



Fachhochschule Köln
Cologne University of Applied Sciences

FACHHOCHSCHULE KÖLN FAKULTÄT FÜR INFORMATIK UND
INGENIEURWISSENSCHAFTEN

ENTWICKLUNGSPROJEKT INTERAKTIVE SYSTEME

Meilenstein 4

Campus Gummersbach
im Studiengang
Medieninformatik

Betreut von:

Prof. Dr. Kristian Fischer
Prof. Dr. Gerhard Hartmann
Robert Gabriel, B. Sc.

ausgearbeitet von:

DERYA ERGUEL
SINEM KAYA

6. Juni 2015

Inhaltsverzeichnis

1 Datenstrukturen	2
1.1 XML-Dateien	3
1.2 XML-Schemata	5
1.3 Entity-Relationship-Modellierung	7
2 WBA-Modellierung	8
2.1 Ressourcen	8
2.1.1 Profil	8
2.1.2 Karte	8
2.1.3 Nachricht	9
2.1.4 Wetter	9
2.1.5 Route	10
2.1.6 Ressourcentabelle	10
3 Prototyp UI	12
3.1 Gestaltungslösungen	12
3.1.1 Paperbased-Prototyping	12
3.2 Fazit zu der Gestaltungslösung	23
3.2.1 Funktionale Anforderung	25
3.2.2 Organisatorische Anforderung	25
3.2.3 Qualitative Anforderung	26
4 Projektplan	27
5 Literaturverzeichnis	29

1 Datenstrukturen

In diesem Kapitel wird die Strukturierung der Daten ausführlich besprochen, die notwendig sind, um die technische Umsetzung zu realisieren und das Repräsentieren der Daten gerecht realisieren zu können.

Ein Benutzer gibt verschiedene Daten von sich preis. Ausserdem enthalten die Karten und Routen Informationen und Daten, genauso wie die Wetterfunktion. Um welche Daten es sich hierbei handelt soll die folgende Tabelle darstellen:

Funktion	Daten
Benutzer	Benutzer ID Vorname, Nachname E-Mail Bild Passwort
Karte	Geodaten (Longitude, Latitude) Standort Anfangsposition Zielposition Umgebungsdaten Zeit Reitstil Bericht
Wetter	Ort Temperatur Niederschlag Windgeschwindigkeit
Nachrichten	NachrichtenID Textnachricht

Tabelle .1: Daten innerhalb der Anwendung

1.1 XML-Dateien

Da eine Android-Applikation entwickelt werden soll, ist die Repräsentation der Daten im Zusammenhang mit dem Repräsentationsformat XML die beste Möglichkeit für dieses Projekt, da das Format in der Entwicklung bereits involviert ist und in dem Modul 'Web-basierte Anwendungen 2' praktisch umgesetzt wurde. Aus diesem Grund soll die Strukturierung der Daten anhand von XML-Schemata dargestellt und erläutert werden. Hierfür werden zunächst die XML-Dateien präsentiert werden. Im weiteren Verlauf werden zu den XML-Dateien zugehörigen XML-Schemata dargestellt und erläutert werden.

Die Sprache XML wurde entwickelt, um Daten zu beschreiben. Bevor die erstellten XML-Dateien dargestellt und erläutert werden, soll kurz auf die Wetterfunktion eingegangen werden. Die XML-Datei, die für die Wetterrepräsentation benötigt und entnommen wird kann unter der Referenz (2) aufgerufen werden. Bei der vorgegebenen URL handelt es sich dabei um die Wetterdaten für die Stadt Frechen. Da die XML-Datei bereits vorgegeben ist und abgerufen werden kann, wird nicht mehr weiter darauf eingegangen.

Die XML-Datei der Karte ist am umfangreichsten, weshalb diese beschrieben und erläutert werden soll. Die XML-Datei zur Route und zu Profil wird nicht weiter erläutert, da es einen ähnlichen einfacheren Aufbau enthält.

```
<?xml version="1.0" encoding="utf-8"?>
<Karte>
    <Geodaten>
        <Standortposition Längengrad="50.916672" Breitengrad="6.81667"/>
    </Geodaten>
    <Umgebungsinformationen>
        <Höfe>
            <Name>Reithof Am Sandberg</Name>
            <Position Längengrad="50.916672" Breitengrad="6.81667"/>
            <Beschreibung>Der Reithof Am Sandberg bietet für Anfänger einen idealen Einstieg an....</Beschreibung>
        </Höfe>
        <Höfe>
            <Name>Reithof Idylle</Name>
            <Position Längengrad="52.979672" Breitengrad="7.86767"/>
            <Beschreibung>.. Von 9 bis 19 Uhr geöffnet..</Beschreibung>
        </Höfe>
        <Vegetation>
            <Pflanzen>
                <Name>Fliegenpilz</Name>
                <Position Längengrad="50.364757" Breitengrad="6.264783"/>
                <Bericht>Achtung gerade ein Fliegenpilz entdeckt...</Bericht>
            </Pflanzen>
            <Pflanzen>
                <Name>Alpenveilchen</Name>
                <Position Längengrad="50.323457" Breitengrad="6.2454483"/>
                <Bericht>Alpenveilchen unterwegs gefunden. Richtig giftig passt auf..</Bericht>
            </Pflanzen>
            <Barrieren>
                <Beschreibung>Achtung Straßenschäden nach Sturm letzte Nacht...</Beschreibung>
                <Position Längengrad="51.934572" Breitengrad="7.36467"/>
            </Barrieren>
        </Vegetation>
    </Umgebungsinformationen>
</Karte>
```

