swp15-lib

Entwurfsbeschreibung des Gesamtprojektes

Projektleiter: Christian Blecha

Christian Blecha
26.05.2015

Inhaltsverzeichnis

1 Allgemeines

Ziel des Projektes ist es eine mobile App zu entwickeln, die einerseits das Austeilen von Informationsmaterialien unterstützt und andererseits auf mehreren Plattformen verfügbar ist. Der Überblick über die schon belieferten Häuser soll dadurch erleichtert werden. Ein weiteres Feature der App ist die Navigation. Man kann sich entlang seiner Route navigieren lassen, wenn man z.B. noch neu in der Stadt ist bzw. die Austragungstrecke nicht kennt. Die App ist so angelegt, dass die Daten einer Route am Anfang heruntergeladen werden müssen, man dann ohne aktive Internetverbindung die Route abarbeiten und am Ende, wenn man z.B. wieder zu Hause ist, die neugewonnenen Informationen wieder mit dem Server synchronisieren kann. Außerdem bildet unser Projekt eine Grundlage für die Planung der Routen. Man kann z.B. mit den Daten ermitteln, wie viele Materialien ein Verteiler wirklich benötigt, denn der Verteiler kann in der App vermerken, wenn sich z.B. die Gesamtanzahl der Briefkästen oder die Anzahl der „Keine Werbung“-Aufkleber geändert hat. Dazu wird dem Verteiler die Möglichkeit gegeben neuere Informationen über ein Haus bzw. generell neue Häuser einzutragen.

2 Produktübersicht

Das Produkt besteht aus einer MySQL-Datenbank, 2 PHP-Skripten und der App. Die MySQL-DB speichert alle Daten, wie z.B. die Login-Daten für die Benutzer, die Routeninformationen und die Informationen über alle Häuser. Die PHP-Skript werden dafür verwendet, dass das Smartphone eine Verbindung zur MySQL-Datenbank aufbauen kann. Die einzige Funktionalität, die sie bieten, ist die Verbindung zur Datenbank aufzubauen und die Daten weiterzuleiten und wenn nötig eine Fehlermeldung zurückzugeben. Die MySQL-DB und die PHP-Skripte können auf jedem System installiert werden, auf dem man root-Rechte für MySQL besitzt sowie einen Webserver mit PHP-Unterstüzung installiert hat. Dafür würde sich z.B. ein Rasberry-Pi anbieten. Die App wird für mehrere System entwickelt, wie z.B. Android und Windows Phone. Sie enthält die gesamte Funktionalität. Sie bietet z.B. eine Karte, in der 10 Häuser durch Marker dargestellt werden, wobei 4 davon später die zuletzt abgehakten und die anderen 6 die nächsten noch offenen Häuser auf der Route beschreiben.

3 Grundsätzliche Struktur- und Entwurfsprinzipien

Das Produkt kann in 3 Teile gegliedert werden: die Datenbank, die Verbindung der App zur Datenbank und natürlich die App selbst.
Nähere Informationen zur DB sind in Kapitel 4.1 auf Seite 5 zu finden.
Bei der Verwendung von Qt sind Treiber und Bibliotheken vorhanden, welche eine Verbindung zur MySQL-DB realisierbar machen sollten. Dadurch konnten wir eine Verbindung zu einer lokalen MySQL-DB von der für Desktop kompilierten App herstellen, um einen ersten Test der Funktionalität auszuführen. Leider gelang es uns nicht (nach zahlreichen Arbeitsstunden), diese Verbindung bei der für Android kompilierten App aufzubauen.

Grund dafür war eine fehlerhafte Kompilierung der MySQL-Lib für die ARM-Architektur der Android-Geräte. Die Ursachen für diesen Fehler waren u.a. falsche Toolchains, Kompiler, etc. Letztendlich entschieden wir uns gegen die Direktverbindung zwischen der App und MySQL-DB über Qt, da trotz enormen Arbeitsaufwand keine Lösung zustande kam. Unser Ausweichplan lässt nun 2 PHP-Skript2 zwischen App und MySQL-DB vermitteln. Dieses bauen die Verbindung zur DB auf, leiten das SQL-Statement von der App an die DB weiter und die Ergebnisse wieder an die App zurück.
Die App besteht im Grunde aus mehreren Screens: z.B. einem für den Login des Verteilers, einem für die Anzeige der Karte und einem für das Eintragen neuer Informationen sowie ein paar weiteren Screens für z.B. Einstellungen. In Abb. 1 auf Seite 4 sind ein Entwurf für den Login- und für den Kartenscreen abgebildet, da einerseits der Kartenscreen der Hauptscreen sein wird und andererseits alle anderen Screens nur für die Auswahl von Optionen und das Anzeigen/ Eintragen von Informationen dienen, wodurch sie sich kaum vom Loginscreen unterscheiden werden.

