HTL - Perg
Höhere Abteilung für EDV und Organisation

Diplomarbeit

Mobile Dashcard

(Mobile Anwendung zur Darstellung von Kennzahlen)

Projektteam: Thomas Hasl
Theodor Seiser
Projektbetreuer: Dipl.-Ing. Michael Stumpfl

In Zusammenarbeit mit Logicx consulting & workflow integration GmbH
Betreuer: Ing. Robert Stenitzer, Florian Bachinger

Bearbeitungszeitraum: 01.10.2014 – 13.05.2015

Eidesstattliche Erklärung

Hiermit versichern wir, die vorliegende Arbeit selbstständig, ohne fremde Hilfe und ohne Benutzung anderer als von den von uns angegebene Quellen angefertigt zu haben. Alle Stellen, welche wörtlich oder sinngemäß aus fremden Quellen direkt oder indirekt übernommen wurden, sind als solche gekennzeichnet.

Perg, _____ Unterschrift _____

Thomas Hasl

Perg, _____ Unterschrift _____

Theodor Seiser

Danksagung

Wir bedanken uns recht herzlich bei allen Personen, die uns bei der Erstellung dieser Diplomarbeit unterstütz haben.

Besonderer Dank gilt dabei unserem Betreuungslehrer Professor Dipl.-Ing. Michael Stumpfl, der uns tatkräftig bei der schriftlichen Dokumentation unserer Arbeit geholfen hat.

Hervorzuheben ist auch die exzellente Zusammenarbeit mit den Mitarbeitern der Logicx GmbH, welche uns mit ihrer wertvollen Zeit und Energie beiseite standen. Besonders zu erwähnen sind die beiden Ansprechpersonen Robert Stenitzer und Florian Bachinger, die bei jeder Frage schnelle und kompetente Antworten lieferten.

Zusammenfassung

Mobile Dashcard ist eine Diplomarbeit der Höheren Technischen Bundeslehranstalt für Informatik in Perg, die im Zuge der Matura im Jahr 2015 verfasst wurde.

Mobile Dashcard (MDC) ist eine mobile Anwendung, welche vom Auftraggeber verwendet wird, um projektbezogene Kennzahlen anschaulich darzustellen. Sie ermöglicht dem Personal des Unternehmens eine Gegenüberstellung der geplanten und tatsächlich aufgewendeten Personentage für Projekte und Meilensteine. Zusätzlich kann mit Hilfe der App eine graphische Darstellung der Zusammensetzung der Aufwände eingesehen werden. Darüber hinaus bietet die Anwendung Filtermechanismen für die Auflistungen der Projekte, Meilensteine und Issues.

Die Applikation steht in hybrider Form zur Verfügung, wodurch sie auf den Plattformen Android, iOS und Windows Phone installiert und verwendet werden kann.

Abstract

Mobile Dashcard is a diploma thesis of the HTL in Perg, written as part of the Matura in 2015 in cooperation with the company Logicx consulting & workflow GmbH.

Mobile Dashcard (MDC) is a mobile application developed with the aid of Cordova and the use of standard Web technologies including HTML, JavaScript and CSS. The contractee uses this application to display project-related key figures vividly. It presents the company's staff a comparison of the planned and actual person-days for projects and milestones. Moreover, the app can view graphs of the composition of expenses. In addition, the application provides filtering mechanisms for the listings of projects, milestones and Issues.

The application is available as a hybrid, which means it is executable on the platforms Android, iOS and Windows Phone.

The document should give an insight in the development process, the problems we faced and the experiences we made during the creation of this thesis.

Inhaltsverzeichnis

Einleitung

1.1 Auftraggeber

Die in Ansfelden und Klagenfurt angesiedelte Logicx consulting & workflow GmbH ist ein Dienstleister in der IT Branche. Das Unternehmen betreut mehrere Projekte mit Kunden in den verschiedensten Tätigkeitsbereichen.

Neben Projekten zur Datenverwaltung für internationale Unternehmen versucht sich die Gesellschaft auch in innovativen Produkten zur visuellen Darstellung von Produktlinien mit fortschrittlicher Touch-Bedienung.

Im Zuge dieser Diplomarbeit unterstützten uns der technische Projektleiter Robert Stenitzer und der Entwickler Florian Bachinger.

1.2 Ausgangssituation

Das Unternehmen verwendet die hauseigene Zeiterfassungsanwendung Time um ihre Aufwände aufzuzeichnen. Time speichert die Daten in eine eigene Datenbank. Über das Berichtssystem Cognos der Firma IBM können ausführliche Auswertungen der Informationen aus der Datenbank erstellt werden. Diese Berichte stellen die Aufwände für die Projekte, Meilensteine und Issues in tabellarischer Form dar. Intern unterteilt das Unternehmen seine Projekte in Meilensteine (auch Arbeitspakete genannt), welche wiederum in einzelne Issues gegliedert werden. Ein Issue beschreibt genau eine Tätigkeit für einen Entwickler. Da Cognos nur innerhalb des Firmennetzes erreichbar ist, benötigen die Mitarbeiter eine VPN (Virtual Private Network) Verbindung, wenn sie außerhalb der Firma auf die Daten zugreifen möchten. Außerdem steht dem Unternehmen das Berichtssystem nur auf Desktop Geräten zur Verfügung.

1.3 Ziel der Arbeit

Das Ziel dieser Diplomarbeit ist es, eine mobile Anwendung zu entwickeln, die es den Projektleitern der Logicx GmbH ermöglicht, die Aufwandsdaten jederzeit und überall einzusehen. Voraussetzung dafür ist es, eine Möglichkeit zu schaffen, die es erlaubt, außerhalb des Firmennetzes die Daten auch ohne VPN Verbindung einzusehen. Außerdem sollen einige zusätzliche Funktionen realisiert werden. Darunter zum einen eine statistische Auswertung mittels Diagrammen zur übersichtlichen Darstellung der Zusammensetzung der Aufwände. Weiters sollen Überschreitungen der geplanten Zeit so hervorgehoben werden, sodass die Übertretung schnell ins Auge sticht.

2 Technologien

2.1 ADO.NET

ADO.NET stellt eine Vielzahl von Klassen für den Zugriff auf Daten im Bereich der .NET Framework Programmierung zur Verfügung. Somit wird es einem ermöglicht, sich zu einer Datenquelle zu verbinden und neue Daten zu erzeugen oder die bereits vorhandenen Daten lesen, löschen oder verändern zu können. (vgl. [MICROSOFT, MSDN, 2015])

2.1.1 ADO.NET Architektur

Die zwei Hauptkomponenten von ADO.NET sind der .NET Framework Data Provider und das DataSet.

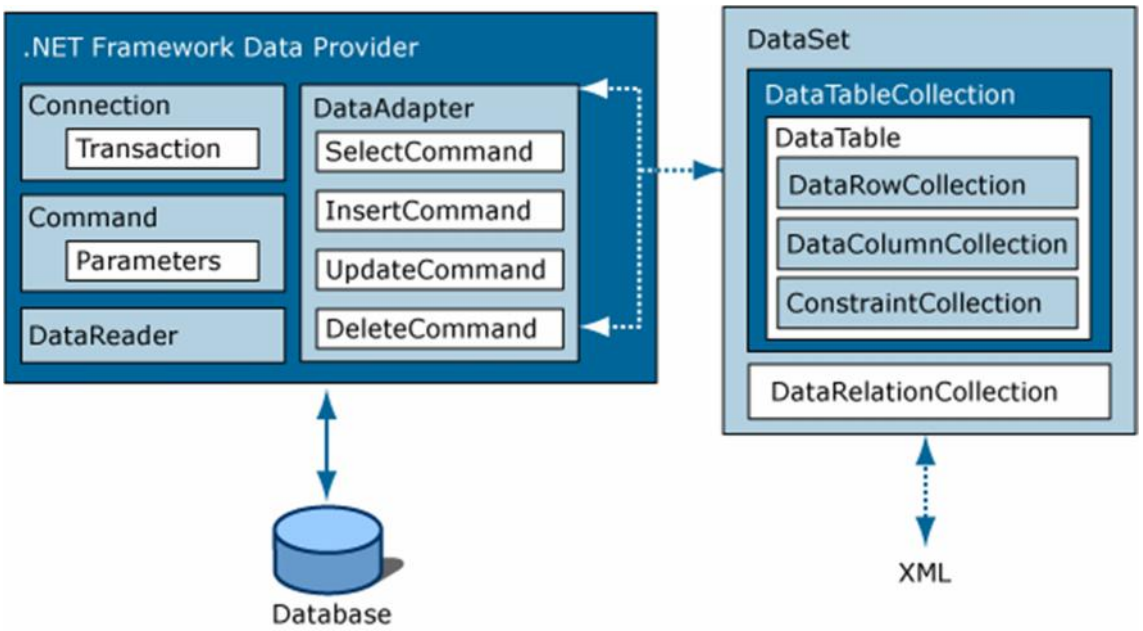

Abbildung 1: ADO.NET Architektur [1]

2.1.2 ADO.NET DataSet

Mit Hilfe des ADO.NET-DataSets kann man unabhängig von der Datenquelle auf beliebige unterschiedliche Informationsquellen zugreifen. Ein DataSet besteht aus einem oder mehreren DataTables, welche die eigentlichen Informationen speichern.

2.1.3 ADO.NET Framework Data Provider

Die ADO.NET Framework Data Provider wurden für effiziente Datenbearbeitung konzipiert. Je nach Provider bieten die Objekte Connection und Command Funktionalität für das Verbinden zur Datenquelle und das Absetzten von Befehlen. Das Adapter Objekt dient zur Verarbeitung der Informationen in den erstellten DataSets. (vgl. [MICROSOFT, MSDN, 2015])

2.2 ADO.NET Entity Framework

Beim klassischen ADO.NET werden die Daten mit Hilfe des DataSets aus einer Datenquelle ausgelesen und in tabellarischer Form gespeichert.

ADO.NET Entity Framework ist eine Erweiterung von ADO.NET und stellt Technologien zur Verfügung, um relationale Datenmodelle in objektorientierte Klassenstrukturen abzubilden und bietet die Funktionalität, um diese Objekte erzeugen, verändern und löschen zu können. Dieser Vorgang des Abbildens der relationalen Datenbanken wird auch Mapping genannt, weshalb auch oft der Begriff Object-relational Mapping (ORM) im Zusammenhang mit ADO.NET Entity Framework auftaucht.

Ergebnisse aus Abfragen auf die jeweilige Datenquelle werden nicht mehr in Tabellenstrukturen gespeichert, wie es beim klassischen ADO.NET der Fall ist, sondern werden in Objekte überführt. Für jeden Datensatz aus dem Resultat wird ein neues Objekt erzeugt. (vgl. [KÜHNEL, Andreas, 2013])

2.2.1 Entity Data Model

Das Herzstück des ADO.NET Entity Framework ist das Entity Data Model (EDM). Es dient als Verbindung zwischen der konzeptionellen Schicht, also dem Datenmodell, und der Datenbankstruktur, auch logische Schicht genannt. Dies setzt die Zuordnungsschicht des Entity Data Models um. Weiters stellt es auch die Verbindung zur Datenbank her, erstellt und führt die Abfragen aus und liefert die Ergebnisse. (vgl. [KÜHNEL, Andreas, 2013])

2.2.2 Techniken zur Datenabfrage

Bei der Verwendung eines Entity Data Model stehen einem drei verschiedene Möglichkeiten zur Auswahl, um auf die Daten zuzugreifen:

LINQ to Entities

LINQ to Entities ermöglicht es, unter Verwendung einer eigenen Syntax Abfragen für die Objektstruktur des Entity Data Model zusammenzustellen, um gefilterte Daten zu erhalten.

Die verwendete Syntax wird im Unterkapitel 2.3 genauer erläutert.

Entity SQL

Entity SQL ist ein SQL-Dialekt, der unabhängig von der darunterliegenden Datenbank mit dem konzeptionellen Modell des Entity Data Models arbeitet. Die Syntax ist zu vergleichen mit dem vom SQL Server bekannten T-SQL (Transact-SQL). (vgl. [KÜHNEL, Andreas, 2013])

EntityClient-Provider

Der EntityClient-Provider stellt lediglich Funktionen zum Lesen von Daten aus dem konzeptionellen Modell zur Verfügung. Ein weiterer großer Unterschied zu den beiden vorher genannten Methoden ist die Form, in der die Daten zurückgegeben werden. Das Resultat der Abfrage wird in Form von Zeilen und Spalten und nicht, wie benötigt, in Form von Objekten retourniert. (vgl. [KÜHNEL, Andreas, 2013])

2.3 Language Integrated Query (LINQ)

Language Integrated Query (LINQ) ergänzt das .NET Framework um Funktionalitäten, welche die Abfrage von beliebigen Datenquellen vereinfachen soll. Die Queries werden nicht als Zeichenketten gespeichert und dann einer Methode übergeben, die diese interpretiert und dann Anfragen auf eine Datenquelle generiert, sondern die Abfragen sind direkt als Code in C# und anderen .NET Sprachen integriert. (vgl. [KÜHNEL, Andreas, 2013])

Die Syntax und einige Sprachelemente sind vergleichbar mit SQL und dessen Schlüsselwörter wie *select*, *where* und *from*.

2.3.1 LINQ-Syntax

LINQ Abfragen kann man auf zwei verschieden Schreibweisen implementieren. Zum einen die SQL ähnliche Abfrage-Syntax und zum anderen die Erweiterungsmethoden-Syntax.

Abfrage-Syntax

Eine LINQ-Abfrage besteht immer aus den drei Schlüsselwörter *from*, in und *select*. Darüber hinaus kann durch das Hinzufügen weiterer Schlüsselwörter wie *where* oder *join* das Ergebnis genauer eingegrenzt oder um mehrere Objekte erweitert werden.

```
1.  var query = from pers in persons
2.              select pers;
```

Beispiel 1: Simple LINQ-Abfrage

From

Mit dem *from* Befehl, der immer zu Beginn einer LINQ-Abfrage steht, wird die verwendete Datenquelle angegeben, welche dann sequentiell durchlaufen wird. Eine solche Datenquelle muss das IEnumerable Interface implementieren, um in einer LINQ-Abfrage verwendet zu werden.

Der Ausdruck direkt nach dem Schlüsselwort *from*, in diesem Beispiel der Ausdruck „pers“ (siehe Beispiel 1), bestimmt die Variable, die das aktuelle Element der Datenquelle speichert. Auf die Werte dieses Elements kann dann, zum Beispiel bei der Filterung in der where-Klausel, zugegriffen werden.

Select

Das Schlüsselwort *select* bestimmt das Objekt, das tatsächlich zur Ergebnismenge hinzugefügt wird.

Es kann auch nur ein Attribut der Variable aus der Datenquelle zur Ergebnisliste hinzugefügt werden (siehe Beispiel 2).

```
1.  var query = from pers in persons
2.              select pers.Name;
```

Beispiel 2: LINQ-Abfrage mit Attribut

Where

Der Ausdruck *where* dient zur Bestimmung der Filterregeln die bei der LINQ-Abfrage angewendet werden sollen, ähnlich der Where-Klausel bei einer SQL Select Query.

Folgende LINQ-Abfrage soll alle Personen liefern, die älter als 50 Jahre sind (siehe Beispiel 3).

```
1.  var query = from pers in persons
2.              where pers.Age > 50
3.              select pers;
```

Beispiel 3: LINQ-Abfrage mit Where-Klausel

Erweiterungsmethoden-Syntax

Neben der Abfrage-Syntax gibt es noch die Möglichkeit seine LINQ-Abfragen in der Erweiterungsmethoden-Syntax zu implementieren. Hierbei werden die Schlüsselwörter von vorhin (siehe Kapitel 2.3.1), zum Beispiel der „where“ Ausdruck, als Methoden der Datenquelle angesehen und können mit dem Punktoperator aufgerufen werden.

Das folgende Quelltextbeispiel (siehe Beispiel 4) zeigt die LINQ-Abfrage aus dem Beispiel 3 mit der Erweiterungsmethoden-Syntax.

```
1.  var query = persons
2.              .Where(pers => pers.Age > 50);
```

Beispiel 4: LINQ-Abfrage mit Erweiterungsmethoden-Syntax

2.4 Windows Communication Foundation (WCF)

Windows Communication Foundation (WCF) ist ein Framework, welches Werkzeuge zur Erstellung von dienstorientierten Anwendungen bereitstellt, um Daten als asynchrone Nachrichten von einem Dienstendpunkt (service endpoint) zum anderen übertragen zu können. Diese Applikationen können Teil eines permanent erreichbaren von IIS gehosteten Dienstes sein oder in einer Anwendung gehostet werden. (vgl. [MICROSOFT, MSDN, 2015])

WCF bietet eine Vielzahl von Funktionalitäten zur Implementierung von dienstorientierte Anwendungen. Im Folgenden wird auf die Funktionen eingegangen, welche für die Erstellung dieser Diplomarbeit maßgeblich waren.

Sicherheit

Meist haben Unternehmen, die Webdienste bereitstellen, schon vorhandene Sicherheitsinfrastrukturen oder Authentifizierungsmodelle. So kann WCF in bereits bestehende Transportsicherheitsmodelle integriert werden und die vorhandenen Infrastrukturen nutzen.

Unterstützung für ReST

Eine WCF Anwendung kann so konfiguriert werden, dass einfache XML-Daten versendet werden, ohne sie in einem SOAP (Simple Object Acces Protocol) Rahmen zu verpacken. Weiters kann WCF soweit ergänzt werden, dass spezifische XML-Formate oder gar Nicht-XML-Formate wie JavaScript Object Notation (JSON) verwendet werden können. (vgl. [MICROSOFT, MSDN, 2015])

2.5 Internet Information Services (IIS)

Internet Information Services (IIS) ist der auf Windows Betriebssystemen, wie Windows 8.1 oder Windows Server 2012, eingesetzte Webserver. Er dient zur Bereitstellung von

statischen oder dynamischen Websites, sowie Web Services und andere Diensten, die über das Internet beziehungsweise Intranet erreichbar sein sollen.

Zuerst muss der Aufbau von IIS erklärt werden und wie die Begriffe Website, Anwendungen (Application), Anwendungspool (Application Pool) und virtuelle Verzeichnisse (Virtual Directory) zusammenhängen.

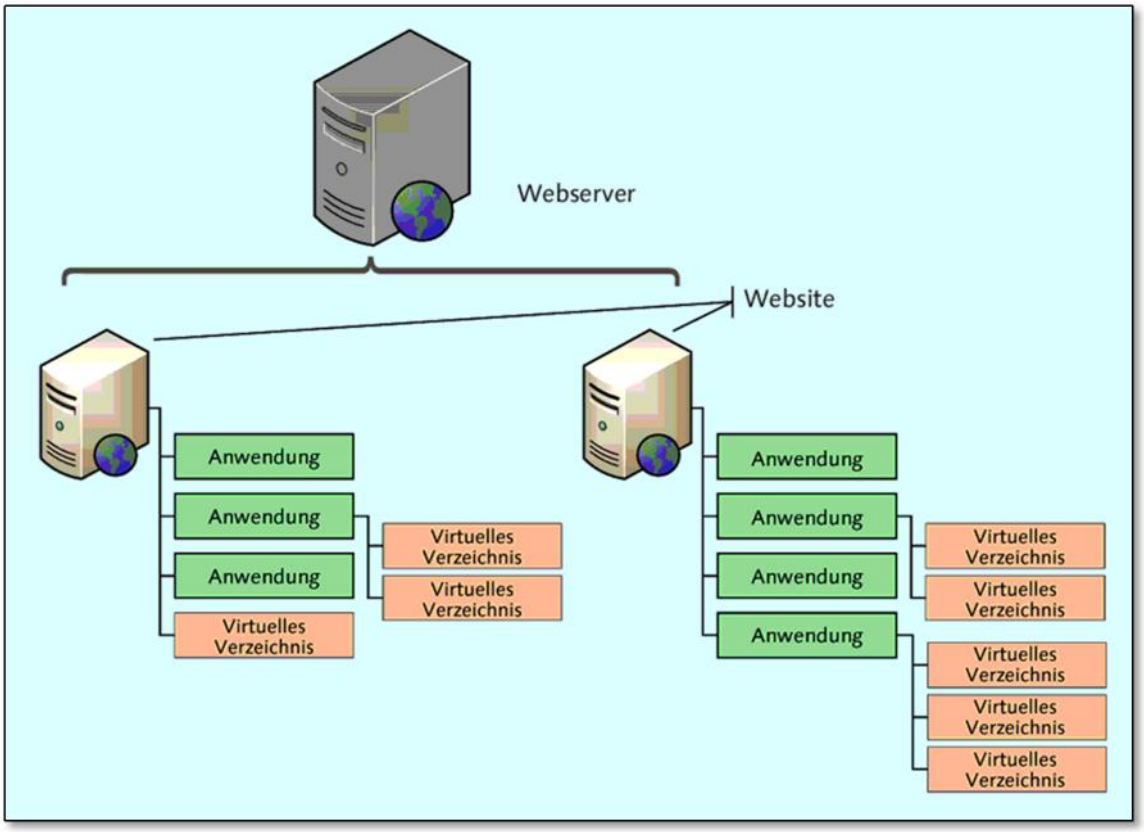

Abbildung 2: Aufbau des IIS [2]

IIS Aufbau

An oberster Stelle der Hierarchie steht der Webserver (siehe Abbildung 2), also der Rechner auf dem der IIS installiert ist. Auf einem Web Server können mehrere Websites installiert sein. Eine Website kann wiederum mehrere Anwendungen enthalten.

Eine Anwendung ist nichts anderes als ein Verzeichnis, welches die Dateien einer Webapplikation beinhaltet.

Man kann einer Website beziehungsweise einer Anwendung auch noch virtuelle Verzeichnisse zuweisen. Ein solches virtuelles Verzeichnis verweist auf ein Verzeichnis, welches nicht im Stammverzeichnis der Website oder Anwendung liegt. (vgl. [BODDENBERG, Ulrich, 2014])

Authentifizierung

Eine essentielle Funktion vom IIS ist die Authentifizierung. Falls es notwendig ist Inhalte einer Website beziehungsweise Anwendung nicht für jedermann im Internet zugänglich zu machen, so muss festgestellt werden, wer den Inhalt verwenden darf und wer nicht.

In dieser Diplomarbeit wurde das Authentifizierungsverfahren Standardauthentifizierung (Basic Authentication) verwendet, weshalb dieses im Folgenden genauer beschrieben wird.

Standardauthentifizierung

Die Standardauthentifizierung ist die simpelste Methode um sich gegen ein Active Directory zu authentifizieren.

Der Ablauf der Authentifizierung sieht bei dieser Methode folgendermaßen aus:

Falls der Benutzer mittels einer HTTP-GET-Anforderung den Inhalt einer Datei betrachten will und sich diese Datei in einer Website oder Anwendung befindet, welche durch die Standardauthentifizierung geschützt ist, so sendet der Webserver eine Antwort mit dem Statuscode 401 (unauthorized). In diesem Antwort-Paket wird mit dem Feld WWWAuthenticate im Header angegeben, welche Authentifizierungsmethode vorgegeben wird. Im Falle der Standardauthentifizierung wäre das „Basic“.

Diese Nachricht interpretiert der Browser und öffnet ein Dialogfenster zur Eingabe der Anmeldeinformationen.

Nach der Eingabe sendet der Browser erneut eine HTTP-GET-Anforderung an die Datei, jedoch befindet sich bei dieser Anfrage das Feld AUTHORIZATION im Header, in dem sich die Authentifizierungsmethode (Basic) und die Anmeldeinformationen in Base64 codierter Form befinden.