Abbildung .1: XML-Datei 'Karte'

Die Karte enthält eine Menge von Daten, u.A. die Umgebungsinformationen. Aus der Abbildung 1 kann entnommen werden, dass das root-Element 'Karte'

ist. Die Karte hat zwei Kindelemente und zwar 'Geodaten' für die aktuelle Standortposition und die 'Umgebungsinformationen'. Das Element 'Geodaten' hat ein weiteres Kindelement 'Standortposition' mit den Attributen 'Längengrad' und 'Breitengrad'. Da der aktuelle Standort beim Öffnen der Karte automatisch angezeigt wird, sind diese Angaben Pflicht. Umgebungsinformationen beinhalten die Reiterhöfe und die Vegetationen. Deshalb sind auch diese zwei Kindelemente zu dem Element 'Umgebungsinformationen' vorhanden. Zu den Höfen werden 'Name', 'Position' und eine Beschreibung erfasst. Die Position wird anhand des Breiten- und Längengrades bestimmt, weshalb das Element 'Position' die Attribute 'Längengrad' und 'Breitengrad' enthält. Die Vegetation unterteilt sich wiederum in 'Pflanzen' und 'Barrieren'. Zu den Pflanzen gehören folgende Angaben: Name, Position, Beschreibung. Das Element 'Barriere' kennzeichnet sich auch durch die Position und einer Beschreibung.

```
<?xml version="1.0" encoding="UTF-8"?>
<Routenplanung>
    <route>
        <Geodaten>
            <Startposition Längengrad="50.916672" Breitengrad="6.81667"/>
            <Zielposition Längengrad="50.916672" Breitengrad="6.81667"/>
        </Geodaten>
        <Streckenlänge Einheit="km" Länge="3.4"/>
        <Dauer Einheit="h" Stunden="2"/>
    </route>
</Routenplanung>
```

Abbildung .2: XML-Datei 'Route'

```
<?xml version="1.0" encoding="utf-8"?>
<Profil>
    <Benutzer>
        <Vorname>Josef</Vorname>
        <Nachname>Groß</Nachname>
        <EMail>JosefGroß@josefGroß.de</EMail>
        <Status>Profi-Reiter</Status>
    </Benutzer>
    <Benutzer>
        <Vorname>Marie</Vorname>
        <Nachname>Brings</Nachname>
        <EMail>MarieBrings@MarieBrings.de</EMail>
        <Status>Amateur-Reiter</Status>
    </Benutzer>
</Profil>
```

Abbildung .3: XML-Datei 'Profil'

1.2 XML-Schemata

Anhand der XML-Schemata ist die Strukturierung der Daten nochmals deutlich zu erkennen. Hier kann man die einzelnen Elemente und die zugehörigen Attribute entnehmen. So sind die Attribute Längengrad und Breitengrad beispielsweise vom Typ float. Wichtig ist auch, dass z.B die Anzahl der Höfe durch **maxOccurs** und **minOccurs** festgelegt ist. Das Element 'Höfe' kann garnicht oder mehrmals auftreten.

```

<xss:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xss="http://www.w3.org/2001/XMLSchema">
  <xss:element name="Karte">
    <xss:complexType>
      <xss:sequence>
        <xss:element name="Geodaten">
          <xss:complexType>
            <xss:sequence>
              <xss:element name="Standortposition">
                <xss:complexType>
                  <xss:simpleContent>
                    <xss:extension base="xs:string">
                      <xss:attribute type="xs:float" name="Längengrad"/>
                      <xss:attribute type="xs:float" name="Breitengrad"/>
                    </xss:extension>
                  </xss:simpleContent>
                </xss:complexType>
              </xss:element>
            </xss:sequence>
          </xss:complexType>
        </xss:element>
      </xss:sequence>
    </xss:complexType>
  </xss:element>
  <xss:element name="Umgebungsinformationen">
    <xss:complexType>
      <xss:sequence>
        <xss:element name="Höfe" maxOccurs="unbounded" minOccurs="0">
          <xss:complexType>
            <xss:sequence>
              <xss:element type="xs:string" name="Name"/>
              <xss:element name="Position">
                <xss:complexType>
                  <xss:simpleContent>
                    <xss:extension base="xs:string">
                      <xss:attribute type="xs:float" name="Längengrad" use="optional"/>
                      <xss:attribute type="xs:float" name="Breitengrad" use="optional"/>
                    </xss:extension>
                  </xss:simpleContent>
                </xss:complexType>
              </xss:element>
              <xss:element type="xs:string" name="Beschreibung"/>
            </xss:sequence>
          </xss:complexType>
        </xss:element>
      </xss:sequence>
    </xss:complexType>
  </xss:element>
</xss:schema>

```

```
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="Vegetation">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Pflanzen" maxOccurs="unbounded" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element type="xs:string" name="Name"/>
            <xs:element name="Position">
              <xs:complexType>
                <xs:simpleContent>
                  <xs:extension base="xs:string">
                    <xs:attribute type="xs:float" name="Längengrad"/>
                    <xs:attribute type="xs:float" name="Breitengrad"/>
                  </xs:extension>
                </xs:simpleContent>
              </xs:complexType>
            </xs:element>
            <xs:element type="xs:string" name="Bericht"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="Barrieren">
        <xs:complexType>
          <xs:sequence>
            <xs:element type="xs:string" name="Beschreibung"/>
            <xs:element name="Position">
              <xs:complexType>
                <xs:simpleContent>
                  <xs:extension base="xs:string">
                    <xs:attribute type="xs:float" name="Längengrad"/>
                    <xs:attribute type="xs:float" name="Breitengrad"/>
                  </xs:extension>
                </xs:simpleContent>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

