

# IMSE Projektabgabe

## Meilenstein 3

### Team 12

Schöndorfer Roman

Nikitina Olena

Rabizo Daniil

Fuchs Andreas

26. Juni 2017

## Übersicht

### Programmiersprache

Java, Java Servlets, JSP, JSTL, SQL, Mongo Shell, Javascript

### Server

Apache Tomcat (ab Version 8)

### Datenbank

MongoDB (Meilenstein 2), MySQL (Meilenstein 1)

### Git Repository

<https://github.com/juksu/IMSE>

### Use-Case Verantwortliche

Roman Schöndorfer:	Benutzerkontos erstellen und ändern.
Olena Nikitina:	Produkt suchen und Verfügbarkeit prüfen.
Daniil Rabizo:	Administration im Backoffice (Produkt erstellen, Lagerstand korrigieren)
Andreas Fuchs:	Produkt bestellen, Bestellung abschließen und Bezahlung vornehmen.

## Kurzbeschreibung

Das Projekt hatte als Ziel das von Roman Schöndorfer in Meilenstein 1 spezifizierte Projekt Ethical European Business Fashion [e<sup>2</sup>BF] umzusetzen - siehe

das Dokument *UseCase-Spezifikation.pdf*. Um dies zu erreichen wurden die Use Cases aufgeteilt und von den jeweiligen Teammitgliedern implementiert.

Umgesetzt wurde das Projekt in Java unter Verwendung von Java Servlets und JSP auf einem Apache Tomcat Server laufend. Das Projekt baut auf einer bereits bestehenden Entwicklung aus Datenbanksystemen und Softwareengineering und wurde wo nötig angepasst und im Funktionsumfang vervollständigt.

Meilenstein 2 hatte MySQL als Datenbank, in diesem Meilenstein war es nun das Ziel das System auf eine MongoDB-Datenbank zu migrieren. Dies war auch mit einigen Schwierigkeiten verbunden auf die später noch mehr eingegangen wird.

Alle Sourcefiles sowie die Dokumentation sind auf dem git-Repository des Teams zu finden:

<https://github.com/juksu/IMSE>.

## MongoDB

Es war natürlich nicht ganz einfach in die andere Struktur und Abfragesprache von MongoDB kennen und verstehen zu lernen. Um zunächst einen groben Überblick über eine mögliche Datenbankstruktur zu bekommen wurde das Programm Mongify ([http://mongify.com/getting\\_started.html](http://mongify.com/getting_started.html)) zur Hilfe genommen. Jedoch war das Ergebnis recht enttäuschend da diese eine reine 1:1 Abbildung macht ohne die verschiedene Struktur von MongoDB und Json wie Embedded Documents, Referenzen,... sich zu Nutzen zu machen. So einfach ging es also dann nicht.

Ein weitere Schwierigkeit waren IDs. Da die von MongoDB generierte ID ein hexadezimal String ist stand zunächst die Idee im Raum die hexadezimal Zahl auf einen Ganzzahldatentyp zu übertragen. Jedoch ist kein Datentyp groß genug um die ID aufzunehmen. Den String als ID zu verwenden war ebenfalls keine Option da intern das Programm mit int operiert und String-Vergleiche auch nicht sehr effizient sind. Es wurde schließlich dazu entschieden eine zweite id in der Datenbank mitzuführen und in der Programmlogik sicher zu gehen dass diese einzigartig in der Collection ist.

Manche Entscheidungen hinsichtlich der Datenbankstruktur war schwierig zu treffen - manche weniger. So war die Entscheidung Bestellpositionen als embedded Document in Bestellung aufzunehmen schnell getroffen da schon im Relationalen Modell Bestellposition eine Weak Entity ist und davon ausgegangen werden kann dass es nur einige Bestellpositionen je Bestellung gibt. Dies war jedoch bei dem Verhältnis zwischen Bestellung und Produkt nicht der Fall - es kann für ein Produkt viele Bestellungen geben. Außerdem ist ein Produktdokument größer womit die Collection schnell anwachsen würde. Deswegen wurden hier Referenzen verwendet.

### Beispiel Bestellung:

```

{
  "_id" : -,
  "oid" : -,
  "datum" : -,
  "bestellstatus" : [...],
  "paypalTNR" : -,
  "kunde" : DBRef( "benutzerkonto", ObjectId() ),
  "bestellposition" : [
    {
      "produkt" : -,
      DBRef( "produkt" : ObjectId() ),
      "menge" : -,
      "preisprostueck" : -,
    },
    ...
  ]
}

```

**Algorithm 1:** Struktur der Collection *bestellung*

## Anhang

eeBF-Spezifikation.pdf: Spezifikation des Software Projekts mit Use-Case Beschreibung, UML-Struktur- und Verhaltensdiagrammen sowie Datenbankentwurf.

Meilenstein 1.pdf: Meilenstein 1 Dokument.

Arbeitsprotokoll.pdf: Gesammeltes Arbeitsprotokoll (nicht vollständig).

sql/: Ordner mit .sql files zum Anlegen und Verwalten der Datenbank.

mongodb/: MongoDB Konfiguration und einige Script Files.

eeBF/: Java Projekt Source Files.