



Fakultät Elektrotechnik

Studiengang: Mechatronik/Fahrzeugmechatronik

PRAKTIKUMSBERICHT

Thema:

Entwicklung eines Messalgorithmus für ein
Batterie-Management-System und
Entwicklung einer Web-Visualisierung für Darstellung von Messdaten

Bearbeiter:	Oskar Engler
Matrikelnummer:	34431
Bearbeitungszeitraum:	01.10.14 bis 28.02.15
Ort, Datum der Abgabe:	Dresden, xx.3.2015
Betreuer:	Dipl. Ing. Lars Mademann
Verantwortliche. Hochschullehrer:	Prof. Dr.-Ing. Ralf Boden

Sperrvermerk

Dieser Praktikumsbericht enthält vertrauliche Informationen, die der Geheimhaltung unterliegen. Sie dürfen nur für die interne Verwendung und zur Kontrolle durch den verantwortlichen Hochschullehrer genutzt werden. Eine, auch nur teilweise, Veröffentlichung der Belegarbeit darf nur mit Zustimmung der BELECTRIC GmbH, Zweigstelle Dresden, Industriestraße 65, 01129 Dresden erfolgen.

Dresden, DATUM

Inhaltsverzeichnis

Abkürzungsverzeichnis	I
Symbolverzeichnis	IV
Tabellenverzeichnis	V
1 Einleitung	1
2 Entwicklung des BMS-Algorithmus	3
2.1 Einführung in das Batterie Management System	3
2.2 Erstellung eines BMS-Planes	5
2.3 Entwicklung der Algorithmus	7
2.4 Test Durchführung	11
2.5 Fazit	11
3 Web-Visualisierung	12
3.1 Zielstellung	12
3.2 Konzipierung	13
3.3 Website - Entwicklung	13
3.4 Die SQL-Datenbank	22
3.5 Datenvisualisierung - Charts	24
3.6 PHP	27
3.7 Test Durchführung	28
3.8 Fazit	29

Literaturverzeichnis	30
Anhang	31
Eidesstattliche Erklärung	32

Abkürzungsverzeichnis

BMS	Batterie-Management-System
EBU	Energy Buffer Unit
B&R	Bernecker + Rainer Industrie-Elektronik
ST	Strukturierter Text
SQL	Structured Query Language
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
PHP	Hypertext Preprocessor
CSV	Comma-separated values
JSON	JavaScript Object Notation
FTP	File Transfer Protocol

Abbildungsverzeichnis

2.1.1 BMS - Visualisierung [1]	4
2.2.1 Plan mit Beschriftung	5
2.2.2 Die Legende zum Plan	6
2.2.3 Verkabelung von Platinen	7
2.3.1 Erster Teil des Algorithmus	10
3.2.1 Login-Seite	13
3.2.2 Liste mit Länder und Anlagen	14
3.3.1 Login-Seite mit Bootstrap	14
3.3.2 Liste mit Amerika	16
3.3.3 Liste mit Deutschlands Kraftwerken	17
3.3.4 jQuery Skript für öffnen von Amerika Kontinent Liste . .	17
3.3.5 jQuery Leaflet Karte mit Pointer Beispielen	18
3.3.6 Die Karte im Seite mit neuen Layout	18
3.3.7 Die erste Umsetzung mit Tabellen als Beispiel	19
3.3.8 Ein Teil der Test-Dashboard ohne Navigationsleisten . .	19
3.3.9 Modulen-Menü links	21
3.4.1 Ein Teil der implementiertes SQL Quellcode	22
3.4.2 Ansicht der Metadatenbank im Adminer	23
3.5.1 dyGraphs Beispiel mit Beispielsdaten	24
3.5.2 Amstockcharts Beispiel mit Beispielsdaten	25
3.5.3 dyGraphs Beispiel-Diagramm	26
3.5.4 Amstockcharts Beispiel-Diagramm	26
3.6.1 CSV Parser für die Diagramme	28

3.6.2 Menü über PHP	28
3.7.1 Ladezeiten vor dem Reduzieren	29
3.7.2 Ladezeiten nach dem Reduzieren	29

Symbolverzeichnis

Tabellenverzeichnis

2.1	Tabelle mit Trogverbund und Reihenordnung	8
2.2	Tabelle mit Indexen	9

1 Einleitung

Die Firma Belectric GmbH wurde im 2001 gegründet und hat seinen Standort im Kolitzheim. Seit dem wurden über 1,5 GWp Solarleistung weltweit installiert. Die Belectric GmbH wurde damit zu einem Weltmarktführer im Bereich der Installation von Freiflächensolarkraftwerken. Es werden neue und innovative Technologien bei der Installation umgesetzt. Weltweit hat die Belectric über 1600 Beschäftigte, die in den Bereichen von Wartung und Anlagenbau bis hin zu Forschung und Entwicklung tätig sind.