```

        </xs:complexType>
    </xs:sequence>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:sequence>
</xs:element>
</xs:schema>

```

Abbildung .4: XML-Schema 'Karte'

```

<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="Routenplanung">
<xs:complexType>
<xs:sequence>
<xs:element name="route">
<xs:complexType>
<xs:sequence>
<xs:element name="Geodaten">
<xs:complexType>
<xs:sequence>
<xs:element name="Startposition">
<xs:complexType>
<xs:simpleContent>
<xs:extension base="xs:string">
<xs:attribute type="xs:float" name="Längengrad"/>
<xs:attribute type="xs:float" name="Breitengrad"/>
</xs:extension>
</xs:simpleContent>
</xs:complexType>
<xs:element name="Zielposition">
<xs:complexType>
<xs:simpleContent>
<xs:extension base="xs:string">
<xs:attribute type="xs:float" name="Längengrad"/>
<xs:attribute type="xs:float" name="Breitengrad"/>
</xs:extension>
</xs:simpleContent>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
<xs:element name="Streckenlänge">
<xs:complexType>
<xs:simpleContent>
<xs:extension base="xs:string">
<xs:attribute type="xs:string" name="Einheit"/>
<xs:attribute type="xs:float" name="Länge"/>
</xs:extension>
</xs:simpleContent>
</xs:complexType>
<xs:element name="Dauer">
<xs:complexType>
<xs:simpleContent>
<xs:extension base="xs:string">
<xs:attribute type="xs:string" name="Einheit"/>
<xs:attribute type="xs:byte" name="Stunden"/>
</xs:extension>
</xs:simpleContent>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>

```

Abbildung .5: XML-Schema 'Route'

