1 wird um die technische Modul-Package-Zuordnung ergänzt. Die organisatorische Zuordnung der Untergruppen zu den Modulen ist im Project-Charter geregelt.

ID	Stakeholder	Beschreibung	Interesse / Implementierungsrolle
S1	Auftraggeber	Prof. Dr. Gerd Marmitt	Lehrziele erreicht, Abnahmekriterien erfüllt; Prüfer des Dokuments
S2	Entwicklungsteam	SE1 Team 2 (Gruppen E, F, G, H)	Erfolgreiche Umsetzung; siehe Implementierungsrollen unten
S3	Endnutzer	spätere Anwender (kleines Unternehmen)	Funktionierender Fakturierungsablauf

Modul	Realisierungspaket	Bezug
Produktverwaltung	`de.hsmannheim.faktura.produkt`	Kap. 4.1
Kundenverwaltung	`de.hsmannheim.faktura.kunde`	Kap. 4.2

Abbildung 2: Systemkontextdiagramm – Abgrenzung System gegen Umgebung.

Abbildung 2: Abbildung 2: Systemkontextdiagramm – Abgrenzung System gegen Umgebung.

Modul	Realisierungspaket	Bezug
Dokumentenprozess	`de.hsmannheim.faktura.beleg`	Kap. 4.3
Programmoberfläche (GUI)	`de.hsmannheim.faktura.ui`	Kap. 4.4

3.2 Benutzerrolle

Das System kennt – wie im Lastenheft Kap. 3.2 festgelegt – ausschließlich die Rolle **Anwender**. Eine Authentifizierungs- oder Rechtekonzept-Rolle entfällt (vgl. Kap. 1.4). Der Anwender legt Produkt- und Kundenstammdaten an, erstellt Belege entlang der Dokumentenkette und exportiert Belege als PDF.

3.3 Systemkontextdiagramm

Abbildung 2 grenzt das System gegen seine Umgebung ab. Das System interagiert mit dem Anwender über die GUI, mit dem lokalen Dateisystem zur Persistenz und mit einer PDF-Bibliothek zur Erzeugung druckbarer Belege. Externe System- oder Netzanbindungen bestehen nicht (vgl. Nicht-Ziele).

---

4. Funktionale Anforderungen

Jede funktionale Anforderung folgt der Satzschablone des Pflichtenhefts:

> **F-<Kategorie>-<Nummer>:** Das System MUSS / SOLLTE / KANN es dem Anwender ermöglichen, <Ziel>, indem <konkretes Systemverhalten>.

Die Modalverben entsprechen der Priorität (Muss → MUSS, Soll → SOLLTE, Kann → KANN). Jede Anforderung führt ID, Beschreibung, Priorität, Herkunft (Lastenheft-ID) und Akzeptanzkriterium. Die Zuordnung zu Testfällen erfolgt in Kapitel 8 und in der Traceability-Matrix (Kap. 9.4).

4.1 Modul Produktverwaltung (F-PV)

F-PV-01 – Produkt anlegen (Muss)

Das System MUSS es dem Anwender ermöglichen, ein neues Produkt anzulegen, indem es eine Eingabemaske bereitstellt, die Pflichtattribute `bezeichnung` (String), `einzelpreisNetto` (BigDecimal) und `mehrwertsteuersatz` (BigDecimal) validiert, eine systemintern fortlaufende `produktId` (long) vergibt und das Produkt über den `ProduktService` persistiert sowie in der Produktübersicht anzeigt.

- **Priorität:** Muss
- **Herkunft:** BA-PV-01
- **Akzeptanzkriterium:** Ein Produkt mit vollständigen Pflichtattributen wird gespeichert und erscheint in der Übersicht. Fehlt ein Pflichtfeld (insbesondere `mehrwertsteuersatz`), wird eine Fehlermeldung angezeigt und es erfolgt keine Speicherung.

F-PV-02 – Produkt bearbeiten (Muss)

Das System MUSS es dem Anwender ermöglichen, bestehende Produktdaten zu bearbeiten, indem es das ausgewählte Produkt lädt, geänderte Werte validiert und über den `ProduktService` persistiert; bereits bestehende Belegpositionen behalten ihren zum Erstellungszeitpunkt festgehaltenen `einzelpreisNetto` (GR-04).

- **Priorität:** Muss
- **Herkunft:** BA-PV-02 (i. V. m. GR-04)
- **Akzeptanzkriterium:** Geänderte Daten sind sofort in den Produktdetails sichtbar und nach erneutem Öffnen weiterhin vorhanden. Eine Preisänderung verändert die Preise bestehender Angebote nicht.

F-PV-03 – Produkt suchen (Muss)

Das System MUSS es dem Anwender ermöglichen, nach Produkten zu suchen, indem es die Eingabe im Suchfeld gegen die Felder `bezeichnung` und `artikelnummer` auswertet und Treffer tabellarisch anzeigt.

- **Priorität:** Muss
- **Herkunft:** BA-PV-03
- **Akzeptanzkriterium:** Produkte werden über Bezeichnung oder Artikelnummer gefunden. Bei ungültiger Eingabe erscheint der Hinweis „Kein Produkt gefunden".

F-PV-04 – Produkt löschen (Muss)

Das System MUSS es dem Anwender ermöglichen, ein Produkt zu löschen, indem es nach einer Sicherheitsabfrage prüft, ob das Produkt in einem Angebot, einer Auftragsbestätigung, einem Lieferschein oder einer Rechnung referenziert ist (GR-05), und es nur bei fehlender Referenz aus Bestand, Übersicht und Suche entfernt.

- **Priorität:** Muss
- **Herkunft:** BA-PV-04 (i. V. m. GR-05)
- **Akzeptanzkriterium:** Nach Bestätigung der Sicherheitsabfrage wird ein nicht referenziertes Produkt entfernt und ist nicht mehr auffindbar. Ein referenziertes Produkt wird mit Hinweis abgewiesen.

4.2 Modul Kundenverwaltung (F-KV)

F-KV-01 – Kunde anlegen (Muss)

Das System MUSS es dem Anwender ermöglichen, einen neuen Kunden anzulegen, indem es die Pflichtattribute (`firmenname` oder `nachname`, `strasse`, `plz`, `ort`) validiert, eine fortlaufende `kundeId` (long) vergibt, eine angegebene `email` auf gültiges Format prüft und den Kunden über den `KundeService` persistiert.

- **Priorität:** Muss
- **Herkunft:** BA-KV-01
- **Akzeptanzkriterium:** Ein Kunde mit vollständigen Pflichtattributen wird gespeichert und erscheint in der Kundenliste. Fehlt ein Pflichtfeld oder ist eine angegebene E-Mail-Adresse ungültig formatiert, wird eine Fehlermeldung angezeigt und nicht gespeichert.

F-KV-02 – Kunde bearbeiten (Muss)

Das System MUSS es dem Anwender ermöglichen, bestehende Kundendaten zu bearbeiten, indem es den ausgewählten Kunden lädt, geänderte Werte validiert und über den `KundeService` persistiert.

- **Priorität:** Muss
- **Herkunft:** BA-KV-02
- **Akzeptanzkriterium:** Geänderte Daten werden gespeichert, sofort angezeigt und sind nach erneutem Öffnen des Kunden weiterhin vorhanden.

F-KV-03 – Kunde suchen (Muss)

Das System MUSS es dem Anwender ermöglichen, nach Kunden zu suchen, indem es die Eingabe im Suchfeld gegen Name (`firmenname`/`nachname`) und Kundennummer (`kundeId`) auswertet und Treffer tabellarisch anzeigt.

- **Priorität:** Muss
- **Herkunft:** BA-KV-03
- **Akzeptanzkriterium:** Kunden werden über Namen oder Kundennummer gefunden. Bei ungültiger Eingabe erscheint der Hinweis „Kein Kunde gefunden".

F-KV-04 – Kunde löschen (Muss)

Das System MUSS es dem Anwender ermöglichen, einen Kunden zu löschen, indem es nach einer Sicherheitsabfrage prüft, ob der Kunde in einem Beleg referenziert ist (GR-05) bzw. ob gesetzliche Aufbewahrungspflichten bestehen (NF-SEC-02), und ihn nur bei zulässiger Löschung aus Liste und Suche entfernt.

- **Priorität:** Muss
- **Herkunft:** BA-KV-04 (i. V. m. GR-05, NF-SEC-02)
- **Akzeptanzkriterium:** Ein nicht referenzierter Kunde ohne Aufbewahrungspflicht wird nach Bestätigung entfernt und ist nicht mehr auffindbar. Ein referenzierter Kunde wird mit Hinweis abgewiesen.

4.3 Modul Dokumentenprozess (F-DP)

F-DP-01 – Angebot erstellen (Muss)

Das System MUSS es dem Anwender ermöglichen, ein Angebot für einen Kunden zu erstellen, indem es Kundenreferenz, `datum` (LocalDate) und mindestens eine `Belegposition` aufnimmt, `gesamtBetragNetto` und `gesamtBetragBrutto` gemäß GR-06 mit BigDecimal berechnet und das Angebot mit `status = AngebotStatus.OFFEN` persistiert.

