Technische Spezifikation

HTW-Berlin
Smart Mini Camper

Autor: Gregor Rechtenwald, Philipp Würfel, Marius Lüders
Letzte Änderung: 7. Juni 2019
Dateiname: SmartMiniCamper_Technische_Spezifikationen

Version: 1.0

Inhaltsverzeichnis

Abbildungsverzeichnis

Version Historie

Version:	*Datum:*	*Verantwortlich*	*Änderung*
0.1	06.06.2019	Marius Lüders	Initiale Dokumenterstellung, Überarbeitung der Vorlage, Kapitel für Restful
0.2	06.06.2019	Philipp Würfel	Bearbeitung Kapitel Messsystem Raspberry Pi Kapitel Komponenten und Schnittstellen Kapitel Prozessüberblick Kapitel Beschreibung und Implementierung Strukturierung Dokument
0.3	06.06.2019	Gregor Rechtenwald	Workflow, Kapitel Android App
1.0	07.06.2019	Philipp Würfel	Fritzing Schaltplan

Vorhandene Dokumente

Alle für die vorliegende Spezifikation ergänzenden Unterlagen müssen hier aufgeführt werden

Dokument	Autor	Datum
Smart_Mini_Camper_Lastenheft.doc	Gregor Rechtenwald, Philipp Würfel, Marius Lüders	25.04.2019
Smart_Mini_Camper_Pflichtenheft.doc	Gregor Rechtenwald, Philipp Würfel, Marius Lüders	17.05.2019

