



Audit 3 zu JetSet Food

Entwicklungsprojekt WS2020/21

13.12.2020

Sophia Johannsen

Seite 1

Markus Mohr-Dama

Technology
Arts Sciences
TH Köln

Inhalt

1. Interne Datenstruktur
2. MongoDB
3. Durchgeführte Proof of Concepts
4. Rapid Vertical Prototype

13.12.2020

Seite 2

Technology
Arts Sciences
TH Köln

Interne Datenstruktur

- Speicherung von Daten in JSON-Dateien
- MongoDB
- Benutzereingabe Obst / Gemüse
- Rückgabe an den Benutzer: Season Kalender mit Karte
- Die Länder sind als GeoJSONs gespeichert

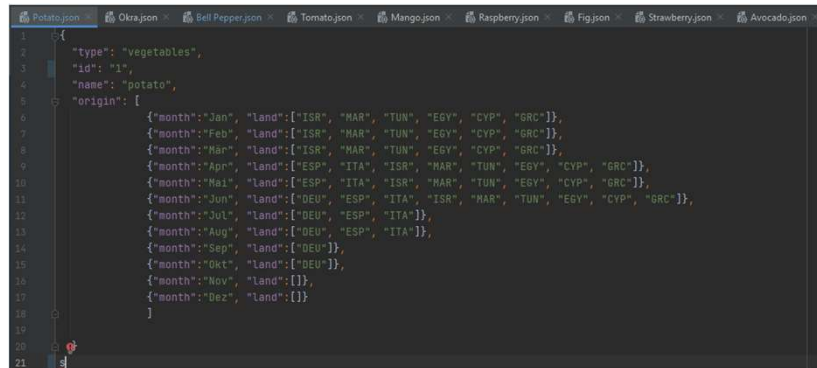
10.01.2021

Seite 3

Technology
Arts Sciences
TH Köln

Nach der Eingabe des Benutzers wird das gewünschte Ergebnis in Form einer JSON Datei übergeben, die den Namen, den Typ sowie den Season Kalender enthält. In den Monaten, in denen das Produkt importiert wird, ist das Land durch ein 3 Buchstabenkürzel abgebildet. Jedes Land ist als Geo-JASON gespeichert. Dem Benutzer wird das Import Land, sowie Deutschland, Visuell markiert und auf einer Karte dargestellt. Außerdem wird eine Route mit der Entfernung angezeigt.

Interne Datenstruktur



```
1 {
2   "type": "vegetables",
3   "id": "1",
4   "name": "potato",
5   "origin": [
6     {"month": "Jan", "land": ["ISR", "MAR", "TUN", "EGY", "CYP", "GRC"]},
7     {"month": "Feb", "land": ["ISR", "MAR", "TUN", "EGY", "CYP", "GRC"]},
8     {"month": "Mar", "land": ["ISR", "MAR", "TUN", "EGY", "CYP", "GRC"]},
9     {"month": "Apr", "land": ["ESP", "ITA", "ISR", "MAR", "TUN", "EGY", "CYP", "GRC"]},
10    {"month": "Mai", "land": ["ESP", "ITA", "ISR", "MAR", "TUN", "EGY", "CYP", "GRC"]},
11    {"month": "Jun", "land": ["DEU", "ESP", "ITA", "ISR", "MAR", "TUN", "EGY", "CYP", "GRC"]},
12    {"month": "Jul", "land": ["DEU", "ESP", "ITA"]},
13    {"month": "Aug", "land": ["DEU", "ESP", "ITA"]},
14    {"month": "Sep", "land": ["DEU"]},
15    {"month": "Okt", "land": ["DEU"]},
16    {"month": "Nov", "land": []},
17    {"month": "Dez", "land": []}
18  ]
19 }
20
21
```

10.01.2021

Seite 4

Technology
Arts Sciences
TH Köln

Auf diesem Bild ist die genau Darstellung einer JSON Datei für Gemüse zu sehen

MongoDB

- dokumentbasierte, verteilte Datenbank für Anwendungsentwicklung
- Cloud MongoDB Atlas
- MongoDB Realm
- Einrichtung von Realm Sync benötigt neueste MongoDB Version
- Upgrade der neuesten Version nicht kostenlos möglich
- Daher sind die Daten für den Prototyp lokal gespeichert