Falls die Anmeldeinformationen korrekt sind, erhält der Browser die angeforderte Datei mit dem Statuscode 200.

2.6 Representational State Transfer (ReST)

Bei ReST (REST) handelt es sich um einen Architektur-Stil der den Fokus auf Skalierbarkeit, Erweiterbarkeit und Interoperabilität setzt. Diese Architektur wurde von Roy Thomas Fielding im Jahre 2000 in seiner Dissertation „Architectural Styles and the Design of Network-based Software Architectures“ veröffentlicht und beschreibt eine Variante zum Entwerfen von verteilten Applikationen. Ein Beispiel für eine solche Applikation mit ReSTful Design wäre das World Wide Web und das HTTP Protokoll, an dessen Entwicklung Roy Fielding beteiligt war. Ein Benutzer navigiert mittels verschiedenen Links zwischen den Inhalten und Ressourcen. (vgl. [DAZER, Michael])

Ein wichtiger Aspekt bei diesem Design ist ein effizientes Zusammenspiel mit Standardtechnologien wie HTML oder XML (Extensible Markup Language).

Im Namen des Designs steckt ein weiterer Schlüsselpunkt. Bei einer ReSTful Architektur muss jede Transaktion einen repräsentativen Zustand zurückgeben. Das bedeutet das Ergebnis einer Anfrage muss alle nötigen Daten umfassen, damit der Client, ohne weiteres Wissen über den Zustand der Verbindung, mit dem Ergebnis arbeiten kann. Dieses Verhalten steht direkt mit dem Schlüsselwort Zustandslosigkeit in Verbindung. (vgl. [DAZER, Michael])

ReST fordert zusätzlich eine einheitliche Schnittstelle in der Ressourcen eindeutig durch URIs (Uniform Ressource Identifier) identifiziert werden können. Weiters soll durch die Verwendung der Standard HTML Funktionen GET, POST, PUT und DELETE verschiedene Funktionalität für das Hinzufügen, Verändern und Löschen von Ressourcen geboten werden. Zur Darstellung des Ergebnisses sollen auch verschiedene Präsentationen einer Ressource möglich sein. Zum Beispiel in Form von HTML, XML oder JSON. (vgl. [DAZER, Michael])

Folgende Grafik (siehe Abbildung 3) soll die eindeutige Identifizierung von Objekten in einer ReSTful Architektur darstellen. Jede Ressource kann mittels einem eindeutigen URI erreicht werden.

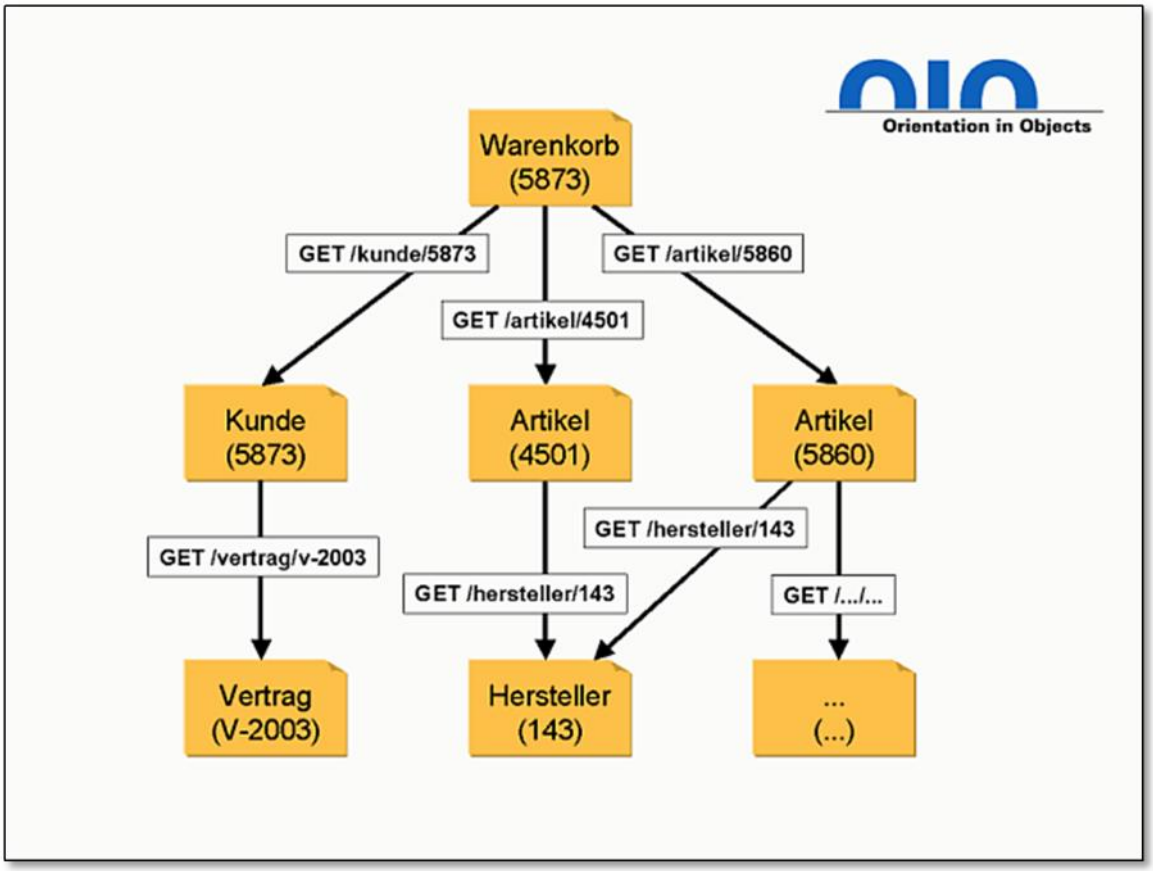

Abbildung 3: ReST Ressourcen Adressierung [3]

2.7 Secure Socket Layer (SSL)/Transport Layer Security (TLS)

TLS, besser bekannt unter dem Namen des Vorgängerprotokolls SSL, ist ein Verschlüsselungsprotokoll zur sicheren Ende-zu-Ende Datenübertragung im Internet. Seit der SSL Version 3.0 wird dieses Protokoll unter dem neuen Namen TLS weiterentwickelt. Die SSL Version 3.1 entspricht dabei der TLS Version 1.0. In dieser Arbeit wird die Abkürzung SSL für beide Protokolle verwendet, da diese noch weiter verbreitet ist.

Heutzutage wird das Protokoll hauptsächlich beim Anwendungsprotokoll HTTPS zur Herstellung von Vertraulichkeit und Integrität, durch Verschlüsselung und Authentifizierung, verwendet.

SSL befindet sich zwischen Anwendungsschicht und Transportschicht und kann deshalb bei mehreren verschiedenen Protokollen der Anwendungsschicht verwendet werden. (vgl. [MICROSOFT, 2015])

Authentifizierung

Die Authentifizierung der Parteien erfolgt bei SSL mit Hilfe eines X.509 Zertifikates. Ein solches Zertifikat ist eine digitale Form der Identifizierung einer Partei und wird meist von einer dritten Partei ausgestellt, einer sogenannten Certification Authority (CA). Eine solche CA überprüft vor dem Ausstellen eines Zertifikates die Identität des Anfragestellers und weist ihm einen öffentlichen Schlüssel zu. Ein Zertifikat beinhaltet eine Gültigkeitsdauer, den öffentlichen Schlüssel, eine Seriennummer und die digitale Signatur des Ausstellers.

Wenn ein Client mit einem Server kommunizieren will und dessen Identität überprüfen will, so sieht er in seiner Liste der zugelassenen CAs nach. Wenn das Zertifikat des Servers von einem der CAs auf der Liste ausgestellt wurde, kann der Client sicher sein, dass der Server vertrauenswürdig ist. (vgl. [MICROSOFT, 2015])

Verschlüsselung

SSL verwendet symmetrische sowie asymmetrische Verschlüsselungsverfahren. Bei der Authentifizierung vom Server, oder auch vom Client, und beim Erzeugen eines Sitzungsschlüssels werden asymmetrische Verschlüsselungsverfahren angewandt. Bei der Verschlüsselung der zu übertragenden Daten wird ein symmetrisches Verschlüsselungsverfahren mit dem Sitzungsschlüssel verwendet. Dieses Vorgehen verbindet die Geschwindigkeit der symmetrischen Verfahren mit der einfachen Authentifizierung bei asymmetrischen Verfahren. (vgl. [MICROSOFT, 2015])

Hashfunktion

Im Laufe des Verbindungsaufbaus beim SSL Protokoll wird zwischen den beiden Parteien ein Hashalgorithmus ausgehandelt zum Erzeugen eines Fingerabdrucks der Nachricht. Dieser Fingerabdruck soll die Integrität der Daten sicherstellen. SSL verwendet das Hashverfahren MAC (Message Authentication Code), die neuere Version TLS verwendet das sicherere Verfahren HMAC (Keyed-Hash Message Authentication Code). (vgl. [MICROSOFT, 2015])

2.8 JavaScript Object Notation (JSON)

JavaScript Object Notation ist ein textbasiertes Format zum Austausch von Daten. Dabei werden Objekte als Zeichenketten serialisiert, welche oft über Netzwerke versendet oder in Dateien gespeichert werden. Verglichen mit der Alternative XML, die ebenfalls ein maschinenlesbares Klartextformat ist, ist JSON weitaus kompakter. Für unsere Zwecke bot sich JSON als Übertragungsformat an, da es von der JavaScript Objekt Struktur ableitet und daher von JavaScript direkt unterstützt wird. JSON verwendet die JavaScript Datenstrukturen (Objekte, Arrays, Zeichenketten, Zahlen und boolesche Werte). (vgl. [MICROSOFT, MSDN, 2015])

JSON Struktur

Jede JSON Zeichenkette ist entweder ein Objekt oder ein Array.

Objekt

Ein Objekt wird durch geschwungene Klammern zu Beginn und am Ende gekennzeichnet. Dazwischen sind beliebig viele Name-Wert Paare, die durch Beistriche getrennt werden. Der Name ist immer eine Zeichenkette mit doppelten Hochkommas, der Wert kann jeden beliebigen JSON Datentyp haben, kann also auch ein weiteres Objekt oder ein Array sein. Name und Wert werden durch einen Doppelpunkt getrennt. Ein JSON Objekt könnte wie nachfolgend aussehen.

```
1. {
2. "name" : "Max Mustermann",
3. "sex"  : "male",
4. "age"  : 53
5. }
```

Beispiel 5: JSON Objekt

Array

Ein Array beginnt mit einer eckigen Klammer auf ([) und endet mit einer eckigen Klammer zu (]). Ein Array unterscheidet sich von einem Objekt dadurch, dass die Werte ohne Namen gelistet werden. Auch hier kann der Wert jeden JSON Datentyp annehmen. Beispiel 5 wird um einen Eintrag erweitert, bei dem der Wert ein Array ist.

```
1. {
2. "name" : "Max Mustermann",
3. "sex"  : "male",
4. "age"  : 53,
5. "kids" : ["Hanna", "David"]
6. }
```

Beispiel 6:JSON Array

3 Entwicklungssysteme

In diesem Kapitel werden alle Entwicklungssysteme beschrieben, die im Zuge dieser Diplomarbeit verwendet wurden.

3.1 Visual Studio 2013

Visual Studio ist eine von Microsoft geschaffene Entwicklungsumgebung (Integrated Development Environment – IDE) welche die Erstellung von Webanwendungen und Webdiensten sowie Desktopanwendungen und mobile Anwendungen auf Windows Basis ermöglicht. (vgl. [MICROSOFT, MSDN, 2015])

In dieser Diplomarbeit wurde diese IDE zur Entwicklung der App beziehungsweise zum Implementieren der Standardwebtechnologien HTML, CSS und JavaScript, sowie der Entwicklung des Services verwendet. Visual Studio bietet neben dem Syntax Highlighting für die verwendeten Sprachen HTML, CSS und JavaScript auch das automatische Quellcode Vervollständigungswerkzeug IntelliSense.

Damit IntelliSense auch bei selbst erstellten JavaScript Objekten, Funktionen und Methoden funktioniert, muss man bei einer JavaScript Datei auf alle JavaScript Dateien referenzieren, damit IntelliSense auf den dort implementierten Code zugreifen kann. Ein solcher Verweis auf eine andere JavaScript Datei sieht im Quellcode folgendermaßen aus (siehe Beispiel 7):

```
1. /// <reference path="JavaScriptFile.js" />
```

Beispiel 7: JavaScript IntelliSense Referenz

Um die automatische Vervollständigung über all die Dateien zu ermöglichen, wurde in einer zentralen Datei auf alle vorhandenen Dateien verwiesen. Alle anderen Dateien verweisen dann auf diese zentrale Datei. Damit wird IntelliSense ermöglicht auf alle Objekte, Methoden und Variablen der referenzierten Dateien zuzugreifen.

Da es bei einem Cordova Projekt keine Solution und kein Projekt gibt, welches geöffnet werden kann, um alle Dateien zu bearbeiten, muss improvisiert werden. Das Problem kann umgangen werden, indem ein Visual Studio Projekt erstellt wird, welche den www Ordner (siehe Unterkapitel 4.1.2 Basiskomponenten) als lokale Website öffnet. Dazu muss man lediglich den „Open Web Site“ Dialog, wie in folgender Abbildung gezeigt, öffnen und im Dateisystem den Ordner auswählen und anschließend die Solution speichern (Microsoft Visual Studio Solution - .sln Datei).

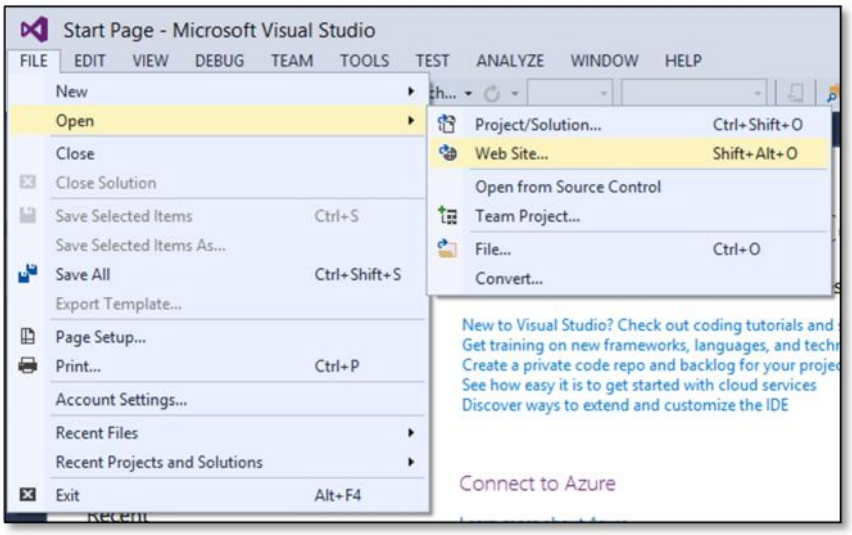

Abbildung 4: Öffnen des "Open Web Site" Dialogs

3.2 Microsoft SQL Server 2012

Microsoft SQL Server ist ein relationales Datenbank Management und Analyse System zum Erzeugen, Bearbeiten, Verwalten und Bereitstellen von Datenbanken. (vgl. [MICROSOFT, MSDN, 2015])

3.3 SQL Server 2012 Management Studio

Das SQL Server 2012 Management Studio ist eine Umgebung von Microsoft um die Infrastruktur des SQL Servers zu verwalten. Es bietet graphische Werkzeuge zum Lesen, Konfigurieren, Aktualisieren und Erzeugen von jeglichen SQL-Server Objekten und Komponenten zum Implementieren von SQL Abfragen sowie SQL beziehungsweise T-SQL Skripts. (vgl. [MICROSOFT, MSDN, 2015])

4 Development Kits

Ein Software Development Kit (SDK) ist ein Satz aus Werkzeugen zur Programmierung von Anwendungen für ein bestimmtes Betriebssystem oder eine Programmiersprache. Es besteht meist aus einer ausführlichen Dokumentation und den nötigen Bibliotheken und beinhaltet oftmals auch Laufzeitumgebungen, Entwicklungsumgebungen oder Simulatoren (vgl. [DATACOM BUCHVERLAG]).

In diesem Kapitel wird näher auf das Framework Cordova eingegangen. Die SDKs der verschiedenen Plattformen, wie Android SDK, Windows Phone SDK und iOS SDK (iPhone SDK) werden in den Unterkapiteln 10.1.2, 10.2.2**Fehler! Verweisquelle konnte nicht gefunden werden.** und 10.3.2 des Kapitels Konfiguration der Entwicklungsumgebungen beschrieben.

4.1 Cordova (PhoneGap)

Apache Cordova, auch PhoneGap genannt, ist ein Open-Source Framework zur Entwicklung von mobilen Anwendungen unter der Verwendung von Standard Web Technologien wie HTML, CSS und JavaScript. Diese Anwendungen sind Cross-Platform Applikationen. Cross-Platform bedeutet in diesem Fall, dass aufgrund einer Codebasis mit Hilfe von verschiedenen Build Systemen Anwendungen für mehrere verschiedene Plattformen erstellt werden können. Weiters erlaubt das Framework dem Entwickler durch Cordova Plugins aus JavaScript Code auf Funktionalitäten des Gerätes, wie zum Beispiel die Kamera zuzugreifen. (vgl. [APACHE SOFTWARE FOUNDATION, 2015])

Anfangs war das Framework unter dem Namen PhoneGap bekannt und wurde von dem Unternehmen Nitobi Software entwickelt. Als das Unternehmen von Adobe Systems gekauft wurde, übergab Adobe Systems PhoneGap an die Apache Software Foundation, welche das Projekt unter dem Namen Apache Cordova weiter führt. Deshalb ist es heute unter den beiden Namen (Apache) Cordova und PhoneGap bekannt.

In diesem Dokument wird für dieses Framework durchgehend der Begriff Cordova verwendet, um mögliche Verwechslungen auszuschließen.

4.1.1 Funktionalität

Wie bereits erwähnt bietet Cordova die Möglichkeit den Code für die Applikation einmalig zu implementieren und diesen dann mit Hilfe von verschiedenen Build Mechanismen für mehrere Plattformen verfügbar zu machen. Ermöglicht wird dies, indem Cordova auf die jeweilige WebView Komponente der einzelnen Plattformen aufbaut und die HTML Seiten dann in dieser anzeigt. Die WebView Komponente greift dabei auf die Rendering Engine des jeweiligen Browsers der Plattform zu, um den HTML Code darstellen zu können.

Bei der Verwendung von Cordova kann man zwischen zwei Methoden der Erstellung eines Projektes wählen. Eine dieser Möglichkeiten nennt sich „platform-centered workflow“. Bei dieser Methode erstellt man ein Projekt für eine bestimmte Plattform, indem man in der dafür vorgesehenen Entwicklungsumgebung (bei Windows Phone zum Beispiel Visual Studio) mit Projektvorlagen von Cordova manuell ein Projekt erstellt. Dieses umständliche Verfahren wird verwendet, wenn man nur für eine Plattform entwickeln will und dabei auch Veränderungen im nativen Code durchführen muss.

Wenn man eine App für so viele verschiedene Plattformen wie möglich entwickeln will und wenige Änderungen im nativen Code machen muss, ist jedoch der sogenannte „cross-plattform workflow“ besser geeignet. Bei diesem Ansatz wird das Kommandozeilenprogramm cordova (Command-Line Interface – CLI) verwendet, um Projekte für mehrere Plattformen erzeugen und gleichzeitig kompilieren zu können. Beim Build des gesamten Projektes werden dabei die implementierten Dateien aus dem www Ordner in die Unterordner von platforms kopiert sowie die Konfigurationen übernommen und die Binaries generiert. Einige der erwähnten Begriffe werden im folgenden Unterkapitel näher erklärt (siehe Unterkapitel 4.1.2). (vgl. [APACHE SOFTWARE FOUNDATION, 2015])

Bei dieser Diplomarbeit wurde die Entwicklung mit dem CLI gewählt, da die Applikation auf möglichst vielen verschiedenen Plattformen lauffähig sein sollte. Die Befehle des CLI, die für die Erstellung dieses Projektes notwendig waren, werden im Kapitel 10 behandelt.

4.1.2 Basiskomponenten

Ein einfaches Cordova Projekt beinhaltet nach dem Erzeugen folgende Ordnerstruktur:

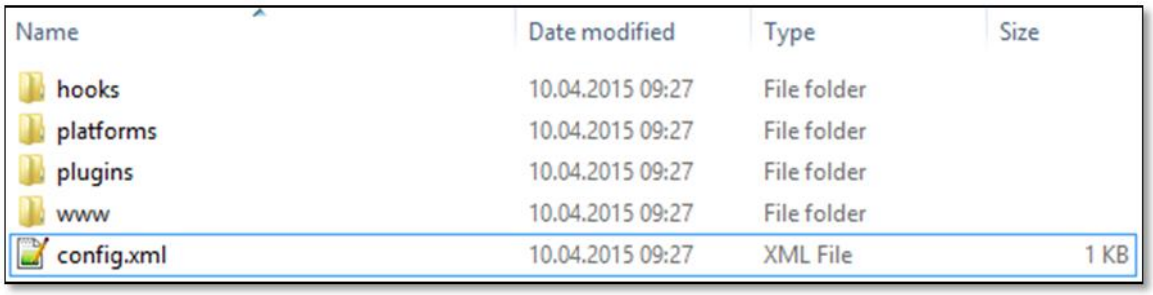

Abbildung 5: Cordova Projekt Ordnerstruktur

config.xml

Alle Konfigurationen für ein Cordova Projekt befinden sich in der config.xml Datei. In dieser werden jegliche Information über die App gespeichert und durch bestimmte Parameter kann das Verhalten der App beeinflusst werden.

Das folgende Beispiel 8 bietet einen Überblick über die config.xml Datei direkt nach dem Erzeugen des Projektes.

```
1.  <?xml version='1.0' encoding='utf-8'?>
2.  <widget id="com.example.test" version="0.0.1"
    xmlns="http://www.w3.org/ns/widgets"
    xmlns:cdv="http://cordova.apache.org/ns/1.0">
3.      <name>Test</name>
4.      <description>
5.        A sample Apache Cordova application that responds to the deviceready
    event.
6.      </description>
7.      <author email="dev@cordova.apache.org" href="http://cordova.io">
8.          Apache Cordova Team
9.      </author>
10.     <content src="index.html" />
11.     <access origin="*" />
12. </widget>
```

Beispiel 8: Basiskonfiguration eines Cordova Projektes

Hier werden folgende Attribute der App konfiguriert:

- **<name>:** Spezifiziert den Namen der Applikation, der am Home Screen des Gerätes und im App Store dargestellt wird.
- **<description>/<author>:** Definieren Metainformationen und Kontakt für die Beschreibung im App Store.
- **<content>:** Bestimmt welche HTML-Datei beim Start der App angezeigt wird. Standardmäßig ist index.html eingestellt.
- **<access>:** Dieses Tag bestimmt die externen Domänen, mit denen die App kommunizieren kann. Der Standardwert „*“ bedeutet, dass alle Domänen erlaubt sind.

Das Verhalten der Apps kann mit dem Element „<preference>“ beeinflusst werden. Hierbei unterscheidet man zwischen Einstellungen (preferences) die für alle Plattformen gültig sind, sogenannte Global Preferences, und jene, die nur für einige bestimmte Plattformen anwendbar sind, welche die Bezeichnung Multi-Platform Preferences tragen.

