Technische Hochschule Köln, Fakultät für Informatik und Ingenieurwissenschaften

Datenbanken 2 Praktikum SS2022 MS01 Gruppe\_47  
Projektdokumentation

Projektname:

Bücherwurm  
Ein interaktiver online Buchhandel

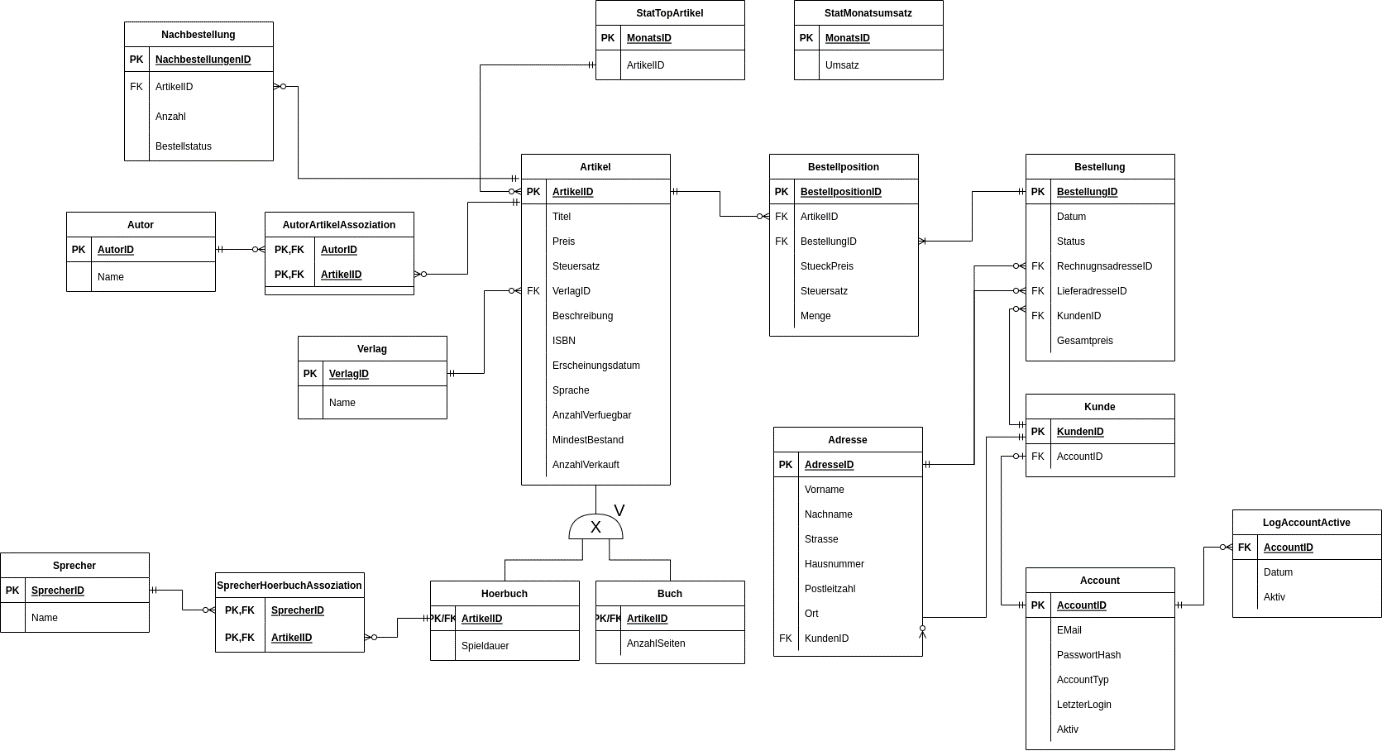
Ein Datenbankenprojekt von:

Dennis Kliewer (AI)  
Jesaja Storm (AI)  
Caner Özer (WI)  
Leo Petersberg (WI)  
Teamleiter:  
Sven Heiter (AI)

