

Icerberg Lounge Online-Shop für Skiausrüstung

Milestone 1: Requirements Analysis & Conceptual Design

Der Kunde erstellt sein Useraccount im Webshop und gibt Informationen wie Namen, Adresse, Bankverbindung, Telefonnummer(optional), Benutzername, E-Mail, Passwort an. Der Kunde ist eine Person und bekommt eine PersonID zugewiesen. Jeder Kunde hat dann einen Benutzeraccount mit Passwort, Benutzername, E-Mail, Benutzer_ID und einem Guthaben.

Eine Bestellung wird mit einer Bestellnummer bzw. einem Bestelldatum beschrieben und besteht aus mehreren Bestellungsartikeln. Ein bestellter Artikel muss einer Bestellung zugewiesen werden. Bei einer Bestellung muss ein Artikel und die bestellende Person angegeben werden. Ein Artikel wird beschrieben durch eine Artikelnummer, Preis, Bezeichnung, Bilder-files und Produkttyp. Ein Warenkorb wird immer einer Bestellung bzw. einem Kunden zugewiesen. Der Warenkorb wird mit der Warenkorbnummer, den Artikeln und der Menge beschrieben.

Die Artikel werden von einem Verkäufer mit einer bestimmten Menge bereitgestellt. Der Verkäufer hat eine UID, Adresse, Namen und Branche. Die Artikel werden in einem Lager mit einer Lagernummer, Anschrift und Lagerkapazität gelagert.

Falls der Kunde seinen Artikel zurückgeben möchte, muss er eine Rücksendegenehmigung erhalten. Diese wird dann von einem Mitarbeiter bearbeitet. Ein Mitarbeiter ist eine Person und hat ein Einkommen und eine Sozialversicherungsnummer. Er kann auch einen Vorgesetzten haben.

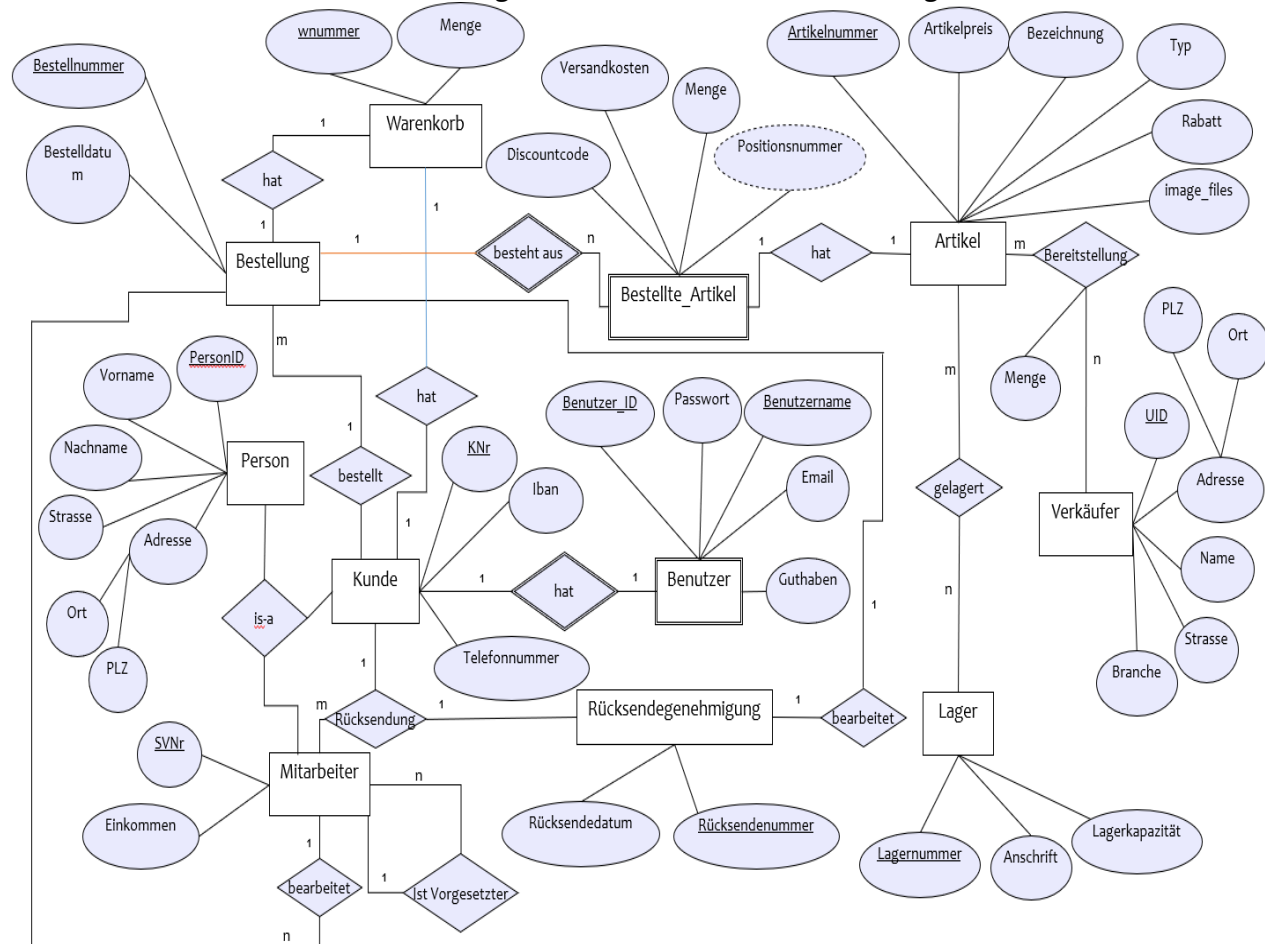


Figure 1: Entity Relationship Diagram

Milestone 2: Logical Desgin

Bestellung (Bestellnummer, Bestelldatum, Besteller_KNr, Warenkorb)