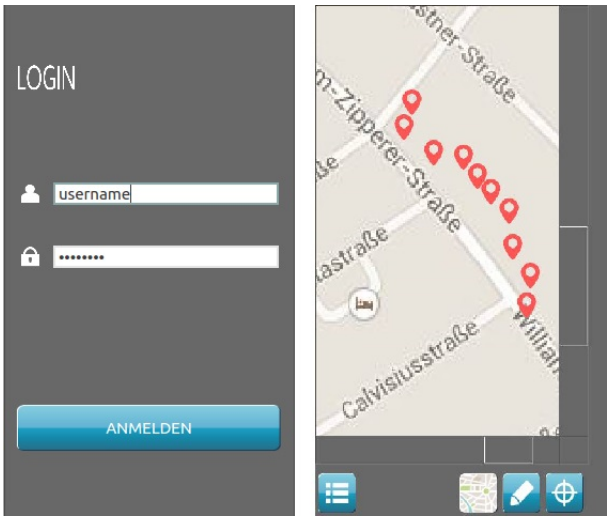

Abbildung 1: Die GUI. (links: Der Loginscreen, rechts: Der Kartenscreen)

Alle Screens wurden an dieses Schema angelehnt, was die Art der Informationsdarstellung und -eintragung sowie die Art des Farbschemas und der Gestalltung der Buttons betrifft. Der Kartenscreen in Abb. 1 auf Seite 4 ist in zwei Teile geteilt: der obere dient der Darstellung der Karte und dem Anzeigen der Route sowie der Standpunkte der Häuser. Die dort aufgeführte Karte soll in zwei Versionen verfügbar sein: einmal als schematische Darstellung und einmal als Satellitenbild. Der Nutzer kann dann in den Optionen auswählen, welche Variante er bevorzugt. Das Satellitenbild wurde eingefügt, weil es anhand von diesen Aufnahmen einfacher ist z.B. einen Eingang in einen Innenhof oder die genaue Position der Briefkästen zu finden, wie mit einer schematischen Darstellung. Dia App ist nicht auf den dauerhaften Gebrauch von GPS ausgelegt. Möchte man seine aktuelle Position auf der Karte markieren, drückt man den entsprechenden Button in der App. Die Navigation ist dadurch gewährleistet, dass nur eine Reihe von Häusern angezeigt wer-

den. Arbeitet man diese ab, werden die nächsten Häuser in der Reihe angezeigt. Dadurch wird man entlang der Route navigiert. Kennt man die Route schon und muss auch nichts neues einfügen, kann man diese auch durch das Beenden der Route gleich als bearbeitet markieren.
Der Loginscreen dient zur Eingabe des Benutzernamens und des Passworts des Verteilers, damit sich dieser mit der DB auf dem Server verbinden kann, um z.B. neue Karten herunterzuladen, neue Informationen hochzuladen bzw. um dem Server mitzuteilen, dass er Routen abgearbeitet hat. Sobald die Daten auf dem Smartphone sind, kann der Verteiler die Internetverbindung trennen und mit seiner Arbeit beginnen. Möchte der Verteiler im späteren Verlauf z.B. Informationen zu einem Haus ändern, dann kann er vom Kartenscreen in den entsprechenden Update-/ Insert-Screen über eine Druck auf den entsprechenden Hausmarker oder eine freie Stelle auf der Karte wechseln. In diesem Screen werden dann schon vorhandenen Informationen zu einem Haus in Textboxen angezeigt. In diesen kann der Verteiler sogleich die Änderungen oder Neuerungen eintragen oder er legt generell ein neues Haus an. Diese Informationen werden dann bis zu einer erneuten Anmeldung auf dem Server sowie der Resynchronisation der Daten mit diesem, auf dem Smartphone gespeichert. Über eine Menüstruktur ist z.B. auch ein Optionsscreen erreichbar sein.
Um eine Kartendarstellung zu ermöglichen, ist eine Bildquelle für den benötigten Kartenausschnitt notwendig. Da Google Maps sowohl Satelliten- als auch Gelände-Bilder zur Verfügung stellt, war die erste Idee, eine der Google Maps APIs zu verwenden. Die Static Maps API (https://developers.google.com/maps/documentation/staticmaps/?hl=de) schien auf den ersten Blick wie geschaffen für unsere Anforderungen, da wir per URL Parameter Bilddateien gewünschter Kartenausschnitte zurückbekommen. Jedoch verwarfen wir diese Idee recht zügig, da die kostenlose Nutzung dieser API lediglich Bilder der Größe 640x640 Pixel (bzw. 1280 x 1280 Pixel mit 2 Zoomstufen mit jeweils 640x640 Pixel) zurückgibt, was für unsere Ansprüche nicht genügt. Unser derzeitiger Lösungsansatz besteht darin, einen hochaufgelösten Screenshot von Google Maps (https://www.google.de/maps/) des gewünschten Kartenausschnitts manuell aufzunehmen und zu verwenden, mit den zugehörigen GPS Koordinaten der Eckpunkte links-unten und recht-oben. Diese Punkte bilden einen Offset für die GPS Koordinaten der Briefkästen, um diese im Nachhinein auf unsere Anzeige der Bilddatei einzufügen.