Die Firma Adensis GmbH, mit dem Standort im Dresden, gehört zu den Entwicklungs- und Forschungsgruppen der Belectric GmbH. Sie wurde 2006 gegründet und ist seit dem ein Forschungszentrum auf dem Gebiet Photovoltaik. Über 70 Mitarbeiter sind in den Bereichen Elektrotechnik, Maschinenbau, Physik und Chemie angestellt. Einen größeren Teil der Mitarbeiter bilden Studenten und ehemalige Studenten. Zu den Aufgabenfeldern der Adensis GmbH gehören das Durchführen von Tests und Analysen, Entwicklung neuer Technologien und Produkte sowie Kraftwerksbau.

Das 20-Wöchiges Praktikum wurde in der Abteilung Kraftwerkstechnik der Firma Adensis absolviert. Das Pflichtpraktikum war in zwei Hauptgebiete geteilt. Im Oktober wurde mit einem Batterie-Management-System (BMS), welches in der Adensis entwickelt wurde, gearbeitet. Die Aufgabe bestand in der Entwicklung eines Algorithmus für das BMS. Dieser Algorithmus soll die gemessenen Daten der jeweiligen Batteriezellen in bestimmte Reihenfolgen sortieren. Damit die Daten später in richtiger Reihenfolge visualisiert werden können.

Ab November wurde die Aufgabe die Visualisierung von gesendeten Daten aus einer

Kraftwerksanlage. Dieses soll mittels einer Website, mit jeweiligen grafischen Mitteln realisiert werden. Es wurde erforderlich, dass die meisten Daten in Diagramme dargestellt werden. Das Endprodukt soll vor allem den Angestellten dienen, um die Daten eines Kraftwerkes zu analysieren.

2 Entwicklung des BMS-Algorithmus

2.1 Einführung in das Batterie Management System

Das Batterie-Management-System (BMS) ist ein System, welches in der Abteilung Embedded Systems zur Überwachung von Batteriezellen entwickelt wurde. Das BMS misst Spannung, Temperatur und Leitwert an den jeweiligen Zellen. Eine BMS-Platine kann man mit 12 Zellen verbinden und von allen den Spannungswert auslesen. Das BMS wurde in dem Batteriespeicher in Alt Daber im Brandenburg in Betrieb genommen. Es soll vor allem als Überwachungssystem für die Firma dienen, damit man die Spannungen und andere Werten kontrollieren kann. Dazu wird, für eine bessere Übersichtlichkeit und Orientierung, eine Visualisierung der Energy-Buffer-Unit (EBU) ¹ angefertigt. (siehe Abbildung 2.1.1). Die Daten der jeweiligen Batteriezellen sind in einer unpassenden Anordnung gespeichert. Dies ist der konstruktionsbedingten Verlegung des Kommunikationskabels geschuldet. Dabei ist die Aufgabe ein Algorithmus zu entwickeln, der die Daten in die richtige, von uns festgestellte, Anordnung sortiert. Damit sollen die Messwerte richtig in der Visualisierung gezeigt werden.

¹Energy-Buffer-Unit ist ein Teil des Kraftwerkssystems. Es handelt sich um einen Container, in dem die Batterien gelagert sind.