Global Preferences sind zum Beispiel die Einstellung für die Orientierung der App, welche die Werte default, landscape oder portrait haben kann. Ein solches Tag sieht dann in der config.xml Datei folgendermaßen aus:

```
1.  <preference name="Orientation" value="default" />
```

Beispiel 9: Preference Tag Beispiel

Es besteht auch die Möglichkeit bestimmte Einstellungen nur für eine Plattform einzurichten. Dies ist möglich durch das Einfügen des gewünschten <preference> Elementes zwischen die zwei <platform> Tags, welche im Attribut „name“ die gewünschte Plattform beinhaltet. Um die Orientierung der App nur für Android konfigurieren zu können muss der Code der folgenden Abbildung zur config.xml Datei hinzugefügt werden (siehe Beispiel 10).

```
1. <platform name="android">
2.     <preference name="Orientation" value="default" />
3. </platform>
```

Beispiel 10: Konfiguration für nur eine Plattform

www

Im Unterordner www sind alle HTML, CSS und JavaScript Dateien sowie alle Ressourcen enthalten, die zur Darstellung der App dienen. In diesem Ordner muss auch die HTML Datei vorhanden sein, die in der config.xml Datei im <content> Element definiert wird und beim Start der App angezeigt werden soll. Nach dem Erzeugen des Projektes befinden sich Testdateien in diesem Ordner.

plugins

Im Verzeichnis plugins befinden sich alle installierten Plugins, sowie Metadaten die beschreiben, welche Plugnis für welche Plattformen verwendet werden.

Plugins sind Bibliotheken die zusätzliche Funktionalität zum Projekt hinzufügen. Solche Plugins können sowohl nur reinen JavaScript Code enthalten, aber auch eigene Klassen für jede einzelne Plattform, die dem Entwickler ermöglichen, von JavaScript Code auf native Komponenten zuzugreifen. Seit der Version 3.0 wird jegliche Funktionalität von Cordova, die es ermöglicht mit nativen Komponenten zu kommunizieren, in Form von Plugins angeboten. Standardmäßig sind keine Plugins installiert, um den Umfang einer App möglichst gering zu halten. (vgl. [APACHE SOFTWARE FOUNDATION, 2015])

platforms

Der Ordner platforms beinhaltet die nativen Projekte der jeweiligen Plattformen, die zum Projekt hinzugefügt wurden.

5 Bibliotheken

5.1 Angular.js

AngularJS ist eine JavaScript Bibliothek der Google Inc., die es ermöglicht sehr einfach Single-page Webanwendungen zu entwerfen. Außerdem bietet sie eine Hilfestellung um das Model-View-Controller (MVC) Entwurfsmuster in der Webentwicklung umzusetzen.

Installation

Um in der Website AngularJS verwenden zu können muss die JavaScript Datei angular.js eingebunden werden. Aufgrund des Befehls in nachfolgenden Quellcodebeispiel lädt der Webbrowser die Datei und die gewünschte Funktionalität steht zur Verfügung.

```
1. <script type="text/javascript" src="angular.js"></script>
```

Beispiel 11: AngularJS Einfügung

Die Datei kann auf der Website von AngularJS heruntergeladen werden. [4]

Funktionsweise

Modul

Das zentrale Element bei der Entwicklung mit AngularJS ist das sogenannte Angular Modul. Das Modul muss im JavaScript Code erstellt werden. Das folgende Quellcodebeispiel zeigt wie das Angular Modul erzeugt wird:

```
1. var module = angular.module('app', []);
```

Beispiel 12: Angular Modul erzeugen

Alle weiteren Angular spezifischen Elemente werden über dieses Modul erstellt.

Controller

Der Controller (die Steuerung) in Angular ist streng ausgerichtet auf die Controller Komponente im MVC. Das heißt er bietet Funktionalität, die bei Benutzeraktivitäten aufgerufen wird. Jeder Controller in AngularJS erhält einen eigenen Gültigkeitsbereich ($scope) auf den die Anzeigeelemente, die diesem Controller zugeordnet sind, zugreifen können. In diesem Gültigkeitsbereich werden Funktionen und Variablen definiert, mit welchen dann die View Funktionalität und Daten, da diese ja selbst nicht auf das Modell zugreift, erhält.

Routing

Das Routing ist nicht direkt in AngularJS integriert. Es ist ein Aufsatz, der sich ngRoute nennt und eigens in die Seite eingebunden werden muss. Allerdings wird es in der Dokumentation als Bestandteil von AngularJS gehandhabt, daher wird es auch hier nicht als eigene Bibliothek aufgeführt. Um die Funktionalität von ngRoute nutzen zu können muss sie bei der Definition des Moduls angegeben werden. (vgl. [GOOGLE, 2015])

Mit Hilfe des Routing können Teile des HTML Codes in andere Dateien ausgelagert werden, ohne dass dabei eine neue Seite erstellt wird. Diese Dateien werden in Angular Templates genannt. Beim Routing wird jedem Pfad in der URL eine Route zugeordnet. Eine Route besteht aus einem Template und einem Controller. Somit erhält jedes Template einen Controller, der genau die Funktionalität bereitstellt, die benötigt wird. (vgl. [GOOGLE, 2015])

Direktive

Eine AngularJS Direktive ist vergleichbar mit einem HTML Attribut. Es erweitert ein HTML Element um bestimmte Funktionen. AngularJS bietet eine Vielzahl von vordefinierten Direktiven. Für spezielle Anforderungen ist es allerdings oft sinnvoll, eigene Direktiven zu implementieren. Dazu muss eine JavaScript Funktion erstellt werden, über dessen Rückgabewert der Angular Compiler erfährt, wie Elemente mit dieser Direktive behandelt werden sollen. Über diese Direktiven können zum Beispiel Vorlagen für Muster, die mehrfach vorkommen, erstellt werden. Ebenso können dem Element Styles oder Event-Listeners hinzugefügt werden. Für die Namensgebung von Direktiven gibt es einige Konventionen, die zu beachten sind. Während die vorgefertigten mit dem Kürzel ng beginnen, sollte für die eigenen ein anderes Präfix verwendet werden. Bei der Definition im JavaScript Code sollte der Name ohne Trennzeichen an das Präfix gehängt werden. Um den Name vom Präfix zu unterscheiden, wird der erste Buchstabe groß geschrieben, ein Konzept, das sich camelCase nennt. Da HTML nicht zwischen Groß- und Kleinbuchstaben unterschiedet, soll im HTML Code dem Präfix ein Bindestrich als Trennzeichen folgen. (vgl. [GOOGLE, 2015])

Data Binding

Es gibt zwei verschiede Arten Data Binding mit AngularJS zu spezifizieren. In beiden Fällen wird es über eine Variable im Gültigkeitsbereich eines Controllers erreicht. Die erste Variante ist es, den Variablennamen zwischen zwei geschwungenen Klammern zu setzen. Eine weitere Möglichkeit ist es, in einer beliebigen Angular Direktive die Variable zu verwenden. Hervorzuheben sind die explizite Binding Direktive ng-bind und die Direktive ng-model, mit der two way Data Binding bei HTML Input Feldern realisiert wird. Eine weitere wichtige Direktive in diesem Kontext ist ng-repeat. Mit dieser kann Data Binding auf Arrays angewandt werden. Das Element, in dem ng-repeat steht, dient als eine Vorlage, von der für jedes Element aus dem Array ein HTML Element erstellt wird. Über eine Schleifenvariable kann dann auf die Werte zugegriffen werden.

5.2 Hammer

Hammer bietet eine vorgefertigte Lösung, um auf Gesten des Benutzers auf einem Gerät mit Touch-Bedienung zu reagieren. Um die Funktionen nutzen zu können, wird die Bibliothek in Form einer JavaScript Datei in das Projekt eingebunden. Danach kann ein Objekt von Hammer erzeugt werden, mit welchem Event Listener generiert werden können. In dieser Arbeit werden die Gesten nach links Wischen und nach rechts Wischen benötigt. (vgl. [TANGELDER, Jorik])

5.3 Swipe

Für das Umschalten zwischen verschiedenen Ansichten der Applikation wird oft die in der mobilen Entwicklung beliebte Methode Swipe verwendet. Von einem Swipe spricht man, wenn die aktuelle Ansicht durch eine animierte Transformation in eine Richtung verschwindet und gleichzeitig eine neue Ansicht aus der anderen Richtung erscheint. Dadurch entsteht für den Benutzer der Eindruck, dass die beiden Ansichten nebeneinander sind und gemeinsam verschoben werden. Ein Swipe wird üblicherweise durch eine Wischgeste ausgelöst.

Diese Transformation kann mit der gleichnamigen Bibliothek Swipe erreicht werden. Dafür wird ein Objekt von Swipe mit einigen Optionen erzeugt. Mit diesem Objekt kann mit den Methoden prev() und next() zu der vorherigen und nachfolgenden Ansicht auf diese Weise umgeschaltet werden. (vgl. [GITHUB, 2015])

6 Cloud Service

6.1 Google Charts

Mit Google Charts können verschiedene Diagramme auf Websites erzeugt werden. Damit Google Charts geladen werden kann, wird zuerst der sogenannte Google API loader benötigt. Dafür ist es nötig im Kopf des HTML Dokuments folgende Zeile hinzuzufügen:

```
1. <script type="text/javascript" src="https://www.google.com/jsapi"></script>
```

Beispiel 13: Google API loader einbinden

Danach können die benötigten Module geladen werden. Für Google Charts ist es das Modul „visualization“ sowie das Paket „corechart“. Das anschließende Codebeispiel zeigt den Befehl, über den das Modul geladen werden kann. (vgl. [GOOGLE, 2014])

```
1. google.load('visualization', '1.0', { 'packages': ['corechart'] });
```

Beispiel 14: Visualisierungsmodul laden

Die zu visualisierenden Daten werden in eine Tabelle eingetragen. Bei Tortendiagrammen wird in die erste Spalte die Bezeichnung eingetragen, die für den jeweiligen Sektor angezeigt wird, in die zweite Spalte der Wert. Außerdem können noch verschiedene Optionen, wie die Größe, der Titel oder die Farben für die Sektoren mitgegeben werden. Diese Informationen werden dann an den Google Charts Service gesendet und das erzeugte Diagramm in ein Containerelement gezeichnet. Der nachfolgende Quellcode zeigt welche Aufrufe dafür nötig sind. (vgl. [GOOGLE, 2015])

```
1. var chart =
2.       new google.visualization.PieChart(document.getElementById('chart'));
3. chart.draw(dataTable, options);
```

Beispiel 15: Google Chart zeichnen

Google Charts verwendet für das Aufbauen von Diagrammen normalerweise Scalable Vector Graphics (SVG) also Vektorgrafiken, die wie alle anderen Vektorgrafiken beliebig ohne Qualitätsverlust skalierbar sind. SVG ist neu in HTML5 und noch nicht in jedem Browser umgesetzt. Für ältere Internet Explorer Versionen wird stattdessen Vector Markup Language (VML) verwendet. (vgl. [GOOGLE, 2015])

7 Architekturen

7.1 Drei-Schichten-Architektur

In der Software Entwicklung empfiehlt es sich, die Benutzeroberfläche und die Geschäftslogik zu trennen. Ebenso sollte die Datenhaltung von der restlichen Anwendung ausgelagert werden. Durch diese Anforderungen ergibt sich eine Architektur mit drei Schichten. Die drei Schichten sind hierarchisch nach ihrem Abstraktionsniveau aufgebaut. Dabei kann immer die höher gelegene Schicht auf die darunterliegende zugreifen. (vgl. [BALZERT, Helmut, 2010])

7.1.1 Präsentationsschicht

Die Präsentationsschicht enthält alle Elemente die benötigt werden, damit der Benutzer mit dem System interagieren kann. Dazu gehört in erster Linie die Benutzeroberfläche. Dies fasst alle Komponenten zusammen, die entweder zur Anzeige von Informationen oder zur Eingabe des Benutzers dienen. Außerdem wird eine Lösung benötigt um die Daten der darunterliegenden Schichten zu konsumieren. Diese werden danach so bearbeitet, damit sie von der Benutzeroberfläche dargestellt werden können. (vgl. [MICROSOFT, MSDN, 2015])

7.1.2 Geschäftslogikschicht

Die Geschäftslogikschicht ist das Verbindungsstück von Präsentations- und Datenhaltungsschicht. Sie benötigt daher auf der einen Seite eine Komponente, die es erlaubt auf die Daten zuzugreifen. Andererseits muss der Präsentation eine Möglichkeit geboten werden ihrerseits auf die Geschäftslogik zuzugreifen. Dafür soll eine Schnittstelle erstellt werden, die die gesamte benötigte Funktionalität zur Verfügung stellt, aber dennoch einfach gehalten wird. Die Präsentationsschicht hat keinen Zugriff auf die Details dieser Schicht. Dadurch verringert sich die Abhängigkeit der beiden Schichten, so können beispielsweise Änderungen an der Geschäftslogik vorgenommen werden ohne die Präsentationsschicht anpassen zu müssen. (vgl. [MICROSOFT, MSDN, 2015])

7.1.3 Datenhaltungsschicht

Die Datenhaltungsschicht sorgt dafür, dass die Informationen, die in den oberen Schichten nur im Arbeitsspeicher gehalten werden, dauerhaft gespeichert werden. Damit bestehen sie auch nach Ende der Laufzeit des Programmes und man spricht von persistenten Daten. Die Datenhaltung wird meist über eine Datenbank umgesetzt.

7.2 Model-View-Controller Entwurfsmuster

Das Model-View-Controller Entwurfsmuster teilt die Objekte einer Applikation in drei Rollen auf. Das Model, auf Deutsch Modell, die View oder Präsentation und der Controller, auf Deutsch Steuerung. Diese sollen möglichst unabhängig voneinander sein und nur durch definierte Schnittstellen miteinander kommunizieren.

7.2.1 Model

Das Model stellt die Daten für die Applikation zur Verfügung. Zu den Aufgaben zählt neben dem Beziehen auch die Prüfung der Daten sowie das Bündeln von zusammengehörigen Daten. Wie die Daten konkret dargestellt werden, liegt nicht in der Verantwortung des Models. (vgl. [CAKE SOFTWARE, 2014])

7.2.2 View

In der View sind alle Objekte die für den Benutzer sichtbar sind. Dazu gehören neben Benutzeroberflächenelementen auch Bilder oder Videos. Die View sollte so von den anderen Objekten getrennt sein, dass sie ohne viel Aufwand austauschbar ist. Wenn verschiedene Benutzerschnittstellen für verschiedene Plattformen benötigt werden, können Model und Controller wiederverwendet werden. (vgl. [APPLE, 2013])

7.2.3 Controller

Der Controller verarbeitet Benutzeranfragen und regelt die Kommunikation zwischen Model und View. Der Controller sollte sowohl von dem zugrundeliegenden Model als auch von der verwendeten View unabhängig sein. Diese Komponente erkennt und interpretiert Änderungen im Model und leitet diese an die View weiter. Außerdem können die Daten im Model über den Controller verändert werden. (vgl. [APPLE, 2013])

8 Evaluierung

Der Auftraggeber wollte die in Frage kommenden Entwicklungsmethoden auf bestimmte Kategorien untersuchen, bevor die verwendete Entwicklungsmethode festgelegt wird. Deshalb wurde eine ausführliche Recherche durchgeführt. Es wird auf Qt, die native iOS Entwicklung und die Entwicklung mit HTML eingegangen.

8.1 Kriterien

Hier werden kurz die Kriterien beschrieben, die für die Evaluierung im Zuge dieser Diplomarbeit wichtig waren.

8.1.1 Einheitlicher Code für unterschiedliche Auflösungen

Da es viele verschiedene mobile Geräte mit unterschiedlichen Auflösungen gibt und die Anwendung auf jedem Gerät gleich angezeigt werden soll, muss eine Möglichkeit gefunden werden, die Darstellung unabhängig vom Bildschirm zu definieren. Hierfür gibt es für unterschiedliche Entwicklungsmethoden verschiede Ansätze, auf die in den jeweiligen Kapitel eingegangen wird.

8.1.2 Entwicklungsumgebung

Hier wird betrachtet welche Voraussetzungen das System haben soll um mit dieser Methode entwickeln zu können. Es wird beschrieben auf welchem Betriebssystem gearbeitet werden muss. Außerdem wird auf die verwendbaren Entwicklungsumgebungen eingegangen.

8.1.3 Developer-Lizenzen

Hier werden alle Lizenzen aufgelistet die von der Entwicklung der Anwendung über das Testen bis zur Fertigstellung und Veröffentlichung des Produkts benötigt werden.

8.1.4 Community Support

Es soll außerdem recherchiert werden, wie groß die Entwicklergemeinschaft in dem entsprechenden Bereich ist und ob es aktive Foren gibt. Im Besonderen soll beachtet werden ob und welche Bibliotheken verfügbar sind, die es erlauben Diagramme darzustellen.

8.1.5 Wiederverwendbarkeitsgrad

Hier wird bearbeitet in wie weit der Programmcode für verschiedene Plattformen verwendet werden kann. Generell hängt der Wiederverwendbarkeitsgrad vom eingesetz-

ten Datentrennungskonzept ab. Wir sollten hierfür das Model-View-Controller Entwurfsmuster anwenden, daher wird auch die Umsetzung dieses Musters in der zu betrachtenden Technologie untersucht.

8.1.6 Deployment Prozess

Hier werden alle Schritte beschrieben, die nötig sind, um das Produkt auf die Geräte auszuliefern.

8.2 Qt

8.2.1 Einheitlicher Code für unterschiedliche Auflösungen

In Qt gibt es die Möglichkeit die Elemente relativ zu positionieren. Dazu können vorgefertigte Layouts herangezogen werden um Standardlösungen wie Gitter- oder Listenanordnung zu erhalten. (vgl. [QT, 2015]) Sollte das nicht reichen, gibt es in Qt sogenannte Anchors, über die jedes Element in Relation zu anderen Elementen positioniert werden kann. (vgl. [QT, 2015]) Qt bietet auch die Möglichkeit die Abmessungen für bestimmte Komponenten für verschiedene Auflösungen in eigenen Files zu speichern.

8.2.2 Entwicklungsumgebung

Mit der IDE QtCreator kann die App unabhängig von der Zielplattform entwickelt werden. Wenn man jedoch die Apps testen will, muss man verschiedene andere Entwicklungsumgebungen wie Visual Studio und Xcode konfigurieren, um diese anschließend verwenden zu können. Bei iOS muss mit Xcode gearbeitet werden, dafür ist ein Mac PC als Entwicklungsgerät nötig. In Xcode ist ein Simulator bereits integriert. (vgl. [QT, 2015])

Für Windows Phone Apps gibt es ein Visual Studio Plugin von Qt. Dieses muss verwendet werden, da der QtCreator noch kein Debugging auf dem Windows Phone Emulator oder auf Geräten ermöglicht. Dafür benötigt man ein Windows Gerät. (vgl. [QT, 2015])

Nur die Android App kann direkt aus dem QtCreator sowohl auf einem Emulator als auch auf einem Android Gerät getestet werden. Hierbei ist auch das verwendete Betriebssystem beliebig wählbar. (vgl. [QT, 2015])

8.2.3 Developer-Lizenzen

Für das Testen der Applikationen am Gerät sind bei iOS Apps und bei Windows Phone Developer Accounts nötig. Jedoch ist es bei Windows Phone Apps möglich, ein einziges Gerät kostenlos für die Entwicklung freizuschalten.

Auf der Homepage [5] ist eine kostenlose Version des QtCreators zu finden und weitere 3 kostenpflichtige. Laut einer Auflistung auf der Seite ist es mit der freien Community

Version nicht möglich, Apps über den öffentlichen App Store zu verteilen. Eine Lizenz für eine IDE bei der das möglich ist, ist die Mobile Indie Lizenz, die 25 $ pro Monat kostet. Außerdem gibt es noch die Professional Lizenz mit deutlich mehr Funktionen für 174 $ pro Monat. (vgl. [QT, 2015])

8.2.4 Community-Support

Es gibt einige Foren, sowohl in Deutsch als auch in Englisch. Verglichen mit den beiden anderen angesehenen Technologien hat Qt aber die kleinste Community.

Qt-Charts ist in der Professional Version enthalten und ist eine einfache Möglichkeit Diagramme zu erzeugen. Erzeugt werden Diagramme mit Qt Meta Language (QML) Objekten, Objekten mit denen in Qt Komponenten der Benutzerschnittstelle definiert werden (vgl. [QT, 2015]). Diese Objekte erlauben es auf Events, beispielsweise Klicks auf einen Sektor, zu reagieren. (vgl. [QT, 2015])

8.2.5 Wiederverwendbarkeitsgrad

Mit Qt kann der Code großteils für alle Geräte verwendet werden. Für native Zugriffe müssen jedoch einzelne Teile für jede Plattform einzeln implementiert werden.

Qt unterstützt das „model/view“ Entwurfsmuster, welches das Model und den Controller vom MVC zusammenfasst. Obwohl Qt zwischen Model und Controller nicht unterscheidet, sollte der Code getrennt sein. Der Controller kommuniziert mit der View über Signals und Slots. Ein System, das sich mit dem Observer/Observable Pattern vergleichen lässt. Die View, welche die Signale sendet, lässt sich mit dem UI Editor erzeugen. (vgl. [QT, 2015])

8.2.6 Deployment-Prozess

iOS

Hierzu benötigt man Xcode auf einen Mac OS Rechner und den Enterprise Developer Account. Mit qmake kann das Projekt als .xcodeproject exportiert werden. Dieses kann in Xcode importiert werden und wie ein gewöhnliches Xcode Projekt verteilt werden. (vgl. [QT, 2015])

Windows Phone

Mit qmake kann das Projekt in ein Visual Studio Format umgewandelt werden. Danach kann mit Visual Studio die App sowohl auf einen Emulator als auch auf einem Windows Phone ausgeführt werden. (vgl. [QT, 2015])

Android

Um eine App auf Android installieren zu können wird ein verteilbares Android Package (APK) benötigt. Dies kann mit dem QtCreator erstellt werden. (vgl. [QT, 2015])

8.3 iOS Native

8.3.1 Einheitlicher Code für unterschiedliche Auflösungen

Diese Anforderung ist komfortabel gelöst. Es gibt zum Entwickeln ein vom Gerät unabhängiges Koordinatensystem, das vom Gerät auf die Auflösung übertragen wird. Auf diesem können die Elemente angeordnet werden. Bilder sollen in zwei Versionen verwendet werden, zusätzlich einmal in doppelter Größe für Geräte mit höherer Auflösung. (vgl. [APPLE, 2012])

8.3.2 Entwicklungsumgebung

Native iOS Apps sind nur auf iOS Geräten lauffähig. Zum Entwickeln ist ein Mac OS PC mit Xcode notwendig. Die Entwicklungsumgebung Xcode enthält außerdem einen Simulator, auf dem die Apps getestet werden können. Registrierte Entwickler können auch direkt mit Xcode die App signieren und freigeben.

8.3.3 Developer-Lizenzen:

Um die entwickelten Apps für die Geräte freigeben zu können, muss der Entwickler als solcher registriert sein. Hierfür gibt es zwei Möglichkeiten, zum einen die Standard Lizenz, mit der die Apps am App Store angeboten werden können, zum anderen die Enterprise Lizenz. Letztere wird für Unternehmen angeboten, die die Apps im eigenen Unternehmen verteilen wollen. Die Enterprise Version ist mit jährlich 299 $ deutlich teurer als die Standardversion für jährlich 99 $. (vgl. [APPLE, 2015])

8.3.4 Community-Support

Die iOS Entwicklung hat eine sehr große Community, mit vielen Foren sowohl in englischer als auch deutscher Sprache.