1 Systemüberblick

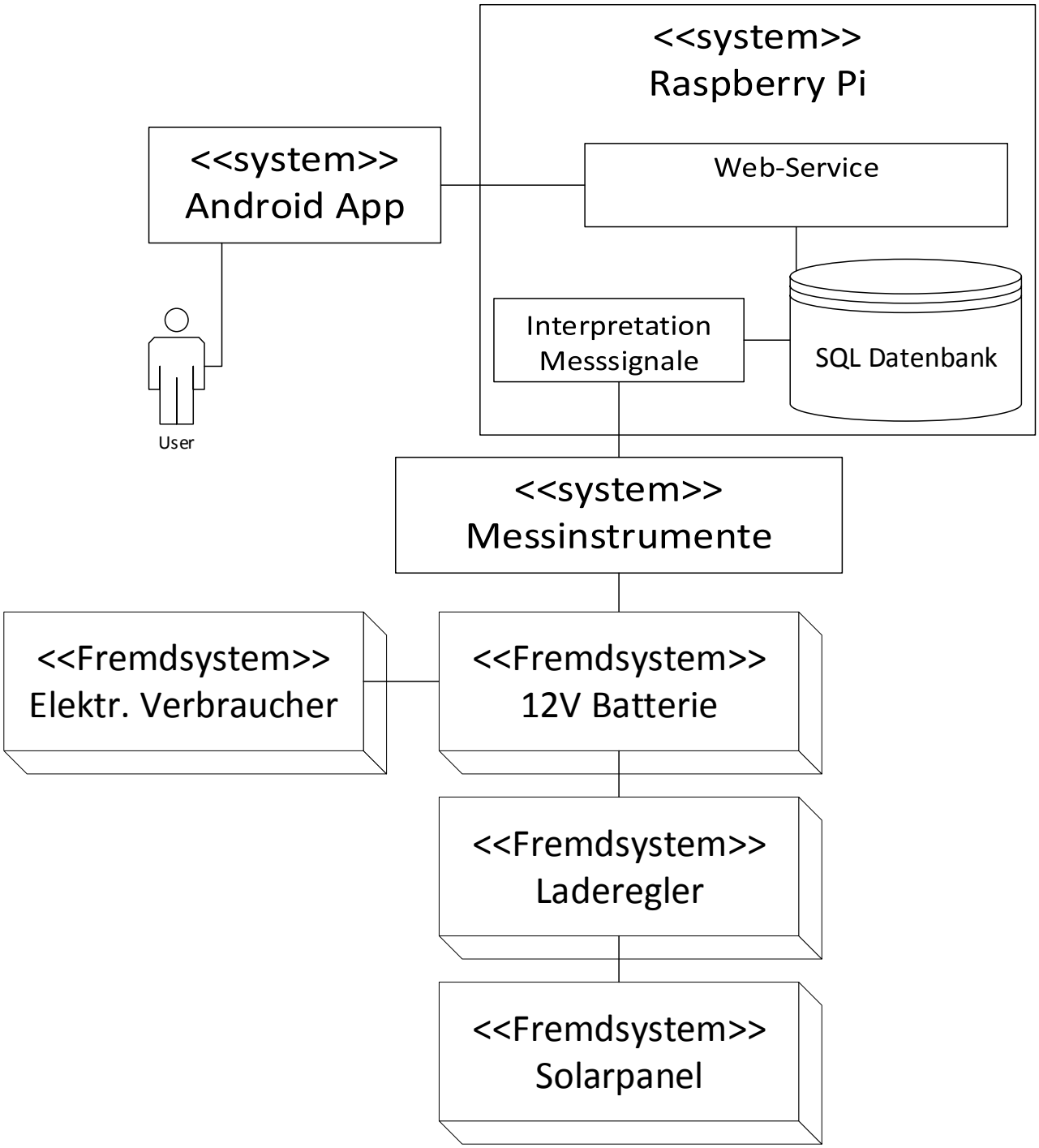


Abbildung 1 Kontextdiagramm zum Smart Mini Camper Projekt

