TH Köln Studiengang
Medieninformatik
EIS Projekt
Sommersemester 2017

Technology
Arts Sciences
TH Köln

Implementationsdokumentation des Projekts HarvestHand

Studierende

Franziska Gonschor
Sergej Atamantschuk

Betreuer

Robert Gabriel
Prof. Dr. Gerhard Hartman
Prof. Dr.Kristian Fischer

Inhaltsverzeichnis

1. Überblick über das System

Im Folgenden wird allgemeiner Überblick über die Funktionsweise des Systems gegeben. Die Interaktion des Systems startet mit dem Login/Signup Fenster, in dem man sich mit der Telefonnummer einloggen oder registrieren kann. Bei der Registration muss der Usertype (Literater/Illiterate) ausgewählt werden. Dadurch kann das System die Lese – und Schreibekompetenzen des Users feststellen und demensprechend die Präsentation der Daten anpassen. Die Präsentationen unterscheiden sich hauptsächlich darin, dass im Profiltype „Illiterate“ keine Verwaltungsfunktionen für Einträge wie Erstellen, Aktualisieren oder Löschen zugänglich sind.

Im Modus „Literate“ kann ein neuer Eintrag erstellt, aktualisiert und gelöscht werden. Bei der Erstellung müssen benötigte Daten eingegeben werden. Die „Location“, „Air temp.“ und „Air humidity“ können von dem System automatisch eingetragen werden um den Prozess der Dateneingabe zu vereinfachen. Diese Funktion war ebenfalls für Attribute „Soil temp.“ und „Soil moisture“ geplant. Leider hat das Entwicklerteam keinen Zugang zu den APIs erhalten, die solche Daten zu Verfügung stellen. Die Kontaktierten Anbieter wollten keinen Zugangs-Key für unsere Zwecke zur Verfügung stellen. Des Weiteren können einem Eintrag andere Benutzer des Systems als Collaborators hinzugefügt werden, damit diese ebenfalls den Zugriff auf den Eintrag bekommen. Dies hat in dem Fall Sinn, wenn ein Benutzer(Landwirt) ein Smartphone besitzt, aber selber nicht in der Lage ist, einen Eintrag für sein Feld zu erstellen. So kann der Helfe einen Eintrag für sein Feld erstellen und den Landwirt als Collaborator hinzufügen. Auf diese Weise erhalten beide den Zugriff auf den gleichen Eintrag.

Nachdem ein Eintrag erstellt und im System gespeichert wurde, wird direkt eine Datenanalyse durchgeführt, um mögliche Probleme einer Pflanze festzustellen. Darauf aufbauend werden interaktive Anbauempfehlungen erstellt. Die Anbauempfehlungen (Tutorials) werden in Form von Visualisierungen und Verbalen Informationen präsentiert. Der Inhalt eines Tutorials hängt davon ab, wie der aktuelle Wert einer Eigenschaft ist. Z.B. das Tutorial für Bodenfeuchtigkeit wird anders aussehen, abhängig davon, ob der Boden aktuell zu feucht oder zu trocken ist. Zu Präsentation dieser Funktionalität werden von dem Entwicklerteam Abbildungen nur zu einigen Eigenschaften erstellt, um die Realisierbarkeit der Funktion zu demonstrieren.

Weiter wichtige Funktionalität des Systems ist die Vorlese-Funktion der bestimmten Inhalte. Die Informationen, die vom System vorgelesen werden können, sind mit dem entsprechenden Icon gekennzeichnet. Auch die Systemmeldungen wie z.B. „Verbindungsfehler“ werden vom System vorgelesen, um die Gebrauchstauglichkeit für Analphabeten zu optimieren. Um diese Funktion umzusetzen, wurde der TextToSpeech Service von Android verwendet. Dieser Service ermöglicht beliebige Texte in verschiedenen Sprachen vorzulesen. Dabei kann auch die Vorlesegeschwindigkeit gewählt werden, was im Kontext des Systems sehr nützlich war, um die Verständlichkeit des vorzulesenden Textes zu verbessern.

Bei der Implementierung wurde darauf geachtet, dass vor allem der Client möglichst effizient und ressourcenschönend funktioniert. Das Android-System ist leider so aufgebaut, dass die verwendeten Bilder enorm viel RAM während der Laufzeit verbrauchen können. Dies liegt daran, dass alle Images entsprechend der Bildschirm – Größe und – Auflösung skaliert werden müssen. Dadurch kann ein PNG - Bild mit Größe von 2Mb ganze 40Mb des Arbeitsspeichers verbrauchen. Da die meisten Informationen des Systems mit Bildern dargestellt werden, wurde darauf geachtet, dass alle Icons und Images möglichst kleine MB – Größe haben. Als kleiner Trick wurde zudem ein großes Backgroundbild in einen „res“ – Ordner verschoben, in dem die Bilder während der Laufzeit nicht skaliert werden.