Um Diagramme zu erzeugen, gibt es die Bibliothek Core-Plot, die einfach gehalten ist und Diagramme mit schlichtem Aussehen erzeugt. Für Tortendiagramme gibt es leider wenig Anpassungs- und Gestaltungsmöglichkeiten. Daneben gibt es noch eine Reihe kostenpflichtiger Bibliotheken, die wesentlich mehr Funktionalität bieten.

8.3.5 Wiederverwendbarkeitsgrad

Da der Code nur für iOS Geräte verwendet werden kann, wird hier nur auf die Design Patterns eingegangen. Apple empfiehlt die Verwendung des MVC Patterns. Da Cocoa Touch, das Framework mit dem mobile Apps gemacht werden können, mit MVC aufgebaut ist, ist es einfach zu verwenden. In Xcode gibt es den Interface Builder, mit dem das UI bequem aufgebaut werden kann. Für jede View, die im UI erzeugt wird, wird im Hintergrund eine Controller Klasse angelegt. Die Daten sollen in Modell Objekten gekapselt werden. (vgl. [APPLE, 2013])

8.3.6 Deployment-Prozess

Voraussetzung sind eine Apple ID, ein gewöhnlicher Login mit E-Mail, und die iOS Developer Enterprise Program Lizenz. Mit dieser kann ein Team erstellt werden, in dem jeder sein Gerät, mit dem er entwickelt, registrieren muss. Danach muss ein „team provisioning profile“ erstellt werden. Mit diesem kann jeder im Team seine Apps signieren, damit sie auf einem Apple Gerät laufen können. Jedes Teammitglied muss dann ein Code Signing Certificate anfordern, die der Administrator bestätigen muss.

MDM: Mit Xcode wird ein Archive-File erzeugt und gemeinsam mit einer Manifest-Datei auf den MDM Server mit einem Downloadlink geladen. Dieser Server verteilt die Apps auf die Geräte der Mitarbeiter. Updates werden automatisch durchgeführt.

Ohne MDM: Die App kann als Archive über E-Mail oder über das Firmennetz verteilt werden. Ohne MDM müssen Updates immer eigens verteilt werden.

8.4 HTML5

Bei HTML5 Apps muss zuerst zwischen zwei verschiedenen Arten der App-Entwicklung unterschieden werden: Zum einen die Entwicklung einer mobilen Website zum andern einer hybriden App.

8.4.1 (Mobile) Website:

Bei dieser Variante wird eine einfache Website entwickelt, welche dann entweder gänzlich auf mobile Geräte ausgerichtet ist oder mittels eines Frameworks für responsives Webdesign sowohl für die Darstellung auf PCs als auch im Browser auf mobilen Geräten optimiert ist.

Hierbei müsste ein Webserver eingerichtet werden, der nur über eine VPN-Verbindung mit dem Firmennetz erreichbar ist, um unberechtigte Zugriffe auf die Daten zu verhindern.

Die HTML-Seiten können auch lokal temporär gespeichert werden, wodurch die Performanz gesteigert wird.

8.4.2 Hybride App:

Eine hybride App ist für einen User nichts anderes als eine normale (native) App, welche vom jeweiligen App Store heruntergeladen werden kann. Lediglich die für jedes Betriebssystem typische Benutzeroberfläche unterscheidet sich, da die Oberfläche rein aus HTML besteht.

Dies ist möglich, da jede Plattform ein Layout-Element zur Verfügung stellt, welches zur Darstellung von HTML mit JavaScript und CSS dient.

Mit Hilfe von Cordova können die HTML-Seiten in die Applikation eingebunden werden. Dies ermöglicht, dass die Apps auch offline, sprich ohne Internetverbindung, verwendet werden können.

8.4.3 Einheitlicher Code für unterschiedliche Auflösungen

In HTML gibt es eine große Palette von Möglichkeiten, wie das Layout den verschiedensten Anforderungen angepasst werden kann. So kann für jedes Element entweder eine fixe Größe festgelegt werden oder die Größe im Verhältnis zum übergeordneten Element. Sollte ein Element eine feste Abmessung benötigen, kann ein anderes Element so formatiert werden, dass es den je nach Bildschirmauflösung größeren oder kleineren restlichen Raum ausfüllt.

8.4.4 Plattform-Unterstützung

Die Unterstützung der einzelnen Plattformen hängt vom verwendeten Browser und dessen Rendering-Engine ab. Die Rendering-Engine ist das Modul eines Browsers, das für die Darstellung der Webinhalte zuständig ist. Dazu parst die Engine das HTML-Dokument und wandelt es gemeinsam mit den Stilinformationen des CSS Codes in die sogenannte Rendering-Struktur um. Danach wird die Rendering-Struktur in UI-Elementen dargestellt. (vgl. [GARSIEL, Tali, Irish, Paul, 2011])

Sowohl Android Geräte, mit einer mobilen Version von Chrome, als auch iOS Geräte mit Safari setzen auf die von Apple entwickelte HTML-Rendering-Engine WebKit auf. Nur Microsoft setzt bei ihrem Windows Phone auf ihre eigene Engine Trident. Die meisten Elemente werden mit beiden Engines gleich dargestellt, daher kann HTML als Plattformunabhängig bezeichnet werden. Allerdings gibt es, vor allem bei neuen Funktionen Unterschiede in der Umsetzung, daher ist bei der Entwicklung zu beachten, diese Verschiedenheiten zu kennen und den Code immer auf allen Geräten zu testen.

8.4.5 Entwicklungsumgebung

Bei der Entwicklung von Websites stehen viele verschieden Entwicklungsumgebungen zur Verfügung, die alle ihre eigenen Vor- und Nachteile haben. Im Bereich der Freeware gibt es aber zurzeit noch keine wirkliche Umgebung, die nur auf die Entwicklung von HTML, CSS und JavaScript ausgerichtet ist.

Bei den Freeware Angeboten sind hierbei Visual Studio, NetBeans (ab Version 7.3), Aptana Studio, Eclipse (in der Enterprise Edition) und Komodo Edit die empfehlenswertesten Umgebungen. Bei den kostenpflichtigen Produkten gibt es noch JetBrains WebStrom 6, welche nur auf die Entwicklung von JavaScript-Anwendungen ausgerichtet ist. Die meisten dieser Entwicklungsumgebungen sind für verschiedene Betriebssysteme verfügbar.

Es gibt aber auch Web-basierte Entwicklungsumgebungen, in denen der Code gleich getestet werden kann. Diese haben auch den Vorteil, dass mehr oder weniger gleichzeitig mehrere Personen am Code arbeiten können. Ein Beispiel hierfür wäre JSFiddle.

Zum Testen dieser Apps wäre natürlich ein Gerät für jedes der Betriebssysteme am besten, doch wenn man dieses nicht besitzt, kann man sich auch die Emulatoren für jedes einzelne Betriebssystem herunterladen. Für Apple Geräte ist unbedingt ein Mac OS Rechner nötig um den in Xcode verwendeten Simulator gebrauchen zu können.

8.4.6 Community-Support

Die Community von Webentwicklern ist sehr gut und man findet viele Foreneinträge, Blogs und Tutorials in allen verschiedenen Foren und über alle möglichen Problemstellungen.

Auch Cordova hat eine aktive Community. Auf der Homepage von Cordova gibt es ein eigenes Community Portal. Weiters ist dort auch noch ein Cordova Wiki und die Dokumentation sowie einige Tutorials zu finden, wobei zu sagen ist, dass diese Tutorials etwas zu wünschen übrig lassen. Jedoch ist im übrigen Web viel zu auftretenden Fehlern und Problemen zu finden.

Eine Vielzahl von Möglichkeiten stehen auch bei der Auswahl von JavaScript Bibliotheken zur Verfügung. Auch für die spezielle Anforderung, eine Bibliothek zur Darstellung von Diagrammen, gibt es viele Angebote. Darunter das bereits beschriebene Google Charts oder Chart.js. Andere Bibliotheken für diesen Einsatzbereich sind jqPlot, GraphUp und CanvasJS.

8.4.7 Wiederverwendbarkeitsgrad

Der Wiederverwendbarkeitsgrad ist bei dieser Methode der App-Entwicklung sehr hoch, da rein in HTML, JavaScript und CSS programmiert wird. Das kann dann ohne weiteres auch auf allen Plattformen dargestellt werden. Es sind möglicherweise kleinere Anpassungen nötig, da sich die Darstellung der verschiedenen Browser auf den Plattformen etwas unterscheidet.

Um die Business-Logik von der Darstellung zu trennen, gibt es in JavaScript mehrere Ansätze. Zum einen kann man eigene JavaScript-Klassen implementieren, welche dann die einzelnen Rollen darstellen. Zum anderen gibt es auch verschiedene Frameworks wie Backbone.js, Ember.js oder AngularJS. Diese stellen eigene Klassen für Model, View und Controller zur Verfügung, beziehungsweise sind diese Patterns oft etwas abgewandelt.

8.5 Fazit

Die Entscheidung ist in Abstimmung mit dem Auftraggeber auf die Entwicklung einer hybriden App mit HTML5 gefallen, da die Fülle an Bibliotheken viel Funktionalität bie-

tet. Darunter etwa eine große Auswahl von Diagrammbibliotheken. Nativ mit iOS entwickeln wäre aufgrund der fehlenden Plattform Unterstützung nur dann sinnvoll, wenn sehr viel auf plattformspezifische Funktionen zugegriffen werden müsste. Wir hatten uns auch gegen Qt ausgesprochen, da wir damit noch keine Erfahrungen gemacht haben und die Unterstützung der Community hier am geringsten ist.

9 Systemanalyse

Im folgenden Kapitel werden die für die Diplomarbeit relevanten Abläufe im Unternehmen Logicx GmbH und die beteiligten Systemteile genauer beschrieben.

Grundlegend unterteilt das Unternehmen seine Projekte in Meilensteine (auch Arbeitspakete genannt), welche wiederum in einzelne Issues gegliedert werden.

Der Aufwand, der für die Erledigung eines der Issues aufgebracht werden muss, wird in acht Kategorien unterteilt. Diese sind:

- AM (Anforderungsmanagement)
- PM (Planungsmanagement)
- QM (Qualitätsmanagement)
- LM (Lösungsmanagement)
- UM (Umsetzung)
- Test
- Nicht verrechenbar
- Reisezeit

Bei den Kategorien „Nicht verrechenbar“ und „Reisezeit“ ist zu beachten, dass dieser Aufwand nicht dem Kunden in Rechnung gestellt werden kann.

Die Verwaltung dieser Projekte, Meilensteine und Issues erfolgt in einzelnen SharePoint Listen. Auch die Verteilung der Issues auf die Mitarbeiter wird mit SharePoint durchgeführt.

9.1 SharePoint

SharePoint ist eine von Microsoft entwickelte Webanwendung die es Organisationen ermöglicht, Dokumente, Daten und Informationen effizient für mehrere Benutzer erreichbar zu machen. Unter anderem beinhaltet es Funktionen zum Verwalten von Projekten und Koordination von Aufgaben.

Die Projekte, Meilensteine und Issues werden in verschiedenen SharePoint Listen verwaltet. Mit Hilfe der Webanwendung von SharePoint können Projekte, Meilensteine und Issues zu diesen Listen hinzugefügt, aber auch verändert oder gelöscht werden.

Für die Einführung der eigens erstellten Zeiterfassungssoftware Time, welche mit den Projekten, Meilensteinen und Issues aus dem SharePoint System arbeitet, mussten Maßnahmen getroffen werden.

Da der Zugriff von anderen Applikationen auf die SharePoint Datenbank problematische Konflikte auslösen kann, musste das Unternehmen eine Zwischenschicht einführen, die die aktualisierten Daten aus dem SharePoint System in einer Datenbank repli-

ziert. Dabei sollten alle Änderungen an den Daten in den SharePoint Listen auch in die Datenbank übernommen werden.

Aus diesem Grund entschied sich das Unternehmen dazu, einen EventReceiver zu den betroffenen Listen hinzuzufügen, der die veränderten Daten einem WCF Service übergibt, der diese wiederum in der Datenbank aktualisiert.

Ein EventReceiver ist eine Anwendung, die zu einem bestehenden SharePoint System hinzugefügt werden kann und dessen Code bei bestimmten Aktionen (Events) auf SharePoint Listen oder Elemente solcher Listen ausgeführt wird.

9.2 Datenmodell

In diesem Unterkapitel sollen die relevanten Tabellen und deren Attribute der Datenbank der Software Time beschrieben werden.

Man kann die relevanten Tabellen in zwei Bereiche unterteilen. Die Tabellen mit den Daten aus dem SharePoint System (SharePoint_Projekte, SharePoint_Meilensteine, SharePoint_Issues) und die Time spezifischen Tabellen, die für diese Diplomarbeit wichtig sind.

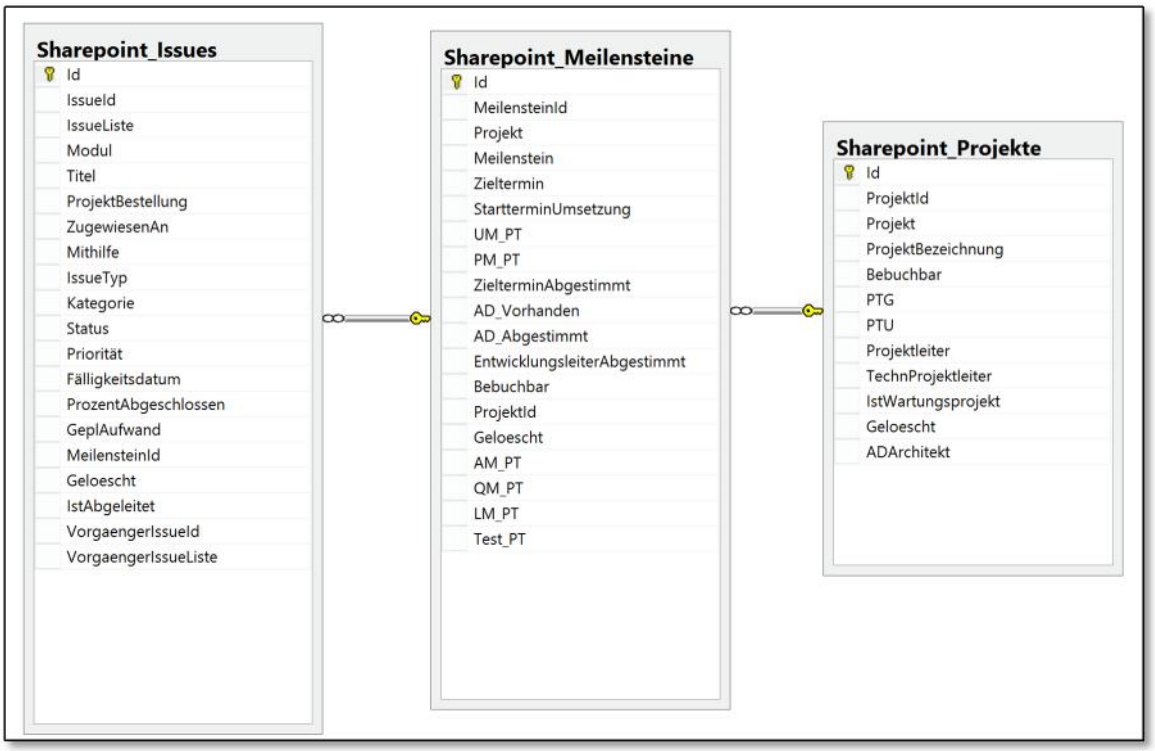

Abbildung 6: Tabellen aus dem SharePoint System

9.2.1 SharePoint_Projekte

In dieser Tabelle befinden sich die Daten der Projekte aus den jeweiligen SharePoint Listen. Es befinden sich lediglich allgemeine Daten des Projektes, wie zum Beispiel die Bezeichnung (Spalten Projekt, ProjektBezeichnung), Projektleiter und Technischer Projektleiter, in dieser Tabelle.

Die für diese Diplomarbeit wichtigen Spalten sind ID, ProjektID, Projekt, Bebuchbar, Projektleiter, TechnProjektleiter, Geloescht.

9.2.2 SharePoint_Meilensteine

Die Tabelle SharePoint_Meilensteine enthält wiederum die Daten aus den SharePoint Listen mit Meilensteinen. Neben allgemeinen Informationen wie der Bezeichnung des Meilensteines (Spalte Meilenstein) befinden sich auch die Planwerte AM_PT, LM_PT, PM_PT, QM_PT, Test_PT und UM_PT in dieser Tabelle.

Diese Planwerte geben den geschätzten Aufwand für die jeweiligen Kategorien an. Der Aufwand für „Nicht verrechenbar“ und „Reisezeit“ wird bei der Schätzung außer Acht gelassen.

In dieser Tabelle sind die Spalten Id, MeilensteinId, Meilenstein, ProjektId, Gelöscht, AM_PT, LM_PT, PM_PT, QM_PT, Test_PT und UM_PT von Bedeutung.

Zu beachten ist das die Spalte ProjektId **nicht** auf die gleichnamige Spalte ProjektId der Tabelle SharePoint_Projekte referenziert, sondern auf die Spalte Id, also den Primärschlüssel. (siehe 11.2.1 SharePoint_Projekte)

9.2.3 SharePoint_Issues

Bei der Tabelle SharePoint_Issues werden einige Daten wie zum Beispiel die genaue Beschreibung der Aufgabe aus den eigentlichen SharePoint Listen der Issues ausgespart und sind nur in selbigen ersichtlich. Der geplante Aufwand für das Issue befindet sich in der Spalte GeplAufwand.

In dieser Tabelle können sich Einträge befinden, die sich lediglich durch die Spalten Id und IssueList unterscheiden. Dies ist darauf zurückzuführen, dass bei der Archivierung von einem der Issues im SharePoint System einfach eine Kopie dieses Issues in eine SharePoint List für archivierte Issues eingetragen wird.

Die für die Diplomarbeit relevanten Daten befinden sich in dieser Tabelle in den Spalten Id, IssuesId, IssueListe, Titel, Kategorie, Status,GeplAufwand, MeilensteinId und Geloescht.

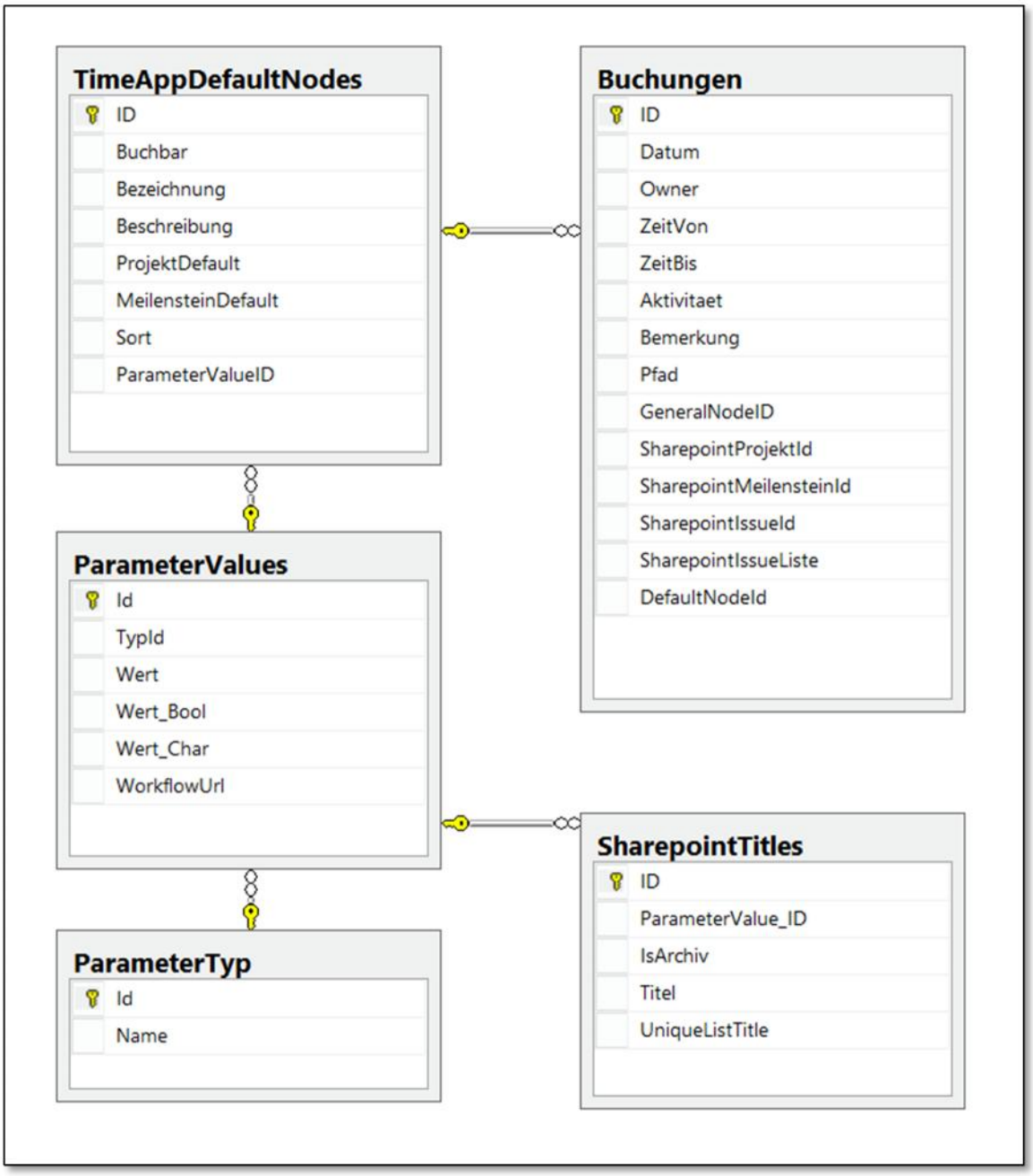

Abbildung 7: Relevante Tabellen von TimeStore

9.2.4 Buchungen

Die Tabelle Buchungen erfasst alle Aufwände, die für die jeweiligen Issues beziehungsweise Meilensteine aufgebracht wurden. Dabei wird der Anfangs- und Endzeitpunkt der Periode eingetragen, in der an einer bestimmten Aufgabe gearbeitet wurde.

Wie genau Zeit für die einzelnen Aufgaben verbucht werden, kann wird im Unterkapitel Zeiterfassungssoftware näher erklärt. (siehe 9.3 Zeiterfassungssoftware)

Für die Diplomarbeit sind die Spalten Datum, ZeitVon, ZeitBis, SharepointProjektId, SharepointMeilensteinId, SharePointIssueId, SharepointIssueListe und DefaultNodeId von Bedeutung.

In dieser Tabelle verweist die Spalte SharepointProjektId auf die Spalte ProjektId der Tabelle SharePoint_Projekte. Die Spalte SharepointMeilensteinId verweist auf die Spalte MeilensteinId in der Tabelle SharePoint_Meilensteine und die Spalte SharepointIssueId auf die Spalte IssueId der Tabelle SharePoint_Issues.

9.2.5 SharePointTitles

In der Tabelle SharePointTitles werden die verschiedenen SharePoint Listen der Projekte, Meilensteine und Issues gepflegt. Ein Issue kann in zwei verschiedenen Listen sein, da er bei der Archivierung in eine Archive Liste hinzugefügt wird. Deshalb müssen die Aufwände beider Listen berücksichtigt werden.