- **Priorität:** Muss
- **Herkunft:** BA-DP-01 (i. V. m. GR-06)
- **Akzeptanzkriterium:** Ein Angebot mit Kundenreferenz, Datum, mindestens einer Position und korrekt berechneter Netto- und Bruttosumme ist gespeichert.

F-DP-02 – Angebotsstatus setzen (Muss)

Das System MUSS es dem Anwender ermöglichen, ein Angebot mit Status `OFFEN` als „angenommen" oder „abgelehnt" zu markieren, indem es `status` auf `AngebotStatus.ANGENOMMEN` bzw. `AngebotStatus.ABGELEHNT` setzt und dauerhaft persistiert (Statusübergänge gemäß Kap. 6.1.7 LH).

- **Priorität:** Muss
- **Herkunft:** BA-DP-02
- **Akzeptanzkriterium:** Der gesetzte Status wird dauerhaft gespeichert und ist auch nach Neustart in der Übersicht sichtbar.

F-DP-03 – Auftragsbestätigung aus Angebot erzeugen (Muss)

Das System MUSS es dem Anwender ermöglichen, ein angenommenes Angebot in eine Auftragsbestätigung umzuwandeln, indem es alle Positionen und die Kundenreferenz übernimmt, die Angebotsreferenz speichert, die Auftragsbestätigung mit `status = AuftragsStatus.BESTAETIGT` anlegt und das Vorgängerangebot automatisch auf `AngebotStatus.UEBERFUEHRT` setzt.

- **Priorität:** Muss
- **Herkunft:** BA-DP-03 (i. V. m. GR-01)
- **Akzeptanzkriterium:** Alle Angebotsdaten sind übernommen, das Vorgängerangebot ist referenziert und sein Status auf „überführt" gesetzt.

F-DP-04 – Lieferschein aus Auftragsbestätigung erzeugen (Muss)

Das System MUSS es dem Anwender ermöglichen, aus einer Auftragsbestätigung (`status = BESTAETIGT`) einen Lieferschein zu erzeugen, indem es Referenz, `lieferdatum` (LocalDate) und alle Positionen übernimmt und den Lieferschein mit `status = LieferscheinStatus.OFFEN` persistiert.

- **Priorität:** Muss
- **Herkunft:** BA-DP-04 (i. V. m. GR-01)
- **Akzeptanzkriterium:** Ein Lieferschein mit Referenz auf die Auftragsbestätigung, Lieferdatum und vollständigen Positionen ist gespeichert.

F-DP-05 – Rechnung aus Lieferschein erzeugen (Muss)

Das System MUSS es dem Anwender ermöglichen, aus einem Lieferschein (`status = GELIEFERT`) eine Rechnung zu erzeugen, indem es eine eindeutige, fortlaufende `rechnungNr` (GR-02) vergibt, alle § 14 UStG-Pflichtangaben aufnimmt, genau einen Lieferschein referenziert (GR-01) und die Rechnung mit `status = RechnungStatus.OFFEN` persistiert.

- **Priorität:** Muss
- **Herkunft:** BA-DP-05 (i. V. m. GR-01, GR-02)
- **Akzeptanzkriterium:** Eine Rechnung mit vollständigen § 14 UStG-Pflichtangaben, eindeutiger Rechnungsnummer und Referenz auf den Lieferschein ist erzeugt.

F-DP-06 – Rechnungsdatum automatisch setzen (Muss)

Das System MUSS es dem Anwender ermöglichen, beim Anlegen einer Rechnung das Rechnungsdatum automatisch zu erhalten, indem es `rechnungsdatum` beim Erzeugen auf `LocalDate.now()` setzt und dauerhaft speichert.

- **Priorität:** Muss
- **Herkunft:** BA-DP-06
- **Akzeptanzkriterium:** Das Rechnungsdatum entspricht beim Anlegen dem aktuellen Systemdatum und bleibt persistent.

F-DP-07 – Beleg als PDF exportieren (Muss)

Das System MUSS es dem Anwender ermöglichen, Angebote, Auftragsbestätigungen, Lieferscheine und Rechnungen als PDF zu exportieren, indem der `PdfExportService` über die Schnittstelle IF-03 eine druckbare PDF-Datei mit allen Beleginhalten erzeugt und unter einem vom Anwender gewählten Pfad speichert; Rechnungs-PDFs enthalten alle § 14 UStG-Pflichtangaben.

- **Priorität:** Muss
- **Herkunft:** BA-DP-07
- **Akzeptanzkriterium:** Für jeden Belegtyp wird eine PDF-Datei erzeugt, deren Inhalt dem Beleg entspricht; die Rechnungs-PDF enthält alle § 14 UStG-Pflichtangaben.

F-DP-08 – Integrität der Dokumentenkette (Muss)

Das System MUSS sicherstellen, dass ein Folgebeleg nur erzeugt werden kann, wenn der Vorgängerbeleg im passenden Status vorliegt, indem es vor jeder Belegerzeugung die Dokumentenkette (Rechnung → Lieferschein → Auftragsbestätigung → Angebot) und den jeweiligen Vorgängerstatus prüft und unzulässige Erzeugungen ablehnt.

- **Priorität:** Muss
- **Herkunft:** GR-01
- **Akzeptanzkriterium:** Der Versuch, eine Rechnung ohne zugehörigen Lieferschein (oder einen sonstigen Folgebeleg ohne gültigen Vorgänger) anzulegen, wird abgewiesen.

F-DP-09 – Eindeutige fortlaufende Rechnungsnummer (Muss)

Das System MUSS sicherstellen, dass jede Rechnung eine eindeutige, fortlaufende und lückenlose Rechnungsnummer erhält, indem es beim Erstellen die nächste freie Nummer vergibt und vergebene Nummern nicht wiederverwendet.

- **Priorität:** Muss
- **Herkunft:** GR-02
- **Akzeptanzkriterium:** Drei aufeinanderfolgend erstellte Rechnungen besitzen fortlaufende, eindeutige Nummern.

F-DP-10 – Unveränderlichkeit festgeschriebener Rechnungen (Muss)

Das System MUSS sicherstellen, dass eine erzeugte und gespeicherte Rechnung inhaltlich nicht mehr geändert werden kann, indem es die Rechnung nach dem Festschreiben als unveränderlich (immutable) behandelt und Bearbeitungsversuche ablehnt.

- **Priorität:** Muss
- **Herkunft:** GR-03
- **Akzeptanzkriterium:** Ein Bearbeitungsversuch an einer gespeicherten Rechnung wird abgewiesen.

F-DP-11 – Preisübernahme (Preis-Snapshot) (Muss)

Das System MUSS sicherstellen, dass der zum Zeitpunkt der Positionsübernahme gültige Einzelpreis in der `Belegposition` festgehalten wird, indem es `einzelpreisNetto` als Kopie in der Position speichert, sodass spätere Produktpreisänderungen bestehende Belege nicht verändern.

- **Priorität:** Muss
- **Herkunft:** GR-04
- **Akzeptanzkriterium:** Eine Preisänderung am Produkt verändert den Preis bestehender Angebote nicht.

F-DP-12 – Summen- und Steuerberechnung (Muss)

Das System MUSS sicherstellen, dass Beträge korrekt berechnet werden, indem es `gesamtpreisNetto` je Position als `menge × einzelpreisNetto`, die Nettosumme als Summe der Positionen, den USt-Betrag je Position als `netto × mehrwertsteuersatz` und die Bruttosumme als `netto + USt` mittels `BigDecimal` (Rundung `HALF_UP`, 2 Nachkommastellen) berechnet.

- **Priorität:** Muss
- **Herkunft:** GR-06
- **Akzeptanzkriterium:** Die manuelle Nachrechnung an drei Beispielen stimmt mit der Systemausgabe überein (centgenau).

4.4 Modul Programmoberfläche / GUI (F-GUI)

F-GUI-01 – Angebotserfassung über grafische Maske (Muss)

Das System MUSS es dem Anwender ermöglichen, Angebote über eine grafische Eingabemaske zu erstellen, indem es Artikel per Dropdown auswählbar macht, die berechnete Gesamtsumme ohne spürbare Verzögerung live aktualisiert und nach dem Speichern eine Erfolgsmeldung anzeigt.

- **Priorität:** Muss
- **Herkunft:** BA-GUI-01
- **Akzeptanzkriterium:** Artikel sind per Dropdown wählbar, die Summe aktualisiert sich live und nach dem Speichern erscheint eine Erfolgsmeldung.

F-GUI-02 – Auftragsbestätigung über GUI (Muss)

Das System MUSS es dem Anwender ermöglichen, ein angenommenes Angebot über die GUI in eine Auftragsbestätigung umzuwandeln, indem es alle Daten ohne erneute Eingabe in die Auftragsbestätigungs-Maske übernimmt, diese mit dem Titel „Auftragsbestätigung" anzeigt und leere Pflichtfelder (Liefertermin) farblich markiert.