4 Struktur- und Entwurfsprinzipien einzelner Pakete

4.1 Die Datenbank

- Das Attribut Material & die Tabelle Haeuser_auf_Route:
  Ein Haus kann grundsätzlich auf mehreren Routen mit unterschiedlichen Materialien liegen. Deshalb wird zu jeder Route in der Tabelle „Routenverteilung“ auch gleichzeitig das dazugehörige Material, welches ausgeliefert werden soll, eingetragen. Man muss darauf achten, dass die Materialbezeichnungen immer gleich geschrieben werden & dass die gleiche Wortwahl für die Bezeichnung eines Materials verwendet wird. Ein Haus darf nicht auf zwei Routen mit dem selben Material lie-

gen. Ein Trigger ist hier schwer realisierbar, da es vom DBMS nicht realisierbar ist, verschiedene Schreibweisen für ein Material bzw. verschiedene Wortwahlen in der Bezeichnung für das selbe Material, zu vergleichen. Deshalb muss man selbst bei der Erstellung bzw. dem Import der Routendaten in die Datenbank darauf achten.

- Der Start & das Ende einer Route sind GPS-Referenzen und keine Routenpunkt-Referenzen:
  Wäre dies nicht gegeben, würde ein Referenz-Zyklus entstehen, welcher die Implementierung der Datenbank auf einem Server mit einem MySQL-DBMS nicht zulässt, denn das DBMS lässt keine Relationen zu, die sich gegenseitig als Fremdschlüsselbeziehungen besitzen. Trotzdem sollten diese Punkte als besondere Punkte einer Route abgespeichert werden. Der Start & das Ende können auch Routenpunkte auf einer Route sein (siehe Abb. 2 auf Seite 7). Weiterhin können Start und Ende einer Route ebenfalls gleich sein (siehe Abb. 3 auf Seite 7).

- Die Verteilertabelle:
  Diese dient nur noch der Organisation & der Haltung der Passwörter für die Administratoren. Sie ermöglicht zusätzlich einen „PW vergessen“-Support & organisiert die max. Anzahl an Routen die ein Verteiler bearbeiten kann. Diese Anzahl wird durch einen speziellen Trigger, welcher die Anzahl aller Routen für einen Verteiler bei jeder neuen Zuteilung eines Verteilers zu einer Route, überprüft.

- Die Routenverteilungstabelle:
  Da auch hier der Fakt eintritt, dass das DBMS die verschiedenen Schreibweisen einer Materialbezeichnung nicht erkennen kann, ist es notwendig, dass beim Einfügen einer neuen Route selbst darauf geachtet wird, dass eine Route mit einem Material nur einem Verteiler zugeordnet wird. Weiterhin soll auch gesagt sein, dass die App später nur Routen aus dieser Tabelle einliest, die noch nicht abgearbeitet wurden. D.h. es kann der Fall eintreten, dass ein Verteiler nach seinem Login keine Routen zur Auswahl in der Liste vorfindet.