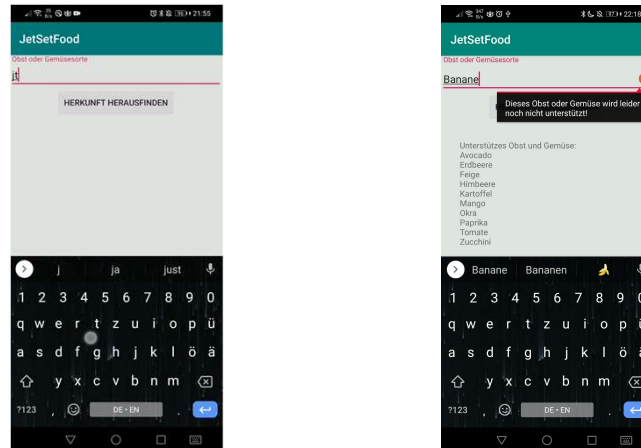
10.01.2021

Seite 5

Technology
Arts Sciences
TH Köln

MongoDB ist eine universelle, dokumentenbasierte, verteilte Datenbank für die moderne Anwendungsentwicklung. Mit dem eigenen Cloud-Dienst MongoDB Atlas und dem Synchronisierungsdienst MongoDB Realm sollen die Daten für die Benutzer jederzeit zur Verfügung stehen. Im Laufe der Konfiguration von MongoDB Realm stellte sich jedoch heraus, dass die Neueste Version der Datenbank benötigt wird, um die Synchronisierung einzurichten. Die neueste Version konnte jedoch mit der kostenlosen Tier Stufe (M0) nicht installiert werden.

Proof of Concept 1 : Nutzereingabe



10.01.2021

Seite 6

Technology
Arts Sciences
TH Köln

Der Nutzer gibt eine Eingabe ein. Die Applikation prüft nun ob diese Eingabe in der Datenbank vorhanden ist. Falls nicht wird der Nutzer darauf aufmerksam gemacht und es werden vorhandene Obst und Gemüsesorten angezeigt. Falls die Eingabe in der Datenbank vorhanden ist, wird dem Nutzer angezeigt, in welchen Ländern das Obst oder Gemüse momentan Saison hat.

```

//Knopf klick auf Anzeigen die Eingabe überprüfen und ggf. das zur eingabe gehörige Datenblatt ausgeben
button.setOnClickListener( R.View()
    if(produceInput.text.toString().isEmpty())
        produceInput.error="Bitte gib eine Obst oder Gemüsesorte ein"

    //prüft ob das Obst oder Gemüse vorhanden ist
    else if(!produceListe.contains(produceInput.text.toString().toLowerCase())) {
        produceInput.error = "Dieses Obst oder Gemüse wird leider noch nicht unterstützt!"
        datenblatt.text="Unterstützte Obst und Gemüse: AnzeigeStrichl "
        datenblatt.visibility= View.VISIBLE
    }

    //liest zur eingabe gehörige Datenblatt ein und wählt die aktuellen Monat aus dem Saisonkalender aus
    else {
        val jsonString = getJsonDataFromAsset( context this, fileName: produceInput.text.toString().toLowerCase() + ".json")
        val data:Produce = gson.fromJson(jsonString,produceType)
        datenblatt.text="Im \$currentMonth+1. Monat im Jahr kann man \$data.name aus \$getCountryName\(data\) kaufen"
        datenblatt.visibility=View.VISIBLE
    }
}

```

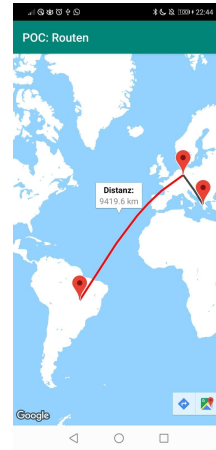
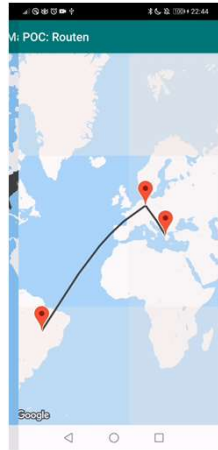
11.01.2021

Seite 7

Technology
Arts Sciences
TH Köln

Es wird überprüft ob die Eingabe des Nutzers in der Datenbank vorhanden ist. Falls nein werden dem Nutzer unterstützte Obst und Gemüsesorten angezeigt und dazu aufgefordert, eine dieser auszuwählen und einzugeben. Falls ja wird die zugehörige Datei eingelesen und im Saisonkalender der aktuelle Monat ausgewählt. Dem Nutzer wird nun angezeigt in welchen Ländern das Obst oder Gemüse saisonal ist.

