„EinkaufsApp“
Angebot

Informationen zum Angebot

An	Frau Prof. Dr. Wieland Gustav-Freytag-Str. 43-45, 04277 Leipzig
Über	Projekt EinkaufsApp
Von	EinkaufsApp

Vorgelegt von	
Projektleiter	Markus Hube
Gruppenmitglieder	Huong Dang Thomas Elias Viktor Fuchs Florian Graupeter Jannis Grohs Michael Hein Moritz Karsten Sebastian Kiepsch Annika Köstler Daniel Sawadenko Moritz Schaub Florian Schmitt Eric Sorgalla
E-Mail	markus.hube@hft-leipzig.de

Inhaltsverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

Abb.	Abbildung
PHP	Hypertext Preprocessor
HTML	Hypertext Markup Language
HfTL	Hochschule für Telekommunikation Leipzig
DWI13	Duales Studium Wirtschaftsinformatik Jahrgang 2013
JS	JavaScript
CSS	Cascading Style Sheets
App	Anwendungsprogramm
EAN	European Article Number
OS	Operating System
iOS	iPhone Operating System
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
Jade	A Rendering Engine

1. Problembeschreibung

Aufgrund der steigenden Angebotsvielfalt von Produkten und Dienstleistungen und dem daraus resultierenden wachsenden Kaufinteresses der Konsumenten, gestaltet sich das Nachverfolgen vergangener Einkäufe immer schwieriger und komplexer.
Die Analysen sind vor allem wichtig, um zukünftige Einkäufe besser planen und sein Ausgabeverhalten besser beschränken zu können.

Eine einfache Rechnung verdeutlicht dies:

Geht man von einem Wocheneinkauf einer Drei-köpfigen Familie von ungefähr 200,00 Euro aus, würden sich die Ausgaben auf 10 400,00 Euro pro Jahr belaufen. Diese können durch gezielte Sparmaßnahmen eingeschränkt werden. Dazu gehören z. B. das Kaufen von Alternativmarken gleicher Produkte zu einem günstigeren Preis oder Beachtung von Sonderangeboten.

Allein eine Ersparnis von zehn Euro pro Woche würden die Jahresausgaben um 520 Euro senken. Das Sparpotential ist darüber hinaus noch erweiterbar. Deswegen ist eine gute Einkaufsplanung wichtig.

Der damit einhergehende zeitliche Aufwand durch eine manuelle Analyse ist im heutigen digitalen Zeitalter nicht mehr notwendig. Mehr als 50 Prozent der Einwohner Deutschlands[1] besitzen ein Smartphone und haben somit die Möglichkeiten mit Hilfe von Apps z.B. Einkäufe einfacher zu analysieren.

Die im Angebot vorgestellte EinkaufsApp soll dem Nutzer die Möglichkeit bieten seine Einkäufe aufzuzeichnen, sie nachzuverfolgen und schlussendlich durch unterschiedliche Auswertungsoptionen zu analysieren und zu optimieren.
Dabei liegt der Fokus darauf, dass nicht nur der eigene Einkauf verwaltet werden kann, sondern auch Gruppeneinkäufe. Was wiederum bedeutet, dass im Laufe eines Einkaufs, für z. B. eine Wohngemeinschaft, die einzelnen Gruppenmitglieder ihren jeweiligen Artikel direkt zugewiesen bekommen können.

[1] http://de.statista.com/statistik/daten/studie/198959/umfrage/anzahl-der-smartphonenutzer-in-deutschland-seit-2010/, zuletzt abgerufen 09.12.2015, 11:37 Uhr

2. Beschreibung der Applikation

Wie bereits erwähnt wird für die Erleichterung des alltäglichen Einkaufserlebnisses die EinkaufsApp konzipiert. Ein Anwendungsprogramm, die das Tracken des finanziellen Aspekts vergangener Einkäufe erleichtert, um somit Sparpotentiale besser zu erkennen.

Mit der angebotenen Lösung werden hauptsächlich Privatnutzer angesprochen und die Nutzung für Geschäftskunden ist aufgrund des Anwendungsbereiches nicht vorgesehen.

Die Hauptfunktionen der App besteht im Wesentlichen aus einer Einkaufs-, Marktauswahl,- Nutzerverwaltungs- und Auswertungsfunktion.

Der Nutzer kann seine gekauften Artikel eines Einkaufs, via eines implementierten Barcodescanners, in eine Liste einpflegen und speichern.