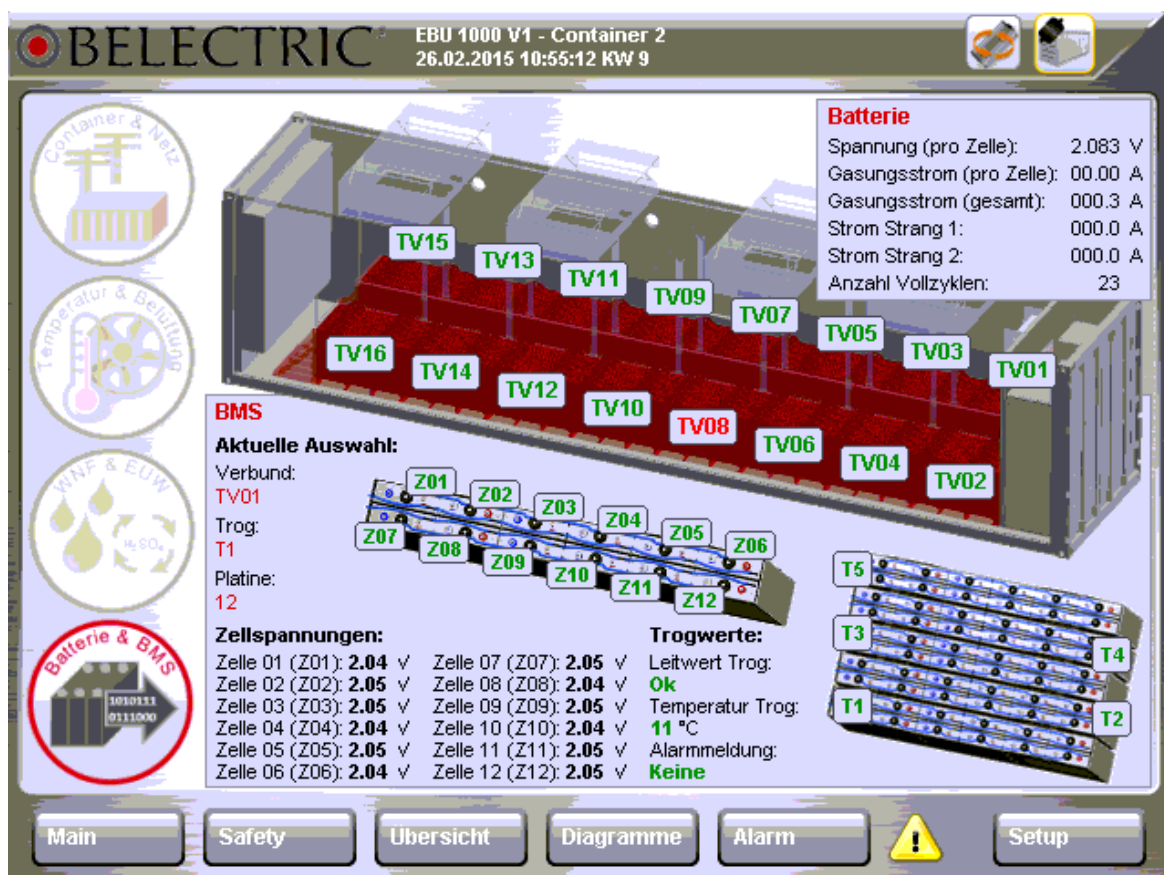


Abbildung 2.1.1: BMS - Visualisierung [1]

2.2 Erstellung eines BMS-Planes

Die Batterien sollen mit dem BMS-System belegt werden. Dafür wurde ein Plan zur Verteilung von BMS Platinen konzipiert. Es existierten mehrere Pläne/Versionen, wie die Platinen verbunden werden können. Wichtige Punkte waren, dass die Kommunikationskabel nicht länger als das festgelegte Maximum werden und dass die Batteriezellen mit dem BMS System verbunden werden. Der Plan sollte übersichtlich sein, weil es der Orientierung in dem EBU dienen soll. Der Plan wurde mit Microsoft Word erstellt. Während einer Gruppenbesprechung wurde die Nummerierung der jeweiligen Teile der EBU festgelegt. In einer EBU gibt es 16 Trog-Verbunde mit jeweils 5 Trögen. Ein Trog hat zwei Zellenblöcke mit jeweils 6 Zellen. Eine Platine kann Werte von zwei Zellenblöcken messen. Die Platine ist allerdings durch die Kabellängen beschränkt und reicht nicht über einen Trog, das heißt, dass die Platine die Werte von zwei benachbarten Zellblöcken messen kann.

In dem ersten Plan (siehe Abbildung 2.2.1 und 2.2.2) ist die Verteilung und Nummerierung zu sehen.