PK: Bestellnummer

FK: Bestellung.Warenkorb \diamond Warenkorb.wnummer

FK: Bestellung.Besteller_KNr \diamond Kunde.KNr

Bestellte_Artikel (Positionsnummer, Bestellnummer, Artikelnummer, Menge, Versandkosten, Code)

PK: Bestellnummer, Positionsnummer

FK: Bestellte_Artikel.Bestellnummer \diamond Bestellung.Bestellnummer

FK: Bestellte_Artikel.Artikelnummer \diamond Artikel.Artikelnummer

Warenkorb (wnummer, Artikelnummer_FK, Kunde_FK, Menge)

PK: wnummer

FK: Warenkorb.Artikelnummer_FK \diamond Artikel.Artikelnummer

FK: Warenkorb.Kunde_FK \diamond Kunde.KNr

Artikel (Artikelnummer, Preis, Bezeichnung, Typ)

PK: Artikelnummer

Rabatt (Typ, Rabattsatz, Rabattcode)

PK: Typ

Artikel_Bild (Art_Bildnr, Artikelnummer_FK, Files)

PK: Art_Bildnr

FK: Artikel_Bild.Artikelnummer_FK \diamond Artikel.Artikelnummer

Person (PersonID, Vorname, Nachname, Strasse, PLZ)

PK: PersonID

PLZ (PLZ, Ort)

PK: PLZ

Mitarbeiter (PersonID, Vorgesetzter, Einkommen, SVNr)

PK: PersonID, SVNr

FK: Mitarbeiter.Vorgesetzter \diamond Mitarbeiter.Personalnummer

FK: Mitarbeiter.PersonID \diamond Person.PersonID

Kunde (PersonID, KNr, Iban, Telefonnummer)

PK: PersonID, KNr

FK: Kunde.PersonID \diamond Person.PersonID

Benutzer (Benutzer_ID, Benutzername, Passwort, Email, Guthaben, Kundennummer_FK)

PK: KNr, Benutzer_ID, Benutzername

FK: Benutzer.Kundennummer_FK \diamond Kunde.KNr

Verkäufer (UID, Name, Branche, Strasse, PLZ)

PK: UID

Bereitstellung (Artikelnummer_FK, VKNr_FK, Menge)

PK: Artikelnummer_FK, VKNr_FK

FK: Bereitstellung.Aritkelnummer \diamond Artikel.Artikelnummer

FK: Bereitstellung.VKNr \diamond Verkäufer.UID

Lager (Lagernummer, Strasse, PLZ, Lagerkapazität)

PK: Lagernummer

Lagert (Artikelnummer, Lagernummer, Menge)

PK: Artikelnummer, Lagernummer

FK: Lagert.Artikelnummer \diamond Artikel.Artikelnummer

FK: Lagert.Lagernummer \diamond Lager.Lagernummer

Rücksendegenehmigung (RS_Nummer, RS_Datum, RS_Bestellnummer)
PK: RS_Nummer
FK: Rücksendegenehmigung.RS_Bestellnummer ◊ Bestellung.Bestellnummer
Rücksendung (MitarbeiterFK, Rücksendenummer, KundeFK)
PK: Personalnummer, Rücksendenummer, KNr
FK: Rücksendung.MitarbeiterFK ◊ Mitarbeiter.SVNr
FK: Rücksendung.Rücksendenummer ◊ Rücksendegenehmigung.Rücksendenummer
FK: Rücksendung.KundeFK ◊ Kunde.KNr

Milestone 4: Implementation

Java

Im Java-Teil wurde der DatabaseHelper vom Studentpackage als Referenz genommen und erweitert. Der DatabaseHelper erstellt eine Verbindung zur Datenbank und umfasst die Funktionen für SELECT, INSERT, UPDATE. Die Inserts erfolgen über eine PreparedStatement und werden mit Batch eingefügt.

Problem: beim Einfügen wirft die Datenbank die Fehlermeldung:

- **ORA-0100:** Maximale Anzahl offener Cursor überschritten
- Beim Einfügen von randomisierten Werten können manchmal doppelte Werte auftreten, deswegen feuert dann die Datenbank einen **Fehler bei einer UNIQUE CONSTRAINT**

Der RandomHelper dient nur zur Generierung von randomisierten Werten wie z.B. die Telefonnummer und die Sozialversicherungsnummer. Im RandomHelper werden auch CSV-Files (Name, E-Mail, IBAN, Straße, PLZ, Ort, usw.) gelesen und in entsprechende Arrays eingefügt. Er wurde auch erweitert, um Files mit Zahlenwerten zu lesen.