```

<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="Profil">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Benutzer" maxOccurs="unbounded" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element type="xs:string" name="Vorname"/>
              <xs:element type="xs:string" name="Nachname"/>
              <xs:element type="xs:string" name="EMail"/>
              <xs:element type="xs:string" name="Status"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>

```

Abbildung .6: XML-Schema 'Profil'

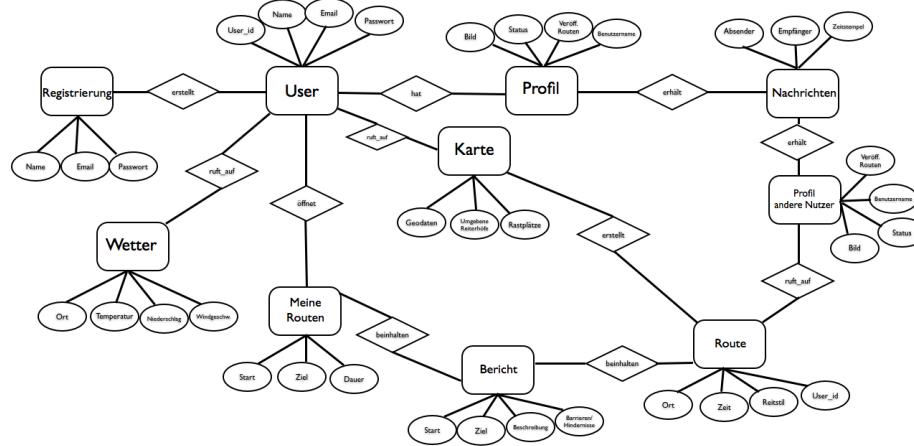


Abbildung .7: ER-Modellierung zur Datenstruktur

1.3 Entity-Relationship-Modellierung

Die gesamte Datenmodellierung der Applikation soll durch ein Entity-Relationship-Diagramm verdeutlicht werden. Die Entität 'User' hat beispielsweise die Attribute 'User id', 'Name', 'E-Mail', 'Passwort' und kann die Entität 'Karte' aufrufen, die die Attribute 'Geodaten', 'Umgebene Reiterhöfe' und 'Rastplätze' enthält. Durch die Modellierung soll dargestellt werden, welche Entitäten vorhanden sind und wie diese in Zusammenhang stehen.

2 WBA-Modellierung

2.1 Ressourcen

Die festgelegten Ressourcen

2.1.1 Profil

Die Profil Ressource enthält alle Profil angaben eines Users. Dabei kann jeder Nutzer das eigene Profil erstellen, ändern, erweitern und löschen. Das Profil beinhaltet Texte, ein Profilbild und veröffentlichte Routen. Über das Profil kann jeder User an andere User Nachrichten schreiben und erhalten. Jedes mal wenn der Nutzer eine Route erstellt erhält dieser anliegende Routen von anderen Nutzern. Somit kann er über die Verlinkung auf der Karte die Profile aufrufen und in Kontakt treten um weiter Informationen zu erhalten oder Erfahrungen auszutauschen.

Methoden: GET,PUT,POST,DELETE

Attribute:

- Name
- Status
- Profilbild
- Veröffentlichte Routen

Einsatz: Erstellung eines Profil,ändern von Angaben, löschen des Accounts

URI: Profil/{ID}

2.1.2 Karte

Die Ressource Karte erlaubt das anzeigen und aufrufen von Standorten auf einer Karte. Der Nutzer hat dabei die Möglichkeit über diese Anzeige Routen zu erstellen. Auf der Karte erkennt der Nutzer die Umgebung und kann sich anhand dieser orientieren. die auf der Karte angezeigten Icons sind beinhalteten Informationen und können übers Display an getippten werden. Sie erhalten damit Texte oder die Nutzer können sich an den Punkt navigier lassen.

Methoden: GET,POST

Attribute:

- Geodaten
- Umgebungsinformationen

- Straßennamen
- Wege
- Rastplätze

Einsatz: Anzeige der aktuellen Position auf einer Karte,navigieren zu einem bestimmten Punkt, abrufen von Informationen zur Umgebung, Umgebung auf einer Karte anzeigen lassen

URI:Karte/{ID}

2.1.3 Nachricht

Die Nachricht Ressource repräsentiert die einzelne Mitteilung die ein Nutzer von einem Nutzer als Text bekommt. Möchte eine Nutzer einem anderen Nutzer etwas mitteilen, kann er die übers Profil des jeweiligen. Eine Nachricht kann abgerufen, erstellt und gelöscht werden.

Methoden: GET,POST,DELETE

Attribute:

- Sender
- Absender
- Inhalt
- Uhrzeit

Einsatz: Nachricht lesen,schreiben,löschen

URI:Nachricht/{ID}

2.1.4 Wetter

Die Ressource Wetter beinhaltet den Dienst, der aktuelle Wetterdaten angibt. Dabei werden diese über Query-Parameter gefiltert.Der User hat die Möglichkeit zu jedem Ort sich die Wetterdaten anzeigen zulassen. Weiterhin kann der Nutzer Vorhersage in Stunden anzeigen lassen.

Methoden: GET

Einsatz: Abrufen von Wetterdaten, Filterung nach Temperatur,Niederschlag und Windgeschwindigkeit,Land,Bewölkung

Query-Parameter: Temperatur=[Celsius], Niederschlag=[prozent],Windgeschwindigkeit=[km/h],

Land=[Land],Bewölkung=[xy]

URI:Wetter/{ID}

2.1.5 Route

Die Ressource Route repräsentiert die Planung der Strecke die ein Reiter mit seinem Pferd vornehmen möchte. Die Ressource beinhaltet Startposition,Zielposition, Streckenlänge und Dauer. Der User kann eine Route erstellen, ändern und löschen.

Methoden: GET,POST,DELETE

Attribute:

- Zielposition
- Startposition
- Streckenlänge
- Dauer

Einsatz: Route erstellen, ändern und löschen

URI:Route/{ID}

2.1.6 Ressourcentabelle

In der folgenden Tabelle werden die ermittelten Ressourcen mit der angewendeten Methode und festgelegten URI dargestellt.