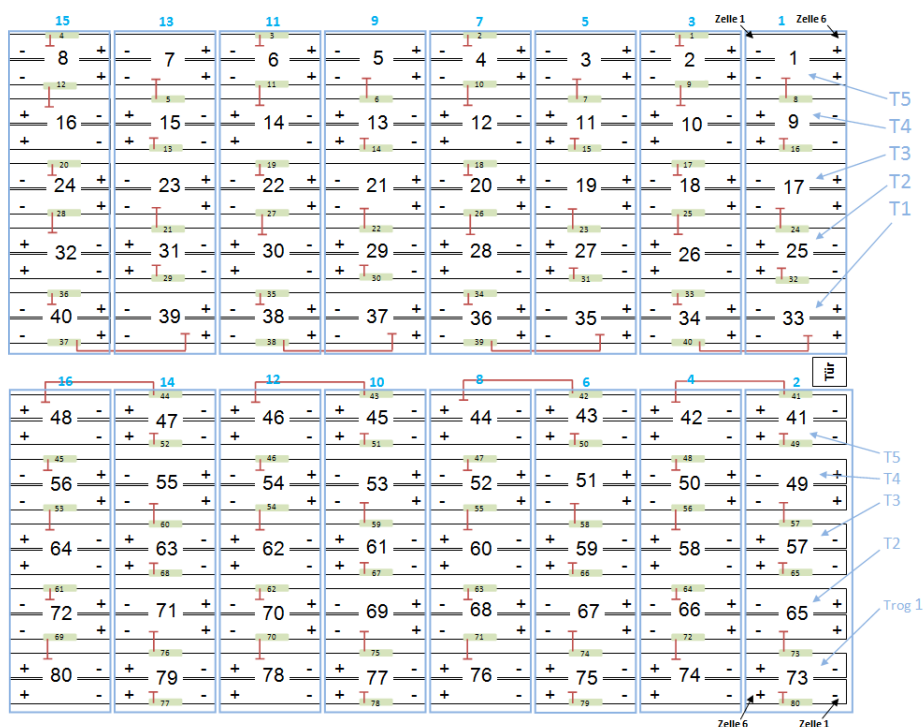


Abbildung 2.2.1: Plan mit Beschriftung

Legende:



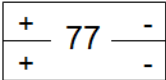



	Platine mit Nummerierung
	Temperatur- und Leitwertsensor
	Nummierierter Träger (= 2 Zellenblöcke = 12 Zellen)
	Zellenblock aus 6 Zellen
	
	Nummerierte Trogverbunde mit 5 Trögen (= 10 Zellenblöcken)

Abbildung 2.2.2: Die Legende zum Plan

In dem zweiten Plan geht es hauptsächlich um die Verkabelung zwischen den BMS-Platinen. Die Platinen wurden mit Kommunikationskabeln verbunden. Dies ist mit blau gekennzeichnet (siehe Abbildung 2.2.3).