Des Weiteren ist eine Gruppenverwaltung vorgesehen, die es ermöglicht Gruppen zu erstellen und Nutzer diesen zuzuordnen. Die vorher vom Nutzer eingescannten Produkte können den einzelnen Gruppen und anschließend den einzelnen Nutzern individuell zugeordnet werden. Am Ende kann jedes Gruppenmitglied in der Auswertungsfunktion seinen finanziellen Ausgabenstand einsehen.

Durch die Funktion der Auswertung ist es dem Verbraucher möglich, die Kaufgewohnheiten zu analysieren und entsprechend auszuwerten.

Folgende Möglichkeiten stehen zur Auswahl:

1. Auswertung nach Kaufhäufigkeit eines Artikels
2. Auswertung von Einkäufen in einem bestimmten Zeitraum
3. Auswertung der Ausgaben innerhalb von Gruppeneinkäufen

Somit kann der Nutzer nun individuell sein eigenes Kaufverhalten analysieren und optimieren.

3. Aussichten

Die EinkaufsApp soll in der ersten Phase der Entwicklung lediglich für Android-fähige Endgeräte verfügbar sein. Aufgrund des hybriden Entwicklungsstils ist eine Implementierung auf iOS und Windows basierten Betriebssystemen ohne großen Aufwand möglich. Ein passendes Framework ist dementsprechend dafür vorgesehen.

Zukünftig, das heißt nach dem Release der EinkaufsApp1.0, ist geplant, dass in der App ein Auswertungsmechanismus implementiert wird, der es dem Verbraucher im ersten Schritt ermöglichen soll, ein vergleichbares Produkt im selben Geschäft und im zweiten Schritt, ein vergleichbares Produkt in diversen Geschäften im Umkreis zu finden. Basierend auf den Kaufgewohnheiten des Nutzers sollen künftig individuell zugeschnittenen Einkaufslisten von der App generiert werden können.

Über die Einführung eines kostenpflichtigen Updates soll zukünftig nachgedacht werden. Um dies zu realisieren, sollen zusätzlich diverse kleinere Optimierungen hinzugefügt und eine Erweiterung der Auswertungsfunktion vorgenommen werden.

.

4. Systemarchitektur

Um die oben genannten Funktionalitäten der Applikation umzusetzen, müssen die genutzten Tools und Methoden klar definiert werden. Im weiteren Verlauf wird zunächst die generelle Struktur der Systemarchitektur beschrieben und dann auf die einzelnen Begrifflichkeiten eingegangen.

Der generelle Aufbau des Softwaresystems sieht folgendermaßen aus.

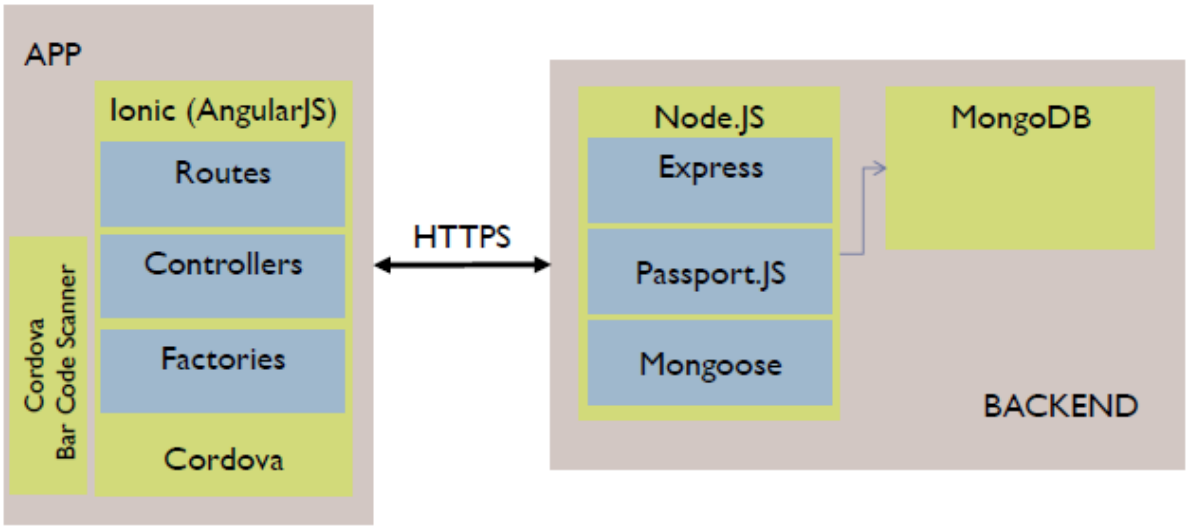

Abbildung 1: Systemarchitektur

Wie in Abbildung 1 ersichtlich, besteht das System aus zwei Komponenten: der eigentlichen App auf Basis des Ionic Frameworks, sowie einem Backend inklusive einer Datenbank.