Ressource	Methode	Semantik
Karte/{ID}	Get	"User ruft die Karte auf "
Route/{ID}	Post	"Der User legt eine Route an "
Route/{ID}	Get	"Der User ruft eine bestehende Route auf"
Route/{ID}	Delete	"Eine bestehende Route wird löschen"
Profil/{ID}	Get	"Der User ruft ein bestehendes Profil aufrufen"
Profil/{ID}	Post	"Es wird ein neues Profil angelegt"
Profil/{ID}	Put	"Der User ändert sein Profil"
Profil/{ID}	Delete	"Der User löscht sein Profil "
Wetter/{ID}	Get	"Der User ruft Wetterdaten auf"
Nachricht/{ID}	Get	"Der User ruft eine vorhandene Nachricht auf"
Nachricht/{ID}	Post	"Der User legt eine Nachricht an"
Nachricht/{ID}	Delete	"Der User löscht eine Nachricht"
liste/	Get	"Der User ruft alle seine vorhandenen Nachrichten auf"
liste/{ID}	Get	"Der User ruft alle seine gespeicherten Routen in einer liste auf"

Tabelle .2: Ressourcen Übersicht

3 Prototyp UI

3.1 Gestaltungslösungen

In Meilenstein 2 wurde festgelegt, dass das Paperbased-Prototyping-Verfahren zum Ermitteln der Gestaltungslösung verwendet werden soll. Die Gründe beruhen auf der Vielfältigkeit des Verfahrens. Es erfordert keinerlei Kosten und ist für allgemeine Änderungen und für Änderungen des Interfaces sehr geeignet, da es sich nur um Zeichnungen auf Papier handelt. Dadurch sind mögliche Iterationen schneller und einfacher umsetzbar.

3.1.1 Paperbased-Prototyping

Der Prototyp baut zum größten Teil auf den Resultaten der Anforderungsanalyse. Es wurde versucht die funktionalen, organisatorischen und qualitativen Funktionen bestmöglich umzusetzen. Im Folgenden werden die erstellten paperbasierten Prototypen in Reihenfolge dargestellt und erläutert. Zudem sollte durch eine kurze und knappe Beschreibung klar werden, welche Abbildung welche Anforderungen erfüllt. Um die Begründung nachvollziehen zu können, sollten die Anforderungen aus Meilenstein 3 vorliegen.



Abbildung .8: Paperbased Prototyp - Login

Ist der Benutzer bereits registriert, kann er durch die Eingabe von Benutzernamen und Passwort in das System gelangen und sich somit einloggen. Ist der Benutzer nicht registriert, so muss er zunächst eine Registrierung vollziehen. Das Ziel hierbei ist es eine schnelle Registrierung bzw. Login zu gewährleisten. In kurzen Schritten kann der Benutzer erfolgreich mit der Nutzung des Systems beginnen.



Abbildung .9: Paperbased Prototyp - Registrierung

Die Registrierung erfolgt durch die Eingabe von Benutzername, Passwort und E-Mail. Über den Benutzernamen kann der User von anderen User angesprochen werden.

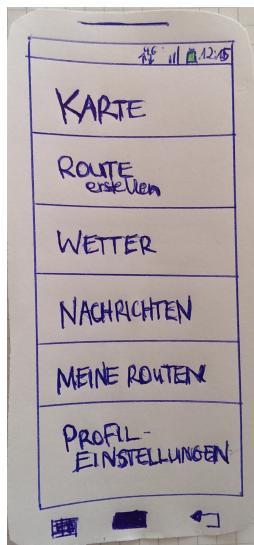


Abbildung .10: Paperbased Prototyp - Hauptmenü

Das Hauptmenü besteht aus 6 Buttons: Karte, Route erstellen, Wetter, Nachrichten, Meine Routen, Profil-Einstellungen. Diese Button stehen für die ein-

zelne Funktionen. Über diese soll der Benutzer zu seinem Nutzungsziel kommen. Q30 erfordert eine einfache und ersichtliche Benutzeroberfläche. Aus dem Grund wurde das User-Interface recht simpel gehalten. Q50 wiederum erfordert die einfache Darstellung der Informationen. Die Buttons sollten deswegen recht einfache und aussagekräftige Titel haben. Eine schnelle und effektive Bedienung (Q70) soll durch die einfache Struktur und der Größe der Buttons gewährleistet werden.

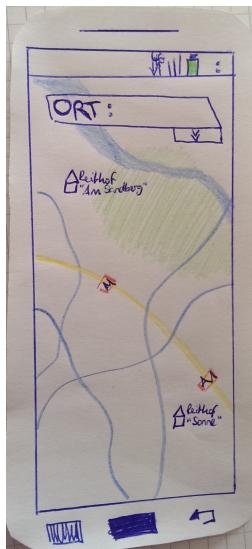


Abbildung .11: Paperbased Prototyp - Karte

Sobald der Benutzer den Button 'Karte' tätigt, öffnet sich die Karte mit den Umgebungsinformationen und dem aktuellen Standort. Zusätzlich kann er andere Orte erforschen, indem er den erwünschten Ort in das Eingabefeld eingibt. Die funktionale Anforderung F10 erfordert die Darstellung der Umgebungsinformationen auf der Karte. Dies wird in diesem Prototyp erfolgreich dargestellt. Das Erfassen des Standorts wird ermöglicht, welches die Anforderung O20 erfordert.

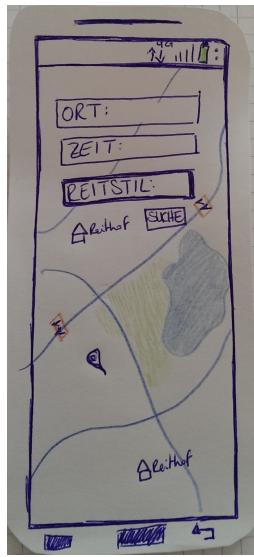


Abbildung .12: Paperbased Prototyp - Route erstellen 1

Tätigt der Benutzer den Button 'Route erstellen', gelangt er ebenfalls auf die Karte. Dort kann er die erwünschte Reitzeit angeben und den Reitstil. Muss der Benutzer eine Route erstellen werden ihm dabei mehrere Kriterien angeboten um die Route dem Bedürfnissen anzupassen. Durch diesen Prototyp wird die funktionale Anforderung F10 (Route erstellen) umgesetzt. Außerdem können werden Umgebungsinformationen dargestellt (F20).