Die Spalten ParameterValue_ID und UniqueListTitle werden von dieser Tabelle für die Diplomarbeit benötigt.

9.2.6 TimeAppDefaultNodes

In der Tabelle TimeAppDefaultNodes werden die Kategorien gespeichert, die für die Darstellung in der Time Applikation verwendet werden.

Für diese Diplomarbeit sind die Spalten Bezeichnung und ParameterValueId dieser Tabelle von Bedeutung.

9.2.7 ParameterValues/ParameterTyp

In den beiden Tabellen ParameterValues und ParameterTyp werden Metadaten für die Time Applikation gespeichert und beschrieben.

9.3 Zeiterfassungssoftware

Die eigens im Unternehmen entwickelte Zeiterfassungssoftware Time bietet den Mitarbeitern die Möglichkeit, die aufgewendete Zeit für eine Tätigkeit auf den dazugehörigen Issues aus dem SharePoint System beziehungsweise aus der Datenbank zu buchen.

Die daraus erhobenen Daten werden dann anschließend für die Verrechnung an den Kunden und für andere Berichte herangezogen.

Die Anwendung bietet eine Auflistung aller Projekte mit den dazugehörigen Meilensteinen und Issues. Die Issues werden aber nicht direkt nach den Meilensteinen aufgelistet, sondern sie werden unterteilt nach ihrem Status (Zu Testen, Zurückgestellt, Wartet auf jemand anderen, Nicht begonnen, In Bearbeitung, Getestet, Abgeschlossen) aufgeführt. Zudem werden bei jedem Meilenstein auch noch die Kategorien (siehe Kapitel 9) aufgeführt.

Nun kann entweder auf einen Issue oder auf eine Kategorie eines Meilensteines gebucht werden. Buchungen auf Projekte oder direkt auf Meilensteine sind nicht möglich.

So ergeben sich nur zwei Fälle, wie die in Punkt 9.2.4 aufgeführten relevanten Spalten gefüllt sein können. Die Spalten ShareponitProjektId und SharepointMeilensteinId müssen immer befüllt sein. Nur bei den Spalten SharepointIssueId, SharepointIssue-Liste und DefaultNodeId muss unterschieden werden: entweder sind die beiden Spalten SharepointIssueId und SharepointIssueListe befüllt und die DefaultNodeId leer, zum Beispiel bei einer Buchung auf ein Issue, oder umgekehrt bei einer Buchung auf eine Kategorie eines Meilensteines.

10 Konfiguration der Entwicklungsumgebungen

In diesem Kapitel soll beschrieben werden, welche Schritte durchgeführt werden müssen, um die eigene Entwicklungsumgebung so zu konfigurieren, dass man bequem mit Cordova arbeiten kann. In den folgenden Absätzen werden jene Abläufe beschrieben die in dieser Arbeit zur Konfiguration der Umgebung angewandt wurden. Es werden auch alternative Möglichkeiten erwähnt. Einige Schritte, die auszuführen sind, bleiben für alle Plattformen gleich, jedoch müssen noch vor der Installation des CLI einige Schritte gemacht werden, wie zum Beispiel die Installation der jeweiligen SDKs.

Die SDKs der jeweiligen Plattform müssen nur dann auf dem System installiert werden, wenn diese Plattform auch auf dem jeweiligen System kompiliert beziehungsweise getestet werden soll. Die Projekte der jeweiligen Plattformen können auch ohne dem SDK erstellt werden.

Dies wird in den folgenden Unterkapiteln 10.1 bis 10.4 beschrieben.

10.1 Windows Phone 8.1

10.1.1 Betriebssystem

Um für Windows Phone 8 Applikationen zu entwickeln, ist eine Windows 8 Umgebung notwendig. Windows Phone 8 Apps können nicht mit einem Windows 7 oder niedrigerem Betriebssystem implementiert werden. Das Betriebssystem muss dabei aber nicht direkt auf dem Rechner installiert sein, sondern kann auch virtualisiert werden. Bei der Entwicklung von Windows Phone 8.1 Apps mit Mac Geräten bestehen daher folgende Möglichkeiten:

Virtuelle Maschine

Zum einen kann man sich im Mac OS (Macintosh Operating System) einen Hypervisor installieren und damit dann das Windows 8 Betriebssystem nachbilden. Solche Hypervisor für Mac Systeme sind zum Beispiel VMWare Fusion, Parallels Desktop oder Virtual Box.

Ein Hypervisor, auch Virtual Machine Monitor genannt, ist eine Software die es ermöglicht, eine virtuelle Umgebung bestehend aus Hardwareressourcen wie CPU, Speicher und Festplattenplatz zu definieren und diese dann als Basis für die Installation eines Gastbetriebssystems zu nutzen.

Boot Camp

Die zweite Möglichkeit um auf Mac Geräten für Windows Phone zu programmieren ist die Installation von Windows 8 neben dem jeweiligen Mac OS. Da dies ohne Hilfsmittel nur mit großem Aufwand zu schaffen ist, stellt Apple den Benutzern das Programm

Boot Camp ab der Mac OS X Version 10.5 zur Verfügung. Diese Software unterstützt den Benutzer bei den Schritten, die für die Installation von Windows auf einem Mac Gerät notwendig sind, wie zum Beispiel Partitionieren und Formatieren. Diese Methode kann jedoch nur angewandt werden, wenn das Gerät einen Intel-Prozessor verwendet.

Im Falle dieser Diplomarbeit wurde auf einem PC mit installiertem Windows 8.1 64-bit entwickelt und getestet.

10.1.2 SDK

Nach der Installation des Betriebssystems ist es notwendig, das Windows Phone 8.0 SDK herunterzuladen und zu installieren. Dieses kann entweder bei einem bestehenden Visual Studio 2013 nachinstalliert werden oder im Zuge der Neuinstallation von Visual Studio 2013 hinzugefügt werden. Bei dem verwendeten Visual Studio muss mindestens das Update 2 installiert sein. Der Dialog zur Installation des SDKs, welcher in folgender Abbildung (siehe Abbildung 8) angedeutet ist, ist bei der Nachinstallation sowie bei der Neuinstallation ähnlich aufgebaut. Zum nachträglichen Hinzufügen kann dieser Dialog über die Windows Funktion „Programme und Features“ erreicht werden. Dazu muss nur nach einem Rechtsklick auf die jeweilige Visual Studio Installation in der Liste der Programme im Kontextmenü der Eintrag „Ändern“ ausgewählt werden. Danach folgt ein Klick auf den Button „Modify“.

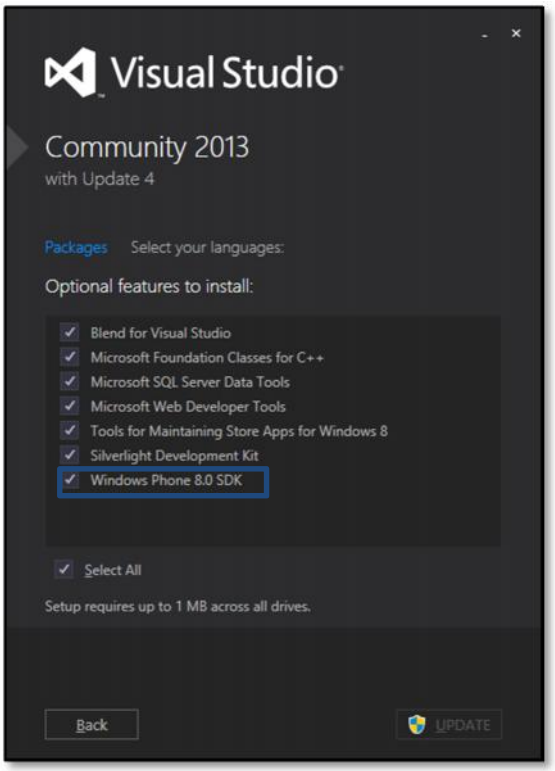

Abbildung 8: Windows Phone 8.0 SDK Installation

Die jeweiligen Versionen des Windows Phone SDKs und die dazugehörigen Emulatoren und Entwicklungsumgebungen sind im Windows Developer Center verfügbar. [6]

10.1.3 Windows Phone Emulator

Bei der Wahl der Windows 8 Version ist zu beachten, dass man den Emulator nur mit einer Windows Version verwenden kann, welche Hyper-V unterstützt. Solche Versionen sind Windows 8 Pro (Enterprise) 64-bit oder Windows Server 2012.

Hyper-V ist ein Hypervisor von Microsoft für Windows Betriebssysteme.

Zusätzlich gibt es auch bei der Hardware Anforderungen, die eingehalten werden müssen, um den von Microsoft mitgelieferten Emulator verwenden zu können. Der Prozessor des verwendeten Gerätes muss Virtualisierung und Second Level Address Translation (SLAT) unterstützen. Weiters sind 6,5 GB an freiem Festplattenspeicher und 4 GB an Hauptspeicher nötig.

10.1.4 Windows Phone Device

Jedoch ist es nicht unbedingt nötig, den Emulator zum Testen der App zu verwenden. Es besteht natürlich auch die Möglichkeit seine erstellten Apps direkt auf einem Gerät mit Windows Phone 8.1 zu testen. Microsoft erlaubt es, ohne weitere Kosten ein einziges Gerät für das Testen von Windows Phone Apps freizuschalten. Dazu ist lediglich ein Microsoft Account, das SDK sowie natürlich ein Windows Phone 8 Gerät notwendig. (vgl. [APACHE SOFTARE FOUNDATION])

Bevor die Apps auf dem Gerät getestet werden können, muss dieses Gerät aber noch freigeschaltet werden. Dazu wurde mit dem SDK ein Programm mitinstalliert, welches die Registrierung des Geräts vornimmt. Dieses Programm kann unter dem Namen „Windows Phone Developer Registration“ gefunden werden. Vor dem Starten des Tools muss das zu registrierende Gerät per USB Kabel mit dem Rechner verbunden und das Display des Gerätes entsperrt werden. Es ist zu beachten, dass auf dem Handy das richtige Datum und die korrekte Uhrzeit eingestellt sind. Ansonsten wird das Gerät nicht erkannt. Nach einem Klick auf den Button „Register“ wird man aufgefordert sich mit seinem Microsoft Account anzumelden. Nach der erfolgreichen Anmeldung sollte es möglich sein Apps auf dem Gerät zu testen. (vgl. [APACHE SOFTARE FOUNDATION])

Anzumerken ist, dass unter Verwendung eines kostenlosen Accounts (DreamSpark oder nur ein Microsoft Account, wie in unserem Fall) lediglich drei verschiedene selbst erstellte Applikationen gleichzeitig auf einem Gerät installiert sein können. Verfügt man über einen regulären Developer Account, so liegt das Limit bei 10 Apps. Microsoft führt diese Möglichkeit unter der Bezeichnung „Sideloaded Apps“. Sie ermöglicht es einem, selbst erstellte Apps direkt von seinem Rechner auf dem Windows Phone zu installieren, ohne die App im App-Store zu veröffentlichen.

Zum Testen der Windows Phone App wurde in dieser Arbeit ein Windows Phone 8.1 verwendet, da aufgrund der nötigen Version von Windows 8 die Anforderungen des Emulators nicht erfüllt werden konnten. Dies bedeutete aber keine Einschränkungen, da das Testen auf einem Gerät in jedem Fall schneller ist und ohnehin in einem Punkt der Entwicklung notwendig wird, um praxisbezogen testen zu können.

10.2 Android

10.2.1 Betriebssystem

Anders als bei Windows Phone können Android Apps mit mehreren verschiedenen Betriebssystemen entwickelt werden. Man kann zwischen Mac OS, Linux und Windows wählen. An diese Betriebssysteme werden von Google auch einige Anforderungen gestellt. So ist bei Windows mindestens die Version XP nötig. Bei Mac OS kann erst ab der Version 10.8.5 Android entwickelt werden und bei Linux ist ein GNOME beziehungsweise KDE Desktop notwendig, sowie die GNU C Bibliothek in der Version 2.15 oder höher. (vgl. [APACHE SOFTWARE FOUNDATION])

Wir haben uns bei der Entwicklung der Android App für Windows entschieden, da wir mit diesem Betriebssystem und dessen Umgebung am besten vertraut sind. Jedoch werden alle weiteren nötigen Schritte auch für Mac OS Rechner beschrieben.

Vor dem Einrichten der SDKs ist es notwendig, das Java Development Kit (JDK) zu installieren, falls dieses noch nicht auf dem Rechner vorhanden ist.

Zur Programmierung von Android Applikationen mit Cordova ist eine JDK Version von 7 oder höher notwendig. [7] (vgl. [APACHE SOFTWARE FOUNDATION])

Die als .exe- oder .dmg-Datei vorliegende Installationsdatei ist auszuführen und installiert das JDK. Nach der Installation des JDKs muss der Installationspfad noch zur PATH Variable hinzugefügt werden. Falls die standardmäßigen Einstellungen während der Installation beibehalten wurden, sieht der Pfad zum bin Verzeichnis des JDKs folgendermaßen aus:

Windows:

C:\Program Files\Java\jdk1.7.x_xx\bin

Mac OS:

/Library/Java/JavaVirtualMachines/jdk1.7.x_xx/Content/Home/bin

Mithilfe folgenden Befehls kann bei Mac OS X der Installationspfad des JDK 1.7 ausgegeben werden:

/usr/libexec/java_home –v 1.7

Bei Windows sollte zusätzlich noch eine weitere Umgebungsvariable angelegt werden. Diese Variable muss die Bezeichnung JAVA_HOME tragen und speichert den Installationspfad des JDKs. Dieser Pfad sieht bei Standardeinstellungen wie folgt aus:

C:\Program Files\Java\jdk1.7.x_xx

Mit Hilfe des Befehls java –version kann in einer Kommandozeile überprüft werden, ob Java richtig installiert wurde und die PATH Variable korrekt editiert wurde. Als Ergebnis sollte die Version des installierten JDKs aufscheinen.

10.2.2 SDK

Das Android SDK kann im Gegensatz zu Windows Phone auch ohne dazugehörige IDE als Stand-alone Tool heruntergeladen werden. Auf der Seite von Android Developers sind die jeweiligen Dateien für die Installation des SDK zugänglich: [8]

Die nötigen Schritte zur Installation bei Windows und Mac OS werden im Folgenden kurz beschrieben:

Windows

Für die Installation des SDKs auf einem Windows Gerät wird von dem oben genannten Download Link die .exe-Datei heruntergeladen und anschließend installiert. Die .exe-Datei überprüft ob bereits ein JDK auf dem Rechner vorhanden ist und installiert dieses gegebenenfalls. Nach der Fertigstellung wird der Android SDK Manager geöffnet.

Falls, wie schon weiter oben im Text beschrieben, das JDK bereits installiert wurde, kann auch die .zip-Datei heruntergeladen werden und in einen geeigneten Ordner entpackt werden.

Mac OS

Zum Installieren des Android SDKs auf Mac OS Geräten muss die dafür vorgesehene .zip-Datei unter dem vorherigen Downloadlink heruntergeladen werden. Anschließend wird die .zip-Datei in einen geeigneten Ordner entpackt.

Damit die CLI von Cordova mit dem SDK arbeiten kann, müssen die beiden Unterverzeichnisse des Android SDK Installationsordners „tools“ und „platform-tools“ zu der PATH Variable hinzugefügt werden. (vgl. [APACHE SOFTWARE FOUNDATION])

Ist das Android SDK erfolgreich installiert worden, so müssen noch die benötigten Tools und Packages nachinstalliert werden. Standardmäßig beinhaltet das Android SDK nicht alle Komponenten, die zur Entwicklung von Android Applikationen benötigt werden. Diese Pakete können mit Hilfe des Android SDK Managers nachgeladen werden. Der Android SDK Manager ist ein Programm, welches einem ermöglicht, die benötigten Tools und Plattformen durch die Bedienung eines User Interfaces bequem nachzuinstallieren. Dieser SDK Manager ist entweder im Unterverzeichnis „tools“ des Installationsordners des SDKs durch einen Doppelklick auf die Datei android.bat zu starten oder kann in der Kommandozeile durch den Befehl „android“ ausgeführt werden, vorausgesetzt die Unterordner befinden sich in der PATH Variable wie vorhin beschrieben.

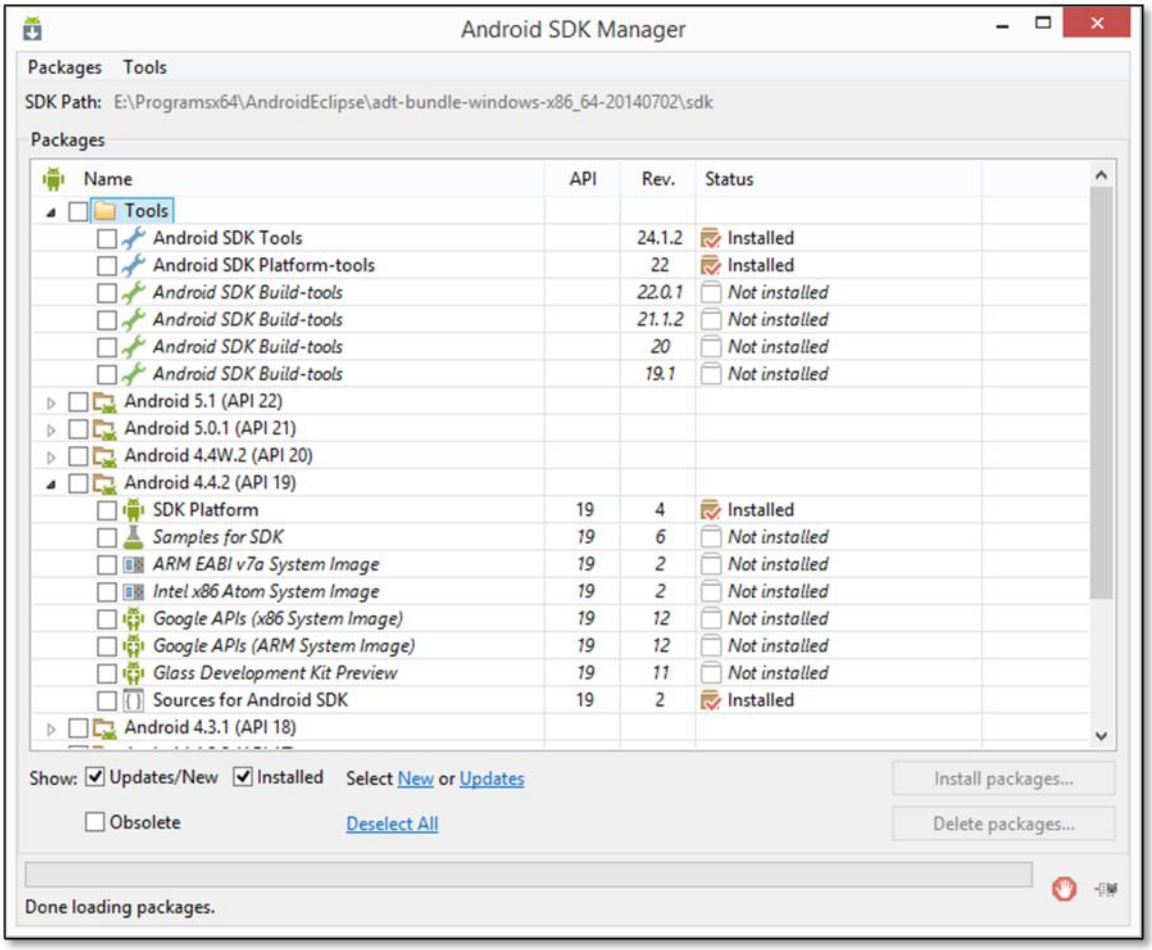

Abbildung 9: Android SDK Manager

Der SDK Manager unterteilt die Pakete in Tools und die einzelnen Plattformen oder API Levels. Eine Plattform oder eine API Level entspricht dabei einer Version von Android. Die Version „KitKat“ entspricht zum Beispiel Android 4.4 bis 4.4.4 und dem API Level 19. Die Pakete SDK Tools und SDK Platform-tools und die jeweilige Android SDK Version müssen zur Programmierung vorhanden sein, wobei die SDK Tools bereits bei der Installation mit heruntergeladen wurden. Die SDK Platform-tools und die SDK Versionen der Plattformen müssen händisch nachinstalliert werden. Hierbei ist zu empfehlen, immer die aktuellste Plattform Version zu verwenden. Es ist immer noch möglich seine Apps mit einer geringeren Plattform Version auszuführen. (vgl. [GOOGLE])

10.2.3 Apache Ant

Apache Ant ist eine Java Bibliothek und gleichzeitig ein Kommandozeilenprogramm zum Steuern von verschiedenen Prozessen, die durch Ziele und Aufgaben definiert werden können. Solche Abläufe werden in Build Dateien beschrieben. Das Apache Ant Project wird hauptsächlich zum Kompilieren und Testen von Java Applikationen verwendet, kann aber auch zum Erzeugen von C oder C++ Applikationen verwendet wer-

den. Cordova verwendet Apache Ant zum Kompilieren des Android Projektes. (vgl. [APACHE SOFTWARE FOUNDATION, 2014])

Nachdem das JDK und das Android SDK installiert sind, muss noch Apache Ant auf dem System konfiguriert werden, um es Cordova zu ermöglichen, die Android Projekte zu kompilieren. Dazu muss lediglich die neueste Version von der offiziellen Apache Ant Project Website heruntergeladen werden. [9]

Die heruntergeladene .zip-Datei muss anschließend entpackt werden und das bin Unterverzeichnis des Installationsordners muss wieder zur PATH Variable hinzugefügt werden. Um die Konfigurationen zu prüfen, kann folgender Befehl in der Kommandozeile ausgeführt werden.

```
ant –version
```

Falls die korrekte Version ausgegeben wird, ist Apache Ant einsatzbereit.

10.2.4 Android Emulator

Das Android SDK liefert bereits einen Emulator für virtuelle Android Geräte mit. Bevor jedoch der Emulator gestartet werden kann, muss zuerst ein sogenanntes Android Virtual Device (AVD) erstellt werden. Ein AVD definiert die Konfigurationen für den Emulator. In einem solchen AVD werden Eigenschaften wie die Zielplattform, der API Level, die Auflösung und so weiter eingestellt. Um ein solches AVD für eine bestimme Plattformversion erzeugen zu können, ist ein System Image eben dieser Plattformversion nötig. Diese System Images können mit Hilfe des SDK Managers für die verschiedenen Plattformversionen nachinstalliert werden. (vgl. [APACHE SOFTWARE FOUNDATION])

10.2.5 Android Device

Um die App auf einem Android Gerät testen zu können, sollte dieses zumindest Android Version 4.0 haben. Obwohl 2.3 auch unterstützt wird, wird dies nicht empfohlen. (vgl. [APACHE SOFTWARE FOUNDATION]) Zuerst muss USB Debugging am Gerät aktiviert werden. Dies kann unter den Entwickleroptionen gemacht werden. Ab Version 4.2 sind diese standardmäßig versteckt. Um sie anzuzeigen muss unter Eistellungen > Telefoninfo sieben Mal auf das Feld Build-Nummer gedrückt werden. Danach erscheinen die Entwickleroptionen in den Einstellungen. Dort kann ein Häkchen bei USB-Debugging gemacht werden. Beim Entwickeln auf einem Windows PC wird ein USB Treiber für das Gerät benötigt, dieser wird vom Gerätehersteller bereitgestellt. Für viele Geräte funktioniert auch der von Google bereitgestellte Treiber der im Android SDK Manager heruntergeladen werden kann. Entwickler mit einem Mac OS X PC können diesen Schritt überspringen. Wenn das Android Gerät das erste Mal an den PC angeschlossen wird, muss am Android bestätigt werden, dass dem PC vertraut wird und Debugging zugelassen wird. (vgl. [GOOGLE])

10.3 iOS

10.3.1 Betriebssystem

iOS Applikationen können nur mit Apples Betriebssystem Mac OS entwickelt werden. Es ist mindestens die Betriebssystemversion Mac OS 10.9 (Mavericks) nötig und ein Intel basierter Rechner, denn erst ab dieser Version ist das Programm Xcode in der Version 6.0 installierbar. (vgl. [APACHE SOFTWARE FOUNDATION])

10.3.2 SDK

Das iOS SDK kann nur gemeinsam mit der IDE Xcode installiert werden. Es bestehen zwei verschiedene Möglichkeiten Xcode auf seinem Rechner zu installieren. Zum einen kann man es sich vom App Store herunterladen oder direkt von den Apple Developer Downloads. Bei letzterer Möglichkeit muss man jedoch schon über einen Apple Developer Account verfügen. Nach der Installation von Xcode ist das iOS SDK einsatzbereit.

