Ablaufbeschreibung

Start der Anwendung: *startseite.jsp*

→ In diesem Startformular gibt der Kunde seine Adresse an und wählt Speisen aus.

- Die Datei ist in der web.xml als *Welcome File* definiert.
- Die Speisekarte wird über den *SpeisekarteServlet* bereitgestellt.
- Jede Speise hat ein Eingabefeld für die gewünschte Anzahl.
- Beim Absenden des Formulars werden die Daten an den *BestellungServlet* gesendet.

Übermittlung der Bestelldaten: *BestellungServlet.java*

→ Das Servlet empfängt die Formulardaten (Kundendaten und Bestellte Mengen) aus *startseite.jsp*, baut daraus ein Bestellung-Objekt und leitet es zur *rechnung.jsp* weiter.

→ Die Daten werden im „Session-Scope" abgelegt, damit sie für die Dauer der Sitzung erhalten bleiben.

- Es erstellt ein *Kunde*-Objekt mit den eingegebenen Adressdaten.
- Es erzeugt ein *Bestellung*-Objekt in dem Kunde, Liste der Bestellpositionen (jede mit einer Speise und Menge), Session-ID und IP-Adresse gespeichert werden.

Erzeugung der Rechnung: *rechnung.jsp*

→ Vom *BestellungServlet* nach erfolgreicher Bestellung aufgerufen (per Forward oder Redirect).

- Es greift auf das gespeicherte *Kunde*-Objekt
- Es greift auf das gespeicherte *Bestellung*-Objekt zu (Session- oder Request-Scope).
- Es berechnet den Gesamtpreis (Summe aus Preis × Menge).
- Es gibt die Rechnung formatiert aus.

Küchenansicht: *kueche.jsp*

→ Es zeigt alle aktiven Bestellungen, die in der Session oder im Application-Scope gespeichert sind.

- Wird über den *SpeisekarteListener* (oder aus ArrayList von Bestellungen) gespeist.
- Dient der Simulation der „Küche", also was gerade bestellt wurde.

Datenmodell / JavaBeans

Klasse	Aufgabe
Kunde.java	Speichert Name, Straße, PLZ, Ort – also Kundendaten
Speise.java	Beschreibt ein Gericht mit Namen und Preis
Bestellposition.java	Verknüpft eine Speise mit einer Menge
Bestellung.java	Enthält Kunde, Liste von Bestellpositionen, IP, Session-ID
Speisekarte.java	Enthält alle verfügbaren Speisen (z. B. in einer ArrayList)
SpeisekarteListener.java	Lädt beim Start der Web-App die Speisekarte in den Application-Scope
StartseiteServlet.java	Bereitet ggf. Daten für startseite.jsp vor (z. B. Menü laden)
BestellungServlet.java	Hauptlogik: erzeugt Bestellung, berechnet Rechnung, leitet weiter
SpeisekarteServlet.java	Stellt evtl. Zugriff auf die Speisekarte bereit

- *Speise.java* → Modelliert eine Speise mit Namen & Preis.
- *Kunde.java* → Enthält vollständige Adressdaten (Name, Straße, PLZ, Ort).
- *Bestellposition.java* → Kombination aus Speise und Menge.
- *Bestellung.java* → Enthält Kunde, Liste der Bestellpositionen, IP-Adresse, Session-ID.
- *Speisekarte.java* → Enthält Sammlung (ArrayList<Speise>) aller verfügbaren Speisen.
- *SpeisekarteListener.java* → Lädt Speisekarte in den Application-Scope beim Start.
- SpeisekarteServlet.java, StartseiteServlet.java, BestellungServlet.java → Controller-Komponenten (MVC).

Scopes

Application-Scope**:** Speisekarte wird beim Start durch SpeisekarteListener geladen → global für alle Nutzer verfügbar.

Session-Scope**:** Bestellung eines Kunden → bleibt während seiner Sitzung erhalten.

Page-Scope**:** Wird in den JSPs für temporäre Daten genutzt (z. B. Anzeige der aktuellen Bestellung).

Ablauf in Kurzform

- Startseite zeigt Speisekarte (über Servlet oder Listener).
- Nutzer gibt Mengen und Kundendaten ein.
- *BestellungServlet* verarbeitet alles, erzeugt Objekte (Bestellung + Kunde) , speichert Session-Daten.
- Weiterleitung auf *rechnung.jsp* → dynamische Rechnungsausgabe
- Weiterleitung auf *kueche.jsp* → zeigt alle Bestellungen

JavaBeans

- Alle Modellklassen sind Beans-konform (private Felder, Getter/Setter, Standard-Konstruktor).
- Jede Bean implements Serializable.

Session-Scope

- Bestellung wird pro Nutzer in der Session gespeichert.
- Session auf null prüfen, um NullPointerExceptions zu vermeiden.

Page-Scope

- JSPs verwenden Page-Scope-Variablen (z. B. zur Anzeige einzelner Werte).
- Über request.setAttribute() Daten an JSP übergeben und dort mit ${pageScope.variable} ausgeben.

Application-Scope

- SpeisekarteListener lädt beim Start die gesamte Speisekarte und legt sie im *Application-Scope* ab.
- In der JSP abrufen mit ${applicationScope.speisekarte}.

Collection

- ArrayList<Speise> in Speisekarte.java
- ArrayList<Bestellposition> in Bestellung.java

Java-Konventionen

- JavaDoc-Kommentare über jeder Klasse und Methode.
- Einheitliche Einrückung (4 Spaces).
- Konstanten (z. B. MwSt-Satz) als `static final` deklarieren.

Gesamtdokumentation

Datei	Zweck	Kommentar-Empfehlung
startseite.jsp	Einstieg, zeigt Menü + Formular	Kurze Beschreibung oben im HTML-Kommentar
BestellungServlet.java	Controller für Bestellungen	JavaDoc: „Verarbeitet POST-Daten aus Startseite und erstellt Bestellung.“
Speisekarte.java	Modell der verfügbaren Speisen	JavaDoc: „Wird global im Application-Scope gehalten.“
Bestellung.java	Haupt-Bestellobjekt	JavaDoc + Methodenbeschreibung
kueche.jsp	Übersicht aller Bestellungen	HTML-Kommentar mit Zweck
rechnung.jsp	Dynamische Rechnung	Hinweis, welche Session-Attribute verwendet werden
web.xml, glassfish-web.xml	Konfiguration	XML-Kommentar oben mit Beschreibung der Rolle
SpeisekarteListener.java	Initialisiert Daten bei App-Start	JavaDoc oben hinzufügen