- Die Briefkastenanzahl & die Anzahl der „Keine Werbung“-Aufkleber bei Häusern:
  Das Neuzählen aller vorhandenen Briefkästen ist in den meisten Fällen nicht notwendig. Man muss nur die Anzahl der „Keine Werbung“-Aufkleber updaten. Dies führt zu einer Reduzierung der Fehler, da die Gesamtanzahl aller Briefkästen & die Anzahl der „Keine Werbung“-Aufkleber jeweils leichter zählbar ist. Außerdem können so die tatsächlich benötigten Materialanzahlen bestimmt werden.

- Kein SSH:
  Da hierauf vom Productowner nicht extra Wert gelegt wurde, reicht für unsere Zwecke die Zugangsberechtigung über das DBMS.

- Stored Procedure „NewCreateUser“:
  Diese Methode erleichtert, wie bereits erwähnt, das Anlegen eines neuen Verteilers. Man übergibt ihr den Benutzernamen des Verteilers, sein Passwort und die Anzahl an Routen, die er max. zugeteilt bekommen möchte. Die „Stored Procedure“ legt

einen neuen Nutzer im DBMS an, speichert die Werte in der Tabelle Verteiler, legt entsprechende Views für den Verteiler an und gibt ihm die entsprechenden Rechte.

- Stored Procedure „DeleteHelp“:
  Diese Methode erleichtert einem das Löschen eines Verteilers, in dem sie den Verteiler und alle zu ihm gehörigen Views und Einträge in der Tabelle Verteiler löscht.

- Stored Procedure „NewCreateAdmin“:
  Diese Methode legt einfach einen neuen Benutzer an und übergibt ihm im selben Zuge die gleichen Rechte wie jeder Admin.

- Die Tabelle Karten:
  Da wir uns dafür entschieden haben zwei Varianten der Karte (schematische Darstellung & Satellitenbild) anzubieten, müssen diese auch seperat in der DB abgelegt werden. Weiterhin tritt der Fakt in den Vordergrund, dass wir 2 Punkte für jede der beiden Karten speichern. Diese Punkte beschreiben dabei die GPS-Koordninaten der linken unteren und der rechten oberen Ecke der jeweiligen Karten und dienen dazu, ein Koordinatensystem über die Karten zu legen, mit deren Hilfe wir die aktuelle GPS-Position und die Häuser auf der Karte eintragen können.

- Die Attribute Position der Tabellen Häuser_auf_Route und Routenpunkt:
  Um auf der Route navigieren zu können, muss man bestimmte Punkte auf der Karte in einer bestimmten Reihenfolge ablaufen. Daher wurde das Attribut Position eingefügt.

- Die Views: Für jeden Benutzer werden während der Einrichtung verschiedene Views in der DB angelegt. Diese Views sind dazu da, dass ein Verteiler auch nur die Daten einsehen kann, die ihm zugeteilt sind bzw. die er auf Grund seiner zugeteilten Routen benötigt.

Abbildung 2: Routenbeispiel 1

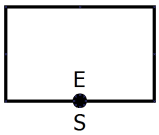

Abbildung 3: Routenbeispiel 2

5 Das Datenmodell

Für das Gesamtprojekt wurden bereits Klassen moddeliert, welche die Daten aus der DB aufnehmen sollen. Dafür wurde ein Mapping der Attribute der Relationen der DB auf die Attribute der Klassen der App erstellt (siehe Abb. 4 auf Seite 9).

6 Das Teskonzept

In unserem Projekt beruht das Testverfahren in der Grundentwicklung der App auf dem Qt Test Framework. Hierfür entwickelt jedes Mitglied seine eigenen, zum jeweiligen Modul passenden Tests, führt diese aus, dokumentiert das Ganze und schickt es dann dem Testverantwortlichen. Im Anschluss daran werden die Tests vom Testverantwortlichen geprüft, dokumentiert und ausgewertet. Des Weiteren sollen mit „TestGui“ GUI Events in unserer Benutzeroberfläche getestet und ausgewertet werden. Hierzu vergleicht Qt zu erwartende und tatsächlich erhaltene Ergebnisse und erkennt somit Fehler in der Verarbeitung.
Für die Anbindung der App an die MySQL Datenbank über PHP werden keine mehrfachen Tests durchgeführt, sondern es wird lediglich ermittelt, ob die abgerufenen Daten auch die sind, die abgerufen werden sollten und ob diese auch in der Art ausgewertet werden, wie dies im weiteren Programmcode nötig ist.
Aufgrund unserer sehr benutzerorientierten Aufgabe wird ein großer Teil der Tests direkt am Gerät durchgeführt. Nur so kann gewährleistet werden, dass das Handling und die Benutzbarkeit bei verschiedensten Umständen sowie die Genauigkeit unserer Map und die auf ihr enthaltenen Events unseren Vorstellungen und Richtlinien entsprechen.