Betreuerin:   
Frau Petra Riemer

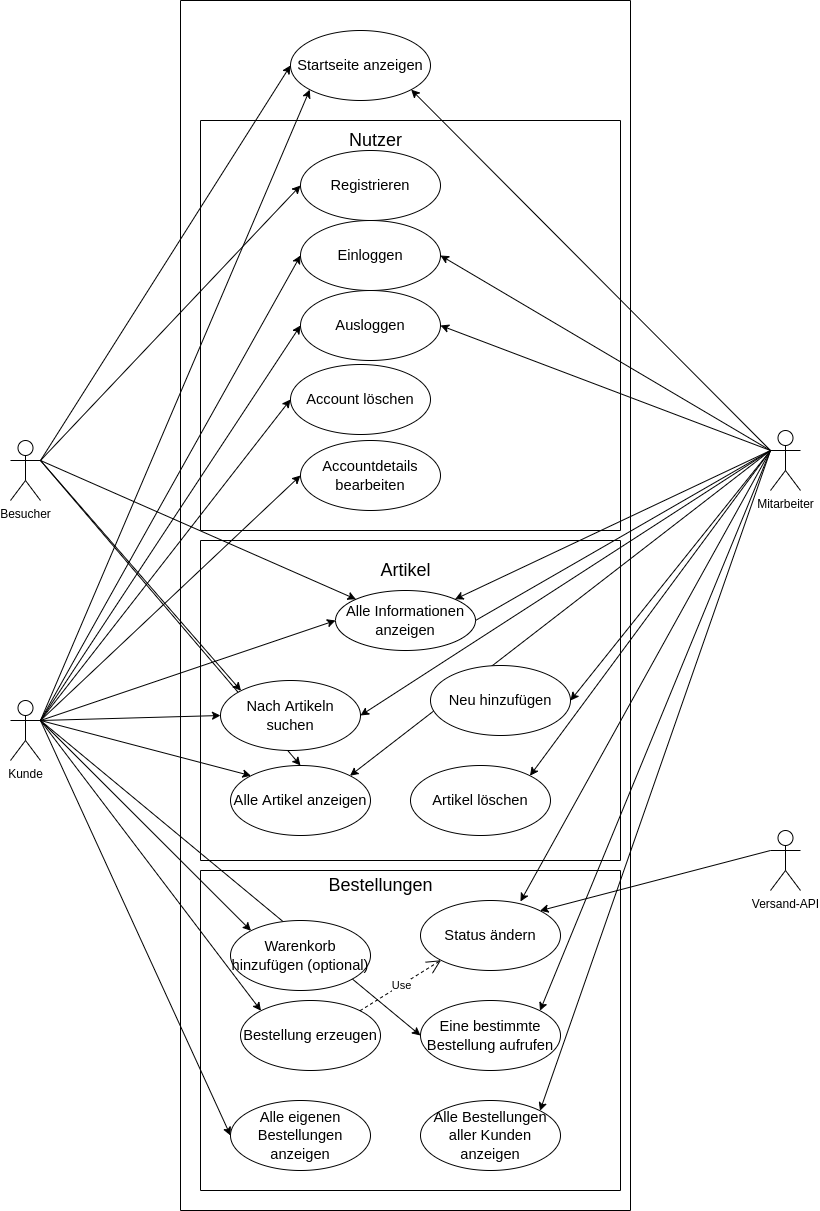
Projektidee

Mit „Bücherwurm“ verfolgen wir die Idee der Realisierung eines Online Buchhändlers in Form einer HTTPS Webseite. Auf dieser soll Kunden ermöglicht werden ganz einfach und bequem auf ihren Endgeräten Bücher, sowie Hörbücher bestellen zu können. Dabei kann jeder sich das Angebot unseres Sortiments anschauen und gezielt nach ihrem Wunschartikel suchen. Um einen Artikel zu kaufen ist es notwendig sich mit seinem selbsterstellten Benutzerprofil anzumelden. Über das personalisierte Benutzerportal bekommen Kunden Einsicht zu Details ihrer aktuellen und bereits abgeschlossenen Bestellungen.  
Aber auch \*\*Datenbankadministratoren/dem Personal\*\* soll die Benutzung des Systems leichtgemacht werden. Deshalb werden sowohl Datenbanktrigger, als auch Funktionen zum Einsatz kommen, um Prozesse, wie den Such-und Bestellvorgang des Kunden, als auch die Werksseitige, an der Marktnachfrage orientierten, Nachbestellung von Artikeln größtenteils zu automatisieren. Mehr zu diesen erfolgt in der weiteren Projektdokumentation.