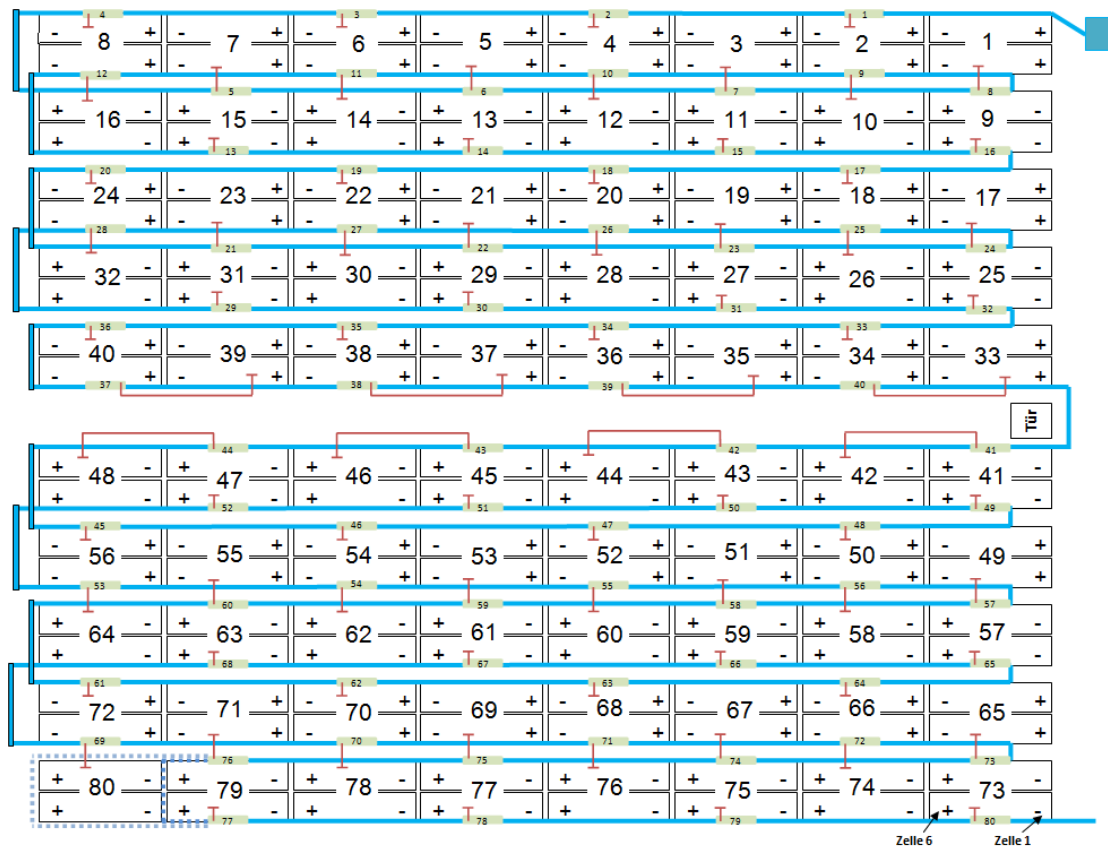


Abbildung 2.2.3: Verkabelung von Platinen

2.3 Entwicklung der Algorithmus

Zu nächst wurden Tabellen im Excel erstellt. Die erste Tabelle beinhaltet Informationen zur Verteilung von Platinen. Aus dieser Tabelle sollte man schnell die Nummer der Platine auslesen, welche zum Beispiel die Batteriezelle Nummer 5 des Troges 2 des Trog-Verbundes 11 messen würde (siehe Tabellen 2.1). Diese Tabelle dient der besseren Orientierung in einer EBU.

In der nächsten Tabelle ist der Zusammenhang zwischen gespeicherte und festgestellte Werte-Indizes beschrieben. Aus dieser Tabelle sollte man zum Beispiel auslesen, dass

Tabelle 2.1: Tabelle mit Trogverbund und Reihenordnung

I	J	K	L	M	N	O	P
Trogverbund	Trognummer	T1 bis T5	Zellenblock	Zelle	Platine	Wert von Platine	Temperatur+Leitwert von Platine
1	1	T5	1	1	1	7	8
				2		8	
				3		9	
				4		10	
				5		11	
				6		12	
			2	7	9	7	
				8		8	
				9		9	
				10		10	
				11		11	
				12		12	
	9	T4	1	1	8	1	16
				2		2	
				3		3	
				4		4	
				5		5	
				6		6	
			2	7	16	1	
				8		2	
				9		3	
				10		4	
				11		5	
				12		6	
	17	T3	1	1	17	7	24
				2		8	
				3		9	
				4		10	
				5		11	
				6		12	
			2	7	25	7	
				8		8	
				9		9	
				10		10	
				11		11	
				12		12	
	25	T2	1	1	24	1	32
				2		2	
				3		3	
				4		4	
				5		5	
				6		6	
			2	7	32	1	
				8		2	
				9		3	
				10		4	
				11		5	
				12		6	
				1		7	
				2		8	

die Platine Nummer 1 auf die Indizes 0 bis 14 die Werten vom 5. Trog des 1. und 3. Trog-Verbundes speichert. Damit es sortiert wird, muss man die Indizes 0 bis 14 mit die festgesetzte Indizes überschreiben, in diesem Fall werden es Indizes von 210 bis 215, von 60 bis 65 und drei Indizes 222, 223 und 224 (siehe Tabelle 2.2). Die Visualisierung wird die festgelegte Anordnung anzeigen.

Tabelle 2.2: Tabelle mit Indexen

1	index	TrogVerbund	Trog	so wird eingeladen	Platine		
3	210	3		0	1		
4	211			1			
5	212			2			
6	213			3			
7	214			4			
8	215			5			
9	60	1		6		1	
10	61			7			
11	62			8			
12	63			9			
13	64			10			
14	65			11			
15	222	nr.2		12			
16	223			13			
17	224			14			
18	510	7		15			
19	511			16			
20	512			17			
21	513			18			
22	514			19			

Nach dem die Fakten gesammelt wurden, kann man den Algorithmus entwickeln. Man muss die Zusammenhänge finden und davon allgemeine Regeln bilden. Diese Regeln müssen alle Werte-Indizes beschreiben. Dann sind die Regeln zusammengefasst und eine Vereinfachung je nach Möglichkeiten erfolgt. Diese Vereinfachungen wurden mit der Programmiersprache *C* geschrieben. Mit If- und Switch-Operatoren und for-Schleifen wurde die Zuordnung der jeweiligen Werte-Indizes auf die von uns fest gestellten Indizes durchgeführt (siehe Abbildung 2.3.1).

```

float r[1199]           //sortierte Werte
float d[1199]           //unsortierte Werte (die aus der BMS kommen)

/*****
                                                                    // alle 80 Platinen. Start beim [210]

    {for(a=0; a≤3;a++ && z=0; z≤45; z+15)
        {d[z] = r[210]+[a*300]
        for (i=z+1; i≤(z+11); i++ && b=1; b++)
            {if ((z+1)≤i≤(z+5))
                {d[i]=r[z+b];}
            else
                {r[i]=r[z+b+150-6];}
            }
        }
    }

/*****
                                                                    //if (z=60 || z=180 || z=300 || z=420)

    {for(a=0; a≤3;a++ && z=60; z≤420; z+120)
        {case a=0:    {r[z]=d[1110]-[165]}
        case a=1:    {r[z]=d[1110]-[165]+[6]}
        case a=2:    {r[z]=d[1110]-[165]-[30]}
        case a=3:    {r[z]=d[1110]-[165]-[24]}
                    for (i=z+1; i≤(z+11); i++ && b=1; b++)
                        {if ((z+1)≤i≤(z+5))
                            {r[i]=r[z+b];}
                        else

