Technische Hochschule Mittelhessen
- Fachbereich MND -

Applikation zur Zeiterfassung mit SAPUI5

Ausarbeitung im Rahmen des Wirtschaftsinformatik-Projekts

vorgelegt von

Christoph Gerold (5036147)
Johanna Schwab (5037463)

Dozent: Prof. Dr. Peter Hohmann

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

1 Einleitung

Diese Ausarbeitung ist im Rahmen des Wirtschaftsinformatikprojektes an der Technischen Hochschule Mittelhessen entstanden. Es behandelt die Entwicklung einer Applikation zur Zeiterfassung mit dem JavaScript-Framework SAPUI5 und dem SAP Netweaver Gateway Service.

1.1 Motivation

Bisher werden webbasierte Anwendungen im Umfeld von SAP mit SAP Web Dynpro erstellt. Dieses Werkzeug wirkt allerdings nicht nur optisch überholt, sondern bietet dem Endnutzer auch nur eine stark eingeschränkte Möglichkeiten dem Endnutzer eine interaktive Web-Applikation zu erstellen. Darüber hinaus können mit SAP Web Dynpro nur bedingt unterschiedliche Endgeräte und damit einhergehende Bildschirmgrößen bedient werden.
Dagegen sollen sich zukünftig mit dem JavaScript-Framework SAPUI5 von der Firma SAP moderne unternehmensweite Web-Anwendungen erstellen lassen, die gleichermaßen für Desktop und mobile Endgeräte optimiert sind. Hierfür setzt das Framework auf den modernen Webtechnologien HTML5, sowie CSS3, auf und bindet zusätzlich gängige und etablierte JavaScript-Bibliotheken ein, die dem Entwickler eine reichhaltige Palette an Funktionen zur Verfügung stellen.

Da SAP zu den größten Unternehmen weltweit gehört und deren Produkte in vielen Konzernen und Betrieben eingesetzt werden, ist es interessant, Erfahrungen mit SAPUI5 zu sammeln und sich gegebenfalls in diesem Bereich eine Expertise aufzubauen. Angesichts des Potentials das damit einhergeht, ist der Aufbau von anfänglichem Know-How in diesem Bereich nicht zu unterschätzen. Daher soll mit diesem Wirtschaftsinformatikprojekt die Chance genutzt werden, grundlegende Kenntnisse über das neue Framework zu erhalten, was sowohl für Informatiker und Wirtschaftsinformatiker gleichermaßen vorteilhaft ist.

1.2 Aufgabenstellung und Zielsetzung

In der Arbeit sollen grundlegende Konzepte und Methoden von SAPUI5 betrachtet, beschrieben und hinsichtlich der Architektur untersucht werden. Die Aufgabe dieses Projekts besteht darin, eine Web-Applikation zur Zeiterfassung mit SAPUI5 zu entwickeln. Dazu gehört die grundlegende Einarbeitung in das JavaScript-Framework und den zu grunde liegenden Webtechnologien HTML5, CSS3 und JavaScript.

Das kurz- bis mittelfristige Ziel ist die Erstellung eines Prototypen, der die Kernanforderungen einer Zeiterfassungs-App beinhaltet. Dazu gehört die Erfassung der Zeit eines Mitarbeiters auf ein Projekt beziehungsweise auf einen Meilenstein. Das Projekt wiederum ist einem Kunden zugeordnet. Eine Master-Detail-Ansicht stellt dabei die Basis der App-Oberfläche dar. Die Daten werden dabei von einem lokalen Mockserver über eine oData-Schnittstelle von der Anwendung konsumiert

Das langfristige Ziel ist eine testgetriebene Anwendung, die über die genannten Kernanforderungen hinausgeht. Es können autorisierte Personen Datensätze für Kunden, Projekte, Meilensteine und Mitarbeiter anlegen sowie graphische Auswertungen zu den erfassten Daten erstellen. Dabei authentifiziert sich die Anwendung gegen das SAP-Backend und konsumiert die an der oData-Schnittstelle bereitgestellten Daten.

1.3 Aufbau dieser Arbeit

Die vorliegende Ausarbeitung ist in vier Kapitel unterteilt:

Einleitung - In diesem Kapitel werden die treibenden Gründe und Aufgaben für diese Ausarbeitung dargelegt.

Technische Grundlagen - Das zweite Kapitel nimmt Bezug auf die bei der Entwicklung eingesetzten Technologien, Techniken und Werkzeuge und erklärt kurz deren Integration und Interaktion.

Applikation zur Zeiterfassung - Im dritten Kapitel wird auf die fachlichen Anforderungen und die technische Umsetzung der Applikation eingegangen, sowie der finale Stand aufgezeigt.

Abschluss - Das vierte Kapitel fasst abschließend die gewonnenen Erfahrungen zusammen und gibt einen kurzen Ausblick über weitere Implementierungsmöglichkeiten.

2 Technische Grundlagen

In diesem Kapitel werden die notwendigen technischen Grundlagen für die Entwicklung mit dem Framework SAPUI5 vorgestellt. Anfangs wird ein Einblick in die Webtechnologien HTML5, CSS3 und JavaScript gegeben und abschließend SAPUI5 sowie der SAP NetWeaver Gateway im Allgemeinen beschrieben. Ein ausführlicher und tiefer Einblick in die Webtechnologien bleibt an dieser Stelle aus, da es den Umfang dieser Ausarbeitung übersteigen würde.

2.1 Verwendete Webtechnologien

2.1.1 HTML5 und CSS3

Die Hypertext Markup Language (HTML) ist eine Auszeichnungssprache, die überwiegend für die Erstellung von Webanwendungen verwendet wird. Dabei wird der HTML-Code individuell von der Rendering Engine des verwendeten Browsers übersetzt, was kleine Unterschiede, je nach verwendetem Browser, in der Darstellung der Webseite zufolge haben kann [Ant14, S. 17]. Das Grundgerüst einer HTML-Datei wird in Listing 2.1 gezeigt.

```
1 <!doctype html>
2 <html>
3   <head>
4   </head>
5   <body>
6   </body>
7 </html>
```

Listing 2.1: HTML-Grundgerüst

HTML5 ist die fünfte Version der Hypertext Markup Language, die am 28.10.2014 in der finalen Spezifikation[1] vom World Wide Web Consortium[2] vorgelegt worden ist. HTML5 wird

[1] http://www.w3.org/TR/html5/
[2] http://www.w3.org/

von allen modernen Browsern unterstützt. Die Besonderheiten an der fünften Fassung sind unter anderem [Meh10]:

- Nativer Audio- und Videoplayer
- Vereinfachte Bildunterschriften mit dem Tag <figcaption>
- Neue Elemente zur besseren Strukturierung von Inhalten

Um eine in HTML5 geschriebene Webseite optisch zu verbessern, kann bei der Implementierung zusätzlich Cascading Style Sheet (CSS) verwendet werden. CSS3 beschreibt die dritte Version von CSS. Ein paar Vorteile dieser dritten Version sind nachfolgend aufgelistet:

- Setzen von abgerundeten Ecken
- Schatten auf Elemente legen
- Definition von Farbverläufen

Bei CSS handelt es sich um eine deklarative Gestaltungssprache [wik], mit der in HTML erstellte Elemente gestaltet und positioniert werden können. Es lassen sich mit Hilfe von CSS Design und Inhalt einer Webseite voneinander getrennt behandeln. Zusätzlich zum modularen Aufbau der Programmiersprache wird dadurch die Übersichtlichkeit und Lesbarkeit des Codes erhöht. Innerhalb einer HTML-Datei verwendeter CSS-Code muss im Head-Bereich untergebracht werden. Auch das Einbinden von externen Dateien, in denen CSS-Code ausgelagert wurde, muss ebenfalls im Kopfbereich des HTML-Quellcodes stattfinden. In Listing 2.2 wird als Anwendungsbeispiel der Hintergrund einer Webseite mit CSS blau gefärbt.

```
1  <!doctype html>
2  <html>
3      <head>
4    <style>
5      body {
6           background-color: #0000ff;
7      }
8    </style>
9      </head>
10 <body>
11 </body>
12 </html>
```

Listing 2.2: Beispielanwendung CSS3

2.1.2 JavaScript und AJAX

Bei JavaScript handelt es sich um eine Skriptsprache, die auf Webseiten eingebunden werden kann, um diese dynamischer zu gestalten. Programme werden clientseitig individuell von Internetbrowsern interpretiert [Kro], wodurch es zu unterschiedlichen Darstellungen der Endanwendung kommen kann. JavaScript bietet sowohl die Möglichkeit prozedural (d.h. methodenbasiert), als auch objektorientiert zu entwickeln. Dafür wird der entsprechende Code mit dem Tag <script> umfasst. In Listing 2.3 wird eine Alert-Box erzeugt, welche dem Benutzer die Ausgabe "Hallo Welt" darstellt.

```
<script type="text/javascript">
    window.alert("Hallo Welt");
</script>
```

Listing 2.3: JavaScript zum Erzeugen einer Alert-Box

Javascript lässt sich auch mit HTML-Elementen verbinden und kann diese an beliebigen Stellen manipulieren. Das Document Object Model (DOM) ist die Schnittstelle zwischen HTML und dynamischem JavaScript. Alle Elemente werden zu Objekten, die dynamisch aufgerufen, verändert, hinzugefügt und gelöscht werden können. Der nachfolgende Programmteil in Listing 2.4 verändert den Text des a-Tags nach einem Klick von "Alt" zu "Neu":

```
// HTML a-Tag
<a href="#" id="click_me"> Alt </a>

// Inline JavaScript
<script type="text/javascript">
    document.getElementById('click_me').onclick = function () {
        document.getElementById('click_me').innerHTML = 'Neu';
    };
</script>
```

Listing 2.4: Mit JavaScript das DOM manipulieren

Die oben gezeigten Listings stellen sogenanntes InlineJavaScript dar - sprich JavaScript, das im HTML-Markup definiert wird. Daneben kann das JavaScript auch in Dateien ausgelagert und bei Bedarf eingebunden werden. Wobei letzteres die empfohlene Variante ist.
Javascript ist vor allem dafür bekannt auf der Clientseite im Browser dynamische Funktionen auszuführen. In den letzten Jahren gab es mit Node.js eine Software die es auch erlaubt JavaScript serverseitig auszuführen.

Durch die zusätzliche Verwendung von Asynchronous Javascript and XML (Ajax) lassen sich Serveranfragen durchführen und umsetzen, ohne dass eine Webseite dafür komplett neu

geladen werden muss [web]. Es werden nur die Webseitenelemente nachgeladen, die sich auf die gestellte Anfrage beziehen. Mit dieser Methoden werden Webseiten optimiert, da ein individuelles Nachladen einzelner Elemente Zeit und Ressourcen spart. Allerdings ist ein Request vom Client zum Server meist eine Einschränkung der Anwendungsperformance. Daher sollte sparsam mit dem Einsatz von Ajax umgegangen werden.

2.2 JavaScript-Framework SAPUI5

Das SAP NetWeaver Development Toolkit für HTML5, kurz SAPUI5 ist ein Framework der Firma SAP, welches auf HTML5, CSS3, JavaScript und jQuery basiert. Es dient der Entwicklung von clientseitigen Webanwendungen. Als Entwicklungsumgebung ist nicht die SAP-Oberfläche notwendig, sondern es können auch externe Programme zur Softwareentwicklung, beispielsweise Eclipse[3] oder PHPStorm[4], verwendet werden. Die Entwicklung wird dadurch flexibler und ist nicht mehr an ein SAP-Backend gebunden. Mit SAPUI5 programmierte Anwendungen sind responsive, das heißt, die Darstellung der Anwendung ist unabhängig von der Bildschirmgröße des Endgerätes. Somit können die Applikationen für Desktop- und Mobile-Geräte verwendet werden, ohne den Programmcode mit großem Aufwand individuell anpassen zu müssen.