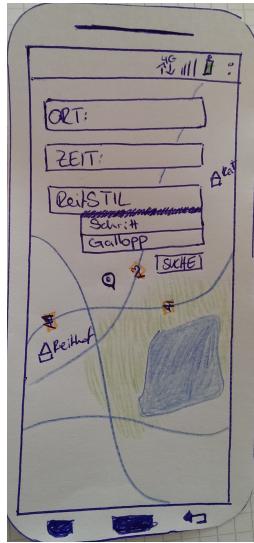


Abbildung .13: Paperbased Prototyp - Route erstellen 2

Durch Tätigkeiten des Reitstils öffnen sich Optionen, die sich in 'Sprint' und 'Gallopp' unterscheiden. Jeder Reitstil enthält konstante Mittel-Geschwindigkeiten (3), die als Kriterien zur Suche der Routen genutzt werden. Durch den Button 'Suche' wird nach der Besten Alternativ-Route gesucht. Der Benutzer kann eine Strecke öfter geritten sein dabei sollte ihm eine Alternative zur Verfügung stehen um neue Strecken zu entdecken.

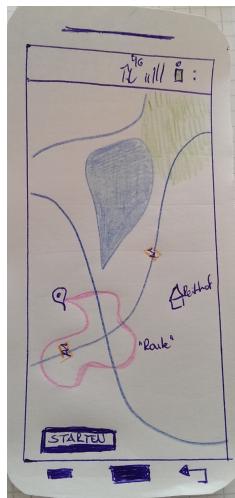


Abbildung .14: Paperbased Prototyp - Route erstellen 3

Die Route wird gefunden und angezeigt. Durch den Button 'Starten' kann die Route gestartet werden. Hier sollten dem Benutzer auch Alternativ-Routen angeboten werden, welches ein Erfordernis der funktionalen Anforderung F80 ist.

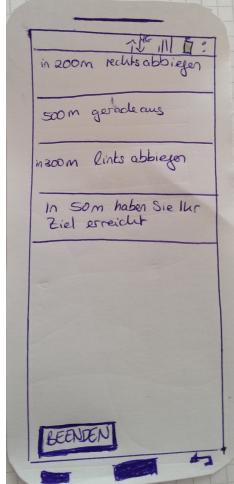


Abbildung .15: Paperbased Prototyp - Route erstellen 4

Die Route wird schriftlich navigiert und gleichzeitig auditiv wiedergegeben. Durch den Button 'Beenden' wird die Route beendet. Während des Reitens hat der Reiter nicht die Möglichkeit sein Smartphone in der Hand zu halten und ständig aufs Display zu schauen. Aus diesem Grund müssen auditive Navigationen zu Verfügung stehen, um dem Reiter ständig zu begleiten und zu warnen. Dadurch wäre die funktionale Anforderung F60 ermöglicht. Durch die Navigation wird dem Benutzer keinerlei Vorkenntnisse zur Ortschaft vorausgesetzt, welches die Anforderung O70 erfordert. Q40 beinhaltet eine einfache und deutliche Sprachführung. Sowohl das textuelle, als auch das auditive Navigieren sollte dementsprechend angepasst werden.

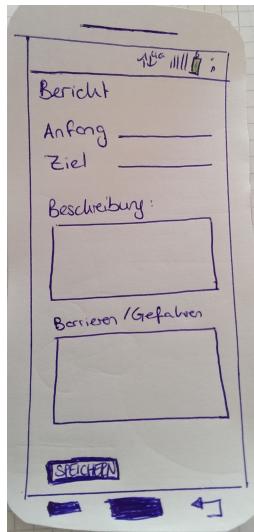


Abbildung .16: Paperbased Prototyp - Route erstellen 5

Nach Beenden der Route erscheint ein Fenster zum Erfassen eines Berichts. Dies ist für die Benutzer optional. Das bedeutet, dass der Benutzer nicht verpflichtet ist eine Eingabe durchzuführen. Handelt es sich um keine neue Route, sondern um eine, die ein anderer Benutzer erstellt und geteilt hat, wird der Benutzer, der die Route erstellt hat über neue Berichte benachrichtigt. Dies umfasst die Realisierung und Umsetzung der Anforderung F100.



Abbildung .17: Paperbased Prototyp - Wetter 1

Tägt der Benutzer im Hauptmenü den Button 'Wetter', so erscheint dieses Fenster. Durch Eingabe des Ortes und anschließend durch das Tätigen des 'Wetter'-Buttons werden die Werte für den erwünschten Ort angezeigt. Die Wetter-Anzeige (F40) muss für den Benutzer unter geeigneten Kriterien angezeigt werden um vorausschauende Planung zu erstellen. Dadurch gelingt es dem Benutzer die Routenplanung unabhängig vom Wetter zu ermöglichen, welches die Anforderung O50 erfordert.

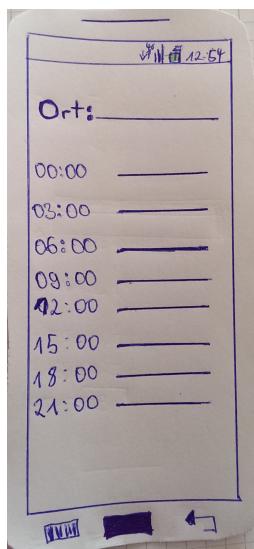


Abbildung .18: Paperbased Prototyp - Wetter 2

Hier wird die Temperatur und der Niederschlag im 3-Stunden Takt angezeigt.

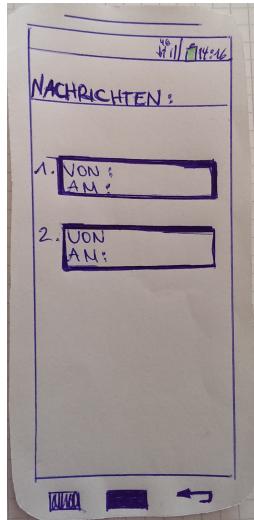


Abbildung .19: Paperbased Prototyp - Nachrichten 1

Tätigt der Benutzer den 'Nachrichten'-Button, so erscheint das Nachrichten-Fenster. Hier kann der Benutzer auf seine Nachrichten zugreifen und die neuesten Nachrichten lesen. Die Benutzer können sich hier untereinander über die Domäne und Routen spezifische Themen austauschen (F30) .

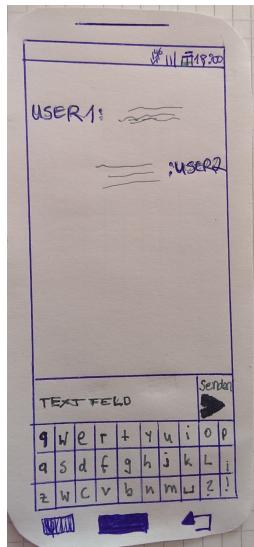


Abbildung .20: Paperbased Prototyp - Nachrichten 2

Abbildung 16 stellt die Kommunikation durch Nachrichtenaustausch zwischen zwei unterschiedlichen Benutzern dar. Dadurch wird die Anforderung F30 die Kommunikation zu anderen Nutzern erfüllt.



Abbildung .21: Paperbased Prototyp - Meine Routen

Durch den 'Meine Routen'-Button im Hauptmenü, werden dem Benutzer die bereits erstellten Routen angezeigt. Der Benutzer kann seine Routen abspeichern und sie für eigene Performance-Zwecke oder gar zur Erinnerung festhalten. Hier kann der Benutzer seine eigenen Routen auch abändern, indem er die Route öffnet, so wie es die Anforderung F50 erfordert.

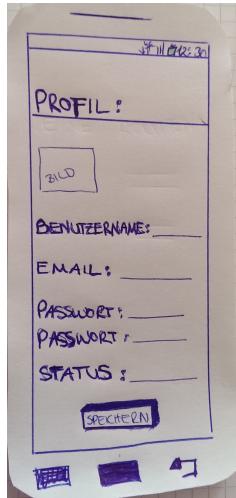


Abbildung .22: Paperbased Prototyp - Profileinstellungen Benutzer Ansicht