- **Priorität:** Muss
- **Herkunft:** BA-GUI-02
- **Akzeptanzkriterium:** Alle Daten werden übernommen, der Maskentitel lautet „Auftragsbestätigung" und leere Pflichtfelder sind farblich markiert.

F-GUI-03 – Rechnungslegung und Statusanzeige (Muss)

Das System MUSS es dem Anwender ermöglichen, über die GUI aus einem Lieferschein eine Rechnung zu erzeugen und deren Status einzusehen, indem es vor dem finalen Speichern eine Druckvorschau anzeigt und den Auftragsstatus in der Übersicht durch ein grünes Icon als „Abgeschlossen" kennzeichnet.

- **Priorität:** Muss
- **Herkunft:** BA-GUI-03
- **Akzeptanzkriterium:** Vor dem Speichern erscheint eine Druckvorschau; nach Abschluss ist der Status in der Übersicht durch ein grünes Icon markiert.

F-GUI-04 – Belegsuche mit Echtzeit-Filterung (Muss)

Das System MUSS es dem Anwender ermöglichen, Belege über eine Suchfunktion zu filtern, indem es die Trefferliste tabellarisch darstellt, sich bei Eingabe von Name oder Nummer sofort aktualisiert und bei einer leeren Trefferliste den Text „Keine Treffer gefunden" anzeigt.

- **Priorität:** Muss
- **Herkunft:** BA-GUI-04
- **Akzeptanzkriterium:** Die Trefferliste aktualisiert sich in Echtzeit; bei einer ungültigen Suche erscheint „Keine Treffer gefunden".

F-GUI-05 – PDF-Export aus der Belegansicht (Muss)

Das System MUSS es dem Anwender ermöglichen, den PDF-Export direkt aus jeder Belegansicht auszulösen, indem es auf der Ansicht eine Schaltfläche „Als PDF exportieren" bereitstellt, beim Klick die Funktion aus F-DP-07 ausführt und eine Erfolgsmeldung mit Speicherpfad anzeigt.

- **Priorität:** Muss
- **Herkunft:** BA-GUI-05
- **Akzeptanzkriterium:** Auf jeder Belegansicht ist die Schaltfläche sichtbar und funktionsfähig; nach dem Export erscheint eine Erfolgsmeldung mit Speicherpfad.

---

5. Nicht-funktionale Anforderungen

Die nicht-funktionalen Anforderungen werden 1:1 aus Lastenheft Kap. 5 übernommen und pflichtenheftseitig mit messbarem Kriterium und Realisierungsbezug geführt. Die IDs bleiben unverändert, um die Traceability eindeutig zu halten. Modalverben entsprechen der Priorität.

5.1 Usability (NF-USE)

ID	Anforderungstext	Prio	Herkunft (LH)	Akzeptanzkriterium
NF-USE-01	Das Anlegen eines neuen Kunden MUSS ohne vorherige Schulung in weniger als 2 Minuten möglich sein (höchstens 1 Fehlbedienung pro 10 Testnutzenden).	Muss	NF-USE-01	Usability-Test mit 5 Probanden; Erfolgsquote > 90 %, Median-Zeit < 120 s.
NF-USE-02	Mindestens 80 % der Testnutzenden MÜSSEN die Aufgabe „aus einem Angebot eine Rechnung erzeugen" im ersten Versuch ohne Hilfetext erfolgreich abschließen.	Muss	NF-USE-02	Usability-Test mit 5 Probanden; mindestens 4 von 5 erfolgreich.
NF-USE-03	Das System MUSS bei jeder validierungspflichtigen Eingabe innerhalb von 1 Sekunde eine sichtbare Rückmeldung (Erfolg oder Fehler) anzeigen.	Muss	NF-USE-03	Manueller Test mit Stoppuhr; Rückmeldezeit < 1 s.

5.2 Performance (NF-PERF)

ID	Anforderungstext	Prio	Herkunft (LH)	Akzeptanzkriterium
NF-PERF-01	Das System MUSS jede Benutzeraktion bei bis zu 1.000 Produkten und 1.000 Kunden innerhalb von 1 Sekunde mit sichtbarer Rückmeldung beantworten.	Muss	NF-PERF-01	Lasttest mit 1.000 Datensätzen je Entität; Antwortzeit < 1 s.
NF-PERF-02	Das System MUSS die Übersichtsliste von Produkten oder Kunden bei bis zu 1.000 Datensätzen innerhalb von 2 Sekunden vollständig anzeigen.	Muss	NF-PERF-02	Lasttest mit 1.000 Einträgen; Anzeigedauer < 2 s.
NF-PERF-03	Das System SOLLTE den Anwendungsstart bis zur bedienbaren Hauptansicht in höchstens 5 Sekunden abschließen (Referenzhardware).	Soll	NF-PERF-03	Kaltstart auf Referenzhardware < 5 s.

5.3 Wartbarkeit (NF-MAINT)

ID	Anforderungstext	Prio	Herkunft (LH)	Akzeptanzkriterium
NF-MAINT-01	Der Quellcode MUSS modular nach den vier Modulen (Produkt-, Kundenverwaltung, Dokumentenprozess, GUI) in eigenen Paketen getrennt sein.	Muss	NF-MAINT-01	Statische Prüfung der Repository- und Paketstruktur.
NF-MAINT-02	Mindestens 80 % der öffentlichen Klassen und Methoden SOLLTEN über kurze Dokumentationskommentare (Zweck, Parameter, Rückgabe) verfügen.	Soll	NF-MAINT-02	Automatisierte Auswertung (JavaDoc-Report) zeigt > 80 %; Stichprobe.
NF-MAINT-03	Das System MUSS einer dokumentierten, dreischichtigen Architektur (Präsentation, Logik, Datenhaltung) folgen.	Muss	NF-MAINT-03	Review des Architekturkapitels (Kap. 7); nachweisbare Schichtentrennung.

5.4 Testbarkeit (NF-TEST)

ID	Anforderungstext	Prio	Herkunft (LH)	Akzeptanzkriterium
NF-TEST-01	Jede Anforderung mit Priorität „Muss" MUSS mindestens einem Testfall in der Traceability-Matrix (Kap. 9.4) zugeordnet sein.	Muss	NF-TEST-01	Vollständigkeitsprüfung der Traceability-Matrix vor Abnahme.
NF-TEST-02	Die Logikschicht MUSS so gestaltet sein, dass sie ohne die GUI durch automatisierte Komponententests (JUnit 5) aufgerufen werden kann.	Muss	NF-TEST-02	Mindestens ein lauffähiger Unit-Test pro Modul ohne GUI-Abhängigkeit.
NF-TEST-03	Mindestens 60 % der Geschäftslogik-Methoden SOLLTEN durch automatisierte Tests abgedeckt sein (Line Coverage).	Soll	NF-TEST-03	Coverage-Report > 60 %.

5.5 Versionierung (NF-VER)

ID	Anforderungstext	Prio	Herkunft (LH)	Akzeptanzkriterium
NF-VER-01	Der gesamte Quellcode MUSS in Gitty unter den im Charter v1.1 genannten Repositories versioniert werden.	Muss	NF-VER-01	Sichtprüfung der Repositories.
NF-VER-02	Jeder Merge in den Hauptbranch MUSS durch mindestens ein Code-Review eines anderen Teammitglieds freigegeben werden.	Muss	NF-VER-02	Review-Historie zeigt für jeden Merge mindestens einen Reviewer.

5.6 Architektur und Datenhaltung (NF-ARCH)

ID	Anforderungstext	Prio	Herkunft (LH)	Akzeptanzkriterium
NF-ARCH-01	Das System MUSS alle Datensätze (Produkte, Kunden, Dokumente) persistent so speichern, dass sie nach einem Neustart vollständig wiederherstellbar sind.	Muss	NF-ARCH-01	Datensätze anlegen, Anwendung neu starten; Datensätze unverändert.
NF-ARCH-02	Die Datenhaltung SOLLTE hinter einer abstrakten Schnittstelle (Repository/DAO) gekapselt sein, sodass die Speichertechnologie austauschbar ist.	Soll	NF-ARCH-02	Code-Review zeigt Repository- bzw. DAO-Abstraktion (IF-02).

5.7 Sicherheit und Datenschutz (NF-SEC)

ID	Anforderungstext	Prio	Herkunft (LH)	Akzeptanzkriterium
NF-SEC-01	Das System SOLLTE personenbezogene Kundendaten so speichern, dass sie ohne Zugriff auf das Anwendungs-Verzeichnis nicht im Klartext einsehbar sind.	Soll	NF-SEC-01	Inspektion aus fremdem Nutzerkontext zeigt keinen lesbaren Klartext.
NF-SEC-02	Das System SOLLTE das Löschen eines Kunden gemäß DSGVO Art. 17 ermöglichen, sofern keine gesetzlichen Aufbewahrungspflichten entgegenstehen.	Soll	NF-SEC-02	Löschanforderung ohne aktive Rechnungen entfernt den Kunden.