Erweitertes Entity-Relationship-Model

Unser EERM mit Markierungen für ein fachliches Datenmodell

Unser Use Case Diagramm

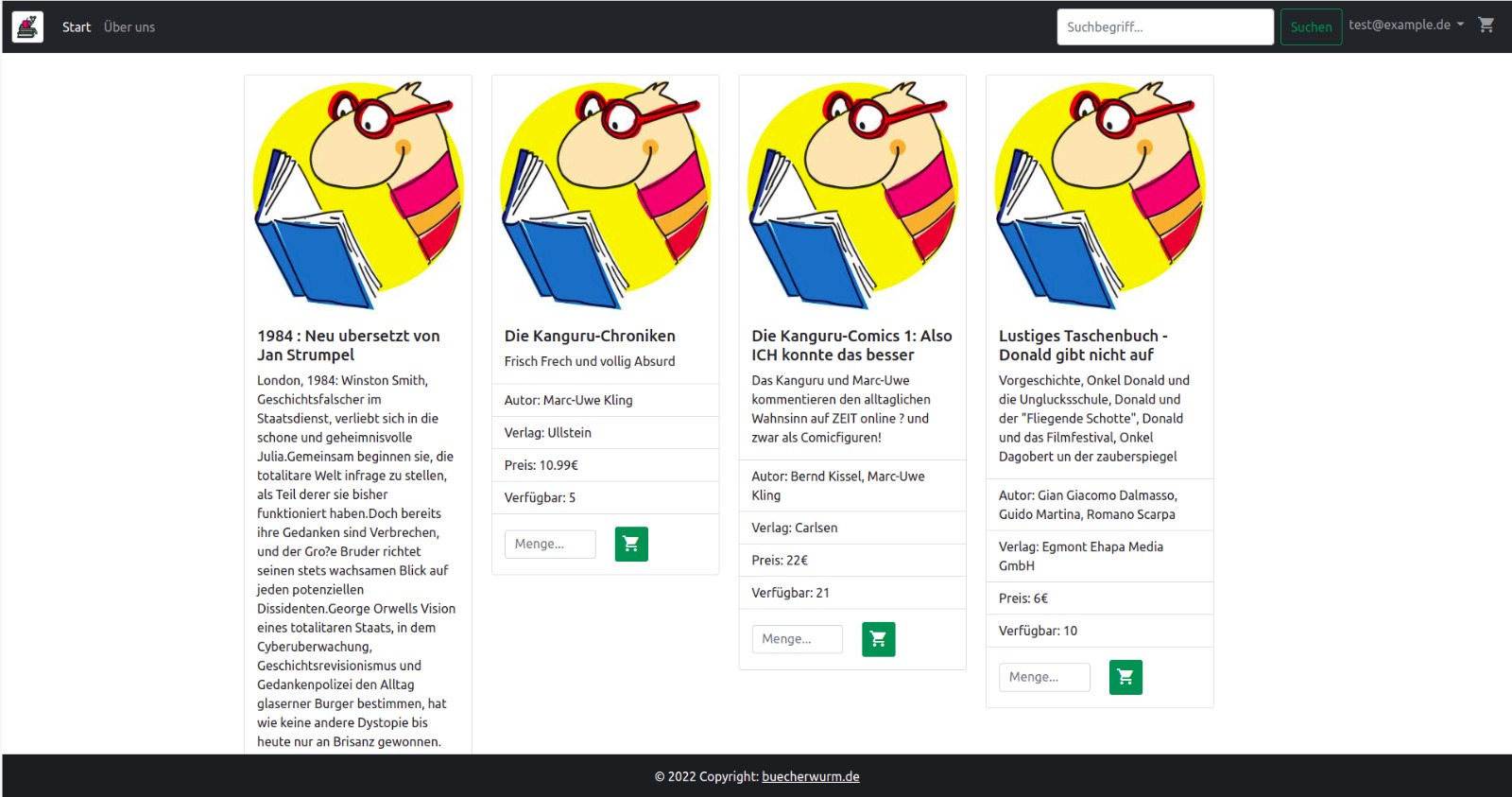
Wie sie ausfolgendem Diagramm entnehmen können, haben wir uns für 3 Benutzergruppen entschieden. Zum einen der Besucher der Bücherwurmseite, der unser Sortiment durchstöbern kann, dann der Kunde, der letzteren um die Kauffunktion erweitert und letztlich der Mitarbeiter, der volle Einsichtnahme genießt, jedoch nicht kaufen kann.

