BWL2 Praktikum

Alex Mantel, Daniel Hofmeister

3. Dezember 2014

Inhaltsverzeichnis

1 Erstes Praktikum

1.1 Wahl unserer Ware

Wir haben uns darauf geeinigt Spaceships, sprich Raumschiffe zu vertreiben. Diese werden aus seperat Lieferbaren Produkten wie beispiels weise Laserkanonen oder Warpantriebseinheiten zusammengesetzt.

1.2 Architekturübersicht

Wir brauchen als Komponenten eine Warenverwaltung, eine Kundenverwaltung, eine Rechnungsverwaltung und eine Businesslogic mit einer Web-GUI.

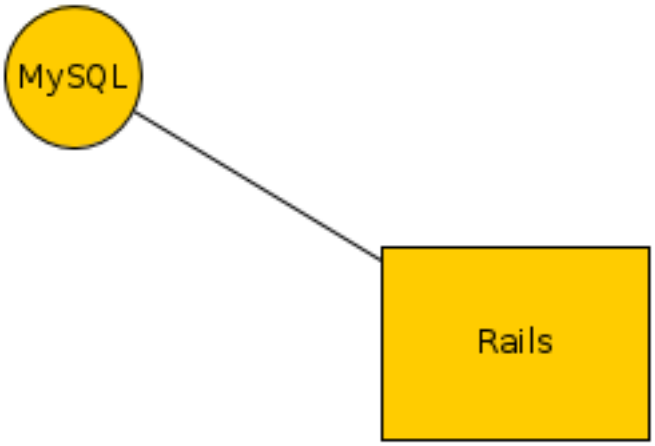
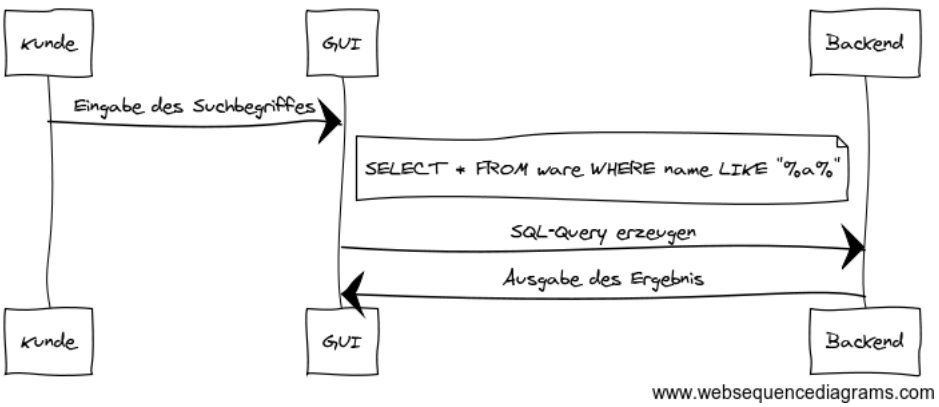


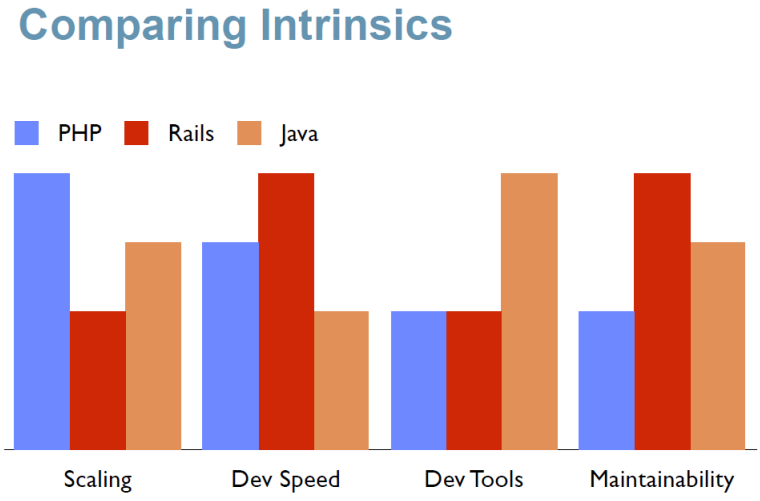
Abbildung 1: Übersicht

1.3 Sequenzdiagramm für eine Warensuche



1.4 Begründung der gewählten Tenchnologien

Zur Debate stand, welche Programmiersprache bzw. welche Scriptsprache, welchen Webserver und welches Datenbankmanagementsystem wir für die Entwicklung des Webshops verwenden. Zur Option stellten wir uns hier aufgrund der Bekanntheit Ruby on Rails und PHP.