1.1 Workflow

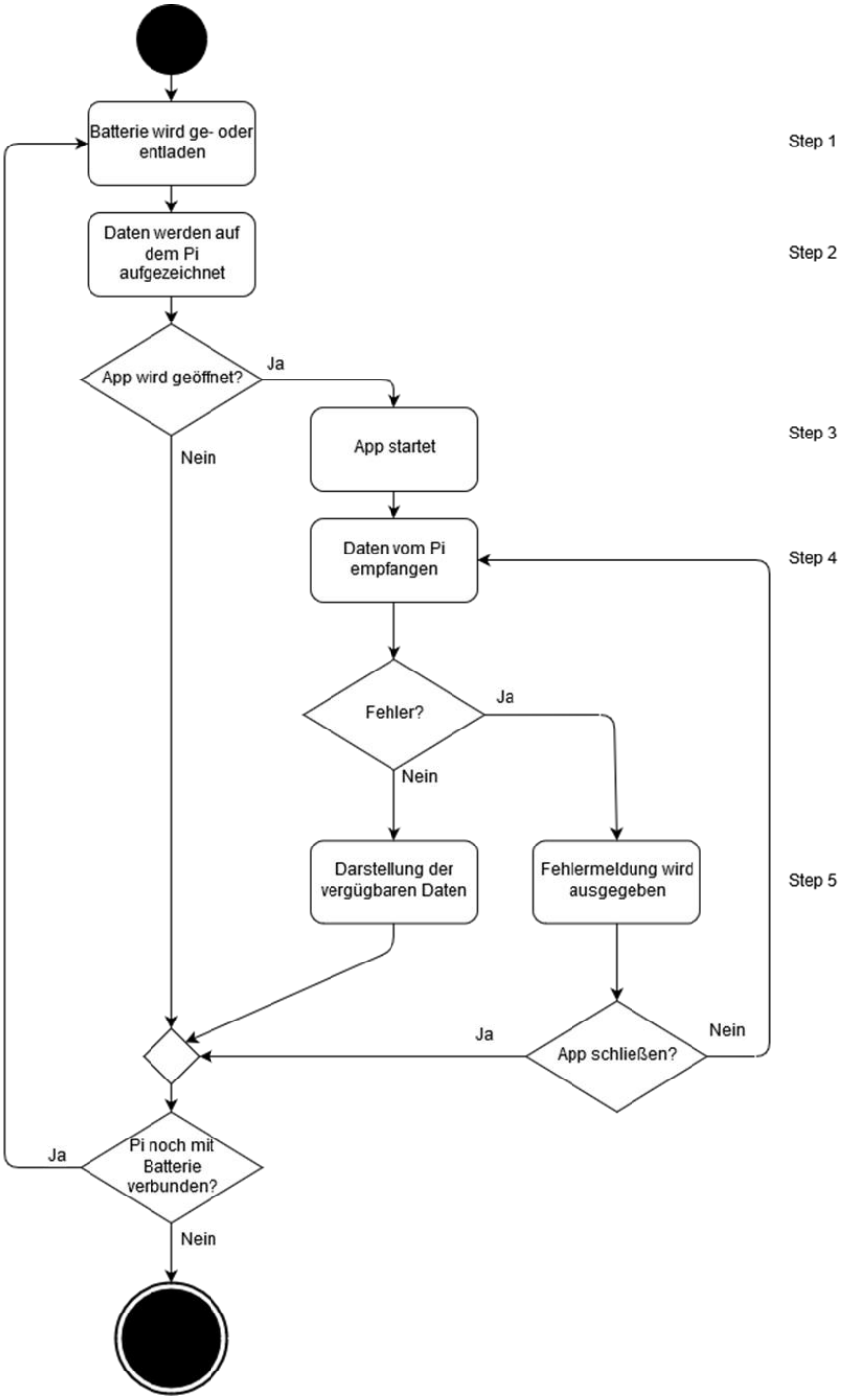
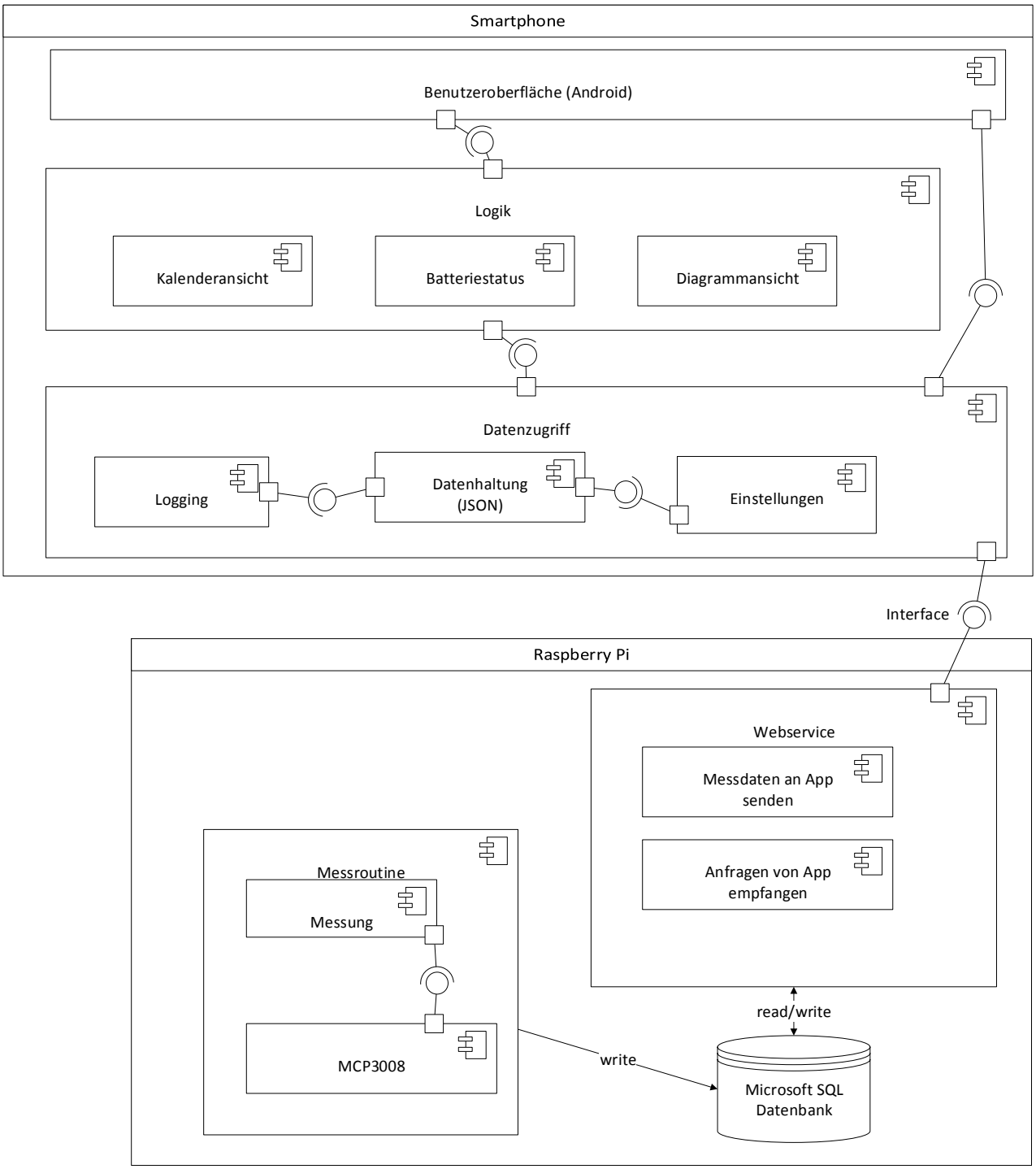