Durch das Tätigen des 'Profil-Einstellungen'- Buttons gelangt der Benutzer in die Einstellungen und kann dort sein Profil-Foto, Benutzername, Passwort und sein Status ändern und speichern. Die angegebenen Daten werden dabei vertraulich und vor Missbrauch geschützt (O10) .

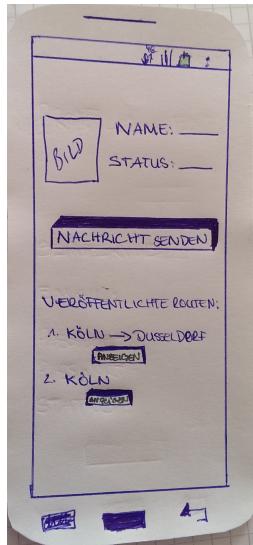


Abbildung .23: Paperbased Prototyp - Profilansicht von anderen Benutzern

Dies ist die Sicht eines Benutzers, der das Profil eines anderen Benutzers aufruft. Durch den Button 'Nachricht senden' kann der Benutzer eine Nachricht senden

(siehe Abbildung 16). Zusätzlich sieht er erstellten Routen des Benutzers, die er auf der Karte abrufen kann.

3.2 Fazit zu der Gestaltungslösung

Die Gestaltungslösung erfüllt die Anforderungen, die in der Anforderungsanalyse festgelegt wurden. Jedoch sind einige funktionalen User-Interfaces entstanden, die zwingend notwendig sind, aber keine Anforderungen aufweisen. Um die Anforderungsanalyse zu vervollständigen und die fehlenden Anforderungen zu identifizieren wurde der gesamte Meilenstein 3 iteriert. Zur Ermittlung der Anforderungen wurden neue präskriptive Szenarien erfasst, woraus die fehlenden Funktionalitäten ermittelt wurden.

Szenario 6: Anke möchte neue Leute kennenlernen

Anke hatte letzte Woche ihre ersten Reitstunden und möchte heute neue Leute kennenlernen, die sie eventuell begleiten möchten. Anke ist begeistert vom Reitsport und hatte einen tollen Einstieg in die Reitwelt. Das einzige was ihr fehlt sind neue Bekannschaften mit Reit-Anfängern wie sie selbst. Jedoch ist es für sie wichtig, dass es sich hierbei ausschließlich um Mädchen/Frauen handelt, da sie einen Freund hat, der sehr eifersüchtig ist. Um diese Situation zu ändern ladet sich Tina die Anwendung auf ihrem Smartphone runter, die sie von einer Freundin letzte Woche erfahren hat. Sie öffnet die Anwendung und es erscheint prompt ein Eingabefeld zum Einloggen und falls noch nicht registriert wurde, ein Button der die Benutzer zur Registrierung weiterleitet. Anke registriert sich und beginnt direkt mit dem Suchen nach neuen Freunden. Sie sucht nach Benutzern, die das selbe Interesse teilen. Es ist sehr wichtig für sie, dass die Benutzer, die sie kennenlernen möchte, auch Reit-Anfänger sind. Sie erforscht einige Benutzerprofile und wird sehr schnell fündig, da der Status den Benutzer kategorisiert. Sie landet aus Zufall auf Zeyneps Profil und bemerkt, dass sie weiblich ist, da sie ein Profilfoto hochgeladen hat. Anke freut sich sehr, da sie bemerkt, dass viele Reit-Anfänger vorhanden sind. Einigen dieser Benutzer, die in der Nähe wohnen, schreibt Anke eine Nachricht, um einen Kontakt zu knüpfen. Als sie die Nachricht versendet und auf das Profil eines Benutzers genauer hinschaut, entdeckt sie, dass ein Reit-Anfänger bereits eine Route geteilt hat. Sie öffnet die Route und liest sich den Bericht zu der Route durch. Anke ist begeistert und motiviert sich umso mehr, sehr bald auch eine eigene Route zu erstellen und diese mit weiteren Benutzern zu teilen, da die Benutzer sich damit gegenseitig informieren können.

Analyse des Szenarios:

Der Reiter möchte neue Leute kennenlernen und dabei unter gewissen Voraussetzungen in Kontakt treten. Der Reiter möchte eigene Routen erstellen und

die gesammelten Informationen in Form von Feedback mit anderen Benutzern teilen. Der Reiter möchte den Erfahrungsgrad eines anderen Reiters erkennen.

Claim Analyse:

+Erhalt von Informationen zu Personen +Kategorisierung der Personen über den Erfahrungsgrad +Orientierung an den Erfahrungsberichten von anderen Benutzer +geschlossenes System durch Registrierung

Begriff	Erklärung
Status	Der Status repräsentiert die reale Reiterfahrung. Dabei wird der Status in 3 Untertitel gegliedert: Professionell, Amateur und Anfänger

Tabelle .3: Glossar für die Anforderungen

Erweiterung der Anforderung anhand des Szenarios 6:

3.2.1 Funktionale Anforderung

- F110 Identifikation der Nutzer**

Das System muss jedem Nutzer eine möglich reale Repräsentation einer Person bereitstellen.

- F120 Entdecken neuer Routen**

Das System muss dem Nutzer die Möglichkeit bieten neue Routen zu entdecken.

- F130 Speicherung der Routen**

Das System muss dem Nutzer die Speicherung und das abrufen von eigenen Routen gewährleisten.