```

Abbildung 2.3.1: Erster Teil des Algorithmus

Die Visualisierung der BMS Daten erfolgt über die B&R Umgebung, welche die Sprache *ST* benutzt. Der Code in *C* wurde in die Sprache *ST* umgewandelt und dann in die Umgebung eingesetzt.

2.4 Test Durchführung

Nach der Anwendung der Visualisierung wurde der Algorithmus getestet. Der Test war dabei wie folgt durchgeführt: Das was die Visualisierung wiedergibt, muss mit den Werten des Messgerätes verglichen werden. Es wurde an allen Zellen getestet, ob die Werte übereinstimmen. Der Test zeigte ein positives Ergebnis, das heißt, dass die Werte auf den richtigen Indizes gespeichert wurden.

2.5 Fazit

Mit der richtigen Zuordnung der Messwerte kann man den Mitarbeitern oder Kunden zeigen, welche Spannungen an den jeweiligen Batteriezellen anliegen. Auch die Temperaturen und Leitwerte sind damit unter Aufsicht. Die ganze Funktionalität wird mittels Visualisierung überwacht. (Screenshots von Johann)

3 Web-Visualisierung

3.1 Zielstellung

Ab November entstand ein neues Projekt mit dem Ziel einer Datenvisualisierung von einem Kraftwerk. Die Visualisierung erfolgt dabei in Form von Diagrammen und Tabellen auf einer eigens eingerichteten Website. Am Anfang wurde die Zielstellung des Projektes in einer Reihe von Besprechungen erarbeitet. In diesen Besprechungen wurden die jeweiligen Aufgaben verteilt. Die Abteilung Eingebettete Systeme sollte einen Webserver mit einer SQL-Datenbank bereitstellen, auf der die Live-Daten des Batteriekraftwerkes bereitgestellt werden. Die Aufgabe war die Visualisierung der Daten. Die visualisierten Daten sollen auf einer Website dargestellt werden.

Die Daten aus dem Kraftwerk sollen übersichtlich und mittels Diagrammen visualisiert werden. Die Eignung der Daten zur Visualisierung wurde mit dem Betreuer abgestimmt. Es wurde ebenfalls besprochen, welche Mitteln für die Website Erstellung geeignet sind. Dazu gehören Bootstrap Frameworks¹, jQuery(weil: „Ohne JavaScript-Bibliothek müssen Entwickler häufig viele Zeilen Code schreiben, um den DOM-Baum (Document Object Model) zu durchlaufen und einzelne Teile der Struktur eines HTML-Dokuments zu finden. Mit jQuery steht den Entwicklern ein robuster und effizienter Selektionsmechanismus zur Verfügung, der es einfach macht, genau den Teil eines Dokuments abzurufen, den Sie untersuchen oder bearbeiten wollen“.[3, S. 7-8]) und dazugehörige Software für die Bearbeitung.

¹Bootstrap ist [...] Framework mit HTML, CSS und JS für die Entwicklung von anpassungsfähigen Projekten für das moderne Web.[2]

3.2 Konzipierung

Es soll ein Konzept für die Website entwickelt werden. Es wird ein Konzept mit dem Grafikprogramm GIMP². erstellt, wo die erste Konzepte für die Anmeldung-Seite (siehe Abbildung 3.2.1) und für die Seite, wo die jeweiligen Länder und Anlagen angezeigt werden (siehe Abbildung 3.2.2), gemacht wurden. Damit die Auswahl der jeweiligen Länder noch übersichtlicher wird, ist unser Team auf der Idee gekommen, dass wir eine große Karte in die Seite implementieren werden, wo der Benutzer aus der jeweiligen Region/Land ein Kraftwerk auswählen kann. Es wurden auch erste Konzepte für die Tabellen, Fehlermeldungen und Diagrammen gemacht.



The image shows a login page for a company named BELECTRIC. At the top, the company logo is displayed, consisting of a red circle with a white dot inside, followed by the word "BELECTRIC" in a bold, sans-serif font, with a trademark symbol (TM) to the right. Below the logo, there are two input fields. The first is labeled "Benutzername:" and the second is labeled "Kennwort:". Below these fields, there is a checkbox labeled "merken?" and a button labeled "Anmelden".