---

6. Daten und Schnittstellen

6.1 Datenobjekte mit Java-Datentypen

Für jedes Datenobjekt aus Lastenheft Kap. 6.1 wird die technische Realisierung mit konkreten Java-Datentypen festgelegt. Geldbeträge und Steuersätze werden ausnahmslos als `BigDecimal` modelliert (niemals `double`/`float`), Datumsangaben als `LocalDate`, Statusfelder als eigene `enum`-Klassen.

BigDecimal wird trotz der in der Vorlesung genannten Nachteile gewählt: höhere Ausführlichkeit (keine arithmetischen Operatoren – Berechnungen über `add()`, `multiply()`, `compareTo()`), höherer Rechen- und Speicheraufwand gegenüber primitiven Typen sowie die bekannte Falle, dass `equals()` die Skala mitvergleicht (Vergleiche daher über `compareTo()`). Diese Nachteile sind hier vertretbar, weil exakte Dezimalarithmetik bei Geldbeträgen Vorrang vor Performance hat (Rundungsfehler von `double/float` sind ausgeschlossen), die Datenmengen klein sind (Einzelplatz, wenige Positionen je Beleg) und das Vorgehen den Vorlesungsbeispielen sowie GR-06 entspricht. Die Alternative – Beträge als `long` in Cent – wurde verworfen, da sie Steuersätze und Prozentrechnung (z. B. 0.19) sowie mehrstufige Rundung umständlicher abbildet.

6.1.1 Produkt

Attribut	Java-Typ	Pflicht/Optional	Beschreibung / Constraint
produktId	long	Pflicht	Systeminterner, fortlaufender Primärschlüssel; eindeutig.
bezeichnung	String	Pflicht	Anzeigename (synonym: Produktname); nicht leer.
einzelpreisNetto	BigDecimal	Pflicht	Nettopreis ≥ 0; Skala 2, Rundung HALF_UP.
mehrwertsteuersatz	BigDecimal	Pflicht	Steuersatz als Dezimalwert (z. B. 0.19); ≥ 0.
artikelnummer	String	Optional	Anwenderkennung; falls leer, abgeleitet aus produktId nach Schema `P-<lfd.Nr.>`.
beschreibung	String	Optional	Freitext.
kategorie	String	Optional	Gruppierungsmerkmal.

6.1.2 Kunde

Attribut	Java-Typ	Pflicht/Optional	Beschreibung / Constraint
kundeId	long	Pflicht	Fortlaufender, eindeutiger Primärschlüssel.
firmenname	String	Pflicht*	Pflicht, sofern kein Nachname angegeben ist (mindestens eines von beiden).
nachname	String	Pflicht*	Pflicht, sofern kein Firmenname angegeben ist.
vorname	String	Optional	–
strasse	String	Pflicht	Straße und Hausnummer.
plz	String	Pflicht	Postleitzahl (String, um führende Nullen zu erhalten).
ort	String	Pflicht	–
telefon	String	Optional	–
email	String	Optional	Wird auf gültiges Format validiert, falls angegeben.
ustIdNr	String	Optional	Umsatzsteuer-Identifikationsnummer.
lieferadresse	String	Optional	Abweichende Lieferadresse.
ansprechpartner	String	Optional	–

*Constraint: `firmenname != null || nachname != null` (mindestens eines der beiden Felder ist befüllt).

6.1.3 Angebot

Attribut	Java-Typ	Pflicht/Optional	Beschreibung / Constraint
angebotNr	String	Pflicht	Format `A-<Jahr>-<lfd.Nr.>`; eindeutig.
kundeId	long	Pflicht	Referenz auf Kunde.
datum	LocalDate	Pflicht	Angebotsdatum.
positionen	List<Belegposition>	Pflicht	Mindestens eine Position.
gesamtBetragNetto	BigDecimal	Pflicht	Berechnet gemäß GR-06; Skala 2.
gesamtBetragBrutto	BigDecimal	Pflicht	Berechnet gemäß GR-06; Skala 2.
status	AngebotStatus	Pflicht	OFFEN, ANGENOMMEN, ABGELEHNT, UEBERFUEHRT.

6.1.4 Belegposition

Attribut	Java-Typ	Pflicht/Optional	Beschreibung / Constraint
produktId	long	Pflicht	Referenz auf Produkt.
bezeichnung	String	Pflicht	Kopie der Produktbezeichnung zum Erstellungszeitpunkt.
menge	int	Pflicht	Stückzahl > 0.
einzelpreisNetto	BigDecimal	Pflicht	Preis-Snapshot gemäß GR-04; durch spätere Änderungen unverändert.
mehrwertsteuersatz	BigDecimal	Pflicht	Steuersatz-Snapshot der Position.
gesamtpreisNetto	BigDecimal	Pflicht	`menge × einzelpreisNetto` (GR-06); Skala 2.

6.1.5 Auftragsbestätigung

Attribut	Java-Typ	Pflicht/Optional	Beschreibung / Constraint
auftragNr	String	Pflicht	Format `AB-<Jahr>-<lfd.Nr.>`; eindeutig.
angebotNr	String	Pflicht	Referenz auf genau ein Angebot.
kundeId	long	Pflicht	Referenz auf Kunde.
datum	LocalDate	Pflicht	Bestätigungsdatum.
positionen	List<Belegposition>	Pflicht	Übernommen aus dem Angebot.
gesamtBetragNetto	BigDecimal	Pflicht	Skala 2.
gesamtBetragBrutto	BigDecimal	Pflicht	Skala 2.
status	AuftragsStatus	Pflicht	BESTAETIGT, GELIEFERT.

6.1.6 Lieferschein

Attribut	Java-Typ	Pflicht/Optional	Beschreibung / Constraint
lieferscheinNr	String	Pflicht	Format `LS-<Jahr>-<lfd.Nr.>`; eindeutig.
auftragNr	String	Pflicht	Referenz auf genau eine Auftragsbestätigung.
kundeId	long	Pflicht	Referenz auf Kunde.
lieferdatum	LocalDate	Pflicht	Datum der Auslieferung.
positionen	List<Belegposition>	Pflicht	Produkt und Menge je Position.
status	LieferscheinStatus	Pflicht	OFFEN, GELIEFERT, FAKTURIERT.

6.1.7 Rechnung

Attribut	Java-Typ	Pflicht/Optional	Beschreibung / Constraint
rechnungNr	String	Pflicht	Eindeutig, fortlaufend, lückenlos (GR-02); Format `R-<Jahr>-<lfd.Nr.>`.
lieferscheinNr	String	Pflicht	Referenz auf genau einen Lieferschein (GR-01).
kundeId	long	Pflicht	Referenz auf Kunde.
rechnungsdatum	LocalDate	Pflicht	Automatisch `LocalDate.now()` beim Anlegen (F-DP-06).
positionen	List<Belegposition>	Pflicht	–
gesamtBetragNetto	BigDecimal	Pflicht	Skala 2.
gesamtBetragUst	BigDecimal	Pflicht	USt-Betrag gemäß GR-06; Skala 2.
gesamtBetragBrutto	BigDecimal	Pflicht	Skala 2.
status	RechnungStatus	Pflicht	OFFEN, BEZAHLT.

6.1.8 Enum-Klassen

Abbildung 3: UML-Klassendiagramm der Datenobjekte mit Java-Typen und Multiplizitäten.

Abbildung 3: Abbildung 3: UML-Klassendiagramm der Datenobjekte mit Java-Typen und Multiplizitäten.

Enum	Konstanten
AngebotStatus	OFFEN, ANGENOMMEN, ABGELEHNT, UEBERFUEHRT
AuftragsStatus	BESTAETIGT, GELIEFERT
LieferscheinStatus	OFFEN, GELIEFERT, FAKTURIERT
RechnungStatus	OFFEN, BEZAHLT

6.1.9 UML-Klassendiagramm der Datenobjekte Abbildung 3 zeigt die Datenobjekte mit ihren Java-Typen und Beziehungen. Ein Kunde besitzt beliebig viele Angebote; jeder Beleg besteht aus mindestens einer Belegposition; jede Position verweist auf genau ein Produkt. Die Belege bilden die Dokumentenkette Angebot → Auftragsbestätigung → Lieferschein → Rechnung mit Multiplizität 1:0..1 (jeder Vorgänger hat höchstens einen Nachfolger).

6.2 Schnittstellen

Die Schnittstellen folgen der Satzschablone: **IF-<Nr>:** Das System MUSS `<Art der Schnittstelle>` bereitstellen, um `<Zweck>`.