Zwei Tests wurden geschrieben:

- InsertTest, bei dem die Arrays vom RandomHelper genommen werden und dann mithilfe des Databasehelpers über eine Schleife die Werte eingefügt.
Die Schlüsselsetzung wurde von der Datenbank übernommen oder mit den inkrementierten Variablen in der Schleife eingesetzt. Um realistische Zusammenhänge zu
Problem: beim Insert in die Tabelle Benutzer bei der Generierung von Passwörtern sind diese nicht „encrypted“ und weisen keinen Hash auf -> Fehler beim LoginPage (login.php verifiziert nur „hashed“ Passwörter)
Das Datenvolumen ist in den Tabellen Personen, Mitarbeiter, Kunde, Benutzer und PLZ.
- SelectTest dient dann zur Überprüfung, ob alle Werte erfolgreich eingefügt wurden

PHP

Verglichen zum Java-Teil wurde die meiste Zeit in den PHP-Teil mitsamt der CSS-Dateien aufgewendet. Die Website wurde mithilfe von Bootstrap gestaltet und verwendet ein paar JavaScripts für Popups (LogIn, SignUp, Passwort ändern), Icons von FontAwesome und zwei CSS-Dateien für die Formatierung eine eigene geschriebene „style.css“ und von Bootstrap „styles.css“.

So sieht die Startseite der Iceberg Lounge aus. Man hat oben im Bild die Navigation Bar, bei der sich der User entweder einloggen oder anmelden kann. Beim Klicken auf den Home Button gelangt der Benutzer wieder auf die index.php Seite.

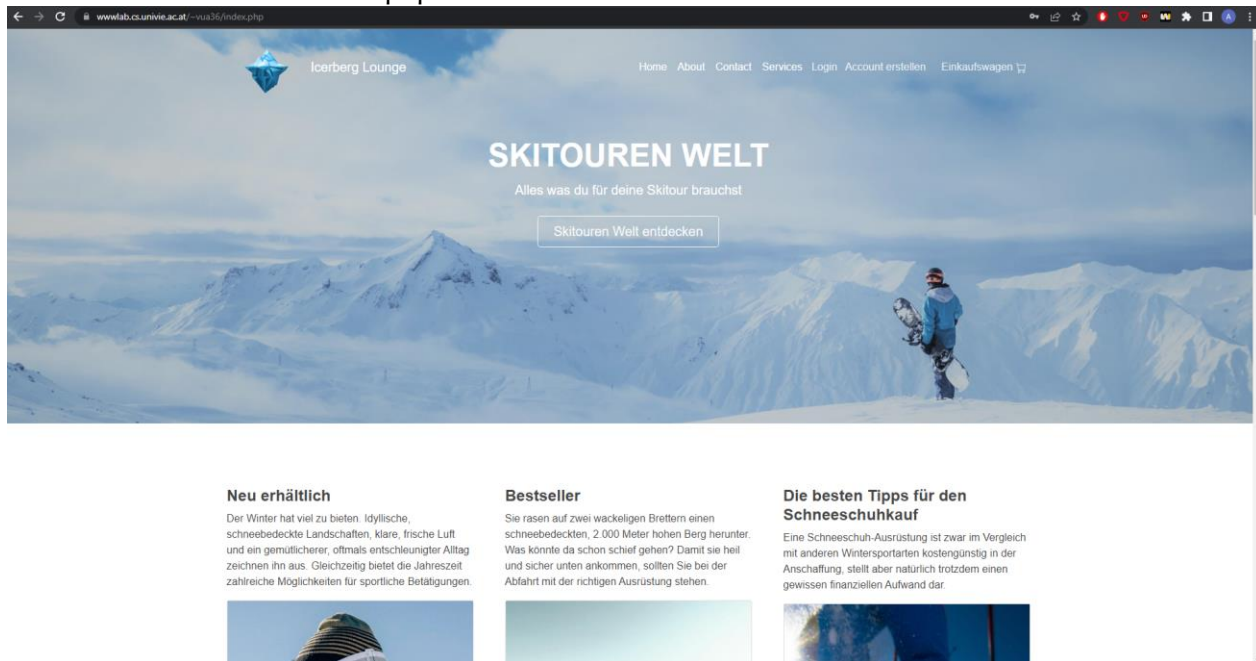


Figure 2: index.php

Falls der User nicht eingeloggt ist und auf den Einkaufswagen klickt, kommt landet er auf die Seite:

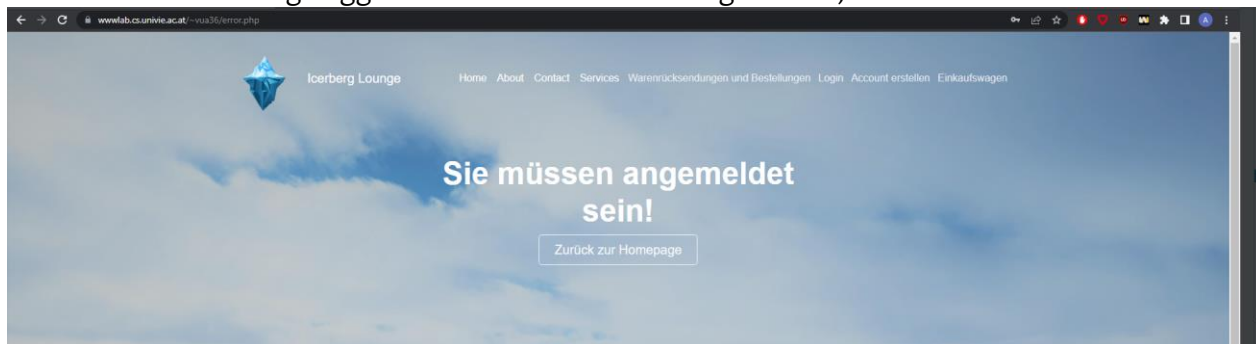


Figure 3: error.php

Wenn der User auf den Login Button klickt öffnet sich ein PopUp für das Login. Beim Eingeben von der Email und dem Password sendet die Seite die Daten zum login.php, welches dann die eingegebenen Inputs mit der Datenbank vergleicht. Hier wird das eingebene Passwort im login.php in einem Hashwert umgewandelt und dann mit den gehashten Passwort in der Datenbank verglichen.

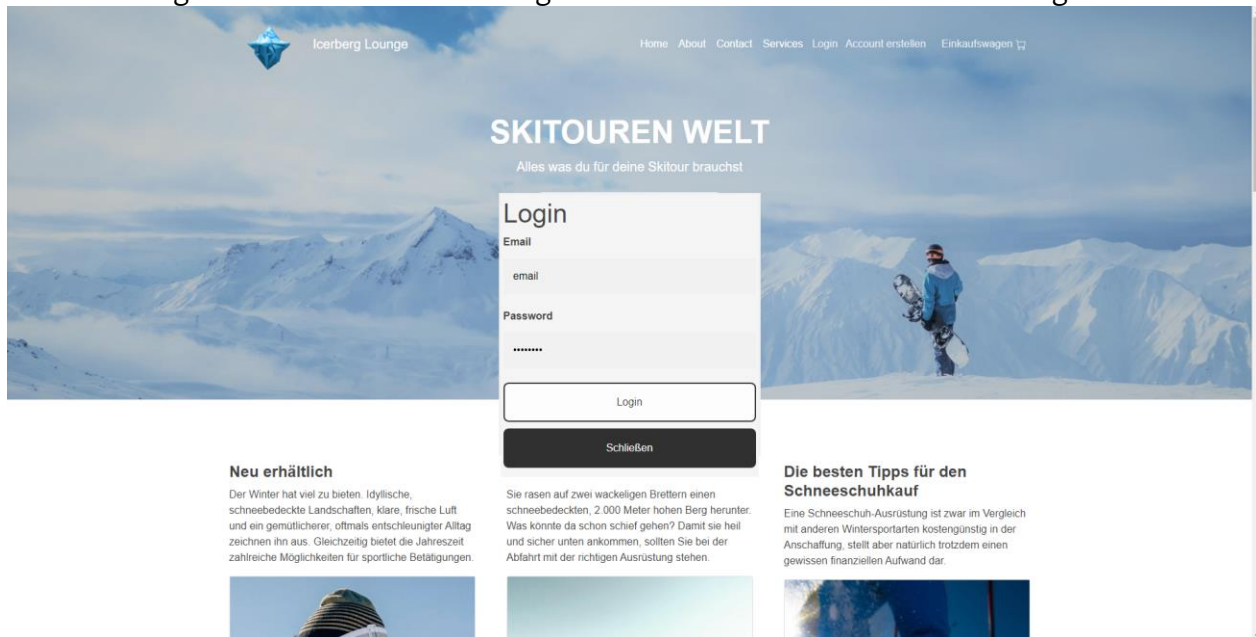


Figure 4: index.php mit LogIn PopUp

Falls sich der User nicht einloggen kann:

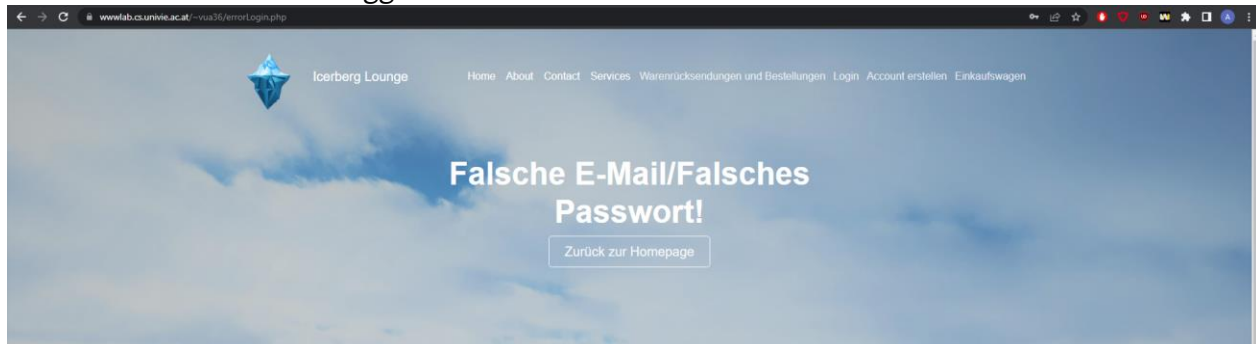


Figure 5: errorLogin.php

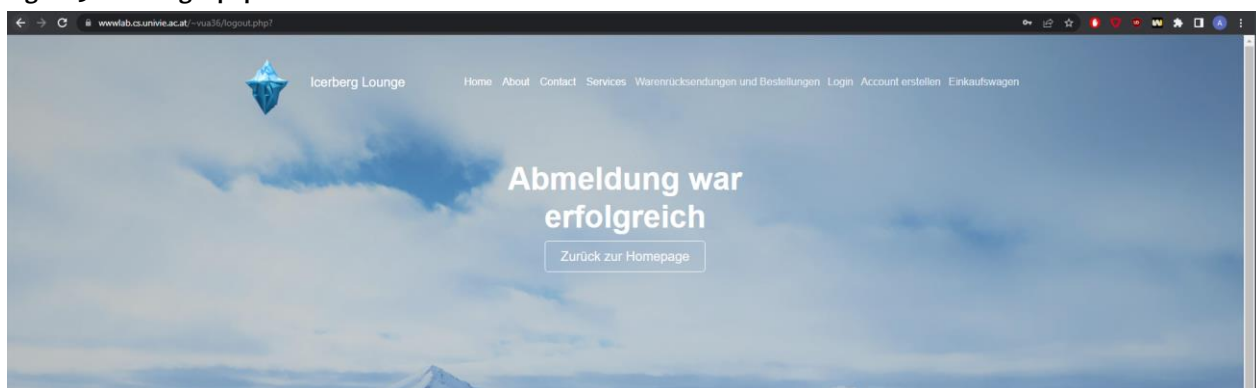


Figure 6: Abmeldung

Wenn der User auf den Button Account erstellen klickt, sieht er wieder eine PopUp für das SignUp. Beim Eingeben der Inputs, gelangt der User mit dem Klicken vom SignUp Button in die signup.php, wo dort die Inputwerte nochmals überprüft werden (Passwörter dieselben sind, E-Mail valide ist, Benutzername nicht vergeben ist, usw). Dort wird dann der User bei der Anmeldung in die Tabelle Person, Kunde und Benutzer eingefügt. Vor dem Einfügen in der Tabelle Person wird der Trigger gefeuert und überprüft, ob es den Vornamen mit dem Nachnamen und der Strasse schon gibt. Falls ja, lehnt der Trigger den Insert ab.

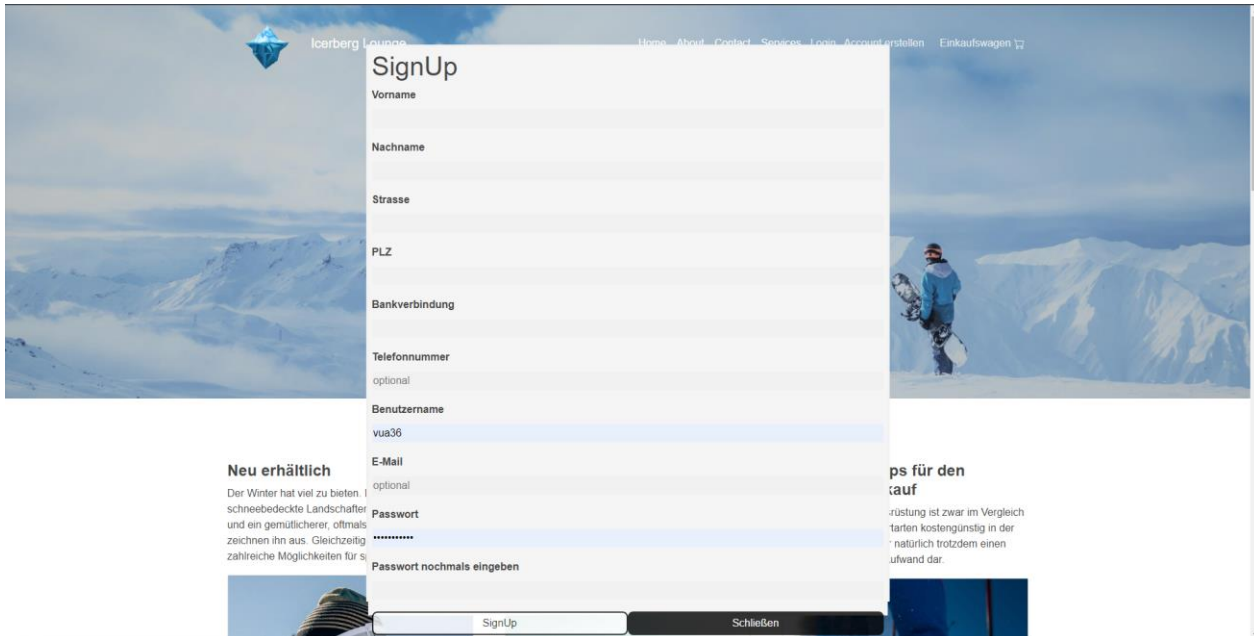


Figure 7: index.php mit SignUp PopUp

Beim Erfolgreichen SignUp bzw. LogIn wird der User auf die logged_in.php weitergeleitet. Der Benutzer sieht nun seinen Benutzernamen (falls der Cursor dort gelangt kann der User sein Passwort ändern oder seinen Account löschen), die Möglichkeit zum Abmelden und kann jetzt auch in den Einkaufswagen schauen.