Abbildung 3.2.1: Login-Seite

3.3 Website - Entwicklung

Nächste Aufgabe war, die entwickelte Konzepte mit statischem HTML, CSS und Javascript umzusetzen. Unser Team hat abgesprochen, dass wir für die Website Entwicklung das Bootstrap Frameworks benutzen werden. Das hat sehr viele Vorteile, als beim Null zu starten. Erstens ist das Bootstrap auch für kommerzielle Benutzung kostenlos, zweitens es beschleunigt die Arbeit, weil es sozusagen schon die CSS-Programmierung

²GIMP ist ein freies, sehr leistungsfähiges Photo- und Bildbearbeitungsprogramm.[4]

liste_Kontinente				liste_Kraftwerk_Anlagen_Blocke			
Europa	(Kenndaten)	bild?	▼	Europa	(Kenndaten)	bild?	▲
Asien	(Kenndaten)	bild	▼	Alt Daber 01 (Kenndaten)	bild	▼	
Amerika	(Kenndaten)	bild	▼	Alt Daber 02 (Kenndaten)	bild	▼	
Afrika	(Kenndaten)	bild	▼	Alt Daber 03 (Kenndaten)	bild	▼	
Australien	(Kenndaten)	bild	▼	Alt Daber 04 (Kenndaten)	bild	▼	
				Asien	(Kenndaten)	bild	▼
				Amerika	(Kenndaten)	bild	▼
				Afrika	(Kenndaten)	bild	▼
				Australien	(Kenndaten)	bild	▼

(a) Vor dem Klick

(b) Nach dem Klick

Abbildung 3.2.2: Liste mit Länder und Anlagen

enthält und wir nutzen nur die vorkonfiguriertes CSS Style für die jeweiligen HTML Elementen und drittens Bootstrap bietet sehr kompatible und responsive Design, das heißt, dass die Seite auch für alle Browsers sowie Handys und Tablets optimiert ist. Für die Programmierung wurde die Software Sublime Text genutzt.

Das erste Konzept der Login-Seite war mit Bootstrap gemacht (siehe Abbildung 3.3.1), eine einfache HTML Seite mit Belectric Logo, Anmeldungsfelder und einen Button 'Sign In'.

Abbildung 3.3.1: Login-Seite mit Bootstrap

Danach wurde ein Konzept für die Auswahl-Liste mittels HTML und Bootstrap realisiert. Mit dem Bootstrap kann man schnell die Liste erstellen und auch die Breite

jedes Listenelements mit mit HTML/CSS definieren. Dafür wird das Grid-System, welches von Bootstrap entwickelt wurde verwendet. Der Grid-System stellt uns eine Zwölfspaltigen-Layout, welche die Breite der jeweiligen HTML Elementen definiert. Dazu gibt es mehrere CSS-Klassen, welche für die Breite des kleinen oder großen Monitors, Handy-Display oder Tablet zugeordnet sind. Man kann damit eine Seite responsive machen, weil man kann die Anzeige an allen verschiedenen Monitors-Größen einstellen. Es funktioniert so, dass die Breite einen erstellten **div** mit **columns** mit unterschiedlicher Größe einstellbar ist. Mit der maximalen Breite 12 wird so geschrieben: **col-lg-12** für große Monitoren bzw. **col-sm-12** für kleine Displays. Das, was in diesem **div** dargestellt wird, wird sich auch über die ganze Breite der Seite strecken. In unserem Fall, war **col-lg-8** für die Kontinente eingestellt und einen **col-lg-offset-1** für die untergeordnete Liste mit Länder (siehe Abbildung 3.3.2), damit die um einen col-1 nach rechts verschoben werden. Mit diesem Fortgang sind auch weitere Liste entstanden.

Wichtig war, dass die wichtige Informationen nach dem Anklicken eines Kraftwerkes angezeigt werden. Das wurde mit einer HTML-Tabelle umgesetzt. Man muss die Zeilen und Spalten definieren und alles in ein definiertes Feld (div) einbetten. Man muss darauf achten, dass die jeweiligen Tabellen den jeweiligen Listenelementen untergeordnet sind. Die HTML **table** ist auch von den Bootstrap stilisiert und verhält sich responsive. Damit es mit Bootstrap CSS funktioniert, muss man in den **class** derjenige Tabellen die richtigen Class-Namen für das Verhalten dieser HTML Element eingeben. Es wurden die Klassen [**class="table-responsive table table-bordered"**] benutzt (siehe Abbildung 3.3.3). Über CSS wurden noch Farben zu Unterscheidung eingestellt und Effekte wie, wenn man über die Listenelementen mit dem Cursor übergeht, bekommt der Element eine andere Farbe.