ID	Schnittstelle	Anforderung (Satzschablone)	
IF-01	Benutzerschnittstelle	Das System MUSS eine grafische Benutzerschnittstelle (GUI ↔ Logikschicht) bereitstellen, um die Interaktion des Anwenders zu ermöglichen.	IF-01
IF-02	Persistenzschnittstelle	Das System MUSS eine Persistenzschnittstelle (Logikschicht ↔ Datenhaltung via Repository/DAO) bereitstellen, um Daten dauerhaft zu speichern und zu laden.	IF-02
IF-03	PDF-Export-Schnittstelle	Das System MUSS eine PDF-Export-Schnittstelle (Logikschicht ↔ PDF-Bibliothek) bereitstellen, um Belege als druckbare PDF auszugeben.	IF-03

6.3 Geschäftsregeln

Die Geschäftsregeln GR-01 bis GR-06 werden aus Lastenheft Kap. 6.3 übernommen und um Java-seitige Umsetzungshinweise ergänzt.

GR-01 – Dokumentenkette. Eine Rechnung verweist auf genau einen Lieferschein, ein Lieferschein auf genau eine Auftragsbestätigung, eine Auftragsbestätigung auf genau ein Angebot. Ein Folgedokument darf nur erzeugt werden, wenn das Vorgängerdokument im passenden Status (Kap. 6.1.7 LH) vorliegt. *Umsetzung:* Statusprüfung im jeweiligen Service vor der Erzeugung; Referenz als Pflicht-Fremdschlüssel (z. B. `lieferscheinNr` in `Rechnung`); Verstoß löst eine fachliche Ausnahme (`IllegalStateException`/eigene `BelegketteException`) aus. Realisiert in F-DP-08.

GR-02 – Eindeutigkeit der Rechnungsnummer. Das System vergibt beim Erstellen die nächste freie, fortlaufende Rechnungsnummer; Nummern werden nicht wiederverwendet. *Umsetzung:* Zentraler Nummernkreis im `RechnungService` (z. B. `AtomicLong` bzw. persistierter Zähler); Format `R-<Jahr>-<lfd.Nr.>`. Realisiert in F-DP-09.

GR-03 – Unveränderlichkeit festgeschriebener Rechnungen. Eine gespeicherte Rechnung ist inhaltlich nicht mehr änderbar. *Umsetzung:* `Rechnung` als unveränderliches Objekt (finale Felder, keine Setter); das `RechnungRepository` bietet keine Update-Operation für festgeschriebene Rechnungen. Realisiert in F-DP-10.

GR-04 – Preisübernahme. Der zum Zeitpunkt der Positionsübernahme gültige Einzelpreis wird in der Position festgehalten und durch spätere Produktpreisänderungen nicht verändert. *Umsetzung:* Beim Hinzufügen einer Position wird `einzelpreisNetto` als Kopie aus dem Produkt in die `Belegposition` übernommen (Snapshot), nicht als Referenz. Realisiert in F-DP-11.

Abbildung 4: Komponentendiagramm – vier Module in der dreischichtigen Architektur.

Abbildung 4: Abbildung 4: Komponentendiagramm – vier Module in der dreischichtigen Architektur.

Abbildung 5: Detailliertes UML-Klassendiagramm der Kernklassen mit Methodensignaturen.

Abbildung 5: Abbildung 5: Detailliertes UML-Klassendiagramm der Kernklassen mit Methodensignaturen.

GR-05 – Stammdatenschutz. Ein Produkt oder Kunde darf nicht gelöscht werden, solange er in einem Beleg referenziert wird. *Umsetzung:* Vor dem Löschen prüft der `ProduktService`/`KundeService` über das `BelegRepository`, ob Referenzen bestehen; bei Treffer wird das Löschen abgewiesen. Realisiert in F-PV-04 und F-KV-04.

GR-06 – Summenberechnung. Nettosumme = SUM(Menge × Einzelpreis); USt-Betrag pro Position = Netto × MwSt-Satz; Bruttosumme = Netto + USt. *Umsetzung:* Berechnung ausschließlich mit `BigDecimal` über `multiply()`/`add()`; Rundung `RoundingMode.HALF_UP` auf 2 Nachkommastellen (`setScale(2, HALF_UP)`). Realisiert in F-DP-12.

---

7. Systemarchitektur

7.1 Dreischichtige Architektur

Das System folgt gemäß NF-MAINT-03 und NF-ARCH-01/02 einer dokumentierten, dreischichtigen Architektur mit klarer Trennung der Verantwortlichkeiten:

Schicht	Aufgabe	Realisierung / Package
Präsentationsschicht	Darstellung, Benutzerinteraktion (GUI)	JavaFX/Swing, `de.hsmannheim.faktura.ui`
Logikschicht	Geschäftslogik, Validierung, Berechnungen, Statuslogik	Service-Klassen, `de.hsmannheim.faktura.service`
Datenhaltungsschicht	Persistenz, gekapselt über Repository/DAO	`de.hsmannheim.faktura.repository`

Die Schichten kommunizieren ausschließlich über die in Kap. 6.2 definierten Schnittstellen (IF-01 GUI ↔ Logik, IF-02 Logik ↔ Datenhaltung, IF-03 Logik ↔ PDF-Bibliothek). Abbildung 4 zeigt die vier fachlichen Module als Komponenten innerhalb der drei Schichten.

7.2 UML-Klassendiagramm der Kernklassen

Abbildung 5 zeigt die Kernklassen der Logik- und Datenhaltungsschicht mit Methodensignaturen (Parameter- und Rückgabetypen) sowie deren Beziehungen. Die Services bilden die Logikschicht, das `ProduktRepository` steht stellvertretend für die als Interface modellierte Persistenzabstraktion (IF-02, NF-ARCH-02).

7.3 UML-Sequenzdiagramm „Angebot → Rechnung"

Abbildung 6 zeigt den Hauptprozess von der Statusänderung eines angenommenen Angebots bis zur Erzeugung der Rechnung. Der Anwender stößt die Aktionen über die GUI an; die Services kapseln die Geschäftslogik (Statusprüfung GR-01, Nummernvergabe GR-02, Summenberechnung GR-06), die Repositories übernehmen die Persistenz.

Abbildung 6: UML-Sequenzdiagramm des Hauptprozesses „Angebot → Rechnung".

Abbildung 6: Abbildung 6: UML-Sequenzdiagramm des Hauptprozesses „Angebot → Rechnung".

8. Testbare Abnahmekriterien

Für jede Muss-Anforderung ist mindestens ein Testfall definiert. Jeder Testfall gibt die Testart an: **Unit** (JUnit-5-Komponententest der Logikschicht ohne GUI), **Integration** (schichtübergreifend, inkl. Persistenz) oder **manuell** (manueller Akzeptanztest über die GUI). Die Testfälle sind so detailliert, dass daraus direkt ein Modultestplan ableitbar ist.

8.1 Produktverwaltung (TC-PV)

TC-ID	Bezeichnung	Vorbedingung	Testeingabe	Erwartetes Ergebnis	PH-Anf.	Testart
TC-PV-01	Produkt mit Pflichtattributen anlegen	Leerer Produktbestand	bezeichnung, einzelpreisNetto=10.00, mwst=0.19	Produkt gespeichert, erscheint in Übersicht, produktId vergeben.	F-PV-01	Integration
TC-PV-02	Anlegen ohne MwSt-Satz	Eingabemaske offen	Pflichtfeld mehrwertsteuersatz leer	Fehlermeldung; keine Speicherung.	F-PV-01	Unit
TC-PV-03	Preis ändern, Angebot unverändert	Produkt in bestehendem Angebot	einzelpreisNetto von 10.00 auf 12.00 ändern	Produktpreis aktualisiert; Position im Bestandsangebot bleibt 10.00.	F-PV-02	Integration
TC-PV-04	Persistenz nach Neustart	Produkt bearbeitet	Anwendung neu starten, Produkt öffnen	Geänderte Daten weiterhin vorhanden.	F-PV-02	Integration
TC-PV-05	Produktsuche Treffer	Produkt „Schraube” vorhanden	Suchbegriff „Schraube”	Produkt erscheint in Trefferliste.	F-PV-03	Unit
TC-PV-06	Produktsuche kein Treffer	Bestand ohne passendes Produkt	Suchbegriff „xyz123”	Hinweis „Kein Produkt gefunden”.	F-PV-03	Unit
TC-PV-07	Löschen referenzierten Produkts	Produkt in Angebot referenziert	Löschen auslösen	Löschen abgewiesen (GR-05); Produkt bleibt erhalten.	F-PV-04	Integration
TC-PV-08	Löschen nicht referenzierten Produkts	Produkt ohne Referenz, Bestätigung	Löschen + Sicherheitsabfrage bestätigen	Produkt entfernt; nicht mehr in Übersicht/Suche.	F-PV-04	Integration