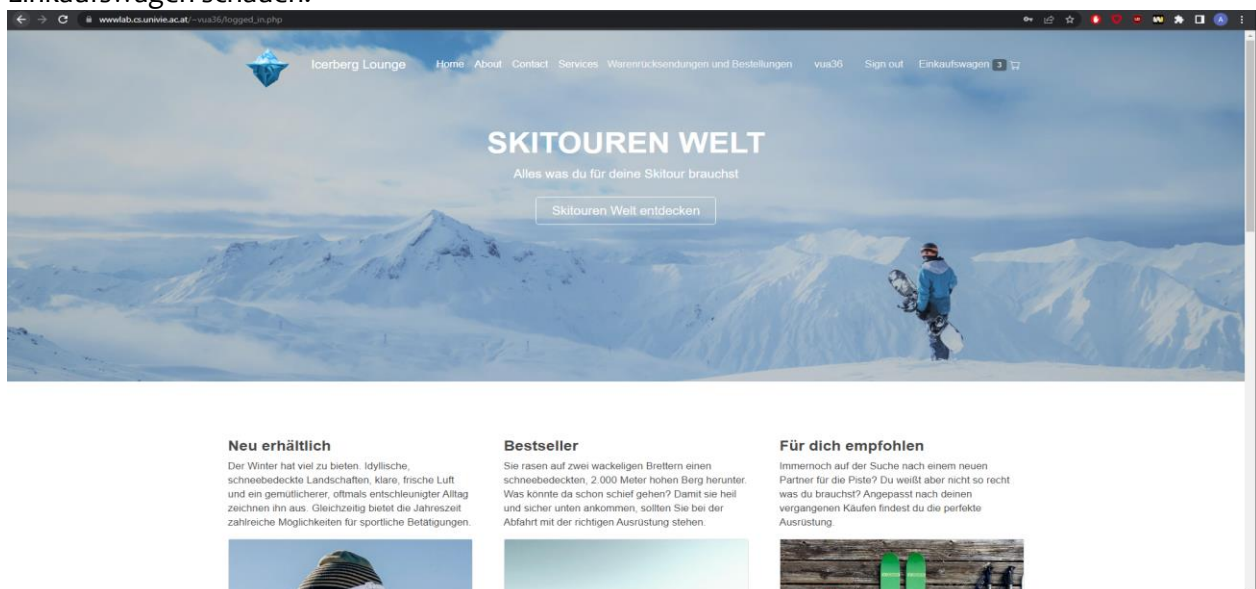


Figure 8: logged_in.php

Beim in den Jetzt entdecken Link bei der Bestseller Sektion gelangt der Benutzer auf die bestseller.php Seite. Dort sieht er jegliche Bestseller Produkte mit dem entsprechenden Preis und der jeweiligen Artikelbezeichnung. PS: beim durchgestrichenen Preis handelt es sich nicht um den Echtprice, da wurde der Originalpreis einfach um 10% erhöht.

Beim Klicken vom „Zum Einkaufswagen hinzufügen“ Button wird das Produkt zum Einkaufswagen hinzugefügt und der itemCount neben der ShoppingCart wird dementsprechend aktualisiert. Die Aktion addShoppingCart.php wird aufgerufen.

Beim Klicken vom „Vergrößern“ Button gelangt der User direkt auf die Produkt Seite, bei der er dann eine ausführlichere Übersicht vom Produkt hat.