2.2.1 Quelloffene Version OpenUI5

Die quelloffene Version OpenUI5[5] ist unter der Apache 2.0 Lizenz verfügbar und kann von der offiziellen Webseite heruntergeladen werden. Der Sourcecode ist größtenteils identisch und es fehlen lediglich einige Bibliotheken und User-Interface-Elemente in der freien Version [con]. Verbesserungen und Fehlerbehebungen innerhalb der Bibliotheken finden in beiden Versionen statt. Im Rahmen dieser Ausarbeitung sprechen wir von SAPUI5, meinen damit aber grundsätzlich beide Varianten.

2.2.2 User-Interface-Elemente und Controls

Das JavaScript-Framework stellt eine Vielzahl an vordefinierten Benutzerelementen bereit. Dazu zählen unter anderem Tabellen, Navigationselemente, Tabs und Aufzählungslisten. Ideen

[3] https://eclipse.org/
[4] PHPIDE__JetBrainsPhpStorm
[5] http://openui5.org/

und Anregungen für weitere mögliche User-Interface-Elementen können den Webapplikationen SAPUI5 Explored[6] oder OpenUI5 Explored[7] entnommen werden.

2.2.3 Unterstützte Datenaustauschformate JSON, XML und OData

Das JavaScript-Framework SAPUI5 unterstützt die drei unterschiedlichen Datenaustauschformate JSON, XML und OData. Während die beiden Formate JSON (Listing 2.5) und XML (Listing 2.6) recht bekannt aus dem Bereich der Webentwicklung sind, wird nachfolgend nur das Format OData näher beschrieben.

```
[
  {
    "Id": 1,
    "Title": "Relaunch SAP CRM"
  },
  {
    "Id": 2,
    "Title": "Support SAP BI"
  }
]
```

Listing 2.5: JSON Beispiel

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<document>
  <project>
    <id>1</id>
    <title>Relaunch SAP CRM</title>
  </project>
</document>
```

Listing 2.6: XML Beispiel

Das Open Data Protokoll[8] (OData) ist ein standardisierte Webprotokoll, das durch den SAP NetWeaver (mehr dazu im Kapitel 2.3) unterstützt wird. Es verwendet das HTTP-Protkoll und somit auch die CRUD-Operationen (anlegen, lesen, ändern und löschen) auf Datensätze. Darüber hinaus ist dsa Protokoll auf der REST-Architektur (Representational State Transfer Architecture) aufgebaut, bei der eine Ressource immer eindeutig identifizierbar ist [Roh]. Die Daten werden, nicht wie bei den vorherigen Models auf Client-Seite, sondern hier auf Server Seite gehalten. Listing 2.10 zeigt, wie ein OData-Model erzeugt werden kann. Die

[6] https://sapui5.netweaver.ondemand.com/sdk/explored.html
[7] https://openui5.hana.ondemand.com/explored.html
[8] www.odata.org

eingebundene URL zeigt im Falle von SAP-Anwendungen zu einem angelegten Service aus dem SAP-Backend.

```
var oModel = new sap.ui.model.odata.ODataModel("<URL zum Service>");
```

Listing 2.7: OData-Model

Ein OData-Model kann sich immer nur auf einen Service beziehen [Spy15]. Falls mehrere Services verwendet werden sollen, muss für jeden ein neues OData-Model geladen werden.

2.2.4 Datenbindung an User-Interface-Elemente

Damit die verwendeten Control-Elemente des User-Interfaces die gewünschten Daten anzeigen können, muss eine korrekte Datenanbindung erfolgen. Die Datenanbindung kann einseitig oder zweiseitig durchgeführt werden. Einseitig bedeutet, dass die Daten aus dem Model nur in der entsprechenden View ausgelesen, beziehungsweise. dargestellt werden können. Unter zweiseitig ist zu verstehen, dass über eine Eingabefläche Daten erfasst und diese an das Model zurückgegeben werden können.
Das Listing 2.8 zeigt, wie eine Instanz eines JSON-Models mit den Daten aus Listing 2.5 erzeugt, und an das UI-Control Table gebunden werden kann [Ant14, S. 159 f.].

```
var oJSONModel = new sap.ui.model.json.JSONModel();

oJSONModel.loadData("<Pfad zur JSON-Datei>");

// Model an UI-Element Table binden
oTabel.setModel(oJSONModel);
oTable.bindRows("<Pfad innerhalb der JSON-Datei>");
```

Listing 2.8: JSON-Model

Die Anbindung an XML-Daten verläuft ähnlich wie beim JSON-Model. In Listing 2.9 ist kurz dargestellt, wie ein XML-Model mit Angabe des Pfades zu den XML-Daten erzeugt und im weiteren Verlauf das User-Interface-Control Table angebunden wird.

```
var oXMLModel = new sap.ui.model.xml.XMLModel();
oXMLModel.loadData("<Pfad zur XML-Datei>");

// Model an UI-Element Table binden
oTabel.setModel(oXMLModel);
oTable.bindRows("<Pfad innerhalb der XML-Datei>");
```

Listing 2.9: XML Model

Auch die OData-Anbindung erfolgt fast nach dem gleichen Vorgehen wie auch schon bei JSON und XML und wird in Listing 2.10 gezeigt. Zuerst wird ein OData-Model mit der gegebenen Servie-URL instanziiert und im Anschluss an das gewünschte User-Interface-Control gebunden.