Beide Komponenten orientieren sich an einer Model – View – Controller Architektur. Zwischen den Komponenten findet eine Kommunikation mit HTTPS statt.

In beiden Komponenten wurden existierende Frameworks zur Entwicklung von mobilen, hybriden Apps mit JavaScript bzw. der Implementierung von Webservern auf Basis von Node.JS verwendet. Mit dieser Programmiersprache wurde ein hohes Augenmerk auf die Betriebssystemneutralität der entstehenden Anwendungen gelegt.

Insbesondere Node.JS eignet sich durch seine Skalierbarkeit und den von Haus aus geringen Overhead besonders gut für Anwendungen, welche wenig auf betriebssystemspezifische Ressourcen zugreifen müssen, sowie hoch performante Webapplikationen. Da Node.JS single-threaded ausgeführt wird, eignet es sich besonders gut für Input-Output-lastige Anwendungen wie z. B. WebAPIs mit geringem Rechenaufwand.

In der App Komponente wurde ein Framework (Ionic Framework) gewählt, welches eine große Palette an vorgefertigten Design Elementen mitliefert und damit den Entwicklungsaufwand erheblich verringert. Das Ionic Framework liefert, neben diesen Elementen, auch ein logisch strukturiertes Programmiermodell, indem es die Komponenten von AngularJS (Routing, Controllers, Factories sowie Services) verwendet. Das Basis Framework, welches die Verbindung zu den Ressourcen des mobilen Betriebssystems (iOS oder Android) herstellt, ermöglicht in dieser App insbesondere das Hinzufügen von Plugins.

4.1 Backend

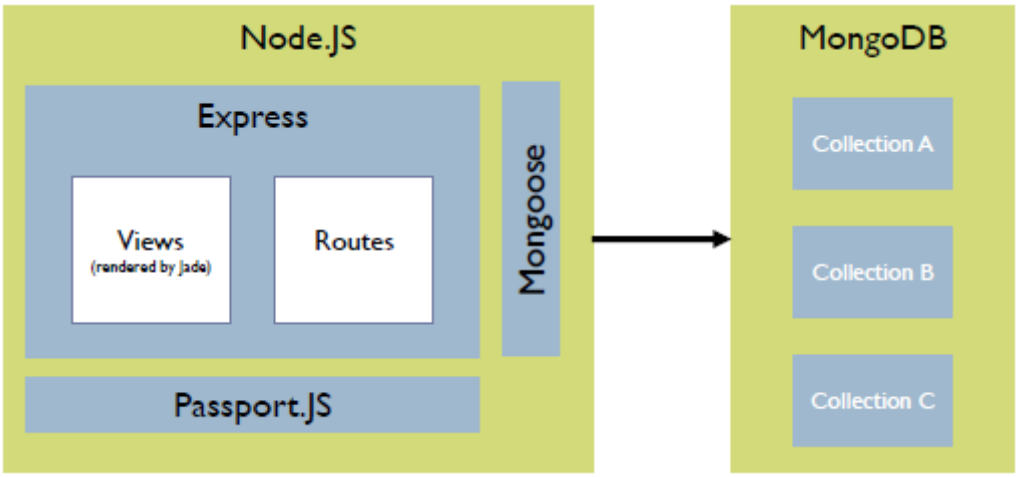

Abbildung 2: Aufbau des Backend

Als Backend wird in diesem Projekt jeglicher serverseitig ausgeführter Code, sowie alle dort befindlichen Dritt-Applikationen bezeichnet. Konkret gliedert sich das Backend in zwei grobe Komponenten auf.

Die Node.JS basierte Webapplikation stellt mithilfe des Express.JS Webframeworks jene Endpunkte bereit, welche durch die mobile Applikation konsumiert werden können. Routen definieren Funktionen, welche bei einer HTTP-Anfrage an einen definierten Endpunkt ausgeführt werden. In der, im Rahmen dieses Projektes erstellten, API ist diese Funktion zumeist das Ausführen einer Datenbankoperation und das anschließende Antworten des Servers auf die Anfrage. Die Antwort erfolgt dabei immer in Form eines JSON Arrays, welches dann vom Empfänger interpretiert werden kann. Um das Produkt zu präsentieren,sowie eine Verwaltung von Login und Registrierung über ein Webinterface außerhalb der mobilen App verfügbar zu machen, wurden Views implementiert. Views sind Ansichten, welche mithilfe der Rendering Engine „Jade“ dynamisch generierte Seiten im HTML Format als Antwort vom Server zum Client senden können.