8.2 Kundenverwaltung (TC-KV)

TC-ID	Bezeichnung	Vorbedingung	Testeingabe	Erwartetes Ergebnis	PH-Anf.	Testart
TC-KV-01	Kunde mit Pflichtattributen anlegen	Leerer Kundenbestand	firmenname, strasse, plz, ort	Kunde gespeichert, erscheint in Kundenliste.	F-KV-01	Integration
TC-KV-02	Ungültige E-Mail abweisen	Eingabemaske offen	email = „abc@”	Fehlermeldung Format; keine Speicherung.	F-KV-01	Unit
TC-KV-03	Telefonnummer ändern	Kunde vorhanden	telefon ändern, speichern	Änderung sofort sichtbar.	F-KV-02	Unit
TC-KV-04	Persistenz nach Neustart	Kunde bearbeitet	Neustart, Kunde öffnen	Geänderte Daten weiterhin vorhanden.	F-KV-02	Integration
TC-KV-05	Kundensuche Treffer	Kunde „Müller GmbH” vorhanden	Suchbegriff „Müller”	Kunde in Trefferliste.	F-KV-03	Unit

TC-ID	Bezeichnung	Vorbedingung	Testeingabe	Erwartetes Ergebnis	PH-Anf.	Testart
TC-KV-06	Kundensuche kein Treffer	kein passender Kunde	Suchbegriff „zzz”	Hinweis „Kein Kunde gefunden”.	F-KV-03	Unit
TC-KV-07	Kunde löschen	Kunde ohne Referenz	Löschen + Bestätigung	Kunde entfernt; nicht mehr auffindbar.	F-KV-04	Integration
TC-KV-08	Löschen referenzierten Kunden	Kunde in Rechnung referenziert	Löschen auslösen	Löschen abgewiesen (GR-05).	F-KV-04	Integration

8.3 Dokumentenprozess (TC-DP)

TC-ID	Bezeichnung	Vorbedingung	Testeingabe	Erwartetes Ergebnis	PH-Anf.	Testart
TC-DP-01	Angebot mit einer Position	1 Kunde, 1 Produkt vorhanden	Position menge=2, einzelpreisNetto=10.00	Angebot gespeichert; Netto=20.00, Brutto=23.80 (GR-06); status=OFFEN.	F-DP-01	Integration
TC-DP-02	Angebotsstatus persistent	Angebot status=OFFEN	Status „angenommen” setzen, Neustart	Status ANGENOMMEN nach Neustart sichtbar.	F-DP-02	Integration
TC-DP-03	Auftragsbestätigung erzeugen	Angebot status=ANGENOMMEN	„Auftragsbestätigung erzeugen”	AB mit allen Daten + Referenz; Angebot status=UEBERFUEHRT.	F-DP-03	Integration
TC-DP-04	Lieferschein erzeugen	AB status=BESTAETIGT	„Lieferschein erzeugen”	Lieferschein referenziert AB; Positionen vollständig.	F-DP-04	Integration
TC-DP-05	Rechnung mit Pflichtangaben	Lieferschein status=GELIEFERT	„Rechnung erzeugen”	Rechnung mit § 14 UStG-Angaben, eindeutiger Nr., Lieferschein-Referenz.	F-DP-05	Integration
TC-DP-06	Automatisches Rechnungsdatum	Rechnungserzeugung möglich	Neue Rechnung anlegen	rechnungsdatum = Systemdatum (LocalDate.now()).	F-DP-06	Unit
TC-DP-07	PDF-Export je Belegtyp	je ein gespeicherter Beleg	„Als PDF exportieren”, Zielpfad wählen	PDF je Typ erzeugt; Inhalt entspricht Beleg; Rechnung enthält § 14-Angaben.	F-DP-07	Integration
TC-DP-08	Rechnung ohne Lieferschein abweisen	kein Lieferschein vorhanden	Rechnung direkt anlegen	Erzeugung abgewiesen (GR-01).	F-DP-08	Unit
TC-DP-09	Fortlaufende Rechnungsnummern	3 abrechnungsreife Lieferscheine	3 Rechnungen nacheinander erzeugen	Drei eindeutige, fortlaufende Nummern; keine Wiederverwendung.	F-DP-09	Unit
TC-DP-10	Unveränderlichkeit der Rechnung	gespeicherte Rechnung	Bearbeitungsversuch	Änderung abgewiesen (GR-03).	F-DP-10	Unit
TC-DP-11	Preis-Snapshot	Angebot mit Position, Preis 10.00	Produktpreis nachträglich auf 12.00 ändern	Position im Angebot bleibt 10.00 (GR-04).	F-DP-11	Unit
TC-DP-12	SummenberechnungBigDecimal	Position menge=3, preis=9.99, mwst=0.19	Summen berechnen	Netto=29.97, USt=5.69, Brutto=35.66 (HALF_UP, GR-06).	F-DP-12	Unit

8.4 Programmoberfläche (TC-GUI)

TC-ID	Bezeichnung	Vorbedingung	Testeingabe	Erwartetes Ergebnis	PH-Anf.	Testart
TC-GUI-01	Live-Summenaktualisierung	Angebotsmaske offen, Produkte	Artikel per Dropdown hinzufügen	Gesamtsumme aktualisiert sich ohne spürbare Verzögerung.	F-GUI-01	manuell
TC-GUI-02	Erfolgsmeldung nach Speichern	gültiges Angebot	Speichern	Erfolgsmeldung erscheint.	F-GUI-01	manuell
TC-GUI-03	Datenübernahme Auftragsbestätigung	Angebot status=ANGENOMMEN	„Auftragsbestätigung" öffnen	Daten übernommen; Titel „Auftragsbestätigung"; Liefertermin markiert.	F-GUI-02	manuell
TC-GUI-04	Druckvorschau vor Speichern	abrechnungsreifer Lieferschein	Rechnung erzeugen	Druckvorschau wird vor finalem Speichern angezeigt.	F-GUI-03	manuell
TC-GUI-05	Status-Icon „Abgeschlossen"	Rechnung erzeugt	Übersicht öffnen	Auftragsstatus durch grünes Icon markiert.	F-GUI-03	manuell
TC-GUI-06	Echtzeit-Filter	mehrere Belege vorhanden	Name/Nummer ins Suchfeld eingeben	Trefferliste aktualisiert sich sofort, tabellarisch.	F-GUI-04	manuell
TC-GUI-07	Hinweis bei leerer Trefferliste	kein passender Beleg	ungültigen Suchbegriff eingeben	Text „Keine Treffer gefunden" erscheint.	F-GUI-04	manuell
TC-GUI-08	PDF-Export-Schaltfläche	Beleg angezeigt	„Als PDF exportieren" klicken	Schaltfläche sichtbar; Export ausgeführt; Erfolgsmeldung mit Pfad.	F-GUI-05	manuell

8.5 Nicht-funktionale Anforderungen (TC-NF)

TC-ID	Bezeichnung	Vorbedingung	Testeingabe / Vorgehen	Erwartetes Ergebnis	PH-Anf.	Testart
TC-NF-01	Kunde anlegen < 2 min	5 Probanden, ungeschult	Aufgabe „Kunde anlegen"	Median < 120 s; Erfolgsquote > 90 %.	NF-USE-01	manuell
TC-NF-02	Angebot $\rightarrow$ Rechnung erstmalig	5 Probanden	Aufgabe ohne Hilfetext	≥ 4 von 5 im ersten Versuch erfolgreich.	NF-USE-02	manuell
TC-NF-03	Rückmeldezeit Validierung	Eingabemaske	Fehlerhafte Eingabe, Stoppuhr	Sichtbare Rückmeldung < 1 s.	NF-USE-03	manuell
TC-NF-04	Antwortzeit bei 1.000 Datensätzen	1.000 Produkte + 1.000 Kunden	Benutzeraktion ausführen	Antwortzeit < 1 s.	NF-PERF-01	Integration
TC-NF-05	Listenanzeige bei 1.000 Einträgen	1.000 Datensätze	Übersichtsliste öffnen	Anzeigedauer < 2 s.	NF-PERF-02	Integration
TC-NF-06	Kaltstart	Referenzhardware	Anwendung kalt starten	Hauptansicht bedienbar < 5 s.	NF-PERF-03	manuell
TC-NF-07	Modultrennung	Repository vorhanden	Statische Paketprüfung	Vier getrennte Modulpakete vorhanden.	NF-MAINT-01	Unit
TC-NF-08	Dokumentationsabdeckung	Quellcode	JavaDoc-Report erzeugen	> 80 % öffentliche Klassen/Methoden kommentiert.	NF-MAINT-02	Unit