7 Glossar

7.1 Abkürzungsverzeichnis

DB: Datenbank

DBMS: Datenbankmanagementsystem

GUI: Graphical User Interface

8 Anhang

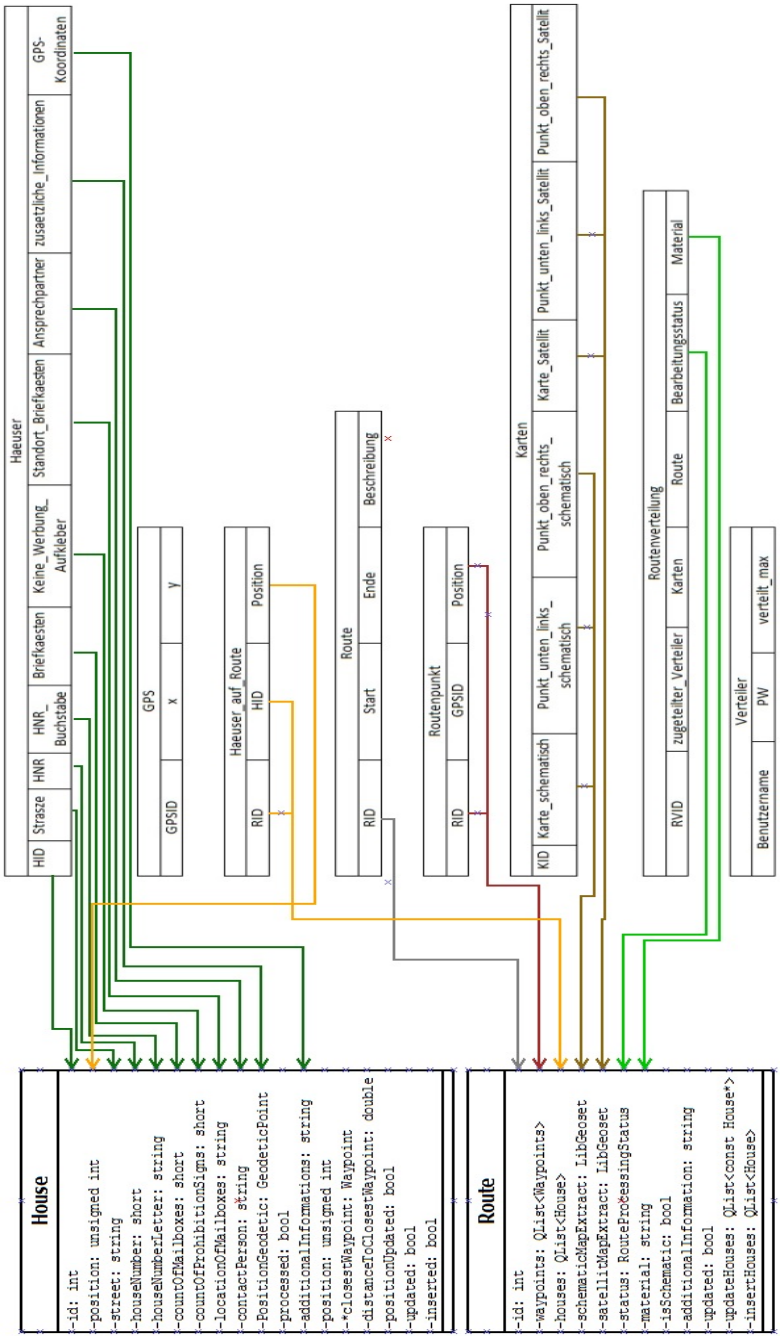

Abbildung 4: Das Mapping der Tabellen der DB auf die Klassen der App.