```
1 var oODataModel = new sap.ui.model.odata.ODataModel("<URL zum
      Service>");
2
3 //Model an UI-Element "Table" binden
4 oTabel.setModel(oODataModel);
5 oTable.bindRows("<Pfad innerhalb des OData-Services>");
```

Listing 2.10: OData Model

2.3 SAP Netweaver Gateway

Der SAP Netweaver Gateway ist die Datenschnittstelle zwischen dem SAP-Backend und einer Webanwendung. Dadurch können Daten einfach verwaltet und ein Zugriff darauf erleichtert werden. Diese flexiblen Strukturen ermöglichen den Zugriff von unterschiedlichen Plattformen auf eine zentrale Datenquelle. Die Abbildung 2.1 zeigt, wo das SAP Gateway im SAP Umfeld einzuordnen ist und welche Komponenten dabei eine Rolle spielen. Der Gateway unterliegt verschiedenen SAP-Systemen und kann über eine OData-Schnittstelle mit Programmen außerhalb des SAP-Umfeldes kommunizieren. Das Gateway selbst besteht aus drei Hauptkomponten: den Werkzeugen, der Core Technologie und der Datenverbindung.
Zu den Tools, die in Verbindung mit dem Gateway verwendet werden, zählt der Service Builder oder auch Eclipse (mit Plugin-Erweiterung möglich). Zudem ist das Gateway, aufgrund der verwendeten Gateway Core Technologie, leicht wartbar, besitzt eine Monitoringfunktion und Sicherheitsstandards. Die Daten, die der Service verwenden soll, können mit dem Business Object Layer (BOL), dem Service Provider Infrastructure (SPI), der Business API (BAPI) oder per Remote Function Call (RFC) übergeben werden.

Durch das verwendete OData-Protokoll wird ein standardisierter Zugriff ermöglicht [Ant14, S. 343], der auf einer REST-Architektur basiert. Aufgrund dieser Architektur arbeitet das Gateway zustandslos und kann nur zustandslose Anwendungen verwalten.
Das Gateway stellt die Services bereit, die von unterschiedlichen Plattformen verwendet werden können. Dabei ist allerdings die Anzahl der Standard-Services, die bereits im System implementiert sind, sehr gering, da diese sich meist auf einen konkreten Anwendungsfall beziehen [Bö14, S. 179]. Es ist aber möglich, eigene Serivces mit dem SAP Gateway Service

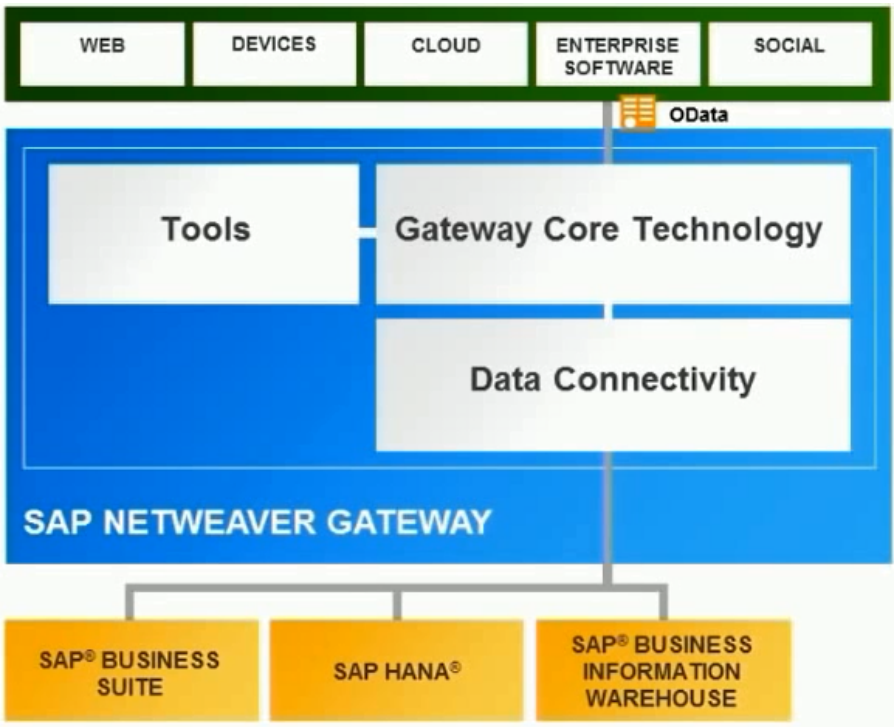

Abbildung 2.1: NetWeaver Gateway

Builder anzulegen und individuell anzupassen.
Dieser Prozess wird in drei Phasen unterteilt [Bö14, S. 182]:

Erste Phase: In der ersten Phasen wird das Datenmodell des Service definiert. Es können die Einstellungen zu Entitätstypen, Entitätsmengen, Assoziationen und weiteren Komponenten vorgenommen werden, welche schließlich vom Service verwendet werden.

Zweite Phase In dieser Phase findet die Serviceimplementierung statt. Hierbei werden die Methoden des Service ausimplementiert.

Dritte Phase Die dritte Phase dient der Serviceverwaltung. Hier wird der Service dem Servicekatalog hinzugefügt und aktiviert. Anschließend kann er vom Client konsumiert werden.

Ein Gateway-Service wird grundsätzlich durch zwei Abap-Klassen implementiert, die, inklusive Unterklassen, nach der ersten Phase automatisch erstellt werden können [SAP14]. Dabei handelt es sich um die Model Provider Klasse (MPC) und die Runtime Data Provider Klasse (DPC). In der MPC wird das verwendete Modell definiert und zugehörige Entitäten und Properties erzeugt. In der DPC werden die CRUDQ-Methoden (Create, Read, Update, Delete

und Query) des Modells ausimplementiert. Auch auf die Entitäten des Services selbst können ebenfalls die einfachen OData-CRUDQ-Operationen angewendet werden. Die Entitäten können so individuell auf einen Anwendungszweck angepasst werden.

2.4 Versionskontrollsystem Git

Git ist ein Versionskontrollsystem für Software und reiht sich zu bekannten weiteren Systemen wie beispielsweise Subversion (SVN) ein. Im Gegensatz zu SVN speichert Git aber keine Differenzen zwischen zwei Versionen einer Datei ab, sonder erstellt Schnappschüsse des momentanen Zustandes der Datei.

Die Arbeit mit Git erfolgt größtenteils lokal. Die Sicherung des momentanen Zustandes in einem lokalen Repository nennt man Commit. Beim Committen ist es möglich eine kurze Nachricht zu verfassen, die beschreibt, was in diesem Commit passiert ist. Erst wenn die Änderungen in ein entferntes Repository gespeichert werden sollen, wird eine Netzverbindung benötigt. Das Sichern wird im Git-Kontext Push genannt.
Git berechnet Prüfsummen für Änderungen und nutzt diese anschließend um die versionierten Änderungen zu referenzieren. Dadurch bekommt Git sofort mit, wenn sich Dateien geändert haben.

Git stellt damit eine dezentrale Versionsverwaltung zur Verfügung, die erst auf einem entfernen Repository zusammenläuft. Die Versionskontrolle ist von Beginn an in diesem Projekt verwendet worden, um das parallele Arbeiten am Projekt zu ermöglichen und jederzeit einen stabilen Stand zu haben beziehungsweise wiederherstellen zu können.

3 Applikation zur Zeiterfassung

In diesem Kapitel werden die fachlichen und technischen Anforderungen für die Applikation zur Zeiterfassung beschrieben. Es werden unter anderem die Use-Cases, Mockups und das Klassendiagramm dargestellt sowie Hintergründe zur Programmierung gegeben.

3.1 Projektplanung

Für die Einarbeitung in das JavaScript-Framework SAPUI5 ist ein Anwendungskontext aufgestellt worden. An diesem orientiert soll das Programm schrittweise entwickelt werden. Ziel diese Projektes ist es, eine Applikation zur Zeiterfassung von Mitarbeitern mit Hilfe des Frameworks zu entwickeln. Die dafür gesetzten notwendigen Anforderungen werden im Abschnitt 3.2 aufgezeigt. Im Anschluss daran wird die technische Realisierung in Abschnitt 3.3 vorgestellt und näher auf die Herausforderungen bei der Implementierung eingegangen. Zu guter Letzt wird in Abschnitt 3.3.5 der finale Entwicklungsstand beschrieben.

3.1.1 Organisation

Das Projekt wird mit der Unterstützung unterschiedlicher Online-Werkzeuge organisiert. Auf der technischen Seite wird Bitbucket[1] verwendet, das neben der Bereitstellung von Git-Repositories auch ein Ticket-System anbietet, mit dem die Aufgaben festgehalten und innerhalb des Projektteams verteilt werden können. In Abbildung 3.1 ist das Dashboard von Bitbucket für eingeloggte Benutzer dargestellt.

Auf der anderen Seite ist Google Docs[2] eingesetzt worden, um die insbesondere die fachlichen Anforderungen an das Projekt im Team festzuhalten, auszuformulieren und bei Bedarf anzupassen. Der Vorteil von Google Docs ist das gleichzeitige Bearbeiten von gemeinsamen Dokumenten durch mehrere Benutzer.

[1] `https://bitbucket.org/`
[2] `https://www.google.de/intl/de/docs/about/`

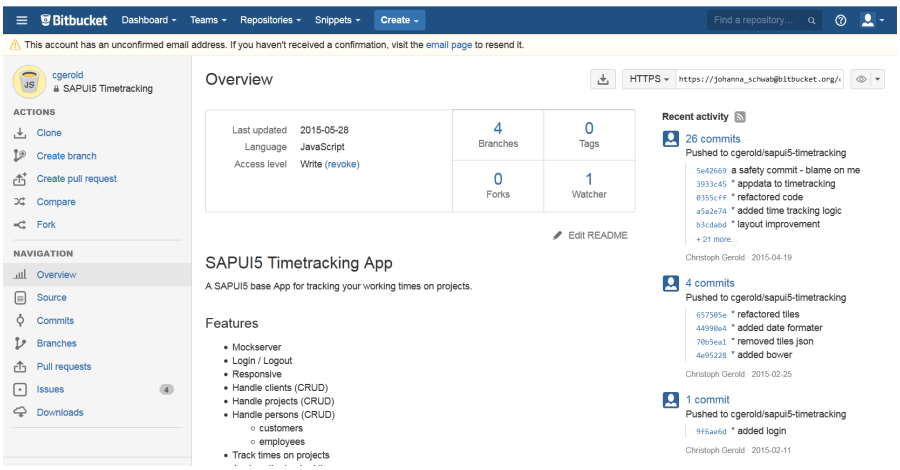

Abbildung 3.1: Dashboardansicht von Bitbucket

3.1.2 Vorgehen und Meilensteine

Da zu Beginn des Projektes noch nicht bekannt war, ob die fachlich gesetzten Anforderungen mit dem JavaScript-Framework umzusetzen sind, wird die Umsetzung des Projektes auf einem prototypischen Ansatz basieren. Mit Hilfe von Prototyping können fachliche Aspekte frühzeitig auf ihre Machbarkeit hin überprüft und gegebenenfalls angepasst werden.
Auf der technischen Seite wird das Prototyping vom Versionskontrollsystem Git unterstützt. Ziel ist es, immer einen Zweig im Version Control System (VCS) zu haben, der einen stabilen und vorzeigbaren Stand aufweist und damit eine Art evolutionäres Prototyping darstellt. Von diesem stabilen Stand ausgehend lassen sich Entwicklungsstränge abzweigen, auf denen dann nach, Art des explorativen Prototypings überprüft wird, ob gewünschte Freatures entsprechend den Vorgaben umzusetzen sind bzw. welche Alternativen es gibt.

Für den zeitlichen Rahmen und die inhaltlichen Ziele sind passend zu dem Prototyping-Ansatz sechs Meilensteine aufgestellt worden. Für jeden Meilenstein ist ein zeitlicher Rahmen geschätzt worden. Erst wenn das im Meilenstein definierte Ziel erreicht wurde, konnte mit der Umsetzung des nächsten Meilensteins begonnen werden. Nachfolgend sind die Meilensteine mit ihren jeweiligen Zielen beschrieben:

1. Meilenstein 27.10 – 23.11

Ziel des ersten Meilstenteins bestand darin, erste grundlegende Kenntnisse von SAPUI5 zu erlangen und eine lauf- und arbeitsfähige Entwicklungsumbebung aufzusetzen. Zu Beginn der Projektarbeit erfolgte deshalb eine grobe Einarbeitung in das Thema. Dabei ging es hauptsächtlich darum, einen ersten Eindruck über das SAPUI5-Framework, die

Architektur und die Komponenten zu bekommen. Nachdem ein erstes Verständnis des Gesamtkonzepts errungen war, wurden auch erste Gedanken über das Datenaustauschformat OData und die Entwicklungsumgebung Eclipse festgehalten.

2. Meilenstein 24.11 – 30.11

Ziel des zweiten Meilensteins war es, ein einheitliches Verständnis über die Struktur der Applikation aufzubauen. Dafür wurde, nach einer ausgiebigen Analyse der Anforderungen, das Grobkonzept entwickelt und Mockups gezeichnet.

3. Meilenstein 08.12 – 18.01

Ziel des dritten Meilensteins war ein präsentierbarer Prototyp auf Basis eines Mockservers, mit dem folgenden Funktionen:

- Zeit auf Projekt buchen
- Kunden auflisten
- Projekte auflisten
- Analyse-Chart der letzten X Tage über die erfassten Arbeitszeiten

Um das Ziel zu erreichen, wurde das Feinkonzept ausgearbeitet und mit agilem Prototyping umgesetzt. Dafür wurden die Aspekte Mockserver und Simulation der OData-Schnittstelle berücksichtigt und ausgiebig getestet. Am Ende des Meilensteins wurde bei der Prototypenentwicklung allerdings die Analyse der Zeiteinträge nicht umgesetzt, da der Aufwand der Implementierung überschätzt wurde.

4. Meilenstein 19.01 – 07.02

Nach diesem Meilenstein sollten die, für die Applikation benötigten Daten im SAP-Backend und ein Service zum Konsumieren, vorhanden sein. Dafür wurden im Backend Datenbanktabellen angelegt und ein NetWeaver Gateway Service modelliert und implementiert.

5. Meilenstein 08.02 – 08.03

Mit dem fünften Meilenstein sollte das Ziel erreicht werden, Daten über die OData-Schnittstelle aus dem SAP-Backend aus der Applikation heraus zu verwenden. Es wurde eine testgetriebene Weiterentwicklung und Qualitätssicherung der Anwendung betrieben und eine Verbindung mit dem Backend hergestellt.