[Bra]

Abbildung 2: Vergleich zwischen Rails und PHP

In Abbildung 2 zu sehen, ist ein Vergleich zwischen Java, Ruby on Rails und PHP. Wir werden aufgrund der Entwicklungsgeschwindigkeit, der Wartbarkeit und dem Grund, dass wir Ruby in Programmieren I verwendeten, Ruby on Rails verwenden. Offen bleibt nun, welches Datenbankmanagementsystem und welchen Webserver wir verwenden. Da Rails nativ einen Webserver bereitstellt, werden wir diesen verwenden. Die Anbindung an ein DMBS gestaltet Rails auch problemlos. Wir stellten uns SQLite und MySQL zur Option. Nach einigen Artikeln, welche diese Vergleichen fällt auf, dass MySQL eher für große Anwendungen geeignet sind, welche auf Skalierbarkeit und Performanz Wert legen. SQLite hingegen soll sehr gut für Prototypen von Datenbanken, eine schnelle Entwicklung geeignet sein. Hierbei legt SQLite keinen Wert auf Nutzerverwaltung und Skalierbarkeit. Nachteile von MySQL ist, dass es eine höhere Komplexität in der Einrichtung aufweist. Beide verwenden offensichtlicher weise SQL. Letztendlich haben wir uns für MySQL entschieden, da wir den Umgang mit einem schwergewichtigen DBMS üeben möchten. In Ruby on Rails werden wir Gems verwenden, welche kleine Erweiterungen für das System sind. Die verwendeten Gems sind `paperclip` und `mysql`.

1.5 Design der Datenbank

Wir wurden gebeten eine Ware zu vertreiben, welche aus anderen Waren zusammengesetzt werden kann. Dieses Modell wird dadurch eine Rekursion enthalten, da wir die Bauteile der Produkte eventuell ebenfalls vertreiben würden. Interessant ist also die Ware mit ihrem Namen, einer Beschreibung, einem Bild der

Ware und ihrer Zusammensetzung. Hierfür verwenden wir eine rekursive Relation von der Ware auf die Ware selbst.

1.6 Dokumentation

1.6.1 Installation der Software Komponenten unter Arch Linux

Das Installieren der Softwarekomponenten hat sich unter Linux als äußerst einfach erwiesen. Unter der Distribution Arch Linux musste man zunächst MySQL, nodejs, ruby und die Gems von Ruby installieren. Die ersten drei waren mit dem Befehl `sudo pacman -S mysql nodejs ruby` abgehandelt. Für MySQL mussten wir zusätzlich `mysql_secure_installation` eingeben um das Passwort von root zu ändern. Nun muss man noch den Daemon mit `sudo systemctl start mysqld` starten. Wenn man möchte, dass der Daemon beim nächsten hochfahren des Rechners automatisch startet, gibt man zusätzlich `sudo systemctl enable mysqld` ein. Für die abschließende Installation der gems nutzten wir `gem install mysql rails paperclip`.

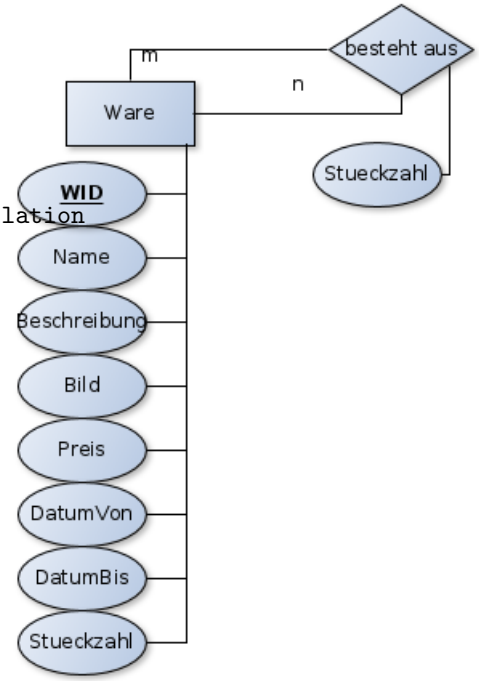


Abbildung 3: ERM der Ware und ihrer Struktur

1.6.2 Umsetzung der Datenbank

Um eine Erzeugung einer Datenbank wird sich in nur indirekt gekümmert, da Rails die Anbindung an Rails vollständig übernimmt. Ebenfalls arbeitet Rails intern mit Relationen, weshalb keine Fremdschlüssel in der Datenbank auftauchen. Desweiteren gestattet uns Rails durch `belongs_to` und `has_many` in den Modellen diese Relationen zu gestallten. Bilder werden in unserer Datenbank mittels Name, Typ, Größe und dem letzten Update gespeichert. Für die Verwaltung dieser haben wir uns für das gem Paperclip entschieden.

1.6.3 Nutzeroberfläche

Die Nutzeroberfläche haben wir uns durch Rails generieren lassen. Eine Validierung der Daten bei der Warenerzeugung findet in den Modellen statt. Wenn diese fehlschlägt, wird der Nutzer aufgefordert seine Eingabe zu korrigeren. Die Validierung überprüft lediglich ob die Daten für das Anlegen neuer Waren vorhanden sind.

Literatur

[Bra] Tim Bray. Issues of web frontends.