Um die Endpunkte der Anwendung vor unbefugten Zugriff zu schützen, wird Passport.JS verwendet. Dieses Node.JS Modul ermöglicht es eine Authentifizierung an der API durchzuführen, welche auf verschiedene Authentifizierungsprovider zurückgreifen kann. In diesem Projekt wurde die lokale Benutzerauthentifizierung auf Basis von in einer Datenbank gespeicherten Credentials gewählt. Die Authentifizierung wird mittels eines temporären Browser-Cookies auf dem Client persistiert.

Weiter oben angesprochene Datenbankoperationen werden mithilfe von Mongoose, eines Daten-Modelling-Tools für Node.JS und MongoDB, durchgeführt. Zuvor wurden in Mongoose Modelle definiert, welche bei Ausführung der Applikation automatisch dem Modell entsprechende Collections in der Datenbank anlegt. Diese Modelle sind in den Routen der Applikationen zugreifbar und können zur Manipulation der Daten in der Datenbank verwendet werden. In den meisten Fällen wurden alle CRUD (Create – Read – Update – Delete) Funktionen als Endpunkte abgebildet (siehe API Dokumentation).

Eine MongoDB als NoSQL Datenbank bildet die zweite Komponente des Backends ab. Die besondere Datenmodellflexibilität ermöglicht eine stark iterativ fokussierte Entwicklungsweise, was dem engen Entwicklungszeitfenster in diesem Projekt zu Gute kam. Auch die hohe Skalierbarkeit und die einfache Implementation, mittels des Tools Mongoose, waren ausschlaggebende Gründe für die Wahl dieser Datenbank.

Um diese Anwendung mit Internet zu hosten wurde OpenShift ausgewählt. Die Cloudplattform von Red Hat bietet im ersten Preismodell eine kostenfreie Möglichkeit Serveranwendungen zu hosten.

4.2 App

In der App Komponente findet das Programmierparadigma MVC in der Umsetzung des Angular JS Frameworks Anwendung, welches die Softwarebasis für die Elemente von Cordova und somit des Ionic Frameworks ist. Im Code wird hierbei zunächst mit sogenannten „Factories“ eine klassenähnliche Struktur geschaffen, welche es ermöglicht mit den definierten Endpunkten des Backends zu kommunizieren. Routen legen die App-internen Endpunkte fest. Diesen definierten Endpunkten werden konkrete Funktionen und Ansichten über Controller zugewiesen. Der Controller ist in dieser Position das Bindeglied zwischen den Datenmodellen und dem View. Die Adaption dieser Begrifflichkeiten auf das MVC-Modell wird in Abbildung 2 zur Verdeutlichung aufgezeigt.

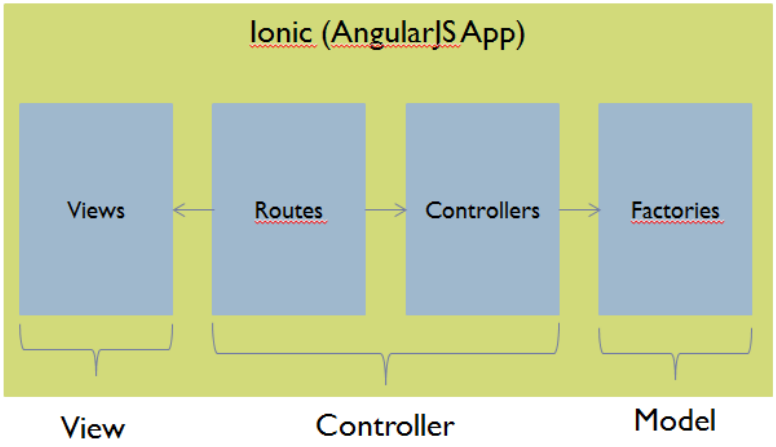

Abbildung 3: Adaption von Model-View-Controller in der App

5. Quellen

Zuletzt geprüft am 19.12.2015

- Openshift Features- https://www.openshift.com/features/
- Ionic - http://ionicframework.com/
- NodeJS - https://nodejs.org/en/
- Why MongoDB - https://www.mongodb.com/blog/post/why-mongodb-popular
- AngularJS - https://de.wikipedia.org/wiki/AngularJS