6. Meilenstein 09.03 – 22.03

Ziel des letzten Meilensteins war eine lauffähige Anwendung zu erhalten, die alle wichtigen Anforderungen erfüllt. Dabei wurde auch Wert auf eine abschließende Dokumentation gelegt.

3.2 Software Requirement Specification

Die Anforderungen an die Anwendung, sowie Abhängigkeiten und Einschränkungen, werden in diesem Abschnitt erklärt.

3.2.1 Allgemeine Beschreibung

3.2.1.1 Akteure

Die Software wird von Unternehmen eingesetzt um die Zeiterfassung der Mitarbeiter zu bewerkstelligen. Die Unternehmen, für die die Software interessant ist, arbeiten hauptsächlich im Projektbetrieb, da einzelne Zeiten direkt auf Projekte gebucht werden können.

3.2.1.2 Anwendung

Die Anwendung kann von verschiedenen Endgeräten und Browsern aufgerufen werden. Wird eine Datenanfrage vom Benutzer gestellt, werden die Daten direkt aus dem SAP-Backend geladen und angezeigt. Dafür wird die Anfrage über die OData-Schnittstelle zum Gateway-service geleitet, der auf die Datenbanken im SAP-Backend zugreift. Ebenso können auch Benutzereingaben langfristig in der Datenbank gespeichert werden. In Abbildung 3.2 ist ein schematischer Aufbau der Datenübertragung dargestellt.

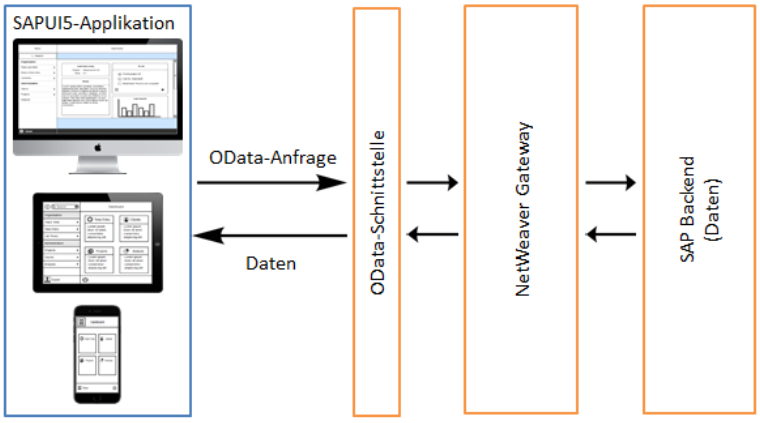

Abbildung 3.2: Datenübertragung der Applikation

3.2.1.3 Funktionsübersicht

Login – Beim Starten der Applikation muss der Benutzer sich mit einem individuellen Benutzernamen und einem Passwort anmelden, damit er zur Dashboardansicht gelangt. Erst bei einer erfolgreichen Anmeldung, kann die Anwendung verwendet werden. Dadurch ist es möglich jedem Anwender eine individuelle Ansicht zu gewährleisten und erfasste Zeiteinträge automatisch den Anwendern zuzuordnen.

Projekte, Kunden und Personen – Mit Hilfe der Applikation zur Zeiterfassung können Projekte, Kunden und Personen verwaltet werden. Die Daten sind für den Benutzer in einer Listenansicht einsehbar und nach vielfältigen Kriterien filter- und gruppierbar. Einzelne Datensätze können jederzeit in einer Detailansicht genauer betrachtet werden und individuell hinzugefügt, gelöscht oder bearbeitet werden. Eine Archivierungsmöglichkeit bietet an, dass die Datensätze bestehender Projekte und Kunden nicht komplett gelöscht, sondern zu einem späteren Zeitpunkt erneut eingesehen und zurück in die Datenbank aufgenommen werden können. Eine, in die Datenbank eingepflegte Person, ist entweder ein Mitarbeiter oder ein Kunde. Wobei ein Mitarbeiter einen Mitarbeiterstundensätzen zugewiesen bekommt, um damit, und aus den gebuchten Zeiten, die Gesamtkosten für ein Projekt bestimmen zu können.

Zeiterfassung – Der Kern der Applikation ist die Zeiterfassung auf Projekte, an denen die Mitarbeiter arbeiten. Diese können einen individuellen Zeiteintrag auf ein Projekt buchen, an dem sie gearbeitet haben. Die Mitarbeiter sind befugt ihre gebuchten Zeiteinträge einzusehen und diese nach vielfältigen Kriterien zu sortieren und zu filtern, um eine bessere Übersicht der Einträge zu erhalten.

Analyse – Die Analyse der erfassten Daten soll dem Anwender eine Übersicht über Zeiten und Kosten von Projekten, Kunden und Personen liefern. Mit einer grafischen und textuellen Auswertung lassen sich wichtige Fakten auf einen Blick erkennen.

Hierbei soll stets gewährleistet sein, dass aufgerufene Seiten gebookmarked und jederzeit erneut über die gespeicherte Adresse aufgerufen werden können.

3.2.2 Spezifische Anforderungen

3.2.2.1 Funktionale Anforderungen

Die Anwendung soll an verschiedene Endgeräte angepasst sein und sich, sowohl auf einem Desktop-Computer, wie auch auf einem kleinen Smartphone-Bildschirm benutzerfreundlich bedienen lassen.

Routing – Ein weiteres wichtiges Kriterium der Applikation ist ein korrektes Routing, mit dem jede Seitenansicht gebookmarkt werden kann. Dazu ist eine individuelle URL notwendig. Mit einer Standard-URL kann man jederzeit auf den Ausgangspunkt der Applikation, das Dashboard, gelangen. Diese URL könnte wie folgt aussehen: `http://openui5.app/index.html#`
Möchte der Anwender zu der Auflistung aller Kunden gelangen, kann er sich diese beispielsweise immer mit der URL `http://openui5.app/index.html#/clients` anzeigen lassen. Eine Detailansicht eines bestimmten Kunden kann mit einem Zahlenzusatz aufgerufen werden: `http://openui5.app/index.html#/clients/3`.
Wird eine Adresse der Applikation aufgerufen, die nicht vorhanden ist, erscheint eine Fehlermeldung, die den Anwender auf den Fehler aufmerksam macht. Die Routingtabelle 3.1 zeigt die in der Applikation verwendeten Routingpfade.

Use Cases und Mockups – Zu Begin der Projektarbeit wurde ein Use-Case gezeichnet, welches in Abbildung 3.3 dargestellt ist. Dafür wurden erste Anforderungen des Benutzers gesammelt. In einem weiteren Schritt wurden Mockups der Applikation mit der Software Lucidchart[3] gezeichnet. Damit wurde ein einheitliches Verständnis der Applikation gewährleistet. Das Mockup zum Dashboard, auf verschiedenen Endgeräten, ist in Abbildung 3.4 dargestellt und zeigt den Einstiegspunkt der Applikation. Von hier aus kann der Anwender alle weiteren Wege durch die Applikation beginnen. Weitere Mockups verschiedenen Ansichten befinden sich im Anhang.

[3] `www.lucidchart.com`

#	Master	Detail	Kontext	URL-Pattern
1	Dashboard	Dashboard	Startseite mit Hauptnavigation	/
2	Client	Dashboard	Benutzer klickt in der Hauptnavigation auf Clients: Clients werden als Liste in der Navigation angezeigt	/clients
3	Client	Client	Clients werden als Liste angezeigt und für einen ausgewählten Client die Detailansicht	/clients/5
4	Client	Project	Der Benutzer befindet sich in der Einzelansicht eines Kunden und klickt auf eines der Kundenprojekte. Navigationsführung im Detail über ein Breadcrumb möglich	/clients/5/projects/10
5	Project	Dashboard	Projekte werden als Liste in der Navigation angezeigt	/projects
6	Project	Project	Projekte werden als Liste in der Navigation und das ausgewähltes Projekt in der Detailansicht angezeigt	/projects/7
7	Dashboard	Tracktime	Mitarbeiter erfasst seine Zeit vom Dashboard	/track-time
8	Project	Tracktime	Mitarbeiter erfasst seine Zeit aus der Einzelansicht eines Projektes	/projects/7/track-time
9	Dashboard	Listtime	Einstiegspunkt, um die erfassten Zeiten anzeigen zu lassen - Ansicht ohne Filterung	/list-times
10	Dashboard	Listtime	Anzeige der erfassten Zeiten mit einer Filterung	/list-times{FILTER?}

Tabelle 3.1: Routingtabelle der Zeiterfassungsapplikation

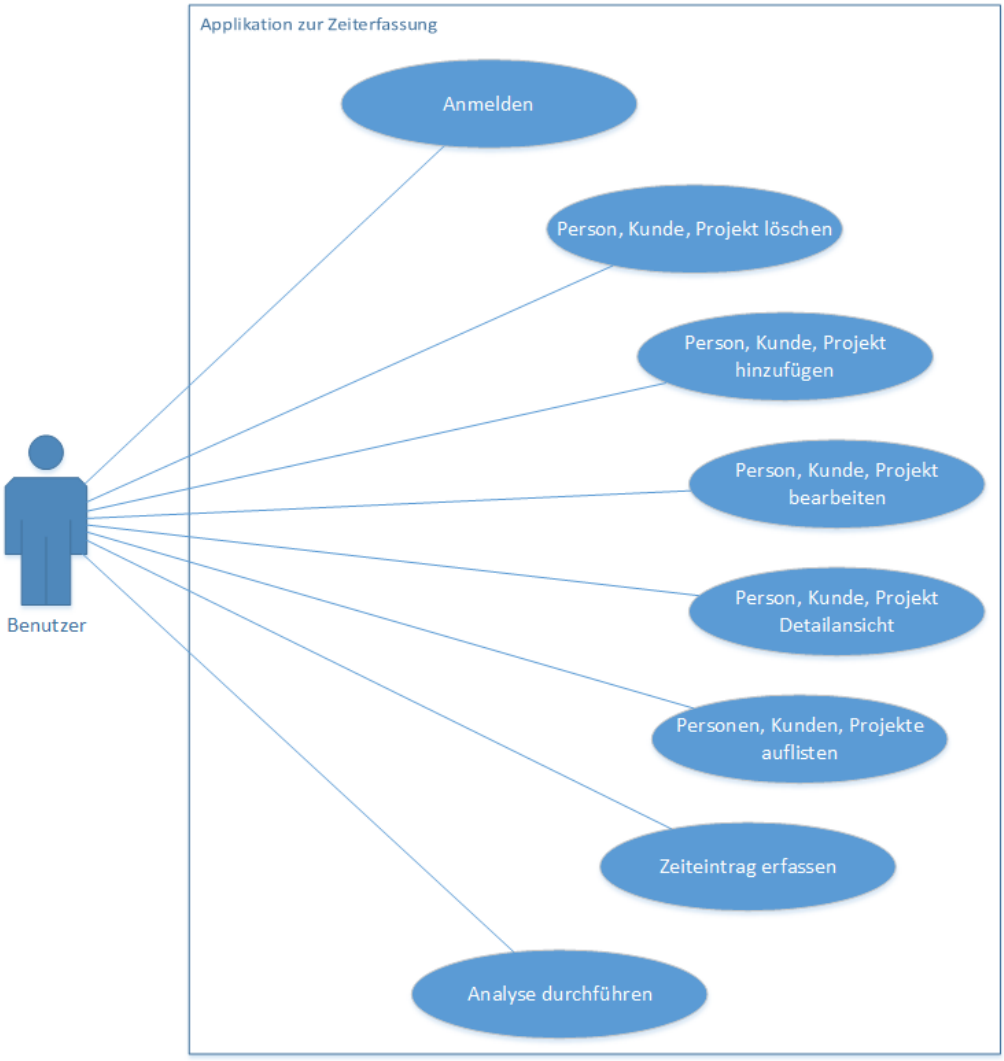

Abbildung 3.3: Use-Case Diagramm

Logische Datenbankstruktur – Aus den identifizierten Objekten und deren Zusammenspiel ergibt sich das in 3.5 abgebildete Klassendiagramm.

Abbildung 3.4: Dashboardansicht auf unterschiedlichen Endgeräten

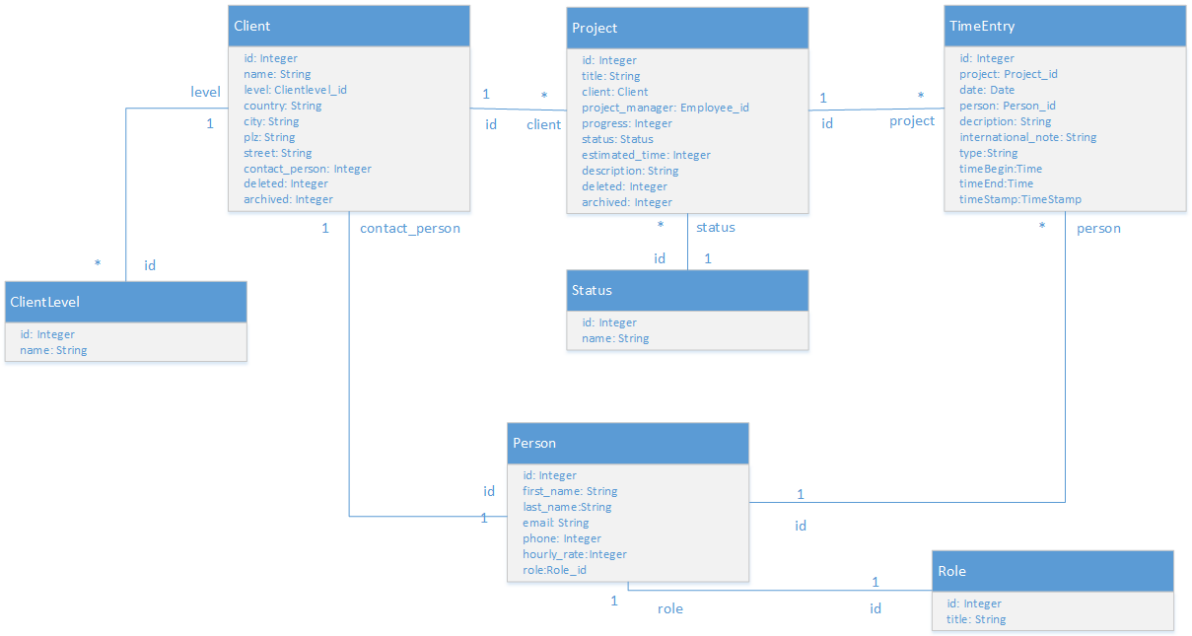

Abbildung 3.5: Klassendiagramm der Applikation zur Zeiterfassung

3.2.2.2 Nicht funktionale Anforderungen

Zu den nicht funktionalen Anforderungen zählt unter anderem die Qualität der Endanwendung. Diese soll jederzeit stabil laufen und auch bei falschen Eingaben von Benutzern nicht abstürzen. Bei Fehlfunktionen wird dem Anwender eine Fehlermeldung über den Grund und mögliche Maßnahmen für die Behebung ausgegeben. Des Weiteren ist für die Anwedung die Performanz nicht zu unterschätzen. Die Benutzer erwarten eine schnelle Ladezeit der Internetseiten, damit sie nicht auf wichtige Inhalte warten müssen. Bei der Zeiterfassungsapplikation soll eine Internetseite deshalb innerhalb von drei Sekunden vollständig geladen sein, damit ein stressfreier Gebrauch gewährleistet werden kann. Zu der schnellen Ladezeit kommen zusätzlich noch die Konsistenz der Daten und deren Sicherheit hinzu. Um beides zu erreichen, werden die Daten über eine OData-Schnittstelle aus dem SAP-Backend geladen.

3.3 Technische Implementierung und Herausforderungen

In diesem Abschnitt wird die technische Konzeption und die damit einhergehende Umsetzung der Anwendung beschrieben. Grundlage hierfür sind die ermittelten Anforderungen aus Abschnitt 3.2.

3.3.1 Entwicklungsrichtlinien

Die Basis einer wartbaren Anwendung und der gemeinsamen Entwicklung im Team ist die Einhaltung von Richtlinien. Daher sind für das Projekte folgende Punkte festgelegt worden, die selbstverständlich auch als Ausgangsbasis für weitere Projekte dienen und den Einstieg für Entwickler vereinfachen sollen.

Coding Guidelines – Grundsätzlich wird sich beim Programmieren an die offiziellen Best Practices[4] des JavaScript-Frameworks gehalten. Darüber hinaus sind notwendige Kommentare im Quellcode entsrechend der JsDoc[5] zu machen. Dies hilft auch Dritten einen schnellen Ein- und Überblick in das Projekt zu bekommen.

Projektstruktur – Die Struktur des Projekts ist auf Dateisystemebene in Abbildung 3.6 abgebildet und basiert ebenfalls auf den Best Practice von SAPUI5.

Version – Es wird die Openui5-Version 1.26.7 verwendet.

[4] https://sapui5.netweaver.ondemand.com/sdk/demoapps.html
[5] http://usejsdoc.org/

3.3.2 MomentJS - Umgang mit Zeit- und Datumsformaten

Die Anwendung beinhaltet einen komplexen Umgang mit Zeit- und Datumsformaten. Insbesondere hinsichtlich der Erfassung von Arbeitszeiten. Um die technische Handhabung möglichst flexibel und übersichtlich zu halten, wird in der Anwendung die JavaScript-Bibliothek *Moment.js*[6] in der Version 2.9.0 eingebunden. Dadurch werden Operationen wie das Validieren, Berechnen oder Manipulieren von Zeit- und Datumswerten vereinfacht. Die Dokumentation der Bibliothek[7] ist umfangreich und übersichtlich.

Ein Beispiel aus dem Projekt soll den Einsatz der Bibliothek verdeutlichen. In Listing 3.1 wird geprüft, ob der Wert aus *timeEntry.TimeDate* dem heutigen Datum entspricht. Dank einer kleinen Änderung an den Codezeilen in Listing 3.2 kann auch geprüft werden, ob sich das Datum für den gemachten Zeiteintrag in der aktuellen Woche liegt.