Anzumerken ist, das zum Veröffentlichen der App die neueste Version von Xcode nötig ist.

10.3.3 iOS Simulator

Der iOS Simulator wird automatisch bei der Installation von Xcode mitinstalliert. Wie schon erwähnt, ist darauf zu achten, dass der Rechner einen Intel Prozessor verwenden muss, um den iOS Simulator ausführen zu können.

10.3.4 iOS Device

Damit die implementierte App auf einem iOS Gerät getestet werden kann, müssen einige Voraussetzungen erfüllt sein. Zum einen muss auf dem verwendeten Gerät mindestens die iOS Version 5.0 installiert sein. Dies ist die niedrigste Version, die von Cordova unterstütz wird. Von Cordova werden alle iPad Modelle unterstützt, sowie alle iPhones ab Version 3GS und die iPod Touch Geräte ab der dritten Generation (3d Generation). Weiters muss man Mitglied des Apple iOS Developer Program sein. Nach der Registrierung, sind einige weitere Schritte in Xcode notwendig. Zum einen muss mittels des Development Provisioning Assistant ein „provisioning profile“ erstellt und in Xcode installiert werden. Danach müssen noch einige Änderungen am Gerät und den Einstellungen des Xcode Projektes durchgeführt werden. Da eine ausführliche Beschreibung der nötigen Schritte den Rahmen dieser Arbeit sprengen würde, wird lediglich auf zwei detailliertere Anleitungen verwiesen. [10], [11] (vgl. [APACHE SOFTWARE FOUNDATION])

10.4 Cordova Projekt

Installation von Cordova

Nachdem die Entwicklungsumgebungen soweit konfiguriert sind, dass Cordova auf jedem System lauffähig ist, muss die Installationsdatei von Cordova noch heruntergeladen und installiert werden. Um Cordova auf dem eigenen System installieren zu können, muss zuvor aber noch das Werkzeug Node.js installiert werden. Node.js ist eine auf der von Chrome verwendeten JavaScript Engine V8 aufbauende Plattform zum einfachen Implementieren von effizienten und skalierbaren verteilten Applikationen. [12]

Nach dem Ausführen des Installationsprogrammes kann durch folgende Befehle im Terminal bei Mac OS oder in der Kommandozeile bei Windows überprüft werden, ob das Programm erfolgreich installiert wurde. Es sollten die Version der installierten Programme ausgegeben werden.

```
npm –v
node –v
```

Falls einer der beiden Befehle fehlschlägt, muss der Installationspfad zu der PATH Variable hinzugefügt werden.

Mit Hilfe des mitinstallierten Kommandozeilenprogrammes npm kann mittels folgenden Befehlen Cordova heruntergeladen und installiert werden:

Windows:

```
npm install –g cordova
```

Mac OS:

```
sudo npm install –g cordova
```

Das Flag “-g” definiert dabei, dass Cordova global installiert werden soll. Falls man diesen Parameter nicht mitgibt, wird es nur im Unterordner „node_modules“ des Node.js Installationsverzeichnisses installiert. Die Korrektheit der Installation kann wiederum mittels folgendem Befehl überprüft werden:

```
cordova –v
```

Erzeugen des Projektes

Nach der erfolgreichen Installation kann das CLI von Cordova zur Ausführung aller weiteren Schritte verwendet werden. Zunächst muss ein Cordova Projekt erstellt werden. Dies erfolget mit dem „create“ Befehl:

```
cordova create MobileDashcard at.logicx.MobileDashcard MobileDashcard
```

Der oben aufgeführte Befehl wurde zum Erzeugen des Cordova Projektes dieser Diplomarbeit verwendet. Er beinhaltet die Bezeichnung des Verzeichnisses als ersten Parameter („MobileDashcard“), den Reverse Domain-Style Identifier („at.logicx.MobileDashcard“) und den Namen der App welcher auch auf dem Home

Screen des Gerätes angezeigt wird, als letzten Parameter. Diese Informationen können im Nachhinein noch in der config.xml Datei beliebig abgeändert werden. Alle weiteren Befehle, die für dieses Projekt angewendet werden sollen, müssen im angegebenen Verzeichnis ausgeführt werden (MobileDashcard). Bevor die App kompiliert werden kann, müssen noch die gewünschten Plattformen hinzugefügt werden. Dies erfolgt mit dem folgenden Befehl:

cordova platform add (Plattform)

Einige Kommandozeilenbefehle für dieses Projekt können ab jetzt entweder für das gesamte Projekt ausgeführt werden, oder nur für eine Plattform. Dazu muss die gewünschte Plattform als Parameter angeführt werden.

Builden des Projektes

cordova build (Plattform)

Testen auf Emulator

cordova emulate (Plattform)

Testen auf Device

cordova run --device (Plattform)

Installation von Plugins

Um Plugins bei einem Cordova Projekt hinzufügen zu können, muss der Client des Versionsverwaltungswerkzeugs Git installiert werden. [13]

Gegebenenfalls muss der Installationspfad wieder zur PATH Variable hinzugefügt werden. Der Befehl zur Installation eines Plugins sieht folgendermaßen aus:

cordova plugin (Plugin)

Für den Platzhalter Plugin muss das Repository des jeweiligen Plugins eingefügt werden (z.B. org.apache.cordova.device). Eine Liste dieser Plugins ist auf der Website von Cordova zu finden. [14]

Bei dem Projekt MobileDashcard wurden die beiden Plugins Device (org.apache.cordova.device) und StatusBar (org.apache.cordova.statusbar) verwendet.

Falls auf verschiedenen Systemen mit ein und demselben Projekt gearbeitet werden will, ist es am besten eine Versionsverwaltung zu verwenden und die einzelnen Projekte für die verschiedenen Plattformen nur auf einem System zu erzeugen, welches für diese eine Plattform auch konfiguriert wurde. Der platforms Ordner kann somit im Repository ignoriert werden.

11 Implementierung

11.1 Umsetzung WCF Service

Die folgenden Unterkapitel sollen die wichtigsten Teile des implementierten Services beschreiben. Aufgrund der Tatsache, dass dieser Service hauptsächlich von einem Client basierend auf Standardwebtechnologien verwenden wird, fiel die Entscheidung, den Service getreu den ReSTful Architektur Anforderungen zu implementieren. Dies bedeutet, dass die Kommunikation zwischen Server und Client über HTTP erfolgt. Ein solcher ReSTful Service wurde mittels WCF implementiert. Es könnte auch einfach ein WCF DataService implementiert werden, welcher das gesamte EntityFramework veröffentlicht, jedoch spricht gegen diese Methode, dass alle Berechnungen der kumulierten Aufwände zentral im Service geschehen sollen. Dies bietet zum einen eine höhere Performance, da JavaScript besonders auf mobilen Geräten weniger Leistung bietet, und zum anderen ist die Businesslogik auf einer Schicht gebündelt. Alle Lesezugriffe auf das EntitiyFramwork Model werden mit LINQ durchgeführt. Wie ein Projekt mit dem normalen WCF Service Application Template von Visual Studio verändert werden muss, damit sie dem Ansprüchen der ReST Architektur entspricht, wird in folgenden Zeilen erklärt.

11.1.1 ServiceContract

Zum einen muss der ServiceContract um einige Felder ergänzt werden. Dieser ServiceContract befindet sich standardmäßig in einem automatisch generierten Interface (IService.cs). In einem ServiceContract werden die Methoden definiert, die dem Client zur Verfügung gestellt werden. Diese Methoden werden mit dem Schlüsselwort [OperationContract] definiert. Diese Methode muss um das WebInvoke Attribut erweitert werden. Dieses Attribut ermöglicht es, dass die Methode mit Hilfe eines URI angesteuert werden kann und mittels den verschiedenen HTTP Methoden unterschieden werden kann, welcher Algorithmus beziehungsweise welche Funktion ausgeführt werden soll.

```
1. [OperationContract]
2. [WebInvoke(
3.     Method = "GET",
4.     BodyStyle = WebMessageBodyStyle.Wrapped,
5.     ResponseFormat = WebMessageFormat.Json,
6.     RequestFormat = WebMessageFormat.Json)]
7. List<SharePointProjekt> GetSharePointProjekte();
```

Beispiel 16: Service Methode mit WebInvoke

Wie im Beispiel 16 zu sehen ist, werden mittels des WebInvoke-Attributs mehrere Parameter für eine Methode definiert. Diese Parameter werden nun etwas näher erklärt:

- **Method:**
  Mit dem Parameter Method wird definiert, auf welche HTTP Methoden eine Antwort gegeben wird. Im Falle dieser Diplomarbeit wurde bei allen Funktionen die HTTP Protokoll Methode GET konfiguriert, da nur Lesezugriffe möglich sind.
- **BodyStyle:**
  Dieser Parameter definiert, welches Format die Daten bei einer Anfrage beziehungsweise bei einer Antwort haben müssen. Man unterscheidet zwischen den beiden Formaten Bare und Wrapped. In dieser Diplomarbeit wurde der BodyStyle Wrapped verwendet.

  Beim Format Wrapped wird das Objekt, welches an den Anfragenden zurückgegeben werden soll, nochmals in ein eigenes Objekt gekapselt. Dieses Objekt hat ein Attribut, dessen Bezeichnung aus dem Namen der aufgerufenen Funktion, in diesem Fall wäre das die Funktion „DoWork“, und dem anschließenden „Result“ besteht. Der Wert dieses Attributes ist das Ergebnis der Anfrage.

```
1. {"DoWorkResult":{"Name":"name","Value":"value"}}
```

Beispiel 17: JSON Response im Wrapped Format

- **ResponseFormat**
  Der Parameter ResponseFormat definiert das Format der Daten einer Antwort des Services. Aufgrund des JavaScript Clients werden die Daten in JSON Format zurückgegeben, da diese ohne weitere Hilfsmittel direkt in JavaScript Objekte umgewandelt werden können.

- **RequestFormat**
  Mittels des Parameters RequestFormat wird bestimmt, welches Datenformat die mitgesendeten Daten einer Anfrage haben müssen. Da der Client dieser Diplomarbeit nur Leseoperationen durchführt, ist dieser Parameter nicht unbedingt nötig, jedoch für etwaige Erweiterungen wird dieses Format vorausgesetzt.

Mittels des WebInvoke-Attributs ist es zusätzlich möglich ein UriTemplate für eine Funktion zu definieren. Ein solches UriTemplate gibt die Struktur eines URIs vor, auf den die Funktion reagiert, und kann zusätzlich Platzhalter für Variablen definieren, die in einem URI übergeben werden können. Folgender Codeausschnitt (siehe Beispiel 18) zeigt ein UriTemplate für die im Service implementierte Funktion GetSharePoint-Meilensteine. Im ersten Teil des Templates vor dem Schrägstrich („/“) wird der Name

der Funktion definiert. In diesem Fall wurde dieselbe Bezeichnung wie bei der eigentlichen Funktion beibehalten. Der Name kann jedoch beliebig gewählt werden. Im zweiten Teil wird der Platzhalter für die Variable projektIdString in den geschwungenen Klammern („{“, „}“) definiert. Die Zeichenkette die sich im URI dann an dieser Stelle befindet, wird an die Funktion als projektIdString übergeben. Hierbei können jedoch nur Zeichenketten übergeben werden, da man im URI nicht zwischen Zahlen und Zeichenketten unterscheiden kann.

```
1. [OperationContract]
2. [WebInvoke(
3.     Method = "GET",
4.     BodyStyle = WebMessageBodyStyle.Wrapped,
5.     ResponseFormat = WebMessageFormat.Json,
6.     RequestFormat = WebMessageFormat.Json,
7.     UriTemplate = "GetSharePointMeilensteine/{projektIdString}")]
8. List<SharePointMeilenstein> GetSharePointMeilensteine(string projektIdString);
```

Beispiel 18: WebInvoke mit UriTemplate

11.1.2 SharePointObjectConverter

Da unser Auftraggeber in einigen Projekten zum Zugriff auf die Datenbank den Objektrational Mapper EntityFramework verwendet und wir empfinden, dass dieser die Handhabung des Datenzugriffes um einiges erleichtert, haben wir uns entschieden dieses Werkzeug ebenfalls für unsere Diplomarbeit zu verwenden. Jedoch stießen wir dabei auf ein kleines Problem. Die Klassen, die von EntityFramework generiert wurden, enthielten Nullable Typen, die nicht serialisiert werden können und somit nicht als Request zurückgegeben werden können. Ein Nullable Typ entspricht dem Wert einer Spalte die NULL Werte beinhalte darf. Darüber hinaus mussten einigen Objekten noch zusätzliche Attribute hinzugefügt werden, um zum Beispiel die kumulierten Aufwandsdaten bequem zu den dazugehörigen Objekten übertragen zu können. Deshalb wurden eigene Klassen für die SharePointProjekte, SharePointMeilensteine und SharePointIssues erstellt, welche die zusätzlichen Attribute enthalten und keine Nullable Typen mehr haben.

Um die Daten der Objekte des EntityFramework Modells in die eigens erstellten Objekte überführen zu können, haben wird die Klasse SharePointObjectConverter implementiert. Diese stellt drei statische Methoden zur Verfügung, die jeweils ein Projekt, einen Meilenstein oder ein Issue konvertieren. Eine der Methoden wird im Beispiel 19 dargestellt. In dieser Methode werden die Werte des EntityFramework Objektes spp des Typs Sharepoint_Projekte in ein neues SharePointProjekt Objekt p geschrieben. Dabei werden mittels des NULL-Sammeloperators („??“) alle Nullable Typen in primitive Datentypen umgewandelt. Der NULL-Sammeloperator funktioniert folgendermaßen: Ist der Wert links vom Operator „??“ nicht NULL, so wird dieser zurückgegeben. Ist der

Wert jedoch NULL, so wird der Wert des Ausdruckes der rechten Seite zurückgegeben, in diesem Fall der Standardwert, welcher bei numerischen Datentypen 0 ist.

```
1.  public static SharePointProjekt ConvertSharePointProjekt(Sharepoint_Projekte
    spp)
2.  {
3.      if (spp == null)
4.      {
5.          return null;
6.      }
7.      SharePointProjekt p = new SharePointProjekt();
8.      p.Id = spp.Id;
9.      p.ProjektId = spp.ProjektId ?? default(int);
10.     p.ProjektBezeichnung = spp.ProjektBezeichnung;
11.     p.Bebuchbar = spp.Bebuchbar ?? default(bool);
12.     p.PTG = spp.PTG ?? default(decimal);
13.     p.PTU = spp.PTU ?? default(decimal);
14.     p.Projektleiter = spp.Projektleiter;
15.     p.TechnProjektleiter = spp.TechnProjektleiter;
16.     p.IstWartungsprojekt = spp.IstWartungsprojekt;
17.     p.Geloescht = spp.Geloescht;
18.     p.ADArchitekt = spp.ADArchitekt;
19.     return p;
20. }
```

Beispiel 19: Konvertierungsmethode des SharePointObjectConverter

11.1.3 DataContract

Dem Service muss noch mitgeteilt werden, welche Daten der selbst implementierten Objekte serialisiert werden sollen und zur Antwort hinzugefügt werden. Dies erfolgt indem man für eine Klasse das Attribut [DataContract] definiert und jeder Variable der Klasse, welche übertragen werden soll, muss das Attribut [DataMember] vorgestellt sein. Im folgenden Beispiel 20 sieht man den DataContract der Klasse SharePointProjekt.

```
1.  [DataContract]
2.  public class SharePointProjekt
3.  {
4.      [DataMember]
5.      public int Id { get; set; }
6.      [DataMember]
7.      public int ProjektId { get; set; }
8.      [DataMember]
9.      public string Projekt { get; set; }
10.     [DataMember]
11.     public string ProjektBezeichnung { get; set; }
12.     [DataMember]
13.     public bool Bebuchbar { get; set; }
14.     [DataMember]
15.     public decimal PTG { get; set; }
16.     [DataMember]
17.     public decimal PTU { get; set; }
18.     [DataMember]
19.     public string Projektleiter { get; set; }
20.     [DataMember]
21.     public string TechnProjektleiter { get; set; }
22.     [DataMember]
23.     public bool IstWartungsprojekt { get; set; }
24.     [DataMember]
25.     public bool Geloescht { get; set; }
26.     [DataMember]
27.     public string ADArchitekt { get; set; }
28.     [DataMember]
29.     public List<string> BebuchteMonate { get; set; }
30. }
```

Beispiel 20: SharePointProjekt DataContract

11.1.4 Web.config

Da die Methoden und Klassen des Services korrekt implementiert und konfiguriert sind, muss nur noch eingestellt werden, wie der IIS den Service hosten soll. Die Web.config Datei bietet dafür die Möglichkeit, die Endpunkte und das Verhalten des Dienstes erst bei der Bereitstellung des Services zu definieren, und nicht schon beim Implementieren des Codes. (vgl. [MICROSOFT, MSDN, 2015])

Der XML Inhalt der Web.config Datei ist dabei in drei wesentliche Hauptabschnitte unterteilt:

- **<services>:**
  Dieser Teil enthält die Spezifikation aller Dienste, die diese Anwendung hostet. Im Falle dieser Diplomarbeit wurde nur ein einziger Service definiert.

```
1. <service name="TimeStoreServiceREST.TimeStoreServiceREST"
2.          behaviorConfiguration="serviceBehavior">
3.     <endpoint address=""
4.               behaviorConfiguration="endpointBehavior"
5.               bindingConfiguration="bindingConfiguration"
6.               binding="webHttpBinding"
7.               contract="TimeStoreServiceREST.ITimeStoreServiceREST" />
8. </service>
```

Beispiel 21: Service Konfiguration

Wie in dem Beispiel 21 zu sehen ist, definiert das Element service die Parameter name und behaviorConfiguration. Der Parameter name gibt dabei die Klasse des implementierten Services an, der aus dem Namespace und einem Punkt gefolgt vom Klassennamen besteht. Der Parameter behaviorConfiguration gibt ein im Teil <behaviors> definiertes Verhalten für diesen Service an.

Ein Service kann mehrere Endpunkte haben. Diese werden mittels des Elementes <endpoint> bestimmt. Ein Endpoint besteht neben der Adresse, also dem URI unter dem dieser erreichbar ist, und dem contract, dem Interface welches den ServiceContract für die Klasse implementiert (wird im Parameter name des <service> Elements definiert, siehe Beispiel 21), auch noch aus der behaviorConfiguration, bindingConfiguration und dem binding. Der Parameter binding definiert eine vom System bereitgestellte Bindung, die die Eigenschaften der Verbindung zum Client beschreibt. Müssen einige Eigenschaften der Bindung abgeändert werden, so wird im Parameter bindingConfiguration eine Konfiguration eingetragen, die im Bereich <bindings> definiert wird. Der Parameter behaviorConfiguration gibt wiederum das Verhalten des Endpunktes an. (vgl. [MICROSOFT, MSDN, 2015])

- **<bindings>**
  Dieses Element enthält alle vom Entwickler definierten Spezifikationen für Bindungen. Für diese Diplomarbeit musste die vom System bereitgestellte Bindung „webHttpBinding“ um die Spezifikation der Sicherheitsmechanismen erweitert werden, da diese nicht definiert sind. Diese Anpassungen sind im Beispiel 22 zu sehen. Da der Service ohne weitere Konfiguration die sensiblen Kundendaten im Klartext versenden würde, musste eine Verschlüsselung der übertragenen Daten eingeführt werden. Deshalb wird im Element <security> der Modus „Transport“ festgelegt. Dieser stellt sicher, dass die Daten nur über eine mit SSL (TSL) verschlüsselte HTTPS Verbindung übertragen werden dürfen. Zu-

sätzlich sind nur die Mitarbeiter des Unternehmens berechtigt, die übertragenen Daten einzusehen, weshalb sich jeder Client vor der Nutzung des Services authentisieren muss. Da das Unternehmen bereits über eine geeignete Active Directory Struktur verfügt, musste die Authentifizierung der Benutzer nur mehr dem IIS überlassen werden. Deshalb haben wir uns entschieden, Standardauthentifizierung zu verwenden, weshalb im Element <transport> der Wert Basic festgelegt ist. Dieser gibt an das sich der Benutzer mittels Standardauthentifizierung (Basic Authentication) authentisieren muss. (vgl. [MICROSOFT, MSDN, 2015])

```
1. <webHttpBinding>
2.     <binding name="bindingConfiguration">
3.         <security mode="Transport">
4.             <transport clientCredentialType="Basic" />
5.         </security>
6.     </binding>
7. </webHttpBinding>
```

Beispiel 22: WebHttpBinding Spezifikation

- **<behaviors>**
  In diesem Bereich der Web.config Datei wird das Verhalten der Services und Endpunkte definiert. Der Teilbereich <endpointBehavior> definiert das Verhalten des Endpunktes. Hierbei wurde durch das Element <webHttp> das WebHttpBehavior für diesen Endpunkt festgelegt, welches es ermöglicht, dass der Endpunkt durch eine simple Anfrage erreicht werden kann (siehe Beispiel 23). (vgl. [MICROSOFT, MSDN, 2015])

```
1. <endpointBehaviors>
2.     <behavior name="endpointBehavior">
3.         <webHttp />
4.     </behavior>
5. </endpointBehaviors>
```

Beispiel 23: Konfiguration des Verhaltens des Endpunktes

Um möglichst wenige Angriffsmöglichkeiten freizugeben, wurde die Anfrage der Metadaten des Services blockiert, sowie die Rückgabe von Fehlermeldungen verhindert. Diese Konfigurationen befinden sich im Unterbereich <serviceBehaviors> (siehe Beispiel 24).

```
1. <serviceBehaviors>
2.     <behavior name="serviceBehavior">
3.         <serviceMetadata httpGetEnabled="false" httpsGetEnabled="false" />
4.         <serviceDebug includeExceptionDetailInFaults="false" />
5.     </behavior>
6. </serviceBehaviors>
```

Beispiel 24: Konfiguration des Verhaltens des Dienstes

11.1.5 Funktionen

Der implementierte Service stellt dem Client drei Funktionen zur Verfügung:

- **GetSharePointProjekte**
  Diese Funktion gibt alle Projekte aus der Datenbank zurück und fügt diesen noch eine Liste jener Monate hinzu, in der von einem Mitarbeiter ein Aufwand für dieses Projekt verbucht wurde.
- **GetSharePointMeilensteine**
  Die Funktion nimmt die ID eines Projektes als Parameter entgegen. Bei einem Aufruf der Funktion GetSharePointMeilensteine erhält der Aufrufende alle Meilensteine des Projektes mit der übergebenen ID zurück. Zusätzlich werden für die einzelnen Meilensteine noch die geplanten und tatsächlichen Aufwände pro Kategorie summiert. Auch die tatsächlich aufgewendete Zeit in Personentagen für das aktuelle Monat wird berechnet.
- **GetSharePointIssues**
  Die Funktion GetSharePointIssues nimmt die ID eines Meilensteines entgegen und gibt dann alle Issues dieses Meilensteines zurück. Für jedes dieser Issues wird auch noch der tatsächliche Ist-Aufwand in Personentagen berechnet.