Abbildung 2 Darstellung des Workflows

1.2 Komponenten und Schnittstellen



1.3 Beschreibung der Implementierung

Detaillierte Beschreibung der notwendigen Entwicklungen/Änderungen pro Komponente zur Durchführung der gewünschten neuen Funktionen

#	Komponentendetail	Erforderliche Arbeiten
T1	Android App Smart Mini Camper	Erstellung der Ansichten. Erstellung eines Diagramms anhand von JSON Daten. Zugriff und Verwendung von Kalenderdaten.
T2	Messsystem Raspberry Pi	An den Raspberry Pi werden über einen ADC Signale von zwei Stromstärkesensoren, und einem Spannungssensor in den Messroutinen interpretiert und abgespeichert. Aus diesen Sensorwerten wird die Lade- und Verbrauchsleistung bestimmt. Messinput: - Stromstärke bei Ladung Batterie (charge_current) - Stromstärke Verbraucher Batterie (load_current) - Batteriespannung (battery_voltage) Berechnung: - Verbrauchsleistung - Ladeleistung Output (Speicherung): - Stromstärke bei Ladung Batterie (charge_current) - Stromstärke Verbraucher Batterie (load_current) - Batteriespannung (battery_voltage) - Verbrauchsleistung - Ladeleistung
T3	Webservice	Der Webservice ermöglicht den Datenaustausch zwischen Android App und der SQL Datenbank auf dem Raspberry Pi.
T4	Datenbank	Wir nutzen eine Sqlite3 Datenbank, es wird nur eine Klasse geben mit 4 Key Value Pairs

2 Android App Smart Mini Camper #T1

Die erstellte App dient dazu von einem Pi gemessene Daten darzustellen. Der Raspberry Pi misst die Ladung und Entladung einer Batterie. Die App ermöglicht die aktuellen Werte zu überprüfen und sich graphisch die Verläufe der Batteriegesamtladung, Entladung und Ladung anzeigen zu lassen.

2.1 Anforderungen

Hardwareanforderungen	Internetverbindung
Softwareanforderungen	API > 23
Entwicklungsplattform	Android Studio V 3.4

2.2 Android App - Funktionen

Die GUI wird als App mit Android Studio erstellt.
Es wird eine GUI mit 3 Aktivitäten implementiert: (API Level, Android Studio Version, Android Version Voraussetzung

- Startfenster
- Diagramm
- Kalender

Verwendete Bibliotheken:

- Kalender: `implementation 'com.squareup:android-times-square:1.6.5@aar'`
- Diagramm: `implementation 'com.jjoe64:graphview:4.2.2'`

Permissions im Android-Manifest:

```
<uses-permission android:name="android.permission.INTERNET" />
<application
    android:usesCleartextTraffic="true"
</application>
```

2.2.1 Startfenster

Im Startfenster wird der aktuelle Ladezustand der Batterie angezeigt.
Außerdem werden in regelmäßigen Abständen die Werte für den eingehenden und ausgehende Strom aktualisiert (Live-Daten)
Über einen Button kann auf die große graphische Darstellung gewechselt werden.