Proof of Concept 2 : Routen



10.01.2021

Seite 8

Technology
Arts Sciences
TH Köln

Die Luftlinie zwischen dem geografischen Mittelpunkt zweier Länder wird angezeigt. Wenn diese angetippt wird, wird sie rot eingefärbt und die Länge der Route errechnet und angezeigt.

Codesnippet dazu

```
//Zwei Polyline zwischen Deutschland und dem Herkunftsland  
//geodesic bedeutet dass die Linie nicht einfach gerade ist, sondern sich an der Krümmung der Erde orientiert  
fun addRoutes(mMap: GoogleMap, herkunft: List<Input>){  
    herkunft.forEach(country->  
        runCatching { mMap.addMarker(MarkerOptions().position(country.coords).title(country.name)) }  
        .onFailure { it: Throwable  
            Toast.makeText(context, this, text: "Markierung konnte nicht hinzugefügt werden", Toast.LENGTH_LONG).show()  
            Log.e(tag: "Marker", msp: "konnte nicht hinzugefügt werden", it)  
        }  
        runCatching { mMap.addPolyline(  
            PolylineOptions().add(germany).add(country.coords).width(10f).color(  
                Color.BLUE).geodesic(true).clickable(true).zIndex(1.0f)) }  
        .onSuccess { it: Polyline  
            it.is=country  
        }  
        .onFailure { it: Throwable  
            Toast.makeText(context, this, text: "Route konnte nicht hinzugefügt werden", Toast.LENGTH_LONG).show()  
            Log.e(tag: "Route", msp: "konnte nicht hinzugefügt werden", it)  
        }  
        //bewegt die Kamera zwischen die einzelnen marker  
        centre= SphericalUtil.interpolate(centre, country.coords, fraction: 0.3)  
    })  
    mMap.moveCamera(CameraUpdateFactory.newLatLng(centre))  
}
```

11.01.2021

Seite 9

Technology
Arts Sciences
TH Köln

Zunächst wird versucht ein Marker im geografischen Zentrum des Landes der Karte hinzuzufügen. Falls dies nicht funktioniert wird dem Nutzer eine Fehlermeldung angezeigt. Danach wird eine Linie zwischen der geografischen Mitte des Herkunftslandes und Deutschland gezogen. Diese soll sich an der Erdkrümmung orientieren und nicht einfach eine schnurgerade Verbindung auf einer flachen Karte sein. Auch hierbei wird bei Fehlern dem Nutzer eine Benachrichtigung gegeben.

Proof of Concept: Länder markieren



10.01.2021

Seite 10

Technology
Arts Sciences
TH Köln

Das Territorium verschiedener Länder wird markiert. Deutschland wird als Aufenthaltsort farblich hervorgehoben, um diese App auch außerhalb von Deutschland nutzbar zu machen, wäre es sinnvoll in einer späteren Iteration den Aufenthaltsort aus der Geolocation des Nutzers zu bestimmen.

Codesnippet dazu

```
//markiert die in herkunft spezifizierten Länder
fun addOutling(map: GoogleMap, herkunft: List<Int>){
    if(herkunft.isEmpty()){
        Toast.makeText(context, this, text="zu diesem Produkt existieren keine Herkunftsinformationen", Toast.LENGTH_LONG).show()
        //Gibt Fehlermeldung aus, das keine Herkunftsinformationen vorliegen
    }
    herkunft.forEach { h: Int
        //Überprüft ob Objekt in der Liste eine GEOJson Datei im richtigen Format ist
        runCatching { this@MainActivity
            GeoJsonLayer(map, it, context, this)
            //Wenn nicht wird ein Toast angezeigt
            }.onFailure { e: Throwable
                Toast.makeText(context, this@MainActivity, text="Land kann nicht markiert werden", Toast.LENGTH_LONG)
                    .show()
                Log.e(tag="GeoJson", msg="Konnte nicht eingelesen werden", it)
            }
            //Wenn schon werden die Daten angezeigt und das Land kann markiert werden
            .onSuccess { layer ->
                layer.defaultPolygonStyle.strokeWidth = 0.0f
                layer.defaultPolygonStyle.fillcolor = Color.DKGRAY
                layer.addToMap()
            }
            if (!layer.isLayerOnMap) {
                Toast.makeText(context, this, text="Land kann nicht markiert werden", Toast.LENGTH_SHORT).show()
                Log.e(tag="GeoJson", msg="Layer konnte nicht zur Karte hinzugefügt werden")
                //Gibt Fehlermeldung aus, das Land nicht markiert werden konnte
            }
        }
    }
}
```