11.1.6 Deployment

Da die Verbindung zwischen Client und Server durch das SSL Protokoll gesichert werden soll und das Unternehmen noch keine Website mit einem HTTPS Binding erstellt hatte, musste eine neue Website mit einem Binding für das Protokoll HTTPS eingerichtet werden. In dieser Website wurde eine neue Applikation erstellt, welche den Service beinhaltet.

Die Einstellungen für die Authentifizierung wurden bei der Applikation getätigt. Es dürfen hierbei nur Standardauthentifizierung (Basic Authentication) und ASP.NET Identitätswechsel (ASP.NET Impersonation) aktiviert sein.

11.1.7 Aufgetretene Probleme

Beim Testen des Services sind wir auf folgendes Problem gestoßen: Ruft man eine Funktion des Services mittels des dazugehörigen URI vom Browser aus auf, so erhält man eine Fehlermeldung als Antwort. Nach einiger Recherche im Internet stießen wir auf die Ursache dieses Problems, welche den Namen Same-Origin-Policy (SOP) trägt. SOP ist ein Sicherheitskonzept das 1996 von Netscape eingeführt worden ist. Es verhindert dass Skriptsprachen wie JavaScript und CSS auf Ressourcen einer anderen Herkunft (Origin) zugreifen können. Eine Origin besteht dabei immer aus dem Protokoll, der Domäne und dem Port (vgl. [MOZILLA DEVELOPER NETWORK, 2015]). Somit kann man im Browser vom JavaScript Code einer Website nicht auf den Kontext einer anderen Seite zugreifen, um zum Beispiel sensible Daten auszulesen. Deshalb war es nicht möglich, die HTML Seite von der Festplatte aus zu öffnen, da dabei die Origin nicht übereinstimmt. Deshalb müssen alle HTML Seiten, welche zum Testen der App dienen, in der Website bereitgestellt werden in der sich die Applikation des Service befindet, da nur dort die Origin übereinstimmt. Auf Apps trifft die SOP Regelung nicht zu, da in solchen nicht mehrere Kontexte von verschiedenen Websites gleichzeitig bestehen können.

11.2 Umsetzung hybride App

11.2.1 HTML Anpassung für mobile Anwendung

Da es bei HTML-Seiten üblich ist, dass die Seite so groß ist wie der Inhalt der angezeigt werden soll, ergibt sich eine gewisse Schwierigkeit beim Entwickeln von HTML-Apps für mobile Geräte. Mobile Browser gehen davon aus, dass die Seite nicht für mobile Geräte bestimmt ist und stellen den Browser- Anzeigebereich (Viewport) so, dass die meisten Websites komplett zu sehen sind. Damit der gesamte Anzeigebereich ausgenutzt wird und die Schrift auf jedem Gerät eine angenehm lesbare Größe hat, sollte der Anzeigebereich auf die Größe des Gerätes festgelegt werden. (vgl. [APPLE, 2014])

```
1. <meta name="viewport"
2.       content="user-scalable=no,
3.         initial-scale=1,
4.         maximum-scale=1,
5.         minimum-scale=1,
6.         width=device-width,
7.         height=device-height,
8.         target-densitydpi=device-dpi" />
```

Beispiel 25: Festlegen des Viewports

Neben der Größe des Viewports werden hier noch weitere Eigenschaften festgelegt. Einerseits wird die Skalierung auf eins, also die Originalgröße, gesetzt. Andererseits

wird verhindert, dass die Seite vom Benutzer skaliert werden kann. Außerdem wird die Pixeldichte der Website auf die Pixeldichte des Gerätes angepasst.

11.2.2 Realisierung MVC

Das MVC Muster wurde mit Hilfe der AngularJS Funktionalitäten umgesetzt. Die Ansichten werden durch sogenannte Templates dargestellt. Das sind HTML Dateien die über Data Binding und Events mit den Controllern kommunizieren. Diese sind in JavaScript Funktionen die als Angular Controller verwendet werden. Das Model ist eine Sammlung von JavaScript Funktionen, die wird unter den Namen timeStoreService erstellt haben. Auf die Funktionalität vom timeStoreService wird im nächsten Kapitel eingegangen. Weder die Controller noch die Ansichten haben Zugriff auf das Model.

11.2.3 timeStoreService

Beim timeStoreService handelt es sich um einen Angular Service. Die gesamten benötigten Daten werden über diesen Service angefordert. Sind die Daten noch nicht vorhanden, werden sie vom Server angefordert und danach im Model gehalten. Der Service bietet Funktionen um eine Liste von Projekten, Meilensteinen oder Issues zu erhalten. Außerdem gibt es eine Funktion, die die gesamten gespeicherten Daten löscht, da daraufhin die Daten neu geladen werden und somit eine Aktualisierung erreicht wird. Im Service befindet sich die Authentifizierung. Diese Methode nimmt die eingegebenen Anmeldedaten Base64 verschlüsselt entgegen. Diese werden an den Standardheader, der für die Kommunikation mit dem Server verwendet wird, angehängt. Daraufhin wird eine Anfrage an den Server gesendet um die Anmeldedaten zu überprüfen. Je nach Erfolgs- oder Fehlercode in der Antwort des Servers wird entweder auf die Projektansicht weitergeleitet oder eine passende Fehlermeldung ausgegeben.

```
1.  this.authenticate = function (credentials) {
2.      $http.defaults.headers.common['Authorization'] = 'Basic ' + credentials;
3.      $http({
4.          method: 'GET',
5.          url: serverUrl + serviceUrl + '',
6.          timeout: 5000,
7.          headers: {
8.              'Content-Type': "application/json"
9.          }
10.     })
11.         .success(function (data, status, headers, config) {
12.             authorized = true;
13.             window.location.hash = "#/Projects"
14.         })
15.         .error(function (data, status, headers, config) {
16.             angular.element(document.getElementById('loginTable'))
17.                 .scope().loggingIn = false;
18.             //Errorhandling mit Variable status
19.         })
20.     ;
21. };
```

Beispiel 26: Authentifizierung

11.2.4 Routing

Hier wird beschrieben wie in der Applikation zwischen den einzelnen Ansichten gewechselt wird. Die erste Ansicht, die geladen wird, ist der Login. Bevor der Login angezeigt wird, wird noch überprüft ob die Adresse des Servers in den Einstellungen gespeichert ist. Ist dies nicht der Fall wird automatisch zu den Einstellungen weitergeleitet. Ansonsten kann vom Login zu den Einstellungen navigiert werden sowie mittels Registrierung zu der Projektauswahl gelangt werden. Von dort sind die Filterung sowie die Detailansicht jedes Projektes zu erreichen. Innerhalb der Projektansicht kann die Statistik sowie die Detailansicht jedes Meilensteins angesteuert werden. In letzterer gibt es wieder eine Statistik.

Beim Wechsel in eine andere Ansicht sind einige Schritte notwendig. Zuerst ist aber zu erwähnen, dass die Ansteuerung einer Ansicht über den Fragmentbezeichner, den letzten Teil der URL, der nach dem Doppelkreuz (#) folgt, erfolgt. Bei jeder Navigation wird also dieser Teil der URL verändert. Außerdem wird der aktuelle Fragmentbezeichner in den History Stack geschrieben. Diese zwei Schritte werden im nachfolgenden Codebeispiel gezeigt.

```
1.  window.location.hash = nextUrl + nextId;
2.  $scope.addToHistoryStack(currUrl, currId);
```

Beispiel 27: Navigation mit History Stack

Im Codebeispiel ist bei jedem Fragment auch eine ID angegeben. Diese wird bei Detailansichten für die Identifizierung genau eines Projektes beziehungsweise Meilensteines benötigt. Ansonsten ist diese ID leer. Nachdem über die URL ein neues Fragment definiert wird, kümmert sich der Route Provider um dessen Auswertung. Er entscheidet anhand des Fragments welches HTML Dokument als Template und welcher Controller geladen wird. Außerdem nimmt er die ID entgegen, lädt anhand dieser die benötigten Daten und stellt sie dem Controller zur Verfügung. Das folgende Codebeispiel zeigt einen Ausschnitt aus der Konfiguration des Route Providers, der benötigt wird, wenn in die Detailansicht eines Projektes gewechselt wird.

```
1.  when('/Milestones/:projectId', {
2.    templateUrl: 'templates/milestones.html',
3.    controller: 'MilestoneController',
4.    resolve: {
5.      milestones: function (TimeStoreService, $route) {
6.        return TimeStoreService.getMeilensteine($route.current.params.projectId);
7.      },
8.      project: function (TimeStoreService, $route) {
9.        return TimeStoreService.getProjekt($route.current.params.projectId);
10.     }
11.   }
12. }).
```

Beispiel 28: Konfiguration Route Provider

Solange die Daten vom Service nicht bereitstehen, wird noch nicht umgeschaltet. Damit der Benutzer dennoch eine Rückmeldung erhält, wird ein Ladebildschirm dargestellt, während seine Eingabe verarbeitet wird. Dazu werden zwei Events zur Hilfe genommen: der Start und der Abschluss des Routing Prozesses. Bei diesen wird der Ladebildschirm ein- beziehungsweise ausgeblendet. Der Ladebildschirm besteht aus einem Icon und einem grauen Hintergrund.

```
1.  $scope.$on('$routeChangeStart', function () {
2.      $scope.initializing = true;
3.      scrollEnabled = false;
4.  });
5.  $scope.$on('$routeChangeSuccess', function () {
6.      $scope.initializing = false;
7.      scrollEnabled = true;
8.  });
```

```
1.  <div ng-show="initializing" class="loadingIcon">
2.  </div>
3.  <div ng-show="initializing" class="greyScreen">
4.  </div>
```

Beispiel 29: Ladebildschirm

Um einen Schritt zurück zu gelangen gibt es zwei Möglichkeiten zum einen über einen Klick auf die Titelzeile oder mittels des Hardware Backbutton des Windows Phones oder Android Gerätes. Beides ruft dieselbe Methode auf, welche den letzten Eintrag aus dem History Stack nimmt und zu diesem navigiert. Sollte sich im History Stack kein Eintrag befinden, heißt das, dass der Benutzer entweder im Login oder in der Projektauswahl ist. In beiden Fällen wird die Applikation beendet. Nachfolgendes Codebeispiel zeigt diese Methode.

```
1. $scope.pageBack = function () {
2.     if (historyStack.length == 0) {
3.         navigator.app.exitApp();
4.     }else{
5.         var url = historyStack.pop();
6.         window.location.hash = url;
7.     }
8. };
```

Beispiel 30: Zurücknavigieren

11.2.5 Hardware Button

Da der Hardware Backbutton nicht das standardmäßig festgelegte Verhalten haben soll, sondern die oben beschriebene Methode aufrufen soll, muss ein Klick auf diesen Button abgefangen werden. Dazu wird ein Event verwendet, welches von Cordova zur Verfügung gestellt wird und sich passenderweise backbutton nennt. Um dieses nutzen zu können, ist es nötig, die Bibliothek cordova.js einzubinden. Nachdem cordova.js fertig geladen ist wird das Event deviceready aufgerufen. Nach Eintreten dieses Events kann auch auf das backbutton Event ein Listener erzeugt werden.

```
1. <script type="text/javascript" src="cordova.js"></script>
2. <script type="text/javascript">
3.    document.addEventListener("deviceready", function() {
4.       document.addEventListener("backbutton", function () {
5.          angular.element(document.getElementById('appDiv')).scope().pageBack();
6.       });
7.    }, false);
8. </script>
```

Beispiel 31: Backbutton Event

11.2.6 Plugins

Bei der Entwicklung dieser App wurden die Cordova Plugins Device und StatusBar verwendet. Device erlaubt es, Detailinformationen über das Gerät, auf dem die App läuft, abzufragen. Verwendet wird es, um die Plattform des Gerätes abzufragen. Wenn es nötig ist, wird dadurch für verschiedene Plattformen unterschiedlichen Code verwendet. Das führt uns zum zweiten Plugin. Bei Windows Phones und bei iOS Geräten

soll die Statusleiste ausgeblendet werden, bei Android Geräten soll sie angezeigt bleiben. Dies kann mit dem Plugin StatusBar erreicht werden.

```
1. if (window.device.platform != 'Android') {
2.     StatusBar.hide();
3. }
```

Beispiel 32: Statusleiste ausblenden

11.2.7 Listenanzeige und Filterung

Die anzuzeigenden Daten sind in einem JavaScript Array vorhanden. Um die gesamte Liste zu durchlaufen wird die Direktive ng-repeat verwendet. Damit wird für jeden Eintrag eine Tabelle erzeugt. In dieser Tabelle wird in der ersten Zeile der Titel angezeigt, darunter noch weitere relevante Informationen. Mit einer Schleifenvariable kann auf jedes Element samt dessen Eigenschaften zugegriffen werden. Für ng-repeat kann ein Filter angegeben werden, welcher im Falle dieser Implementierung eine JavaScript Funktion ist, die für jedes Element der Liste aufgerufen wird und dieses als Parameter erhält. Die Funktion gibt einen booleschen Wert zurück anhand von dem AngularJS das Element in der Iteration beachtet oder nicht. Dieses System wird sowohl bei den Projekten als auch bei den Meilensteinen und Issues verwendet. Bei den Projekten werden in der Filtermethode alle Eigenschaften überprüft, die in der Filteransicht eingetragen werden können. Wird in das Suchfeld ein Wert eingegeben, wird nach diesem Wert ebenfalls gefiltert. Bei den Meilensteinen und Issues wird nur der Fremdschlüssel überprüft, damit nur Meilensteine des richtigen Projekts und Issues des richtigen Meilensteins angezeigt werden. Außerdem können alle Einträge herausgefiltert werden, deren Planwert den Istwert übersteigt. Das Codebeispiel zeigt die Umsetzung dieses System in der Projektansicht.

```
<a ng-repeat="x in filtered = (projects | filter:checkFilter)"
   ng-click="navigateTo('#/Milestones/', x.Id, '#/Projects', '')">
    <table class="grey">
        <tr>
            <th colspan=" 2">
                {{ x.Projekt }}
            </th>
        </tr>
        <tr>
            <td>
                Projektleiter
            </td>
            <td>
                TPL
            </td>
        </tr>
        <tr>
            <td>
                {{x.Projektleiter}}
            </td>
            <td>
                {{x.TechnProjektleiter}}
            </td>
        </tr>
    </table>
</a>
```

Beispiel 33: Projektliste anzeigen

11.2.8 Statistik

Für die statistische Auswertung werden die Aufwände je Kategorie benötigt. Vom Service erhalten wir diese Information sowohl für Plan als auch Ist Daten zu jedem Meilenstein. Auf Meilensteinebene können diese Daten eins zu eins in die Tabelle eingetragen werden, die Google Charts benötigt um ein Diagramm zusammenzustellen. Außerdem wird noch eine Gesamtsumme gebildet, damit falls diese null ist, eine Meldung angezeigt werden kann. Auf Projektebene müssen die gesamten betroffenen Meilensteine durchlaufen werden und die Aufwandsdaten kumuliert werden. Auch hier wird wieder eine Gesamtsumme gebildet. Es werden zwei Tortendiagramme erzeugt, Eines für die Plan-Aufwände, das Andere für die Ist-Aufwände.

11.2.9 Wischgeste

Die Umsetzung der Wischgeste wurde mit den beiden Bibliotheken Swipe und Hammer erreicht. Mit Hammer können die Gesten erkannt werden, um anschließend die Events swiperight und swipeleft aufzurufen. Mit Swipe kann die gewünschte Animation zum Wechsel erreicht werden. Im Codebeispiel werden die beiden Bibliotheken zuerst instanziiert und der Swipe so eingerichtet, dass er nach einer Wischgeste aufgerufen

wird. Beide Exemplare benötigen ein HTML Element auf die sie angewendet werden. Dies ist das äußere Element in dem alle Seiten eingebunden sind.

```
1.  window.mySwipe = Swipe(document.getElementById('mainSwiper'), {
2.      startSlide: 0,
3.      continuous: false,
4.      disableScroll: false,
5.      stopPropagation: false,
6.      callback: function (index, element) { },
7.      transitionEnd: function (index, element) { }
8.  });
9.  if (window.device.platform != 'Android' && window.device.platform != 'iOS') {
10.     var touchControl = new Hammer(document.getElementById('mainSwiper'),
11.         { dragLockToAxis: true, dragBlockHorizontal: true });
12.     touchControl.on('swiperight', function () {
13.         window.mySwipe.prev();
14.     });
15.     touchControl.on('swipeleft', function () {
16.         window.mySwipe.next();
17.     });
18. }
```

Beispiel 34: Wischgeste

Bei Swipe können einige Optionen eingestellt werden. So zum Beispiel welche Seite zuerst angezeigt wird (startSlide) oder ob die Seiten in einer Schleife immer wiederkehren (continuous). Hammer ist so konfiguriert, dass nur waagerechte Wischgesten erkannt werden.

12 Funktionalität

In diesem Kapitel werden die einzelnen Funktionen der Applikation sowie der grobe Ablauf beim erstmaligen Starten und bei der weiteren Verwendung beschrieben.

12.1 Aufbau der App

Die Ansicht der App besteht immer aus drei Teilen: Einem Kopfbereich, den Fußbereich und dem Inhalt, der immer zwischen Kopf- und Fußbereich dargestellt wird. Wie in der Abbildung 10 zu sehen ist, wird im linken Teil des Kopfbereiches der „Zurück“ Button angezeigt, mit dem zur vorherigen Seite zurücknavigiert werden kann, und im rechten Teil die Überschrift der jeweiligen Seite. Der Fußbereich enthält auf der linken Seite immer die Komponenten zur schnellen Filterung des Inhaltes der aktuellen Seite und rechts die Buttons zur Navigation zu anderen Seiten, wie Projektfilterung oder Statistiken, oder der Refresh Button.

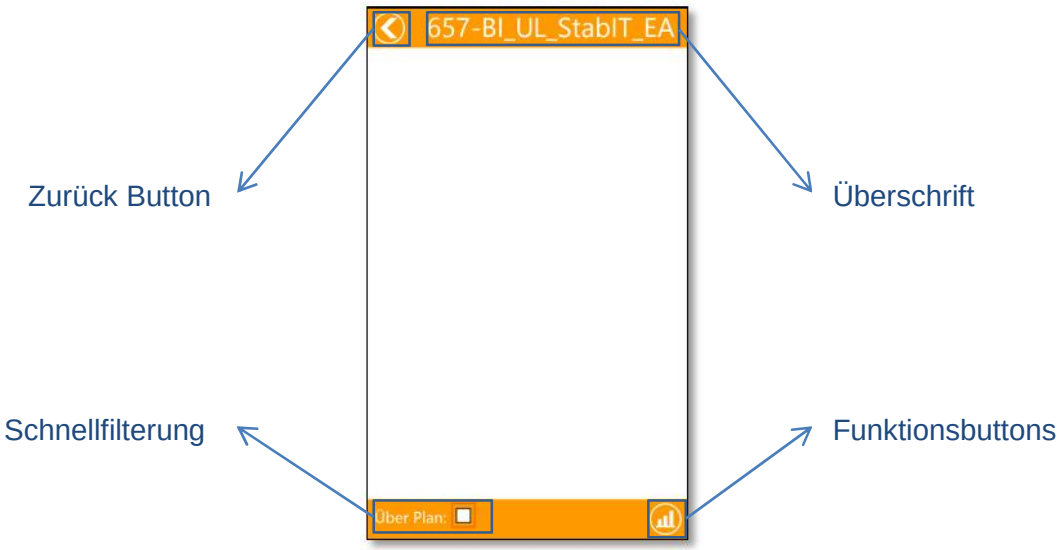

Abbildung 10: Aufbau der App

12.2 Ablauf

Im Zuge des erstmaligen Starts der App erscheint zuerst eine Seite, in der man den URI des Servers und den des Services konfigurieren muss (siehe Abbildung 11).

Abbildung 11: Einstellungen Seite

Diese Seite kann erst verlassen werden, wenn in den beiden Feldern etwas eingetragen wurde. Falls die Werte korrekt in die Textfelder eingetragen wurden und es erfolgt ein Klick auf den Button „Speichern“, werden die Daten auf dem Handy lokal gespeichert und die App navigiert zur Login Seite weiter. Falls keine Werte für Server URL und/oder Service URL eingetragen wurden, erscheint eine Fehlermeldung. Erfolgt beim ersten Start der App ein Klick auf den „Zurück“ Button auf der Einstellungen Seite, so schließt sich die App. Mit Hilfe dieser Seite kann auch nachträglich der Server und Service URL geändert werden.

Abbildung 12: Login Seite

Nach dem Konfigurieren der URLs beim ersten Starten und nach jedem weiteren Start gelangt man zur Login Seite. In dieser Seite muss man seinen Benutzernamen und

das Passwort eingeben, um sich mit dem Service zu verbinden und die Projekte zu laden. Der Benutzername wird dabei nach der erstmaligen Eingabe gespeichert. Mit dem Button im rechten Teil des Fußbereichs kann man zur Einstellungen Seite navigieren. (siehe Abbildung 12)

Abbildung 13: Projektauswahl Seite

Hat man sich erfolgreich authentisiert, folgt nach einem kurzen Ladebildschirm die Projektauswahl (siehe Abbildung 13). In dieser Ansicht werden alle Projekte des Unternehmens aufgelistet und die dazugehörigen Projektleiter und Technischen Projektleiter (TPL) dargestellt. In das Textfeld im linken Teil des Fußbereichs kann man einen Text eingeben, nach dem die Projektbezeichnung gefiltert wird. Durch einen Klick auf den „Aktualisieren“ Button werden alle Daten, die bis zum jetzigen Zeitpunkt heruntergeladen wurden, verworfen und die Projekte erneut heruntergeladen. Der Funktionsbutton „Filterung“ navigiert zur Projektfilterung. Falls man auf ein Element der Liste klickt, also auf ein Projekt, werden alle Meilensteine des Projektes heruntergeladen und es wird zur Meilenstein Ansicht gewechselt. Diese Ansicht hat keinen „Zurück“ Button, da das Zurücknavigieren zum Login nicht möglich sein soll, sondern die App durch einen Klick auf den Hardware Back Button geschlossen werden soll.