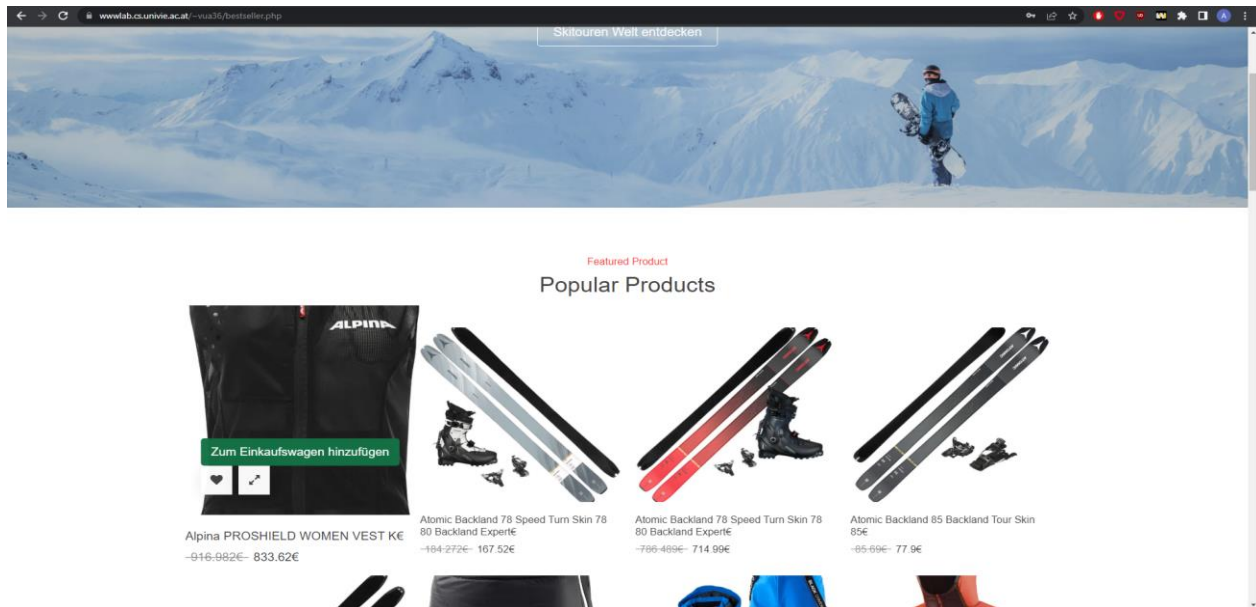


Figure 9: bestseller.php

Die Produktseite mit der Artikelnummer in der URL:

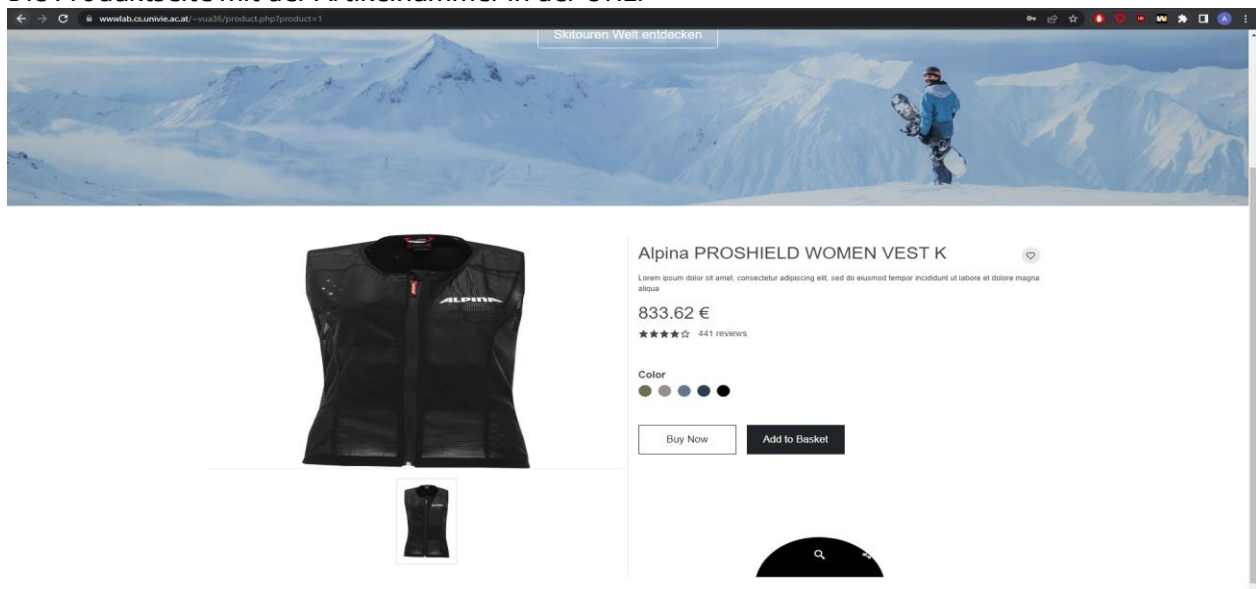


Figure 10: product.php

Die Searchfunktion in der Produktseite funktioniert noch nicht.

Beim ShoppingCart kann der Benutzer seine Artikel samt dem Preis und der Menge sehen. Er kann auf den „-“ Button klicken, welcher dann die Artikel in der Tabelle Warenkorb in der Datenbank updatet. Dabei erfolgt ein Stored Procedure, der die Menge der Artikel in der Tabelle Warenkorb checkt. Falls die Menge 0 ist wird der Artikel dann automatisch aus der Tabelle im Warenkorb gelöscht und der User sieht somit auch das jeweilige Produkt in seinem Warenkorb nicht mehr.