Vorhandene Elemente:

Element	**Umsetzung**	**Funktion**
Überschriften	Textviews	Was wird dargestellt
Trennstriche	Views	Unterteilung der Informationen
Ansichtswechsel	Button	Zum wechseln von der Start- zur Diagrammansicht
Drop-down Menü	Spinner	Einstellung verschiedener Zeiteinheiten für die Vorschau
Stromwerte	Textviews	Anzeigen des ein- und ausgehenden Stroms
Batterieladung	Progressbar	Anzeigen wie sehr die Batterie geladen ist

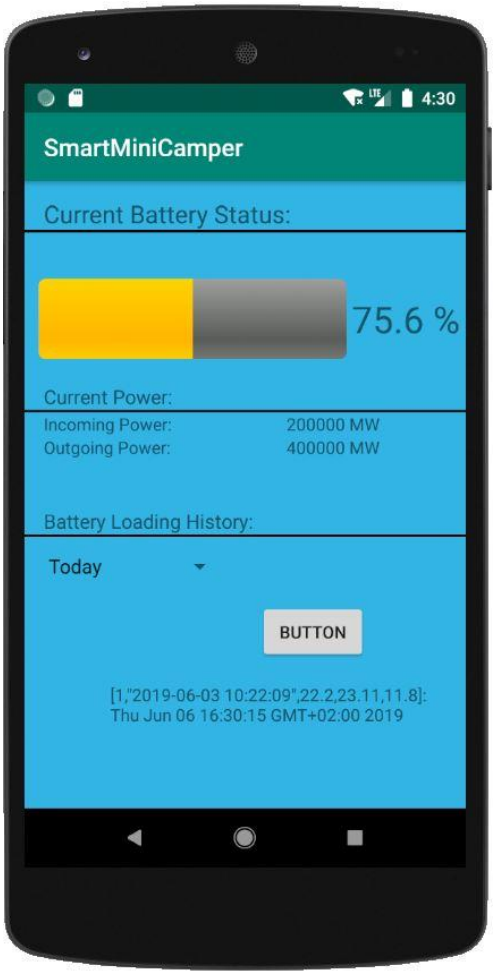


Abbildung 3 Android App Startseite

2.2.2 Graphische Darstellung

Die Graphische Darstellung bekommt von einer JSON-Dateien Werte geliefert und stellt diese in einem Diagramm dar.

Es können gewisse Zeiträume ausgewählt werden.

Über einen weiteren Button kann der Kalender geöffnet werden und zeigt die ausgewählten Daten in der Graphischen Darstellung Aktivität an.

Element	**Umsetzung**	**Funktion**
Datumauswahl	Textviews	Was wird dargestellt
Ansichtswechsel	Button, Backbutton	Button zum Wechseln zur Kalenderansicht, Backbutton zum Wechseln zur Startseite
Drop-down Menü	Spinner	Einstellung verschiedener Zeiteinheiten für das Diagramm
Diagramm	Graphview	Anzeigen des ein- und ausgehenden Stroms