TC-ID	Bezeichnung	Vorbedingung	Testeingabe / Vorgehen	Erwartetes Ergebnis	PH-Anf.	Testart
TC-NF-09	Schichtentrennung	Architekturkapitel	Review Kap. 7	Drei Schichten nachweisbar getrennt.	NF-MAINT-03	manuell
TC-NF-10	Vollständige Traceability	Kap. 9.4 vorhanden	Matrix prüfen	Jede Muss-Anforderung hat $\geq$ 1 Testfall.	NF-TEST-01	manuell
TC-NF-11	Logik ohne GUI testbar	Logikschicht	Unit-Test pro Modul ausführen	Mindestens ein GUI-unabhängiger Unit-Test je Modul.	NF-TEST-02	Unit
TC-NF-12	Testabdeckung	Testsuite	Coverage-Report erzeugen	Line Coverage $>$ 60 %.	NF-TEST-03	Unit
TC-NF-13	Quellcode versioniert	Gitty-Repositories	Sichtprüfung	Gesamter Code in genannten Repos.	NF-VER-01	manuell
TC-NF-14	Code-Review je Merge	Merge-Historie	Review-Historie prüfen	Jeder Merge hat $\geq$ 1 Reviewer.	NF-VER-02	manuell
TC-NF-15	Persistenz nach Neustart	Datensätze angelegt	Anwendung neu starten	Datensätze vollständig wiederhergestellt.	NF-ARCH-01	Integration
TC-NF-16	Repository-Abstraktion	Quellcode	Code-Review	Persistenz hinter Repository/DAO gekapselt (IF-02).	NF-ARCH-02	manuell
TC-NF-17	Klartextschutz	Datendatei geschrieben	Inspektion aus fremdem Nutzerkontext	Keine lesbaren personenbezogenen Daten im Klartext.	NF-SEC-01	manuell
TC-NF-18	DSGVO-Löschung	Kunde ohne aktive Rechnungen	Löschanforderung Art. 17	Kunde entfernt; bei Aufbewahrungspflicht abgewiesen.	NF-SEC-02	Integration

---

9. Anhänge

9.1 Glossar

Übernommen aus Lastenheft Kap. 8.1, ergänzt um pflichtenheftspezifische Begriffe.

Begriff	Bedeutung
Angebot	Unverbindliches Preisangebot an einen Kunden.
Auftragsbestätigung	Verbindliche Bestätigung der Auftragsannahme nach Angebot (Kurzform AB).
Lieferschein	Dokument, das die Auslieferung der Ware bescheinigt.
Rechnung	Forderung nach § 14 UStG mit Steuerangaben.
Anwender	Einzige Benutzerrolle des Systems (keine Authentifizierungsrolle).
Stammdaten	Produkt- und Kundendaten.
Bewegungsdaten	Belege (Angebot, Auftragsbestätigung, Lieferschein, Rechnung).
Repository	Entwurfsmuster zur Kapselung der Datenhaltung; abstrahiert die konkrete Speichertechnologie (IF-02).
DAO	Data Access Object; Objekt, das den Zugriff auf eine Datenquelle kapselt; hier synonym zum Repository.
Service-Klasse	Klasse der Logikschicht, die die Geschäftslogik eines Moduls bündelt (z. B. `KundeService`).
BigDecimal	Java-Klasse für exakte Dezimalarithmetik; verbindlich für alle Geld- und Steuerbeträge (statt double/float).
LocalDate	Java-Klasse für ein Datum ohne Uhrzeit; verbindlich für alle Datumsfelder.

Begriff	Bedeutung
Traceability	Rückverfolgbarkeit von Anforderung zu Realisierung und Test.

9.2 Abkürzungsverzeichnis

Übernommen aus Lastenheft Kap. 8.2, ergänzt um pflichtenheftspezifische Abkürzungen.

Abkürzung	Bedeutung
AB	Auftragsbestätigung
BA	Benutzeranforderung (Lastenheft-Schablone)
DAO	Data Access Object
DSGVO	Datenschutz-Grundverordnung
F	Funktionale Anforderung (Pflichtenheft-Schablone)
GR	Geschäftsregel
GUI	Graphical User Interface
IF	Interface / Schnittstelle
NF	Nicht-funktionale Anforderung
SRS	System Requirements Specification (Pflichtenheft)
TC	Testfall (Test Case)
UStG	Umsatzsteuergesetz
USt-IdNr.	Umsatzsteuer-Identifikationsnummer

9.3 Referenzen

- [**1**] Lastenheft v1.0 (`Lastenheft_v1_0.md`), 15.05.2026
- [**2**] Project Charter v1.1 (`project-charter_v1_1.md`), 15.05.2026
- [**3**] `week7slides.pdf` – Vorlesung Software Engineering 1, Woche 7
- [**4**] § 14 UStG – Umsatzsteuergesetz, Pflichtangaben in Rechnungen
- [**5**] DSGVO – Verordnung (EU) 2016/679

9.4 Traceability-Matrix LH ↔ PH

Jede Anforderung des Lastenhefts v1.0 ist mindestens einer Pflichtenheft-Anforderung und mindestens einem Testfall zugeordnet. Die Matrix ist lückenlos.

LH-Anforderung	Typ	PH-Anforderung(en)	Testfall(e)
BA-PV-01	funktional	F-PV-01	TC-PV-01, TC-PV-02
BA-PV-02	funktional	F-PV-02	TC-PV-03, TC-PV-04
BA-PV-03	funktional	F-PV-03	TC-PV-05, TC-PV-06
BA-PV-04	funktional	F-PV-04	TC-PV-07, TC-PV-08
BA-KV-01	funktional	F-KV-01	TC-KV-01, TC-KV-02
BA-KV-02	funktional	F-KV-02	TC-KV-03, TC-KV-04
BA-KV-03	funktional	F-KV-03	TC-KV-05, TC-KV-06
BA-KV-04	funktional	F-KV-04	TC-KV-07, TC-KV-08
BA-DP-01	funktional	F-DP-01	TC-DP-01
BA-DP-02	funktional	F-DP-02	TC-DP-02
BA-DP-03	funktional	F-DP-03	TC-DP-03
BA-DP-04	funktional	F-DP-04	TC-DP-04
BA-DP-05	funktional	F-DP-05	TC-DP-05
BA-DP-06	funktional	F-DP-06	TC-DP-06
BA-DP-07	funktional	F-DP-07	TC-DP-07
BA-GUI-01	funktional	F-GUI-01	TC-GUI-01, TC-GUI-02
BA-GUI-02	funktional	F-GUI-02	TC-GUI-03

LH-Anforderung	Typ	PH-Anforderung(en)	Testfall(e)
BA-GUI-03	funktional	F-GUI-03	TC-GUI-04, TC-GUI-05
BA-GUI-04	funktional	F-GUI-04	TC-GUI-06, TC-GUI-07
BA-GUI-05	funktional	F-GUI-05	TC-GUI-08
GR-01	Geschäftsregel	F-DP-08	TC-DP-08
GR-02	Geschäftsregel	F-DP-09	TC-DP-09
GR-03	Geschäftsregel	F-DP-10	TC-DP-10
GR-04	Geschäftsregel	F-DP-11	TC-DP-11
GR-05	Geschäftsregel	F-PV-04, F-KV-04	TC-PV-07, TC-KV-08
GR-06	Geschäftsregel	F-DP-12	TC-DP-12, TC-DP-01
NF-USE-01	NF/Usability	NF-USE-01	TC-NF-01
NF-USE-02	NF/Usability	NF-USE-02	TC-NF-02
NF-USE-03	NF/Usability	NF-USE-03	TC-NF-03
NF-PERF-01	NF/Performance	NF-PERF-01	TC-NF-04
NF-PERF-02	NF/Performance	NF-PERF-02	TC-NF-05
NF-PERF-03	NF/Performance	NF-PERF-03	TC-NF-06
NF-MAINT-01	NF/Wartbarkeit	NF-MAINT-01	TC-NF-07
NF-MAINT-02	NF/Wartbarkeit	NF-MAINT-02	TC-NF-08
NF-MAINT-03	NF/Architektur	NF-MAINT-03	TC-NF-09
NF-TEST-01	NF/Testbarkeit	NF-TEST-01	TC-NF-10
NF-TEST-02	NF/Testbarkeit	NF-TEST-02	TC-NF-11
NF-TEST-03	NF/Testbarkeit	NF-TEST-03	TC-NF-12
NF-VER-01	NF/Versionierung	NF-VER-01	TC-NF-13
NF-VER-02	NF/Versionierung	NF-VER-02	TC-NF-14
NF-ARCH-01	NF/Architektur	NF-ARCH-01	TC-NF-15
NF-ARCH-02	NF/Architektur	NF-ARCH-02	TC-NF-16
NF-SEC-01	NF/Security	NF-SEC-01	TC-NF-17
NF-SEC-02	NF/Security	NF-SEC-02	TC-NF-18

Die Traceability-Matrix bildet alle Anforderungen des Lastenhefts v1.0 vollständig auf das Pflichtenheft ab.

9.5 PlantUML-Quellen

Die folgenden Quelltexte erzeugen die Abbildungen 1–6 (gerendert nach `sources/abbX.png`). Sie dienen der Reproduzierbarkeit und sind nicht Teil des Fließtexts.