```
1 if (moment().isSame(timeEntry.TimeDate, 'day')) {
2   [...]
3 }
```

Listing 3.1: Mit Moment.js prüfen ob ein Datum heute ist

```
1 if (moment().isSame(timeEntry.TimeDate, 'week')) {
2   [...]
3 }
```

Listing 3.2: Mit Moment.js prüfen ob ein Datum in der aktuellen Woche liegt

3.3.3 Fehlende Chart-Bibliothek in OpenUI5

Für die Anwendung wird die quelloffenen Version der UI5-JavaScript-Frameworks eingesetzt. Dieses enthält keine eigene Bibliothek, um vorgefertigte Charts zu integrieren. Eine mögliche Alternative hierzu ist die Einbindung einer fremden Bibliothek. Im Kontext dieser Anwendung wird die Carts&Maps-Bibliothek von AmCharts[8] angebunden. Eine Vielzahl an unterschiedlichen Diagrammtypen und eine nützlich aufbereitete Webseite sind die Entscheidungskriterien für diesen Anbieter gewesen. Eingesetzt wird die kostenfreie Version 3.15.2.

Damit ein Diagramm zusammen mit dem SAP-Framework in der Anwendung genutzt werden kann, ist es ratsam ein eigenes User-Interface-Control zu programmieren, das selbst von einem

[6] http://momentjs.com/
[7] http://momentjs.com/docs/
[8] http://www.amcharts.com/

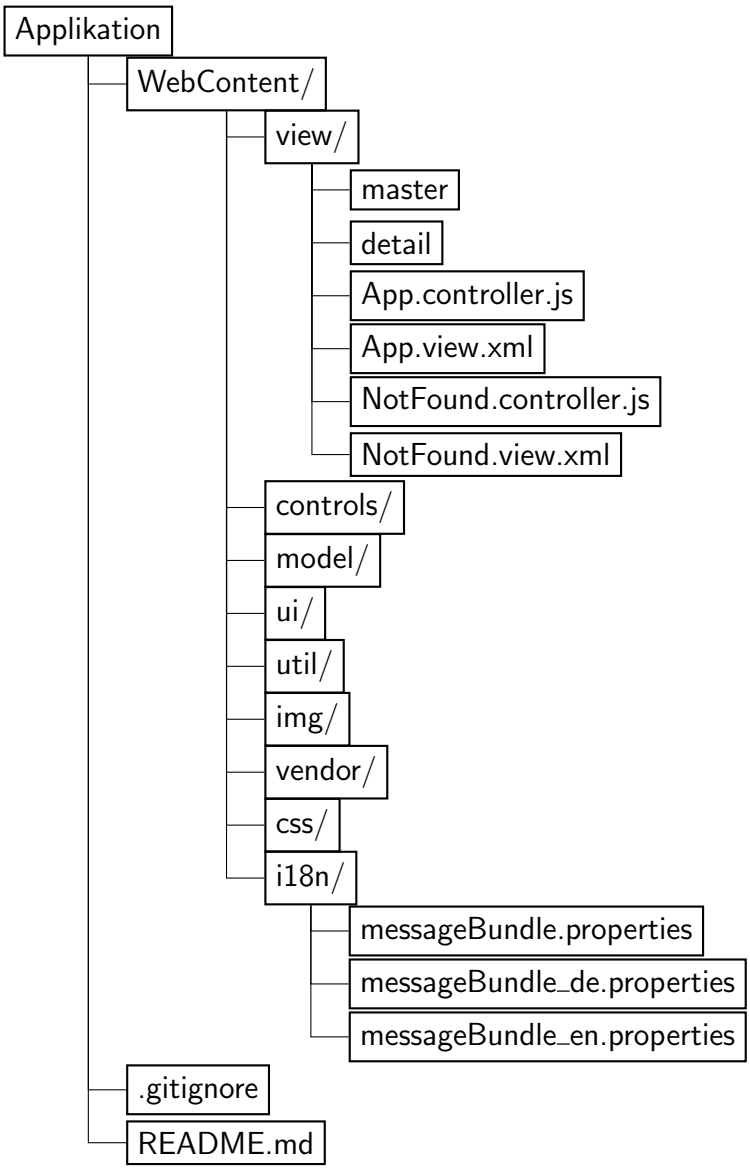

Abbildung 3.6: Ordnerstruktur

Standard-SAPUI5-Control abgeleitet wird. Das Control ist somit die Schnittstelle zwischen der Diagramm-Bibliothek von AmCharts und des JavaScript-Frameworks von SAP.

Die nachfolgenden Codebeispiele sollen den Umgang mit AmCharts etwas verdeutlichen.. Begonnen wird in Listing 3.4 damit, das neue Control von dem SAP-Standard-Control abzuleiten. Dadurch enthält es bereits implementierte Methoden und Attribute auf die das Framework bei der Verwendung des Controls zugreift:

```
/**
 * Custom Control for AmCharts
 *
 * @name AmCharts
 * @namespace cgerold.sapui5.timetracking
 * @extends sap.ui.controller
 */
sap.ui.core.Control.extend('cgerold.sapui5.timetracking.ui.
    AmCharts', {});
```

Listing 3.3: Ein eigenes Control erstellen

Innerhalb der neuen erstellten Klasse muss es eine Render-Methode geben. Diese wird vom Framework aufgerufen, wenn das Control eingesetzt wird. Daher ist es wichtig, die im Control enthaltene Logik dort zu beginnen.

