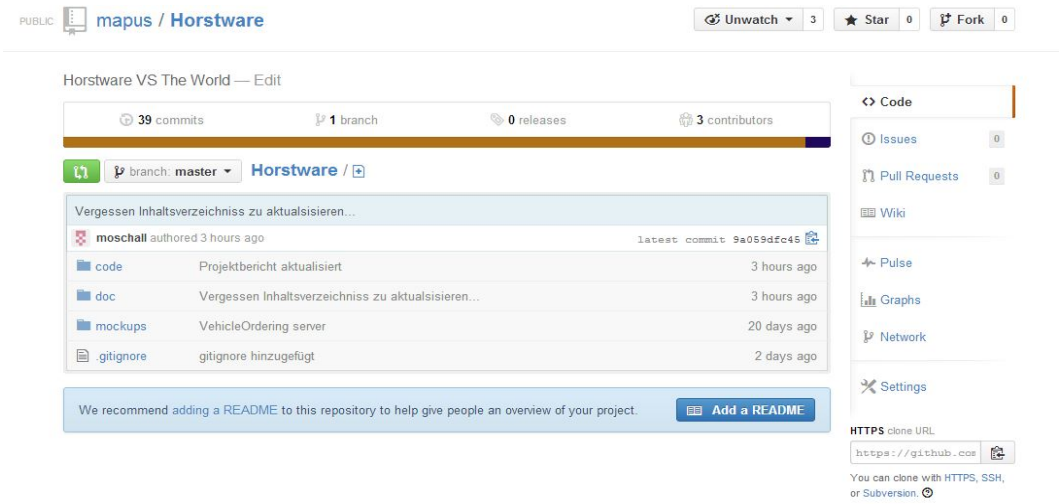
Projektbericht Internetprogrammierung – Stefan Schneider

1. Planung
    1. Erstellen des Repositorys
    2. Erstellen der Projektbeschreibung
2. Implementierung
    1. REST Anbindung unter Android
    2. XML Dateien unter Android Parsen
3. Abschluss

1.Planung

Zur Versionsverwaltung haben wir uns für GitHub entschieden. GitHub bietet die Möglichtkeit ein kostenloses Git-Repository zu erhalten wenn man damit einverstanden ist, dass das Repository öffentlich einsehbar ist.



1.2Erstellen der Projektbeschreibung

Der Nächste Schritt bestand darin die Projektbeschreibung zu schreiben. Dafür mussten die für das Projekt notwendigen anforderungen Disktutiert und ausgearbeitet werden und die Arbeiten aufgeteilt werden.

2.Implementierung

2.1REST Anbindung unter Android

Mein Arbeitsbereich bestand darin eine Schnittstelle zuwischen dem Java-EE Server und dem Android Cliententen bereitzustellen.
Dazu musste ich erst mal die Netzwerkeigenheiten von Android evaluieren, da unter Android mehrere Dinge im Vergleich zu normalem Java anders sind.
Einer diese Besonderheiten ist, dass Android es nicht erlaubt im Hauptthread Netzwerkzugriffe auszuführen was dazu fürht das ich diesen in einen Extra Thread auslagern musste.
Threading unter Android funktioniert allerdings ausschließlich über sogenannte AsyncTasks.
AsyncTasks bieten die Möglichkeit eine lang dauernde Funktion in einer doInBackground() funktion auszuführen und nach dem Ausführen wird eine onPostExcecute() aufgerufen.

Diese Faktoren haben dazu geführt das ich mich für eine Asynchrone Schnittstelle mit Callback-

Interface entschieden habe.

```
public void delete(String url,On RestResponse onResponse){

public void get(String url, OnRestResponse onReponse) {

public void post(String url, String body, OnRestResponse onReponse) throws NetworkErrorException {

public void put(String url, List<NameValuePair> parameters, OnRestResponse onReponse) throws NetworkErrorExcep
```

Der Android-Client erwartet allerdings ein Synchrones Interface. Nach vielen und langen tests hat sich allerdings gezeigt das ein Synchroner Netzwerkaufruf unter Android sogut wie umöglich ist.

2.2**XML Dateien unter Android Parsen**

Anschließend war es noch meine Aufgabe die in XML-Vorliegende Antwort wieder ein die zugehörigen Entitätsklassen umzuwandeln.
Um XML-Datein zu Parsen empfiehlt die Android-Developerseite von Google die verwendung des in Android eingebauten XMLPullParsers.
Nachdem ich diesen für das erste XML File implementiert hatte war relativ schnell klar das dieser Parser viel zu umständlich und unkomfortabel ist.
Nach der Suche auf einer alternative Möglichkeit XML Dateien in Android zu Parsen bin ich auf die Library Simple XML gestoßen.
Nach dem einbinden dieser im Build-Path benötigt diese nur noch ein paar Anotationen über den zu den XML Dateien zugehörigen Entitätenklassen.

```
@Root(name="model")
public class VehicleModel
){
    @Element private String name;
    @Element private VehicleType type;
    @Element private String description;
    @Element private String imageUrl;
    @Element private long priceInEuroCent;
    @Element private long id;
```

Nachdem der Parser funktionsfähig war musste die ganze Schnittstelle noch in dem Clienten implementiert werden was sich aufgrund der zu beginn nicht vorgesehenen Asynchronalität als recht schwierig erwies.

3.Abschluss

Abschließend muss nur noch die Präsentation vorbereitet werden. Dafür gehört unter anderem die Möglichkeit das Bild eines Android Gerätes zu Präsentationszwecken auf einen Laptop zu zeigen.