2. Änderungen der Datenstruktur

In diesem Kapitel werden während der Entwicklung vorgenommene Änderungen beschrieben.

```
var userSchema = mongoose.Schema({
    /*ILLITERATE= 1, LITERATE= 0*/
    user_type: {type: Number, required: true}
    phone_number:{type: Number, unique: true
}

});
```

Als erstes wurde die Datenstruktur eines Users stark vereinfacht, da die Evaluation des Systems ergeben hat, dass eine gewöhnliche Authentifikation des Users mit Email und Passwort im Kontext des Systems nicht optimal ist. Stattdessen wird der Benutzer mit der Telefonnummer im System authentifiziert. Der `user_type` bleibt erhalten. Auch das Modul „Passport“ und „Session“ werden nicht mehr wegwendet. Somit werden auch keine Zustände im System gespeichert.

```
…
location: {
    name: {type: String, required: true},
    countryISOCode: {type: String, required: true},
    city: {type: String, required: true}
},
…
collaborators_id: Array,
collaborators_number: Array,
…
```

Die Struktur eines Entrys wurde um das Objekt `location` erweitert, um die Wetterabfragen nach Location zu ermöglichen. Die Attribute `collaborators_id` und `collaborators_number` wurden ebenfals hinzugefügt. Der Array `collaborators_id` speichert die IDs der Collaborators für den Fall, dass ein Benutzer, der dem Eintrag als Collaborator hinzugefügt wurde, seine Telefonnummer ändert. Da die ID sich dabei nicht ändert bleibt für diesen Benutzer der Zugriff auf den Eintrag weiterhin erhalten. Der Array `collaborators_number` speichert die Nummern der Benutzer. Die Nummern werden auf dem Client in einer Liste dargestellt.

Des Weiteren wurden kleinere Änderungen vorgenommen wie z.B. die Benennung einiger Attribute, um die Lesbarkeit der Datenstrukturen zu verbessern. Diese Änderungen beeinflussen aber nicht die Funktionalität des Systems und werden nicht explizit beschrieben.

3. Ausblick

Während der Entwicklung des Systems müsste das Entwicklungsteam einige neue Technologien kennenlernen. So wurde z.B. die Implementierung der Externen APIs kennengelernt oder auch der TextToSpeech Service von Android. Die effiziente Umsetzung neuer Technologien fordert meistens ein tieferes Verständnis des Problems und längere Einarbeitungszeiten.

Während der Implementierung kam es dazu, dass einige Codeblöcke sich an verschiedenen Stellen in verschiedenen Klassen wiederholen oder mehrere Instanzen einer Klasse erzeugt werden. Daher sind die Codeoptimierungen an manchen Stellen nötig.

Die überflüssigen Instanzen der TextToSpeech Klasse, die nach der Benutzung immer gelöscht werden müssen, werden auf dem Client in mehreren Klassen erzeugt. Auch das Volley Framework von Android wird im Projekt so eingesetzt, dass es zur Codeverdoppelungen kommt. Diese nicht optimale Codestruktur ist in den frühen Phasen der Entwicklung entstanden, da die genaue Funktionsweise dieser Technologinnen nicht bekannt war. In den späteren Phasen der Entwicklung wurden die Codeoptimierungen bereits vorgenommen. Doch aufgrund des dadurch entstehenden Overheads und der knappen Zeit sind Kritische Punkte in der Codestruktur geblieben und sollen in der Zukunft behoben werden.

4. Installationsdokumentation

4.1. Systemanforderungen

Zum ausführen des Systems werden Nodejs und MongoDB unter Windows 7-10 und Internetverbindung gebraucht. Für den Client ist mindestens Android 4.0 notwendig. Server und Client müssen sich in gleichem Netz befinden. Im Folgenden werden benötigte Schritte detailliert beschrieben

4.2. Server

- Projektordner von Github clonen oder herunterladen
- Nodejs herunterladen und installieren
- In den Projektordner wechseln
- Benötigte Packeges mit dem npm – Befehl über Konsole installieren: *npm install*
- MongoDB herunterladen und installieren
- MongoDB zur umgebungsvariablen des Systems hinzufüge
- In den Ordner *EISSS17GonschorAtamantschuk\MS3\HarvestHand\Server* wechseln
- Datenbank mit dem Konsolenbefehl starten: *mongod –dbpath db*
- Server in der Console starten: *node server*

4.3. Client

- HarvestHand.apk ausführen und installieren
- IP Des Servers am Rechner ermitteln und in der App-Einstellungen eintragen
- App benutzen