10.01.2021

Seite 11

Technology
Arts Sciences
TH Köln

Das Territorium der Länder wird in GEOJson Format gespeichert. Dies ist ein standardisiertes Json Format, in welchem geografische Daten gespeichert werden.

In der Applikation wird zunächst überprüft ob die vorhandene GEOJson Datei auch diesem Format entspricht. Falls nicht wird dem Nutzer eine Fehlermeldung angezeigt und die Möglichkeit geboten die Anfrage zu wiederholen. Falls die Datei dem Standard entspricht, wird diese eingelesen und auf der Karte als dunkelgrau ausgefüllte Fläche angezeigt.

Prototyp

- Auswahl von jeweils fünf Obst- und Gemüsesorten
- Nur GeoJSONs der benötigten Importländer
- Android-App Prototypen Eingabe von Obst- oder Gemüsesorte
- Eingabe wird überprüft

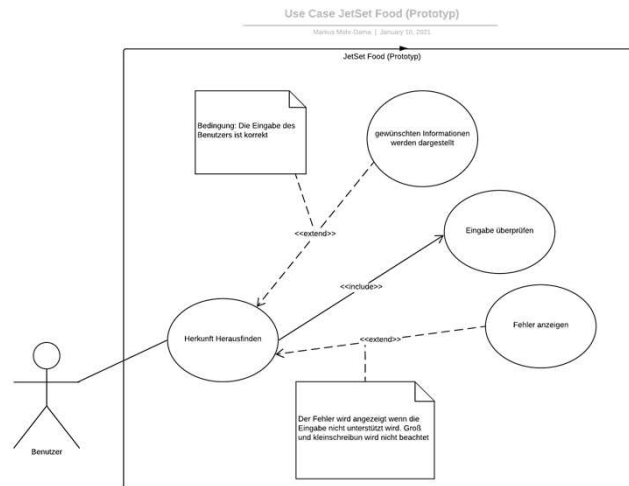
10.01.2021

Seite 12

Technology
Arts Sciences
TH Köln

Der Benutzer kann über den Android-App Prototypen eine Obst- oder Gemüsesorte eingeben und bekommt dann eine Übersicht über den angefragten Season Kalender sowie eine Karte der Importländer. Die Eingabe des Benutzers wird darauf überprüft, ob der angefragte Begriff in der Datenbank zu finden ist. Bei einer ungültigen Eingabe wird der Text „Dieses Obst oder Gemüse wird leider noch nicht unterstützt!“ angezeigt. Die Eingabe des Benutzers wird nicht auf Groß- oder Kleinschreibung überprüft. War die Eingabe erfolgreich wird ein Text ausgegeben aus welchen Ländern das Obst oder Gemüse im Januar importiert wird.

Prototyp



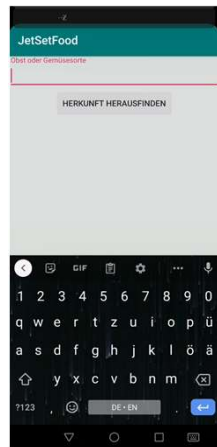
10.01.2021

Seite 13

Technology
Arts Sciences
TH Köln

Dieses Use Case Diagram veranschaulicht nochmal den Funktionsumfang des Prototypen

Prototyp



10.01.2021

Seite 14

Technology
Arts Sciences
TH Köln

Der Prototyp entstand aus der Kombination der Proof of Concepts. Hierbei wurden diese in mehreren Ansichten kombiniert.

Die Karte die verwendet wurde wurde so stilisiert, dass sie nur auf Wasser und Landmassen reduziert ist, dadurch sollen die wesentlichen Informationen nicht durch die Masse an Zusatzinformationen in den Hintergrund rücken. Zur Stilisierung dieser wurde eine speziell formatierte JSON Datei verwendet.