```
    /**
     * Render function for control
     *
     * @param {sap.ui.core.RenderManager} rm
     * @param {cgerold.sapui5.timetracking.ui.AmChart}
         dataContainer
     */
    render: function (rm, dataContainer) {
      this.makeChart(dataContainer.getAmChartOptions() || {});
```

Listing 3.4: Ein eigenes Control erstellen

Obowhl der Chart am Ende des Projektes keine produktiven Daten konsumiert, sondern lediglich anhand von Testdaten erstellt wird, konnte die Lücke einer fehlenden Chart-Bibliothek in OpenUI5 durch ein eines eigens entwickeltes Control geschlossen werden.

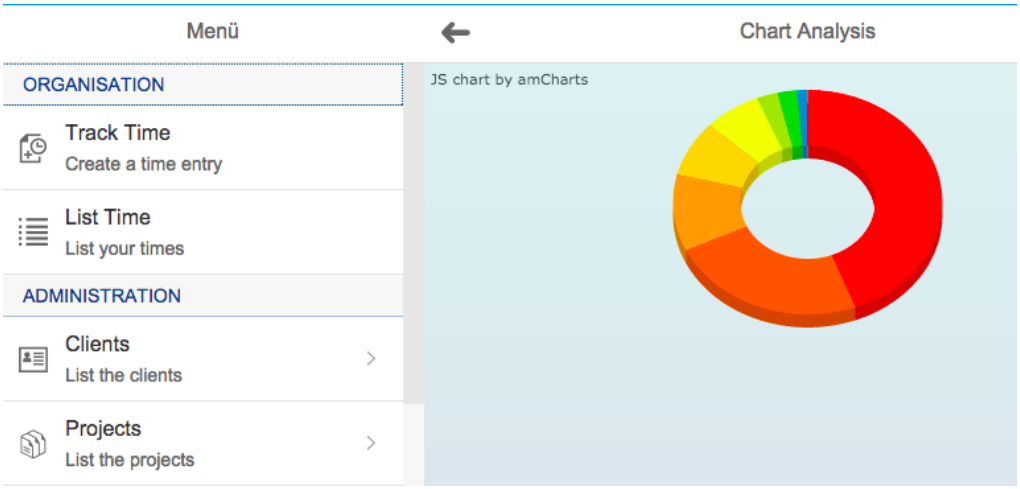

Abbildung 3.7: Anbindung AmCharts an die OpenUI5-Anwendung

3.3.4 NetWeaver Service

Für die Entwicklung der Applikation wurde das derzeit neuste Release des NetWeaver Gateways, Version 7.4, verwendet. Um Daten aus dem SAP-Backend an die Webanwendung zu übermitteln, muss ein Serivce erstellt und eingerichtet werden. Dieser Vorgang kann im SAP NetWeaver Gateway Service Builder, unter der Transaktion SEGW, durchgeführt werden. Die jeweils benötigten Datenbanktabellen müssen in den Service importiert und Methoden für mögliche Operationen auf den Daten ausimplementiert werden. Der, für die Applikation zur Zeiterfassung angelegte Service, ZUI_TIMETRACKING_WP ist in Abbildung 3.8 dargestellt.

Bei der Implementierung der Assoziationszuordnungen traten allerdings Probleme auf, welche bis zur Abgabe des Projekts, nicht gelöst werden konnten. Der Service erlaubt unter der Transaktion SEGW nur die Assoziation von zwei Entitäten in eine Richtung. Die andere Richtung kann nicht korrekt zugewiesen werden, da die Auswahl der Referenzspalten dabei nur auf die Schlüsselattribute beschränkt ist. Letztenendes führte diese Einschränkung dazu, dass bei Projektabgabe komplett auf die Anbindung über das SAP-Backend verzichtet und eine manuell erstellte Metadata für die Applikation verwendet wurde. Der Vollständigkeit halber, wird hier aber dennoch auf die Serviceerstellung und die Aufrufmöglichkeiten eingegangen.

Damit ein angelegter Service auch verwendet werden kann, muss der zugehörige Internet Communication Framework-Knoten (ICF-Knoten) unter der Transaktion /IWFND/MAINT_SERVICE aktiviert werden. Ein angelegter Service kann direkt über eine URL im Gateway Client getestet und die XML-Ausgabe eingesehen werden. Die URL setzt sich aus der Service-Root

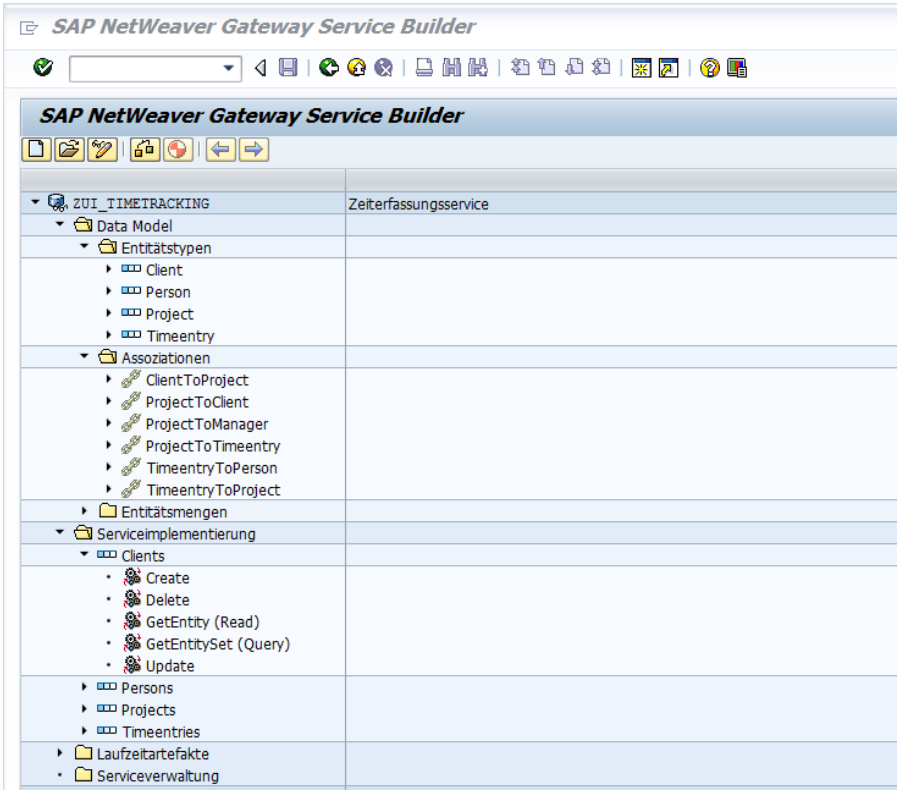

Abbildung 3.8: Transaktion SEGW

URI zusammen und optional aus dem Ressourcen Pfad und den Query-Optionen um die Anfrage zu spezifizieren. In Abbildung 3.9 wird ein Teilausschnitt der metadata.xml des Services im Gateway Client gezeigt, in der alle Informationen über einen Service zusammengefasst sind. Der URL-Aufruf aus dem Browser sieht für die Metadata des erstellen Service wie folgt aus: `http://r87z.hcc.uni-magdeburg.de:8087/sap/opu/odata/sap/ZUI_TIMETRACKING_SRV/$metadata`. Durch zusätzliche Parameter in der URL, können weitere Detailinformationen des Serivce ausgegeben werden. In der Tabelle 3.2 sind möglichen Parameterangaben, ihre Bedeutung und ein jeweiliges Beispiel aufgelistet [oda15].

Parameter	Beschreibung	Beispiel	Beschreibung Beispiel
$format	Ausgabeformat	$format=json	Zeigt die Ausgabe in JSON-Format an
$orderby	Einträge aus der Entität des Serivce werden sortiert aufgelistet	$orderby=name	Sortiert eine Entität nach der Namen-Property
$top	Selektiert bei einer Entität die ersten n Einträgen	$top=5	Zeigt die ersten 5 Einträge der Entität an
$skip	Eine Selektion aller Einträge ab Eintrag n+1	$skip=2	Die ersten zwei Einträge der Entität werden übersprungen und nich angezeigt
$filter	Filterkriterien	$filter=Address/City eq 'Friedberg'	Nur die Einträge anzeigen, deren Adresse mit den Namen Friedberg übereinstimmt

Tabelle 3.2: Parameter der Service-URL

Für Tests ist es auch möglich, alle Einträge der Entität im Browser ausgeben zu lassen. Dafür muss die beschriebene URL mit dem Namen der Entität als Parameter aufgerufen werden. Eine Auflistung aller Projekte wird mit der URL `http://r87z.hcc.uni-magdeburg.de:8087/sap/opu/odata/sap/ZUI_TIMETRACKING_SRV/Projects` erreicht. Ein einzelnes Projekt, hier das mit dem Schlüsselattribut ID=1, kann mit der folgenden URL angezeigt werden: `http://r87z.hcc.uni-magdeburg.de:8087/sap/opu/odata/sap/ZUI_TIMETRACKING_SRV/Projects(1)`. Auch Assoziationen lassen sich so leicht aufrufen. Die URL `http://r87z.hcc.uni-magdeburg.de:8087/sap/opu/odata/sap/ZUI_TIMETRACKING_SRV/Projects(1)/Client` gibt die ID des Clients aus, dem das erste Projekt zugeordnet ist.

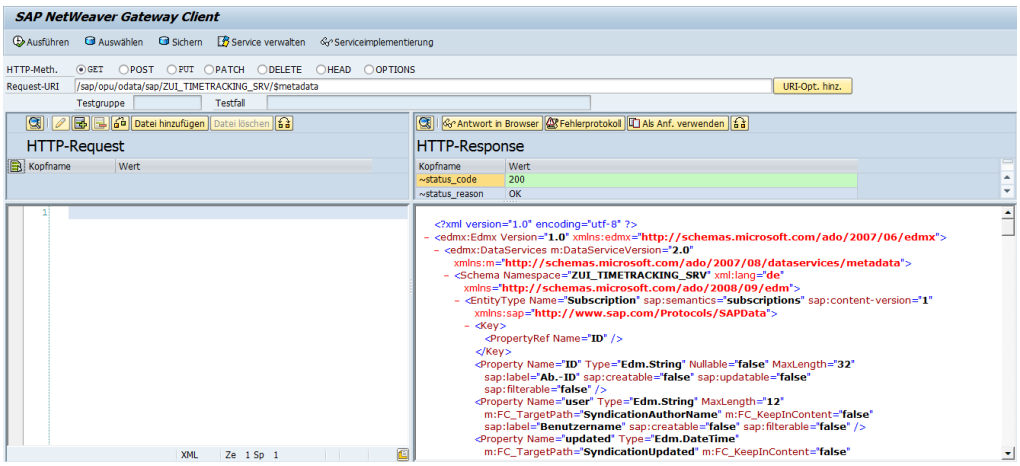

Abbildung 3.9: SAP NetWeaver Gateway Client metadata.xml Ausschnitt

3.3.5 Entwicklungsstand

Da die Entwicklungszeit auf drei Monate beschränkt gewesen ist, konnten bei der Entwicklung nicht alle vorher festgelgten Anforderungen umgesetzt werden. Dennoch sind im aktuellen Entwicklungsstand die wesentlichen Kernpunkte für eine Zeiterfassung implementiert. In Tabelle 3.3 sind die erläuterten Anforderungen zu Beginn dieses Kapitels zusammenfassend aufgelistet und der aktuelle Stand verdeutlicht.

Anforderung	Umgesetzt
Kunden, Projekte und Personen auflisten	✓