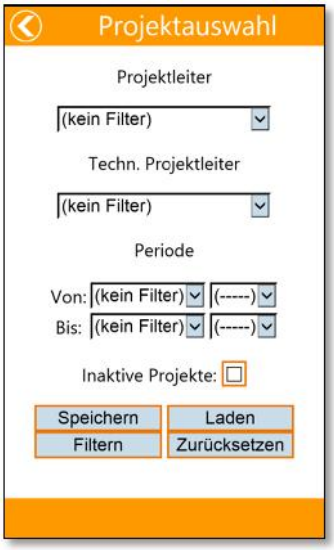

Abbildung 14: Projektfilterung

Die Projektfilterung (siehe Abbildung 14) bietet dem Benutzer die Möglichkeit, die Projekte der Projektauswahl nach bestimmten Kriterien zu filtern. Zum einen ist es möglich durch die ersten beiden Auswahllisten nach dem Projektleiter und dem Technischen Projektleiter zu filtern. Zusätzlich kann eine Periode, in der ein Aufwand auf das Projekt gebucht wurde, mit Anfangs- und Endzeitpunkt in Monaten angegeben werden. Durch das Aktivieren der „Inaktive Projekte“ werden auch jene Projekte angezeigt, auf die kein Aufwand mehr verbucht werden kann. Angewendet wird die Filterung durch einen Klick auf den Button „Filtern“. Der Button „Speichern“ schreibt die aktuell eingegebenen Werte in ein Objekt und speichert diese lokal auf dem Handy. Durch den „Laden“ Button werden diese Werte wieder aus dem Speicher geladen. Der „Zurücksetzen“ Button bewirkt das Löschen des Objektes aus dem Speicher und setzt so die Filterung wieder zurück.

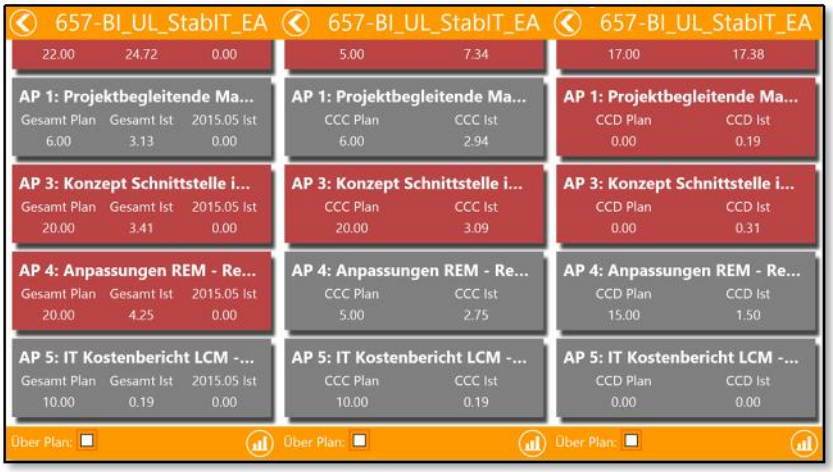

Abbildung 15: Meilensteinansicht

Die Meilensteinansicht (siehe Abbildung 15) bietet eine Übersicht über alle Meilensteine eines Projektes. Die Bezeichnung des ausgewählten Projekts wird dabei in der Überschrift angezeigt. Nach dem Umschalten von Projektauswahl zu Meilensteinansicht werden für jeden Meilenstein die gesamte geplante Zeit (Gesamt Plan), die gesamten tatsächlich aufgewendeten Personentage (Gesamt Ist) und die aufgewendete Zeit im aktuellen Monat dargestellt. Durch horizontale Wischbewegungen ändern sich die dargestellten Kennzahlen. Statt des gesamten Plan- und Ist-Aufwands kann man auch die geplante und tatsächlich aufgewendete Zeit für organisatorische Tätigkeiten (CCC) und umsetzungstechnische Tätigkeiten (CCD) darstellen. Alle Meilensteine bei denen die aufgewendete Zeit die geplante Zeit überschreitet, werden rot markiert. Aktiviert man die Checkbox in der linken unteren Ecke, werden nur jene Meilensteine angezeigt, bei denen die geplanten Zeit überschritten wird. Durch einen Klick auf den Funktionsbutten in der rechten unteren Ecke kommt man zur graphischen Darstellung der Daten. Wählt man wie bei der Projektauswahl einen Meilenstein aus, werden die Issues des Meilensteines heruntergeladen und es wird zur Issueansicht navigiert.

Abbildung 16: Issueansicht

Die Issueansicht (siehe Abbildung 16) ist ähnlich aufgebaut wie die Meilensteinansicht. In der Überschrift wird die Bezeichnung des Meilensteines dargestellt. Sie enthält auch die Checkbox „Über Plan“ und den Button für die statistische Auswertung. Durch das Wischen wird zwischen der Auflistung der geplanten und aufgewendeten Personentage und der Liste mit Status und Kategorie eines jeden Issues gewechselt.

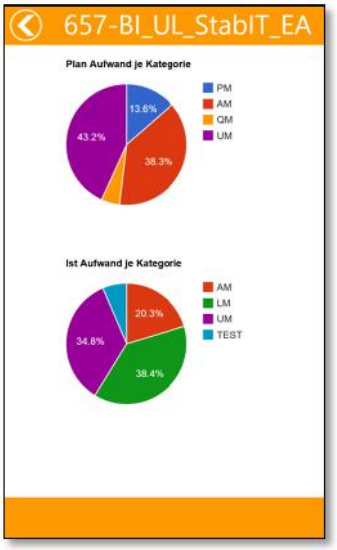

Abbildung 17: Graphische Darstellung

Für ein Projekt und einen Meilenstein kann eine graphische Auswertung der Daten angezeigt werden. Diese wird durch einen Klick auf den Funktionsbutton „Statistik“ angezeigt. Es werden zwei Tortendiagramme angezeigt die zum einen die Zusammensetzung der geplanten Personentage für jede Kategorie, sowie die Zusammensetzung der tatsächlich aufgewendeten Personentage darstellen. Falls noch keine geplanten oder tatsächlichen Aufwände bestehen, wird eine entsprechende Meldung angezeigt. Durch einen Klick auf ein Tortensegment, kann die Prozentzahl jener Kategorie hervorgehoben werden.

13 Zusammenfassung

In diesem Kapitel wird die Diplomarbeit noch einmal kurz zusammengefasst und auf die Ergebnisse dieser Arbeit eingegangen. Zudem wird erläutert, welche Erfahrungen das Team aus dieser Diplomarbeit mitnehmen konnte. Darüber hinaus werden aufgrund der gewonnenen Erfahrungen die Entwicklung von hybriden HTML5 Apps beurteilt und mögliche Fallstricke aufgezeigt.

13.1 Ergebnisse

Im Bearbeitungszeitraum dieser Diplomarbeit wurde die Infrastruktur des Unternehmens um einen Service erweitert und zusätzlich eine hybride App entwickelt, welche auf den Plattformen Android, Windows Phone und iOS getestet wurde.

Konkret wurde ein WCF Service entwickelt, welcher durch einige Anpassungen den Anforderungen der ReSTful Architektur entspricht. Dieser Service greift auf die bereits bestehende Datenbank des Unternehmens mittels EntityFramework und LINQ zu und stellt diese Daten mit zusätzlichen Informationen durch drei Service Funktionen dem Aufrufer zur Verfügung.

Die App wurde mit dem Hilfsmittel Cordova und dem Einsatz von Standardwebtechnologien wie HTML, JavaScript und CSS für mehrere Plattformen entwickelt. Sie bietet dem Benutzer die Möglichkeit die geplanten Aufwände der Projekte, Meilensteine und Issues zu begutachten und zudem mit den kumulierten Werten des tatsächlich erbrachten Aufwandes zu vergleichen.

Der Anwender der App muss sicher vor der Verwendung mit seinem Benutzernamen und Passwort seines Active Directory Benutzers authentisieren. Die Verbindung zwischen App und Service erfolgt über das HTTPS Protokoll und wird durch Verschlüsselung und ein serverseitiges SSL Zertifikat gesichert.

13.2 Erfahrungen

Im Zuge dieser Diplomarbeit wurden viele für das Team neue Technologien verwendet. Bevor man solche Technologien verwendet, ist es unbedingt nötig, sich ausgiebig mit der Dokumentation und Beispielen auseinander zu setzen, um einen möglichst guten Einblick zu erlangen. Vor der Anwendung einer Bibliothek ist es zudem ratsam, einen Prototyp mit den gewünschten Funktionen zu implementieren, um sicherzustellen, dass sich diese auch in der gewünschten Art und Weise verhalten.

Zunächst war bei dieser Diplomarbeit geplant, dass sich die Benutzer mittels eines VPN Clients auf ihrem mobilen Gerät mit dem Netzwerk der Firma verbinden. Jedoch stand nicht auf jeder Plattform eine solche Software zur Verfügung, die mit der VPN

Lösung des Unternehmens kompatibel ist. Aus diesem Grund musste eine neue Website erstellt werden, welche über eine gesicherte Verbindung zugänglich ist. Deshalb ist es essentiell bei der Auswahl der Technologien ausreichend Recherche und Analyse des Systems zu betreiben, um sicherzustellen, dass die angedachte Lösung auch durchführbar ist.

Die Zusammenarbeit, die wir im Bearbeitungszeitrum mit unseren Betreuern der Logicx GmbH pflegten, lehrte uns einiges über die Vorgänge in der Wirtschaft. Das vereinbaren von Projektmeetings und vorbereiten dieser Meetings, sowie die professionelle Durchführung der Treffen gaben uns einen Einblick, wie es im späteren Berufsleben ablaufen wird. Zudem war es eine Herausforderung, das System des Unternehmens zu verstehen und zu analysieren und anschließend eine Erweiterung für eine so umfassende Struktur zu implementieren.

13.3 Beurteilung von hybriden HTML5 Apps

Die Implementierung einer hybriden HTML5 App, welche auf den Plattformen Android, Windows Phone und iOS lauffähig sein sollte, stellte sich als eine Herausforderung heraus. Wie schon im Kapitel 8.4.4 erwähnt, gibt es einige Unterschiede zwischen der Darstellung der einzelnen WebViews. Standard HTML Elemente wie Dropdown Auswahllisten, Checkboxes oder Input Felder werden etwas unterschiedlich dargestellt und auch die JavaScript Popup Boxes (Alert) weisen einige Unterschiede auf. Jedoch kann man dem entgegenwirken und mit Hilfe von eigenen CSS-Files alle gewünschten Styles der Elemente überschreiben. Darüber hinaus ist das Verhalten bei Touch Gesten auch unterschiedlich. Diese Abweichungen sind besonders bei der WebView von Windows Phone zu erkennen, da diese auf den Internet Explorer aufbaut.

Diesem Problem kann zwar durch Bibliotheken wie JQuery Mobile entgegengewirkt werden, jedoch verschlechtern solch große Bibliotheken die Performanz, welche auf mobilen Geräten ohnehin nicht optimal ist. Deshalb haben wir uns entschlossen, so wenige Frameworks wie möglich zu verwenden und wenn dann nur solche, die sich auf eine bestimmte Funktionalität beschränken.

Gesamtheitlich gesehen hängt es von einigen Kriterien ab, ob man sich für eine native oder eine hybride App entscheiden soll. Der wichtigste Punkt dabei ist zu entscheiden, ob es für einen essentiell ist, dass die App auf möglichst vielen Plattformen erhältlich ist oder weitere Plattformen optional sind. Zudem ist es wichtig ob man eine App mit dem „Look and Feel“ der jeweiligen Plattform möchte, oder eine einheitliche Benutzeroberfläche. Fällt die Wahl auf eine App auf möglichst vielen Plattformen mit gleicher Benutzeroberfläche, ist es auf jeden Fall empfehlenswert, eine hybride HTML5 Anwendung zu entwickeln.

Literaturverzeichnis

APACHE SOFTARE FOUNDATION. *Windows Phone 8 Platform Guide*. [online]. [Accessed 29 Apr 2015]. Available from World Wide Web: <http://cordova.apache.org/docs/en/4.0.0/guide_platforms_wp8_index.md.html>

APACHE SOFTWARE FOUNDATION. 2014. *Apache Ant Project - Welcome*. [online]. [Accessed 27 May 2015]. Available from World Wide Web: <http://ant.apache.org/>

APACHE SOFTWARE FOUNDATION. 2015. *Overview*. [online]. [Accessed 06 Apr 2015]. Available from World Wide Web: <http://cordova.apache.org/docs/en/4.0.0/guide_overview_index.md.html>

APACHE SOFTWARE FOUNDATION. *Android Platform Guide*. [online]. [Accessed 29 Apr 2015]. Available from World Wide Web: <http://cordova.apache.org/docs/en/4.0.0/guide_platforms_android_index.md.html>

APACHE SOFTWARE FOUNDATION. *iOS Platform Guide*. [online]. [Accessed 29 Apr 2015]. Available from World Wide Web: <http://cordova.apache.org/docs/en/4.0.0/guide_platforms_ios_index.md.html#iOS%20Platform%20Guide>

APACHE SOFTWARE FOUNDATION. *Security Guide*. [online]. [Accessed 01 May 2015]. Available from World Wide Web: <http://cordova.apache.org/docs/en/4.0.0/guide_appdev_security_index.md.html>

APACHE SOFTWARE FOUNDATION. *The Command-Line Interface*. [online]. [Accessed 29 Apr 2015]. Available from World Wide Web: <http://cordova.apache.org/docs/en/4.0.0/guide_cli_index.md.html>

APPLE. 2012. *iOS Drawing Concepts*. [online]. [Accessed 06 Apr 2015]. Available from World Wide Web: <https://developer.apple.com/library/ios/documentation/2DDrawing/Conceptual/DrawingPrintingiOS/GraphicsDrawingOverview/GraphicsDrawingOverview.html>

APPLE. 2013. *Model-View-Controller*. [online]. [Accessed 21 Apr 2015]. Available from World Wide Web: <https://developer.apple.com/library/ios/documentation/General/Conceptual/DevPedia-CocoaCore/MVC.html>

APPLE. 2014. *Configuring the Viewport*. [online]. [Accessed 02 May 2015]. Available from World Wide Web: <https://developer.apple.com/library/ios/documentation/AppleApplications/Reference/SafariWebContent/UsingtheViewport/UsingtheViewport.html>

APPLE. 2015. *Choosing an iOS Developer Program*. [online]. [Accessed 24 Apr 2015]. Available from World Wide Web: <https://developer.apple.com/programs/start/ios/>

BALZERT, Helmut. 2010. Drei-Schichten-Architektur. *In*: *Java: Objektorientiert programmieren*, W3L GmbH, pp.230-231.

BODDENBERG, Ulrich. 2014. 17 Webserver (IIS). *In*: *Windows Server 2012 R2*, Rheinwerk Computing.

CAKE SOFTWARE. 2014. *Understanding Model-View-Controller*. [online]. [Accessed 30 Apr 2015]. Available from World Wide Web: <http://book.cakephp.org/2.0/en/cakephp-overview/understanding-model-view-controller.html>

DATACOM BUCHVERLAG. *SDK (software development kit)*. [online]. [Accessed 02 May 2015]. Available from World Wide Web: <http://www.itwissen.info/definition/lexikon/software-development-kit-SDK.html>

DAZER, Michael. *RESTful APIs - Eine Übersicht*. [online]. [Accessed 29 Apr 2015]. Available from World Wide Web: <https://www.snet.tu-berlin.de/fileadmin/fg220/courses/WS1112/snet-project/restful-apis_dazer.pdf>

GARSIEL, Tali and Paul IRISH. 2011. [online]. [Accessed 28 Apr 2015]. Available from World Wide Web: <http://www.html5rocks.com/de/tutorials/internals/howbrowserswork/>

GITHUB. 2015. *thebird/Swipe*. [online]. [Accessed 29 Apr 2015]. Available from World Wide Web: <https://github.com/thebird/Swipe>

GOOGLE. 2014. *Google Loader Developer's Guide*. [online]. [Accessed 08 May 2015]. Available from World Wide Web: <https://developers.google.com/loader/>

GOOGLE. 2015. *AngularJS API Reference ngRoute*. [online]. [Accessed 04 Apr 2015]. Available from World Wide Web: <https://docs.angularjs.org/api/ngRoute>

GOOGLE. 2015. *Creating Custom Directives*. [online]. [Accessed 30 Apr 2015]. Available from World Wide Web: <https://docs.angularjs.org/guide/directive>

GOOGLE. 2015. *Getting Started With Charts*. [online]. [Accessed 08 May 2015]. Available from World Wide Web: <https://developers.google.com/chart/image/docs/making_charts>

GOOGLE. 2015. *Tutorial: 7 Routing & Multiple Views*. [online]. [Accessed 30 Apr 2015]. Available from World Wide Web: <https://docs.angularjs.org/tutorial/step_07>

GOOGLE. *SDK Manager*. [online]. [Accessed 29 Apr 2015]. Available from World Wide Web: <https://developer.android.com/tools/help/sdk-manager.html>

GOOGLE. *Using Hardware Devices*. [online]. [Accessed 01 May 2015]. Available from World Wide Web: <http://developer.android.com/tools/device.html>

KÜHNEL, Andreas. 2013. 11 LINQ. *In*: *Visual C# 2012*, Rheinwerk Computing.

KÜHNEL, Andreas. 2013. 37 Einführung in das ADO.NET Entity Framework. *In*: *Visual C# 2012*, Rheinwerk Computing.

KÜHNEL, Andreas. 2013. 38 Datenbankabfragen des Entity Data Models. *In*: *Visual C# 2012*, Rheinwerk Computing.

MICROSOFT. 2015. *Overview of SSL/TLS Encryption*. [online]. [Accessed 29 Apr 2015]. Available from World Wide Web: <https://technet.microsoft.com/de-de/library/cc781476(v=ws.10).aspx>

MICROSOFT, MSDN. 2015. *ADO.NET*. [online]. [Accessed 04 Apr 2015]. Available from World Wide Web: <https://msdn.microsoft.com/de-de/library/e80y5yhx(v=vs.100).aspx>

MICROSOFT, MSDN. 2015. *ADO.NET-Architektur*. [online]. [Accessed 04 Apr 2015]. Available from World Wide Web: <https://msdn.microsoft.com/de-de/library/27y4ybxw(v=vs.100).aspx>

MICROSOFT, MSDN. 2015. *An Introduction to JavaScript Object Notation (JSON) in JavaScript and.NET*. [online]. [Accessed 24 Apr 2015]. Available from World Wide Web: <https://msdn.microsoft.com/en-us/library/bb299886.aspx>

MICROSOFT, MSDN. 2015. *Component Guidelines*. [online]. [Accessed 2015 Apr 30]. Available from World Wide Web: <https://msdn.microsoft.com/en-us/library/ee658121.aspx>

MICROSOFT, MSDN. 2015. *Einführung in Visual Studio*. [online]. [Accessed 12 Apr 2015]. Available from World Wide Web: <https://msdn.microsoft.com/de-de/library/fx6bk1f4(v=vs.90).aspx>

MICROSOFT, MSDN. 2015. *Konfigurieren von Diensten mit Konfigurationsdateien*. [online]. [Accessed 01 May 2015]. Available from World Wide Web: <https://msdn.microsoft.com/de-de/library/ms733932(v=vs.110).aspx>

MICROSOFT, MSDN. 2015. *SQL Server*. [online]. [Accessed 12 Apr 2015]. Available from World Wide Web: <https://msdn.microsoft.com/de-de/sqlserver/aa336270.aspx>

MICROSOFT, MSDN. 2015. *SQL Server Management Studio*. [online]. [Accessed 12 Apr 2015]. Available from World Wide Web: <https://msdn.microsoft.com/de-de/library/hh213248.aspx>

MICROSOFT, MSDN. 2015. *Was ist die Windows Communication Foundation*. [online]. [Accessed 04 Apr 2015]. Available from World Wide Web: <https://msdn.microsoft.com/de-de/library/ms731082(v=vs.110).aspx>

MOZILLA DEVELOPER NETWORK. 2015. *Same-origin policy*. [online]. [Accessed 011 May 2015]. Available from World Wide Web: <https://developer.mozilla.org/en-US/docs/Web/Security/Same-origin_policy>

QT. 2015. *Anchor-based Layout in QML*. [online]. [Accessed 25 Apr 2015]. Available from World Wide Web: <http://doc.qt.io/qt-4.8/qml-anchor-layout.html>

QT. 2015. *Connecting Android Devices*. [online]. [Accessed 25 Apr 2015]. Available from World Wide Web: <http://doc.qt.io/qtcreator/creator-developing-android.html>

QT. 2015. *Deploying Applications to Android Devices*. [online]. [Accessed 25 Apr 2015]. Available from World Wide Web: <http://doc.qt.io/qtcreator/creator-deploying-android.html>

QT. 2015. *Download Qt*. [online]. [Accessed 25 Apr 2015]. Available from World Wide Web: <https://www.qt.io/download/>

QT. 2015. *Layout Management*. [online]. [Accessed 06 Apr 2015]. Available from World Wide Web: <http://doc.qt.io/qt-4.8/layout.html>

QT. 2015. *Model/View Programming*. [online]. [Accessed 25 Apr 2015]. Available from World Wide Web: <http://doc.qt.io/qt-4.8/model-view-programming.html>

QT. 2015. *QML Applications*. [online]. [Accessed 25 Apr 2015]. Available from World Wide Web: <http://doc.qt.io/qt-5/qmlapplications.html>

QT. 2015. *Qt Charts*. [online]. [Accessed 25 Apr 2015]. Available from World Wide Web: <http://doc.qt.io/QtCharts/>

QT. 2015. *Qt for iOS*. [online]. [Accessed 25 Apr 2015]. Available from World Wide Web: <http://doc.qt.io/qt-5/ios-support.html>

QT. 2015. *Qt for WinRT*. [online]. [Accessed 25 Apr 2015]. Available from World Wide Web: <http://doc.qt.io/qt-5/winrt-support.html>

TANGELDER, Jorik. *Getting Started - Hammer.js*. [online]. [Accessed 29 Apr 2015]. Available from World Wide Web: <http://hammerjs.github.io/getting-started/>

Abbildungsverzeichnis

Quelltextverzeichnis

Internetverzeichnis

[1] https://msdn.microsoft.com/de-de/library/27y4ybxw(v=vs.100).aspx
[2] http://openbook.rheinwerk-verlag.de/windows_server_2012r2/17_005.html
[3] http://www.oio.de/public/xml/rest-webservices.htm
[4] www.angularjs.org/
[5] https://www.qt.io/download-open-source/
[6] https://dev.windows.com/en-us/develop/download-phone-sdk
[7] http://www.oracle.com/technetwork/java/javase/downloads/index.html
[8] http://developer.android.com/sdk/index.html
[9] http://ant.apache.org/bindownload.cgi
[10] http://codewithchris.com/deploy-your-app-on-an-iphone/
[11] https://www.bignerdranch.com/we-teach/how-to-prepare/ios-device-provisioning/
[12] https://nodejs.org/
[13] http://git-scm.com/
[14] http://plugins.cordova.io/#/viewAll

Abkürzungsindex

APK ..... *Android Package*
AVD ..... *Android Virtual Device*
CA ..... *Certification Authority*
EDM ..... *Entity Data Model*
HMAC ..... *Keyed-Hash Message Authentication Code*
IDE ..... *Integrated Development Environment*
IIS ..... *Internet Information Services*
JDK ..... *Java Development Kit*
JSON ..... *JavaScript Object Notation*
LINQ ..... *Language Integrated Query*
MAC ..... *Message Authentication Code*
Mac OS ..... *Macintosh Operating System*
MDC ..... *Mobile Dashcard*
MVC ..... *Model-View-Controller*
ORM ..... *Object-relational Mapp(er)ing*
QML ..... *Qt Meta Language*
ReST ..... *Representational State Transfer*
SOAP ..... *Simple Object Access Protocol*
SOP ..... *Same-Origin-Policy*
SSL ..... *Secure Socket Layer*
SVG ..... *Scalable Vector Graphics*
TLS ..... *Transport Layer Security*
URI ..... *Uniform Ressource Identifier*
VML ..... *Vector Markup Language*
VPN ..... *Virtual Private Network*
WCF ..... *Windows Communication Foundation*
XML ..... *Extensible Markup Language*