Datenbanktrigger- und Funktionen

|  |  |  |  |
| --- | --- | --- | --- |
| Art | Name in der Datenbank | Funktionalität | Erarbeitet von |
| Trigger | add\_price\_to\_order | Den Preis einer Bestellposition zur Bestellung hinzufügen BEI: Erstellung einer Bestellposition | Jesaja |
| Trigger | add\_to\_stock\_order | Waren werden der Nachbestellungstabelle hinzugefügt, um wieder über (+20%) den Mindestbestand gehoben zu werden  BEI: zu kleiner Stückzahl durch einen Kauf | Caner |
| Trigger | account\_logging | Gibt alle Inserts und Updates auf der Konsole aus  BEI: Erstellen oder Bearbeiten der Tabelle Account | Leo |
| Trigger | reduce\_stock | Den Bestand bei Bestellung verringern BEI: Aufgeben einer Bestellung (Bestellstatus von „editierbar“„offen“) | Sven |
| Trigger | archive\_deleted\_account | Von Benutzern entfernte Accounts werden archiviert BEI: Löschung eines Accounts durch den Kunden | Dennis |
| Funktion | Adjust\_minimum\_stock | Mindestbestand der Artikel anhand der letzten Verkaufszahlen des letzten Monats erhöhen/vermindern | Jesaja |
| Funktion | create\_surrogate\_key | Generiert einen Surrogate Key für die Bestellung, um an der Bestellnummer direkt erkennen zu können, wann die Bestellung aufgegeben wurde | Caner |
| Funktion | send\_emails | Sendet Marketing E-Mails an Kunden  BEI : langer Inaktivität | Leo |
| Funktion | show\_period\_revenue | Den Umsatz eines bestimmten Zeitraumes zurückgeben | Sven |
| Funktion | get\_gesamtpreis\_median | Gibt den Gesamtpreismedian für einen bestimmten Zeitraum zurück | Dennis |
| View | Buchinformationen | Gibt alle Daten rund um ein Buch aus (Artikel, Buch, Autor, Verlag) | Jesaja |

Mögliche zukünftige Features:  
-Warenkorb  
-View für alle Details rund um ein Hoerbuch

Unsere Webseite



Szenario 1: Artikel zum Warenkorb hinzufügen

1. Zuerst wird eine Menge für einen Artikel eingegeben, der dann zum Warenkorb hinzufgefügt wird
2. Der Browser sendet daraufhin eine Anfrage an den Webserver (PHP)
3. Der Server fragt mittels Funktion „getOrCreateWarenkorb“ bei der Datenbank, ob es bereits eine Bestellung mit der gegebenen Kundennummer (gespeichert in der User Session) und Bestellstatus „editierbar“ gibt. Falls nicht, wird eine angelegt und diese Bestellposition hinzufgefügt
4. Falls es diese Bestellung doch bereits gibt, wird durch eine weitere Funktion „addItemToWarenkorb“ aufgerufen, um die Bestellposition hinzuzufügen
5. Der User Wird letztlich automatisch zum Warenkorb weitergeleitet

Szenario 2 : Registrieren

1. Email und Passwort eingeben und Abfrage absenden
2. Die Eingaben werden vom „register-handling.php“ Skript verarbeitet
   1. Das eingegebene Passwort wird gehasht
   2. Mit der eingegebenen Email und gehashtem Passwort wird ein neuer Kundenaccount der Datenbanktabelle Account hinzugefügt
3. AccountID wird aus der Accounttabelle gelesen
4. Kunde wird mit der AccountID angelegt
5. KundenId wird ausgelesen
6. Kunde wird eingeloggt und KundenID wird in der Session gespeichert
7. Weiterleitung auf Startseite

Szenario 3: Einloggen

1. Im Loginfenster werden Email und Passwort eingegeben
2. Der Browser sendet diese Daten als Anfrage and den PHP Server, welcher diese mit „login-handling.php“ verarbeitet
3. Dieser holt sich das Passworthash und die AccountID aus der Accounttabelle, bei dem die Email übereinstimmt
4. Das eingegebene Passwort wird mit einer Hashfunktion („password\_verify“) geprüft
   1. Bei Fehlerhaftem Passwort wird der User zur Loginseite zurückgebracht und erhält eine Fehlermeldung
5. Bei Erfolg öffnet sich eine neue User Session (KundenID wird zwischengespeichert)
6. Last login Zeitstempel wird in der Datenbank hinterlegt
7. Der User wird zur Startseite weitergeleitet

Gefundene Probleme und ihre Lösung:

Problem 1:   
Sobald wir eine Bestellung erzeugt haben, wude sie dierekt als „offen“ markiert. Dadurch konnten wir nicht zwischen einer Bestellung unterscheiden, der noch Bestellpositionen hinzugefügt werden sollen und einer Bestellung, die tatsächlich aufgegeben wurde.

Lösung zu Problem 1:   
Die Einführung des Status „editierbar“, dadurch konnten wir Bestellungen erkennen, die noch vom Kunden verändert werden und solchen die von Kunden aufgegeben wurden.