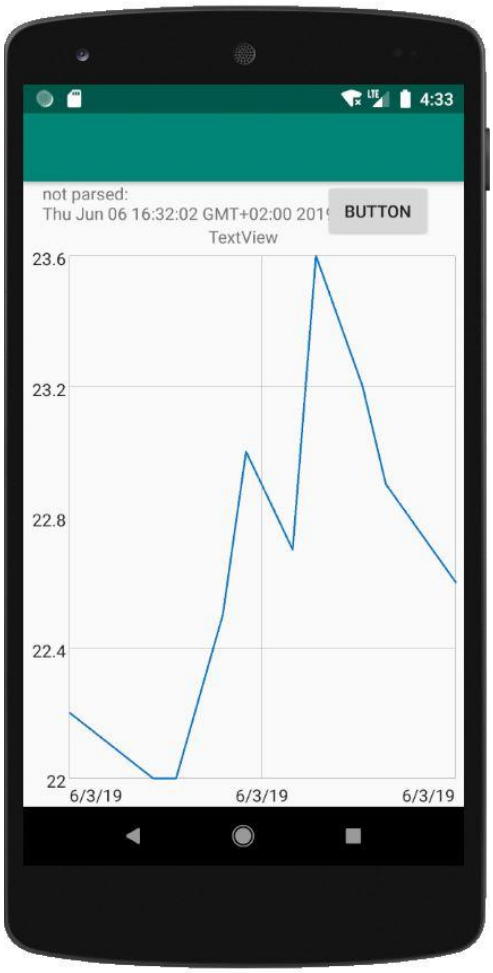


Abbildung 4 Android App Diagrammansicht

2.2.3 Kalender

Die Kalender Aktivität zeigt einen Kalender an in dem ein Datum, oder ein Zeitbereich ausgewählt werden können.

Element	**Umsetzung**	**Funktion**
Datumauswahl	Time Square Kalender	Auswahl des darzustellenden Zeitraums
Bestätigen der Auswahl	Button	Senden der gewählten Daten an die Diagrammansicht und Wechsel zu selbiger
Keine Auswahl	Backbutton	Wechsel zur Diagrammansicht ohne Daten zu senden

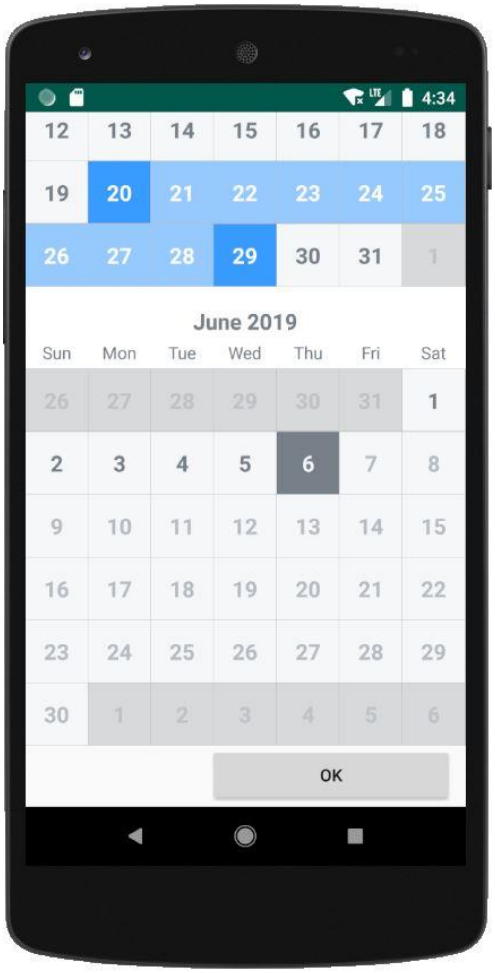


Abbildung 5 Android App Kalenderansicht

Für alle Aktivitäten wurde die Activitybar ausgeschalten und in der Startseite und der Diagrammansicht durch eine Toolbar ersetzt, da diese modularer ist.

3 Raspberry Pi – Messsystem #T2

In diesem Kapitel wird die verwendete Hardware, die Verschaltung und die Implementierung der Messroutinen erläutert.