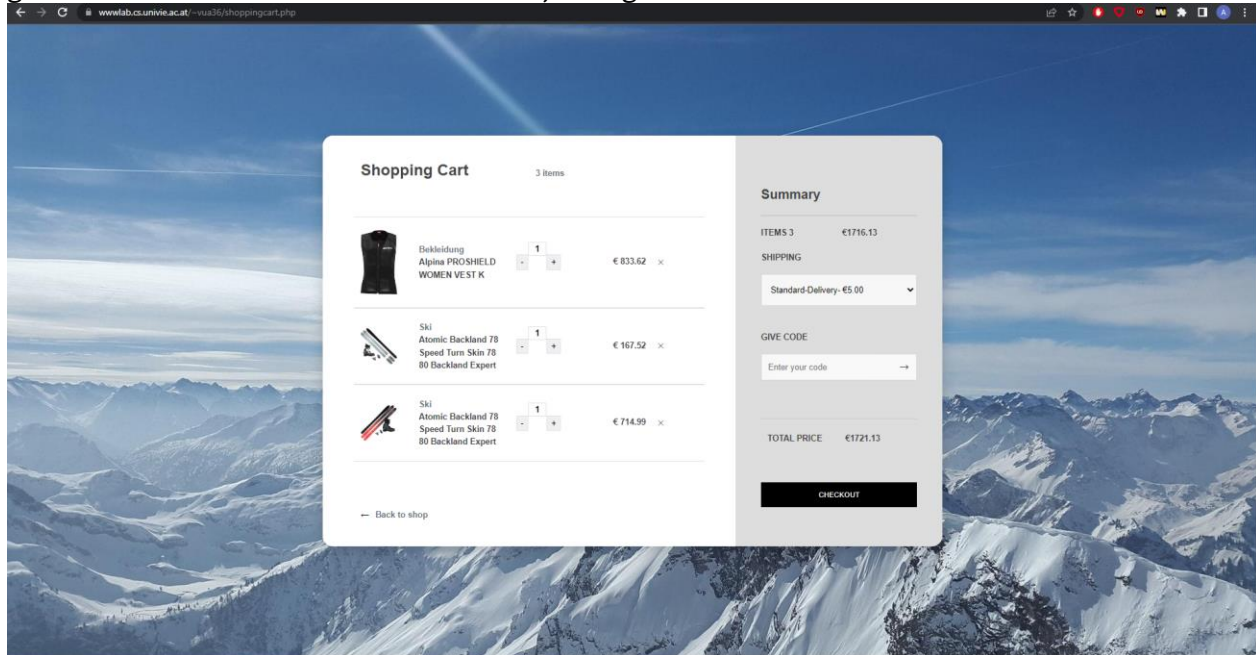


Figure 11: shoppingcart.php

Formatierung:

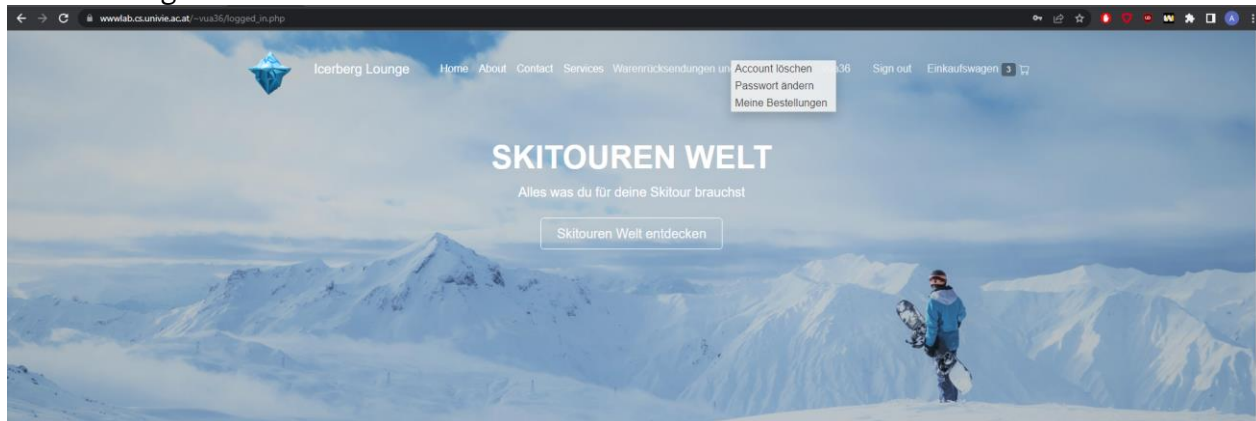


Figure 12: header.php

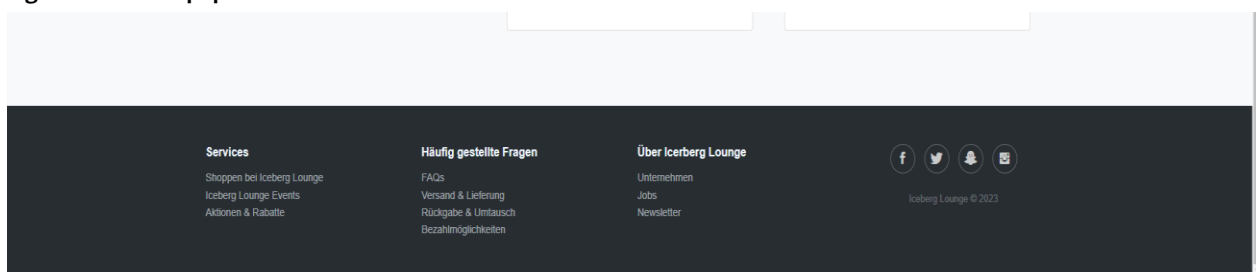


Figure 13: footer.php