Abbildung 1 – Kap. 2.4 – Use-Case-Diagramm

```
@startuml
left to right direction
skinparam packageStyle rectangle
actor "Anwender" as A

rectangle "Fakturierungssystem" {
  usecase "Produkt verwalten\n(anlegen, bearbeiten,\nsuchen, löschen)" as UC1
  usecase "Kunde verwalten\n(anlegen, bearbeiten,\nsuchen, löschen)" as UC2
  usecase "Angebot erstellen" as UC3
  usecase "Auftragsbestätigung erzeugen" as UC4
  usecase "Lieferschein erzeugen" as UC5
  usecase "Rechnung erzeugen" as UC6
  usecase "Beleg suchen" as UC7
  usecase "Beleg als PDF exportieren" as UC8
}

A --> UC1
A --> UC2
```

```
A --> UC3
A --> UC4
A --> UC5
A --> UC6
A --> UC7
A --> UC8

UC4 ..> UC3 : <<precedes>>
UC5 ..> UC4 : <<precedes>>
UC6 ..> UC5 : <<precedes>>
@enduml
```

Abbildung 2 – Kap. 3.3 – Systemkontextdiagramm

```
@startuml
left to right direction
skinparam componentStyle rectangle
actor "Anwender" as A
rectangle "Fakturierungssystem" as SYS
database "Lokales Dateisystem\n(JSON-Persistenz)" as FS
file "PDF-Datei\n(Beleg-Export)" as PDF

A --> SYS : bedient über GUI (IF-01)
SYS --> FS : liest / schreibt Daten (IF-02)
SYS --> PDF : erzeugt Beleg-PDF (IF-03)
@enduml
```

Abbildung 3 – Kap. 6.1.9 – Klassendiagramm der Datenobjekte

```
@startuml
hide empty members
skinparam classAttributeIconSize 0

class Produkt {
  -produktId: long
  -bezeichnung: String
  -einzelpreisNetto: BigDecimal
  -mehrwertsteuersatz: BigDecimal
  -artikelnummer: String
  -beschreibung: String
  -kategorie: String
}

class Kunde {
  -kundeId: long
  -firmenname: String
  -nachname: String
  -strasse: String
  -plz: String
  -ort: String
  -email: String
  -ustIdNr: String
}

class Belegposition {
  -produktId: long
  -bezeichnung: String
  -menge: int
```

```
  -einzelpreisNetto: BigDecimal
  -mehrwertsteuersatz: BigDecimal
  -gesamtpreisNetto: BigDecimal
}

class Angebot {
  -angebotNr: String
  -datum: LocalDate
  -gesamtBetragNetto: BigDecimal
  -gesamtBetragBrutto: BigDecimal
  -status: AngebotStatus
}

class Auftragsbestaetigung {
  -auftragNr: String
  -datum: LocalDate
  -status: AuftragsStatus
}

class Lieferschein {
  -lieferscheinNr: String
  -lieferdatum: LocalDate
  -status: LieferscheinStatus
}

class Rechnung {
  -rechnungNr: String
  -rechnungsdatum: LocalDate
  -gesamtBetragNetto: BigDecimal
  -gesamtBetragUst: BigDecimal
  -gesamtBetragBrutto: BigDecimal
  -status: RechnungStatus
}

enum AngebotStatus {
  OFFEN
  ANGENOMMEN
  ABGELEHNT
  UEBERFUEHRT
}
enum AuftragsStatus {
  BESTAETIGT
  GELIEFERT
}
enum LieferscheinStatus {
  OFFEN
  GELIEFERT
  FAKTURIERT
}
enum RechnungStatus {
  OFFEN
  BEZAHLT
}

Kunde "1" --> "0..*" Angebot
Angebot "1" *-- "1..*" Belegposition
```

```
Auftragsbestaetigung "1" *-- "1..*" Belegposition
Lieferschein "1" *-- "1..*" Belegposition
Rechnung "1" *-- "1..*" Belegposition
Belegposition "0..*" --> "1" Produkt
Angebot "1" --> "0..1" Auftragsbestaetigung
Auftragsbestaetigung "1" --> "0..1" Lieferschein
Lieferschein "1" --> "0..1" Rechnung

Angebot ..> AngebotStatus
Auftragsbestaetigung ..> AuftragsStatus
Lieferschein ..> LieferscheinStatus
Rechnung ..> RechnungStatus
@enduml
```

Abbildung 4 – Kap. 7.1 – Komponentendiagramm

```
@startuml
skinparam componentStyle rectangle

package "Präsentationsschicht (ui)" {
  [ProduktAnsicht]
  [KundeAnsicht]
  [BelegAnsicht]
}

package "Logikschicht (service)" {
  [ProduktService]
  [KundeService]
  [AngebotService]
  [RechnungService]
  [PdfExportService]
}

package "Datenhaltungsschicht (repository)" {
  [ProduktRepository]
  [KundeRepository]
  [BelegRepository]
}

cloud "PDF-Bibliothek" as PDFLIB
database "JSON-Datei" as JSON

[ProduktAnsicht] --> [ProduktService] : IF-01
[KundeAnsicht] --> [KundeService] : IF-01
[BelegAnsicht] --> [AngebotService] : IF-01
[BelegAnsicht] --> [RechnungService] : IF-01
[BelegAnsicht] --> [PdfExportService] : IF-01

[ProduktService] --> [ProduktRepository] : IF-02
[KundeService] --> [KundeRepository] : IF-02
[AngebotService] --> [BelegRepository] : IF-02
[RechnungService] --> [BelegRepository] : IF-02

[ProduktRepository] --> JSON
[KundeRepository] --> JSON
[BelegRepository] --> JSON
[PdfExportService] --> PDFLIB : IF-03
```

```
@enduml
```

Abbildung 5 – Kap. 7.2 – Klassendiagramm der Kernklassen

```
@startuml
hide empty members
skinparam classAttributeIconSize 0

class Produkt {
  -produktId: long
  -bezeichnung: String
  -einzelpreisNetto: BigDecimal
  -mehrwertsteuersatz: BigDecimal
}

class Kunde {
  -kundeId: long
  -firmenname: String
  -nachname: String
}

class Belegposition {
  -menge: int
  -einzelpreisNetto: BigDecimal
  -mehrwertsteuersatz: BigDecimal
  +berechneGesamtpreisNetto(): BigDecimal
}

class Angebot {
  -angebotNr: String
  -status: AngebotStatus
  +berechneSummen(): void
}

class Rechnung {
  -rechnungNr: String
  -rechnungsdatum: LocalDate
  -status: RechnungStatus
}

class AngebotService {
  +erstelleAngebot(kundeId: long, positionen: List<Belegposition>): Angebot
  +setzeStatus(angebotNr: String, status: AngebotStatus): void
  +erzeugeAuftragsbestaetigung(angebotNr: String): Auftragsbestaetigung
}

class KundeService {
  +anlegen(kunde: Kunde): long
  +bearbeiten(kunde: Kunde): void
  +suchen(begriff: String): List<Kunde>
  +loeschen(kundeId: long): void
}

class RechnungService {
  +erzeugeRechnung(lieferscheinNr: String): Rechnung
  +naechsteRechnungsnummer(): String
  +markiereBezahlt(rechnungNr: String): void
```

```
}

class PdfExportService {
  +exportiere(beleg: Beleg, zielpfad: Path): Path
}

interface ProduktRepository {
  +speichern(produkt: Produkt): long
  +findeById(produktId: long): Optional<Produkt>
  +findeAlle(): List<Produkt>
  +loeschen(produktId: long): void
}

AngebotService ..> Angebot
AngebotService ..> Belegposition
RechnungService ..> Rechnung
KundeService ..> Kunde
KundeService ..> ProduktRepository
PdfExportService ..> Rechnung
Angebot "1" *-- "1..*" Belegposition
Belegposition "0..*" --> "1" Produkt
@enduml
```

Abbildung 6 – Kap. 7.3 – Sequenzdiagramm

```
@startuml
actor Anwender
participant "GUI\n(BelegAnsicht)" as GUI
participant AngebotService
participant BelegRepository
participant RechnungService

Anwender -> GUI : "Rechnung erzeugen"
GUI -> AngebotService : erzeugeAuftragsbestaetigung(angebotNr)
AngebotService -> BelegRepository : pruefeStatus(angebot, ANGENOMMEN)
BelegRepository --> AngebotService : ok
AngebotService -> BelegRepository : speichere(auftragsbestaetigung)
AngebotService -> BelegRepository : setze Angebot=UEBERFUEHRT
note right of AngebotService : weiter über Lieferschein\n(GELIEFERT) zur Rechnung
GUI -> RechnungService : erzeugeRechnung(lieferscheinNr)
RechnungService -> BelegRepository : pruefeStatus(lieferschein, GELIEFERT)
BelegRepository --> RechnungService : ok
RechnungService -> RechnungService : naechsteRechnungsnummer() [GR-02]
RechnungService -> RechnungService : berechneSummen() [GR-06]
RechnungService -> BelegRepository : speichere(rechnung)
BelegRepository --> RechnungService : gespeichert
RechnungService --> GUI : Rechnung
GUI --> Anwender : Druckvorschau + Erfolgsmeldung
@enduml
```