3.1 Fritzing Diagramm

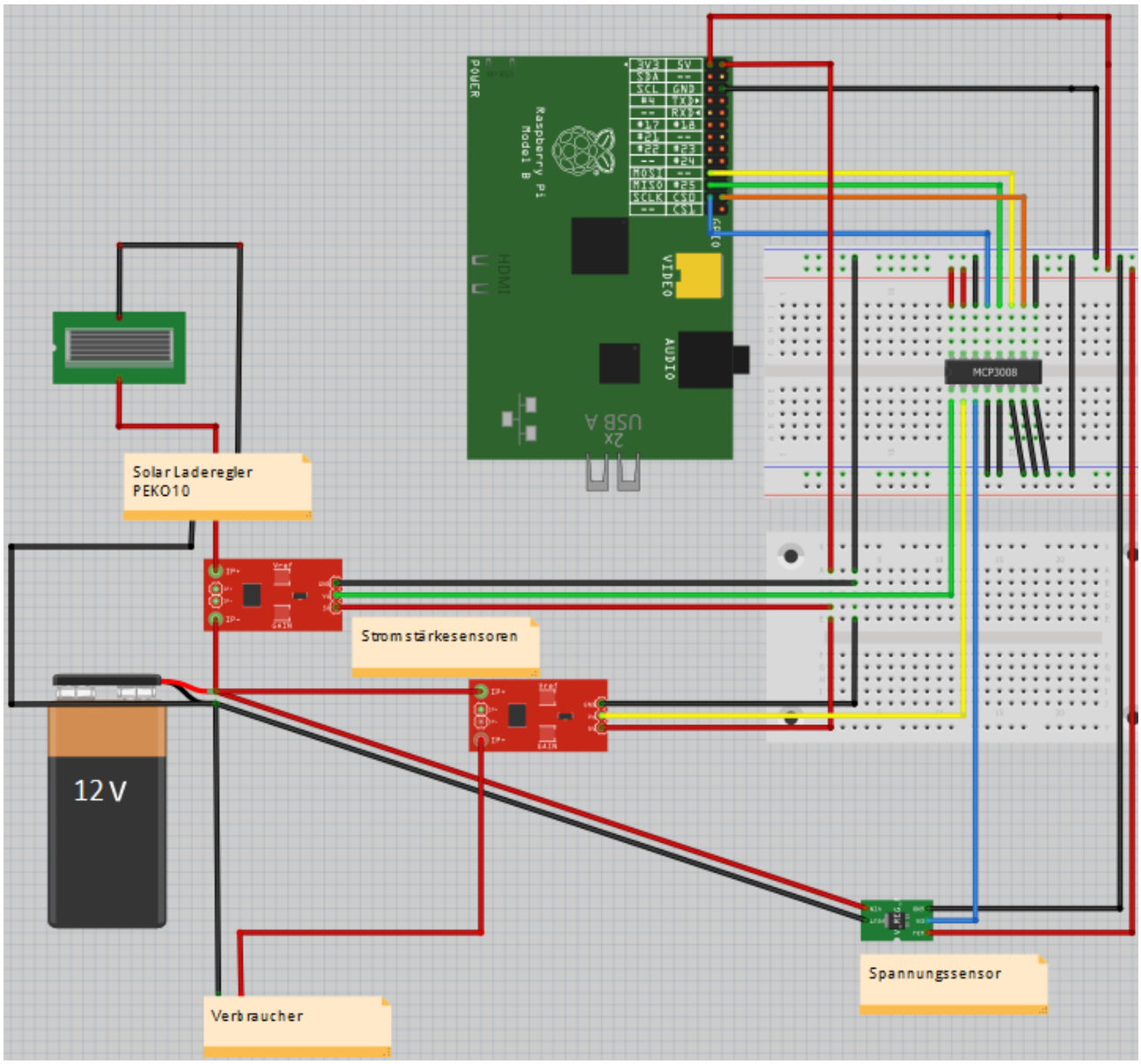


Abbildung 6 Fritzing Schaltplan Messsystem Raspberry Pi

Hinweise bei der Installation:

1. Schaltung am Raspberry Pi getrennt von der Batterie umsetzen
2. Anschluss Batterie an Laderegler: Zuerst Plus von Batterie an Plus vom Laderegler und erst danach Minus von Batterie an Minus vom Laderegler
    a. Bei richtiger Installation blinkt die Info-LED am Laderegler grün
3. Solarpanel eingepackt an den Laderegler anschließen und erst danach Panel aufklappen und positionieren
4. Verbindung der Sensoren Raspberry Pi - Ladesystem

Hinweise zum Schaltplan:

- Stromsensoren ACS712 an 5V Spannung
- ADC MCP3008 und Spannungssensor an 3.3V Spannung, sonst liegt am Raspberry Pi eine zu hohe Spannung an, welche zum Defekt des Raspberry Pi führen kann.

3.2 Verwendete Hardware

#	Name	Beschreibung	Menge
1	Solarpanel WS120SF	120 W Solarpanel zur Stromerzeugung	1
2	Solar Laderegler PEKO10	10 A PWM-Laderegler für 12 V Systeme	1
3	AGM Batterie	12 V AGM Batterie 20 Ah (max. Entladestrom 300A)	1
4	Wechselrichter	150 W Wechselrichter mit 12 V Zigarettensteckdose	1
5	Stromstärkesensor ACS712	0-30 A Input, Passive Stromstärkemessung mit dem Hall-Effekt, analoges Output-Signal	2
6	Spannungssenor	0-25 V Input, passiv gekühlte Spannungsmessung (Spannungsteiler), analoges Output-Signal	1
7	Analog-Digital-Wandler (ADC) MCP3008	8-Channel, 10-Bit ADC mit SPI Interface	1
8	Raspberry Pi 3	Einplatinencomputer für die Messung, Kommunikation und Datenbank + Mechatronik Daten	1

3.3 Einstellungen auf dem Raspberry Pi

SPI Interface anschalten:

```
1. $ sudo raspi-config
```

In den „Interfacing Options“ SPI auf „Enable“ setzen.

3.4 Beschreibung und Verwendung der Messroutinen

Um die Messung zu starten, muss main.py ausgeführt werden. Das Hauptprogramm greift auf die Datei MCP3008.py zu, welche die Verwendung des Analog-Digital-Converters MCP3008 ermöglicht. Im Hauptprogramm (main.py) sind die Channel folgendermaßen eingestellt:

- Channel 1: charge_current – Stromstärke bei Ladung der Batterie
- Channel 2: load_current – Stromstärke der Verbraucher an der Batterie
- Channel 3: battery_voltage – Spannung der Batterie

Mit dem MCP3008 können bis zu 8 Channels für den Input analoger Signale verwendet werden. Die Messdaten für Stromstärke müssen noch über einen Widerstand kalibriert werden. Derzeit wird ein Potentiometer zur Kalibrierung verwendet. Der Widerstand muss so gewählt werden, dass der ADC einen digitalen Wert von 512 ausgibt, wenn der Sensor an 0 A geschaltet ist.

4 RESTFul Webservice #T3

Für die Installation wird ein Raspberry Pi 3 mit Linux Raspbian OS und integrierter Python Distribution verwendet. Zusätzlich wird Flask und SQLite3 benötigt.

Installation von Flask und SQLite3

```
1.   $ sudo apt-get install python3-flask
2.   $ sudo apt-get install sqlite3
```

Im nächsten Schritt wird ein Ordner mit den Python Skripten RESTFul.py und die test.db angelegt.

Außerdem benötigt der Raspberry Pi ein Tool damit dieser als Hotspot für die Verbindung zur Android App dienen kann.

```
1.   $ wget -q https://git.io/voEUQ -O /tmp/raspap && bash /tmp/raspap
```

Hotspot Daten:

- SSID: raspberry Pi
- PW: ChangeMe
- IP: 10.3.141.1
- Webinterface: User: admin PW: secret

5 Datenbank #T4

Datenbankmodell:

Spannungen	
ID (PK)	Int
currentOut	Float
currentIn	Float
currentBat	Float
Date	DateTime