Kunden, Projekte und Personen filterbar	✓
Kunden, Projekte und Personen sortierbar	✓
Kunden, Projekte und Personen Detailansicht	✓
Kunden, Projekte und Personen hinzufügen	✓
Kunden, Projekte und Personen bearbeiten	✗
Kunden, Projekte und Personen löschen	✗
Projekte und Kunden archivieren	✗
Erfassung von Mitarbeiterstunden	✓
Login	✓
Zeiteintrag auf Projekt buchen	✓
Einsehen der gebuchten Zeiteinträge	✓
Zeiteinträge filterbar	✓
Zeiteinträge sortierbar	✗
Analysemöglichkeiten	✗
Routing	✓
Datenanbindung an das SAP-Backend	✗

Tabelle 3.3: Anforderungen

4 Abschluss

Im letzten Kapitel wird zu Beginn eine Reflexion der benötigten Arbeitsstunden beschrieben. Anschließend wird eine Orientierungshilfe für die ersten Schritte mit dem JavaScript-Framework aufgezeigt und ein abschließenden Fazit, sowie ein Ausblick über die mögliche zukünftige Entwicklung der Applikation gegeben.

4.1 Reflexion des Arbeitsaufwandes

Der Aufwand des Projekts wurde für zwei Personen auf 480 Stunden geschätzt, bei einer Projektdauer von drei Monaten und 20 Stunden Arbeitsleistung pro Woche und pro Person. Umgerechnet in Tagen sollten somit ungefähr 60 Tage Arbeit investiert werden. Da während der Projektdauer die bereits erwähnten Hürden auftraten, erhöhte sich der Aufwand und somit auch die investierte Arbeitszeit des Projekts leicht. In der Grafik 4.1 wird die geplante Projektarbeitszeit mit der benötigten, anhand der Meilensteine, gegenübergestellt.
Trotz der anfänglichen Einarbeitungsphase, musste während der Projektarbeit die Literatur und Internetquellen für Problemlösungen bei der Implementierung herangezogen werden. Damit erhöhte sich die anfangs geschätzte Stundenzahl besonders bei der Erstellung des ersten Prototypen. Auch durch die, bei der Weiterentwicklung aufgetretenen Hürden, wie das Rechnen mit Zeiten, die Charterstellung und Einbindung oder die Serviceerstellung, wurde mehr Zeit benötigt, als ursprünglich angedacht.
Letztendlich hat die Applikationserstellung viel Spaß gemacht, wodurch der Mehraufwand und die damit verbundene Aneignung von Wissen durch die Erweiterung der eigenen Wissensbasis entschädigt wurde.

4.2 First Steps mit SAPUI5 / OpenUI5

Bevor im abschließenden Fazit ein Ergebnis hinsichtlich der Umsetzung des Projektes und der Verwendbarkeit des JavaScript-Frameworks gegeben wird, soll der folgende Abschnitt eine kurze Orientierungshilfe für den Einstieg in das JavaScript-Framework geben. Vor allem

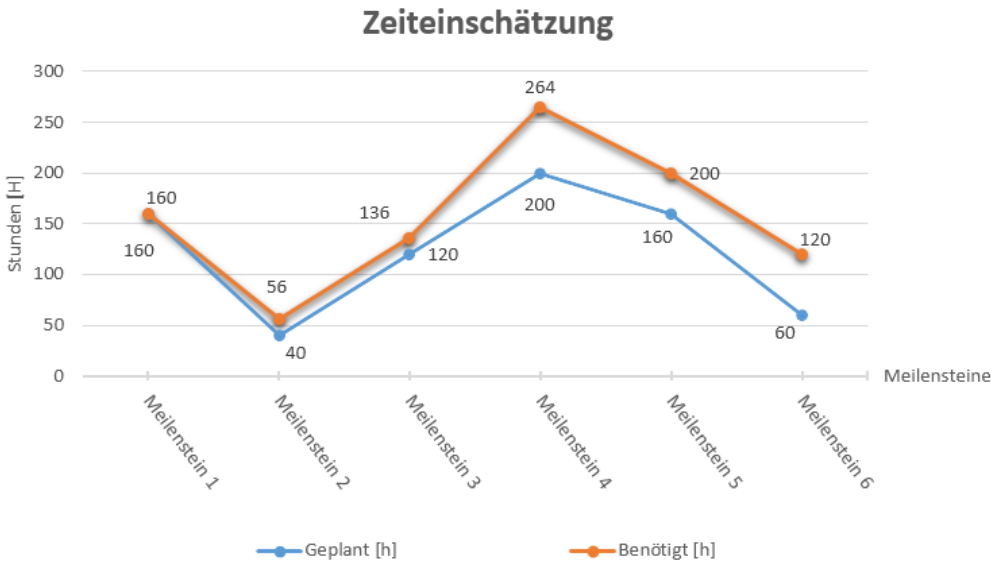

Abbildung 4.1: Zeiteinschätzung

für diejenigen Interessierten, die sich zum ersten Mal mit SAPUI5 beschäftigen und damit arbeiten möchten, sollen die nachfolgenden Worte eine Unterstützung geben.

4.2.1 Literatur

Für das Framework von SAP gibt es zur Zeit nur sehr wenig offizielle Literatur. Zu empfehlen ist hier das Buch „Einführung in SAPUI5 “ [1] von Miroslav Antolovic. Das Buch ist eine hilfreiche Stütze sowohl für Anfänger als auch für erfahrene Entwickler und bietet als Nachschlagewerk immer wieder hilfreiche Dienste.
Daneben ist die der offizielle Developer-Guide[2] für SAPUI5 beziehungsweise OpenUI5 zu erwähnen. Dort werden die Themen Architektur, Datenaustausch, Datenanbindung sowie Lokalisierung und viele weitere Konzepte behandelt.

4.2.2 Entwicklung

In der aktuellsten SAPUI5-Dokumentation, die sich zum Zeitpunkt dieser Ausarbeitung noch in einer Beta-Phase befindet, gibt es einen sehr ausführlicher Walkthrough[3]. Dieser führt den Leser von einer simplen „Hello World!“-Applikation über die Themen Model, View, Controller bis hin zu einer vollständigen Applikation mit Routing und Datenanbindung. Darüber hinaus

[1] https://www.sap-press.com/getting-started-with-sapui5_3565/
[2] https://sapui5.hana.ondemand.com/sdk/#content/Overview.html
[3] https://openui5beta.hana.ondemand.com/#docs/guide/3da5f4be63264db99f2e5b04c5e853db.html

wird auch verstärkt das Thema Tests im Kontext von SAPUI5 angesprochen und vorgestellt. Beim Entwickeln sollte stets die API Referenz[4] für das Framework griffbereit liegen. Zudem lohnt sich ein Blick auf SAPUI5-[5] beziehungsweise OpenUI5-[6]Explored. Dort sind in einer Art Showcase alle im Framework zur Verfügung stehenden UI-Elemente aufgeführt. Zu jedem aufgeführten UI-Element gibt es verschiedene Anwendungsszenarien und es lässt sich der dazugehörige Quellcode anzeigen.

4.2.3 Communities

Unterstützung bei der Entwicklung gibt es vor allem bei zwei Communities: zum einen wäre dort das SAPUI5 Developer Center[7] aus dem SAP Developer Network[8]. Fragen werden hier sehr zügig und ausführlich beantwortet. Zum anderen sind auch mittlerweile sehr viele SAPUI5-Entwickler auf Stackoverflow[9] unterwegs. Dort kann man unter den Tags „sapui5“[10] und „openui5“[11] bereits beantwortete Fragen einsehen und jederzeit neue stellen.

4.3 Fazit