- F140 Feedback zu Routen**

Das System muss dem Nutzer eine Möglichkeit anbieten sich zu Routen zu äußern und diese innerhalb der Anwendung sichtbar für andere Nutzer ist die die selbe Route reiten.

- F150 Kategorisierung der Reiter**

Das System muss den Nutzer ein "Status" bereitstellen anhand der Nutzer sich in die Domäne kategorisieren lässt.

- F160 Wettervorhersage**

Das System muss dem Nutzer eine Wettervorhersage anbieten.

3.2.2 Organisatorische Anforderung

- O80 Registrierung**

Das System darf nur über eine Registrierung den Zugriff in das System gewährleisten.

- O90 Teilen von Informationen**

Das System darf Informationen von Nutzer innerhalb des System anderen Nutzern zu Verfügung stellen.

3.2.3 Qualitative Anforderung

- **Q80 persistente Speicherung**

Das System darf keine Daten mehrfach abspeichern.

- **Q90 Speicherung**

Das System darf die gespeicherten Daten und

- **Q70 Zugriffsrechte**

Das System darf keine Ansichten oder Veränderungen an den Informationen unbefugten dritten Personen gestatten.

Die Zuordnung der Anforderungen zu den jeweiligen Prototypen wird erst in Meilenstein 5 vorgenommen, da die Evaluation in Meilenstein 5 durchgeführt wird und anhand dessen die Verbesserungen zum Prototypen vorgenommen werden.

4 Projektplan

Datum / KW	Aktivität	1. Unteraktivität	2. Unteraktivität	Workload geplant / p.P.	Workload gesamt	Workload tatsächlich	
					Derya Ergue	Sinem Kaya	
14	Exposé	Ideenfindung	Brainstorming	3	3	3	
	Dokumentaufbau	Layout / Struktur	Latex	3	0	4	
	Projektplan			2	6	1,5	
13.04.2015	Milestein 1	Nutzungsproblem		1	1	1	
		Zielsetzung		1	1	1	
		Verteilte Anwendungslogik		1	1	1	
		Wirtschaftliche und gesellschaftliche Relevanz		1	1	1	
27.04.2015	Milestein 2	Zielhierarchie	Strategische Ziele	1	2	2	
			Taktische Ziele	1	2	2	
		related-works	Operative Ziele	1	1	1,5	
			ReiterApp	1	0	1	
			Cavallo-Retcoach	1	1	0	
			sonstige	1	0	1	
		Alleneinstellungsmerkmale		1	1	1	
		Methodischer Rahmen (MCI)		15	11	9	
		Kommunikationsmodell		5	8	3	
		Risiken		3	2	2	
		Spezifikation der POCs		1	2	3	
11.05.2015	Milestein 3	Architekturdigramm/ Architekurbegründung	Architekturdigramm	3	2	3	
			Architekurbegründung	3	0	5	
					MS2 gesamt IST:	32	33,5
01.06.2015	Milestein 4	Datenstrukturen	XML-Schemata	4	0	7	
			ER-Diagramm	2	4	0	
		WBA-Modellierung		10	7	0	
		Prototypen UI		10	11	10	
					MS4 gesamt IST:	22	17
15.06.2015	Milestein 5	funktionale Prototypen	Programmierung gesamt	150	40	66	
		Evaluationsergebnisse UI		10			
		narratives Konzept für filmische Präsentation		10			
29.06.2015	Milestein 6				MS5 gesamt IST:	40	66
		Prozessassessment		10			
		Fazit		4			
		Installationsdokumentation		5			
			Insgesamt Soll	200	Insgesamt Ist:	132	160,5

Abbildung .24: Projektplan zu Meilenstein 3 und 4

Abbildungsverzeichnis

.1	XML-Datei 'Karte'	3
.2	XML-Datei 'Route'	4
.3	XML-Datei 'Profil'	4
.4	XML-Schema 'Karte'	6
.5	XML-Schema 'Route'	6
.6	XML-Schema 'Profil'	7
.7	ER-Modellierung zur Datenstruktur	7
.8	Paperbased Prototyp - Login	12
.9	Paperbased Prototyp - Registrierung	13
.10	Paperbased Prototyp - Hauptmenü	13
.11	Paperbased Prototyp - Karte	14
.12	Paperbased Prototyp - Route erstellen 1	15
.13	Paperbased Prototyp - Route erstellen 2	16
.14	Paperbased Prototyp - Route erstellen 3	16
.15	Paperbased Prototyp - Route erstellen 4	17
.16	Paperbased Prototyp - Route erstellen 5	18
.17	Paperbased Prototyp - Wetter 1	18
.18	Paperbased Prototyp - Wetter 2	19
.19	Paperbased Prototyp - Nachrichten 1	20
.20	Paperbased Prototyp - Nachrichten 2	20
.21	Paperbased Prototyp - Meine Routen	21
.22	Paperbased Prototyp - Profileinstellungen Benutzer Ansicht	22
.23	Paperbased Prototyp - Profilansicht von anderen Benutzern	22
.24	Projektplan zu Meilenstein 3 und 4	27

Tabellenverzeichnis

.1	Daten innerhalb der Anwendung	2
.2	Ressourcen Übersicht	11
.3	Glossar für die Anforderungen	24

5 Literaturverzeichnis

- [1] <http://developer.android.com/reference/org/xmlpull/v1/XmlPullParser.html> - Sichtungsdatum: 21.05.2015
- [2] <http://api.openweathermap.org/data/2.5/forecast/?q=frechen&mode=xml&APPID=fbc548f0ad7993a7b9a589366a0a84fb> - Sichtungsdatum: 25.05.2015
- [3] http://equivetinfo.de/html/eckdaten_pferd.html - Sichtungsdatum: 25.05.2015