Mit der entwickelten Applikation wird die Zeiterfassung der Mitarbeiter vereinfacht. Das übersichtliche Design und die komfortable Bedienung machen es den Benutzern einfach, sich direkt zurechtzufinden. Aufgrund der beschränkten Entwicklungszeit konnten nicht alle, anfangs gesetzten Anforderungen umgesetzt werden. Die Kernfunktionen der Applikation sind vorhanden und eine Erweiterung von Zusatzfeatures ist möglich. Im Allgemeinen ist die Entwicklung von SAPUI5 Applikationen sehr flexibel gehalten. Mit den von SAP bereitgestellten Controls werden bereits ansehnliche Benutzerelemente vorgegeben, die schnell und einfach umzusetzen sind. Die Anwendung kann, ohne großen Programmieraufwand, für unterschiedlichen Endgeräten in einer jeweils passenden Auflösung erzeugt werden. Zu Beginn war der Einstieg in das Thema SAPUI5 aufgrund der großen Flexibilität schwierig. Mit Hilfe von guter Literatur und zahlreichen Internetreferenzen, konnte diese erste Hürde dann aber überwunden werden.

[4] https://sapui5.hana.ondemand.com/sdk/#docs/api/symbols/sap.ui.html
[5] https://sapui5.hana.ondemand.com/explored.html
[6] https://openui5.hana.ondemand.com/explored.html
[7] http://scn.sap.com/community/developer-center/front-end
[8] http://scn.sap.com/
[9] http://stackoverflow.com/
[10] http://stackoverflow.com/questions/tagged/sapui5
[11] http://stackoverflow.com/questions/tagged/openui5

Und obwohl die SAP AG mit dem JavaScript-Framework im Bereich der Webentwicklung keine wirkliche Erneuerung bringt, so ist es dennoch von klarem Vorteil Erfahrungen und Kenntnisse mit dem JavaScript-Framework zu sammeln, da es insbesondere von bestehenden SAP-Kunden in Zukunft verstärkt eingesetzt werden könnte, um den Umgang mit den Anwendungen flexibler und einfacher zu machen.

4.4 Ausblick

Die Applikation umfasst bereits die wichtigsten Kernaspekte zur Zeiterfassung, kann aber, darauf aufbauend, noch erweitert werden. Da es uns nicht möglich war, alle Assoziationen im Service darzustellen, und damit auch die Anbindung der OData-Schnittstelle nicht das gewünschte Ergebnis liefert, könnte man dort Ansetzen um die Applikation mit dem Backend zu verbinden. Des Weiteren können die Analysemöglichkeiten nach belieben Erweitert werden, um den Benutzern einen besseren Überblick über die Zeiteinträge und Aktivitäten zu bieten. Eine weitere Möglichkeit, dem Anwender die Bedienung zu erleichtern, wäre das Einbinden einer Hilfefunktion, die einen schnellen Überblick über den möglichen Ablauf der Zeiterfassung gibt. Dadurch könnte der erste Umgang mit der Software erleichtert und Einstiegshürden möglichst gering gehalten werden. Eine weitere Funktion könnte ein Nachrichtenportal sein, in dem sich alle Anwender des Systems, gegenseitig Nachrichten zukommen lassen können, ähnlich eines Email-Postfachs. Dadurch wird die Kommunikation der Anwender untereinander gefördert und der Kommunikationsweg im Allgemeinen verkürzt. Zudem könnte es zukünftig eine Kalenderfunktion geben, bei der ein Mitarbeiter bereits auf dem Dashboard einen Kalender angezeigt bekommt, in dem die persönlichen Termine und die bereits erfassten Zeiten eingetragen sind. Aufgrund der hohen Flexibilität bei der Programmierung mit SAPUI5 kann die Applikation noch beliebig erweitert werden.

A Anhang

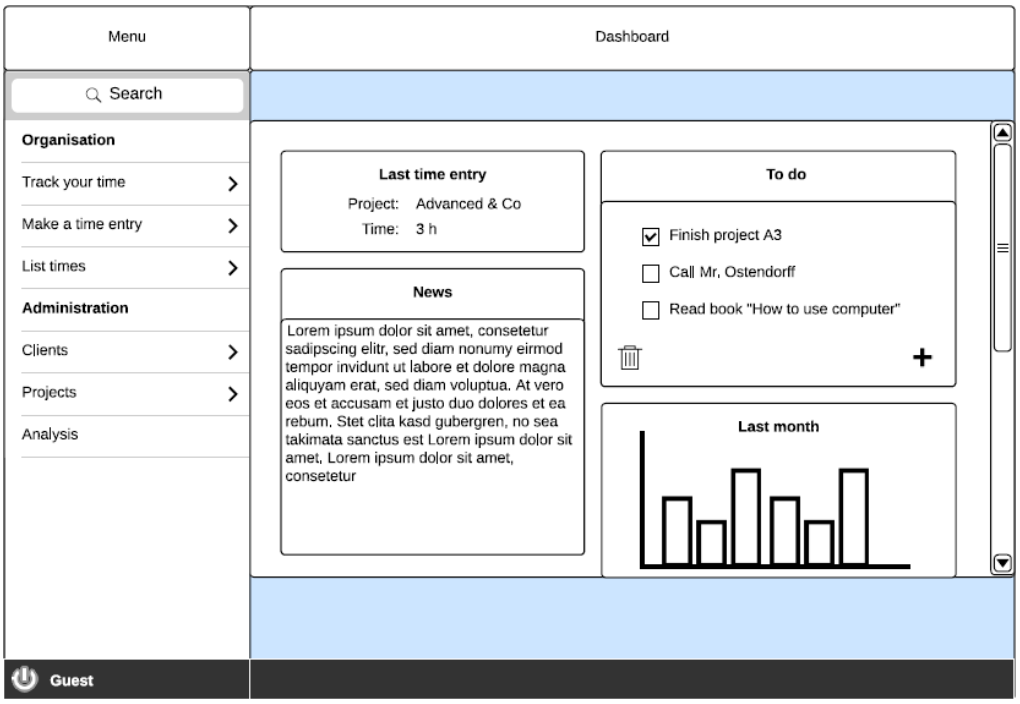

Abbildung A.1: Mockup Dashboard

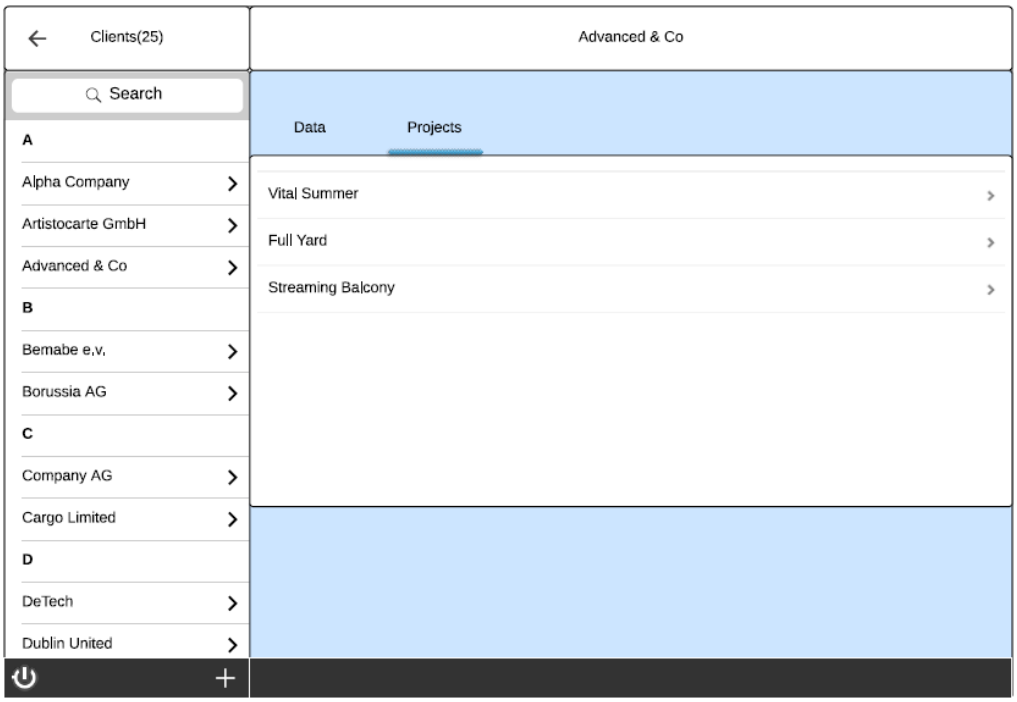

Abbildung A.2: Mockup Client Project

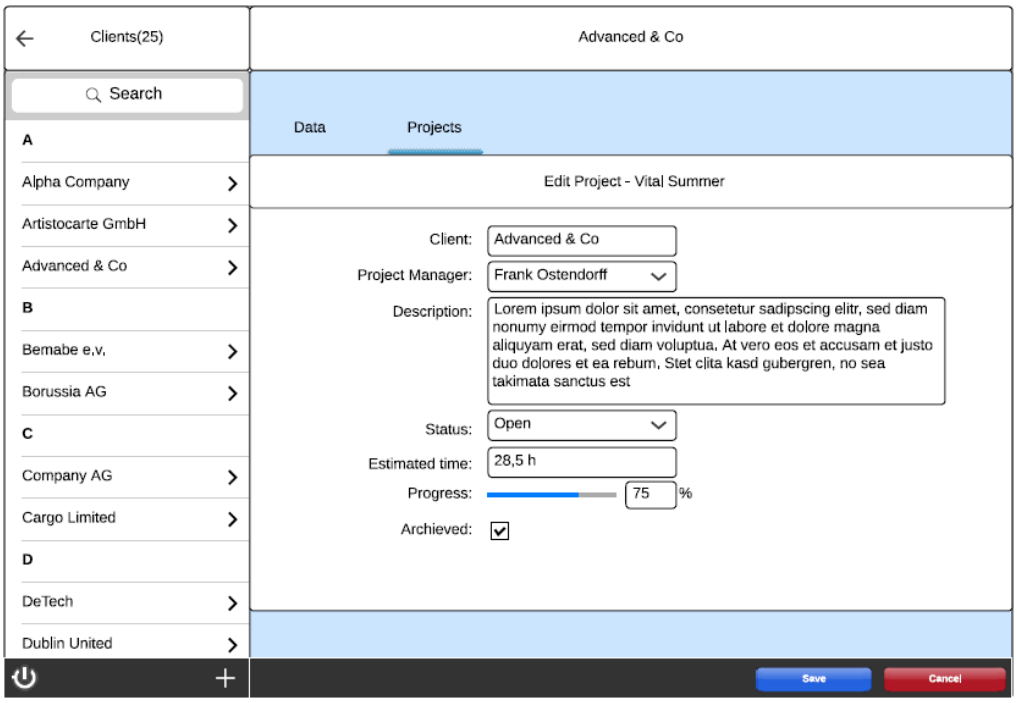

Abbildung A.3: Mockup Projektdetailansicht

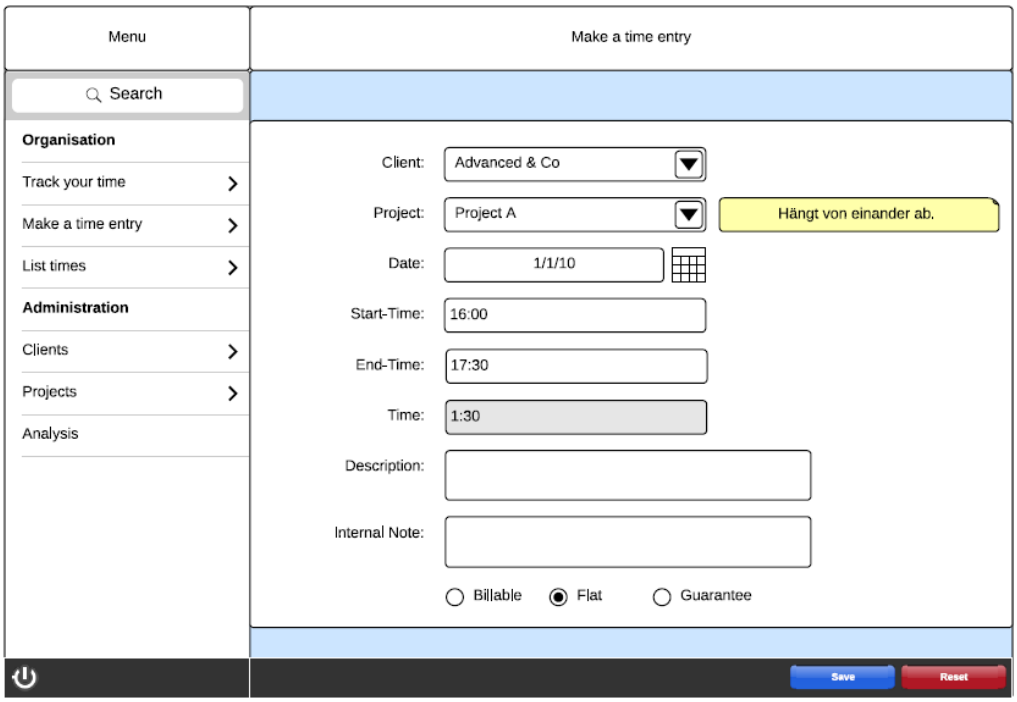

Abbildung A.4: Mockup Zeiteintrag

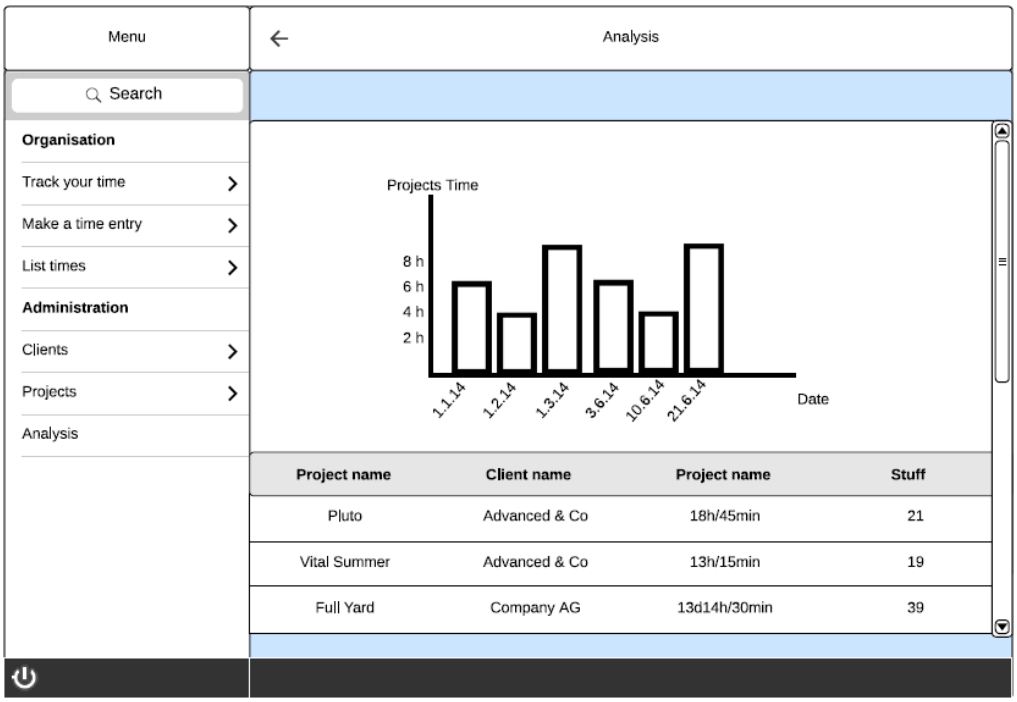

Abbildung A.5: Mockup Analyse

Literaturverzeichnis

[Ant14] Antolovic, Miroslav: *Einführung in SAPUI5*. Bonn : Galileo Press, 2014. – ISBN 3836227533 9783836227537

[Bö14] Bönnen, Carsten: *OData und SAP Gateway: [SAP-Systeme mit Nicht-SAP-Anwendungen integrieren; OData-Services zur User-Interface- und Anwendungsentwicklung generieren; Web-Anwendungen, mobile Apps und UI-Technologien mit Gateway anbinden]*. Bonn : Galileo Press, 2014. – ISBN 3836225387 9783836225380

[con] *SAPUI5*. `http://www.snapconsult.com/portfolio/sap-user-interfaces/sapui5.html`

[Kro] Kropff, Peter: *JavaScript - Einleitung*. `http://www.peterkropff.de/site/javascript/javascript.htm`

[Meh10] Mehl, Walter: *Zukunft des Internets - Mit HTML 5 in das schönere Web*. `http://www.macwelt.de/ratgeber/Zukunft-des-Internets-Mit-HTML-5-in-das-schoenere-Web-4956363.html`. Version: 2010

[oda15] *URI Conventions OData - Version 2.0*. `http://www.odata.org/documentation/odata-version-2-0/uri-conventions/`. Version: 2015

[Roh] Rohe, Klaus: *Java und das Open Data Protocol (OData) - JavaOData280812.pdf*. `http://www.sourcetalk.de/2012/files/2012/09/JavaOData280812.pdf`

[SAP14] *Model provider class and data provider class | SCN*. `http://scn.sap.com/thread/3554185`. Version: 2014

[Spy15] Spyvee: *Documentation/AdvancedTopics/DataBinding/UsageInApp – SAPUI5 Wiki (TIP CORE User Interface)*. `http://www.spyvee.com/SAPHTML5_DemoKit/docs/guide/UsageInApp.html`. Version: März 2015

[web] *Ajax Einführung - Übersicht & Einleitung - Javascript - Ajax - Tutorials, Tipps und Tricks für Webmaster auf Webmasterpro.de*. http://www.webmasterpro.de/coding/article/ajax-einfuehrung-uebersicht.html

[wik] *Cascading Style Sheets – Wikipedia*. http://de.wikipedia.org/wiki/Cascading_Style_Sheets

[Zag] Zaglov, Ilja: *Moment.js: Zeiten parsen und berechnen mit JavaScript*. http://t3n.de/news/momentjs-zeitberechnungen-mint-javascript-562592/