Damit die jeweiligen Listenelementen nach dem Klick immer noch auf einer Seite geöffnet werden, wurde ein jQuery Skript entwickelt. Das Ziel war, dass beim Anklicken das untergeordnete Listenelement bzw. Tabelle nach unten rutscht. Die jQuery biete eine solche Funktion: **slideToggle**, welche der Absicht entspricht. Man muss definieren, welche Elemente dieser Funktion vornehmen sollen (Siehe Abbildung 3.3.4). Deswegen müssen die HTML Elemente mit IDs oder Klassen verbunden sein.


```
<div class="container">
  <div class="col-lg-8">
    <div class="list-group list-responsive">
      <a id="kontinent1" class="list-group-item">America</a>
      <div class="col-md-offset-1">
        <div class="list-responsive list-group-item" id="america">
          <a class="col-sm-13 list-group-item">Chile</a>
          <a class="col-sm-13 list-group-item">Peru</a>
          <a class="col-sm-13 list-group-item">Canada</a>
          <a class="col-sm-13 list-group-item">USA</a>
          <a class="col-sm-13 list-group-item">Argentina</a>
        </div>
      </div>
    </div>
  </div>
</div>
```

(a) HTML - Quellcode



(b) Ansicht im Browser

Abbildung 3.3.2: Liste mit Amerika

Belectric - Analytic Tools

Europe

Germany

Anzahl: 157

table

Alt Daber 1

Baujahr: 2012	Installierte Leistung: 5644.8 [kWp]		
gesamt Erzeugung:	15.767,34 [MWh] - 2.793,25 [MWh/MWp]	Jahres Erzeugung:	5.225,87 [MWh] - 925,78 [MWh/MWp]
Monats Erzeugung:	75.619,58 [kWh] - 75.619,58 [kWh]	Tages Erzeugung:	711,94 [kWh] - 0,13 [kWh/kWp]
Status			

Alt Daber 4

Alt Daber 5

Alt Daber 6

France

Abbildung 3.3.3: Liste mit Deutschlands Kraftwerken

```
$(document).ready(function(){
    $("#k2").click(function(){
        $("#america").slideToggle("slow");
    });
});
```

Abbildung 3.3.4: jQuery Skript für öffnen von Amerika Kontinent Liste

Ein festes Design bzw. Layout wurde gewünscht um die Benutzeroberfläche angenehm zu gestalten, haben wir uns entschieden, dass die Seite mit dem **sb-admin-2** umgebaut wird. Es handelt sich um einen Template³ für Bootstrap. Es hat bereits Layout-Elemente wie eine Navigation-leiste und einem responsive Seitenmenü. Wir nutzen dies für das Umbauen der Seite. Die Login Seite ist geblieben und die anderen Seiten wurden teilweise in das Template implementiert.

Als erste wurde die Idee mit den Karten und der Auswahl von Kraftwerken aus jeweiligen Ländern mit Pointers umgesetzt. Dazu wurde eine Javascript-Bibliothek und die Open-Source Karten benötigt. Für diesen Fall war die Javascript Bibliothek **Leaflet** die Lösung. Es ist leicht anwendbar und auch responsive. Damit es funktionieren kann, braucht man eine Karten-Ebene und die Lagekoordinaten. Dann wird ein Skript zum darstellen dieser Karte geschrieben (Siehe Abbildung 3.3.5). Dieses Skript wurde in die Karte mit div definiert und damit konnte die Karte angezeigt werden. In Abbildung 3.3.6 ist die Ansicht im Browser zu erkennen.

³Eine Schablone oder Muster mit vorgemachten Design und Layout eine Seite.

```

<!-- Leaflet Maps -->
<script>
  L.Icon.Default.imagePath = 'img/leaflet';

  var map = L.map('map').setView([49.797972, 14.589346], 5); // ausgerec
  new L.TileLayer('http://a.tile.openstreetmap.de/tiles/osmde/{z}/{x}/{y}
    { maxZoom: 18, attribution: "Map data &copy; <a href='http://osm.o

  var KRA02 = L.marker([50.046102, 20.079026]).addTo(map);
  KRA02.bindPopup("<b>Krakau 02</b><br>url#");
  var KRA01 = L.marker([51.046102, 21.079026]).addTo(map);
  KRA01.bindPopup("<b>Krakau 01</b><br>url#");

  var ALD02 = L.marker([53.199631, 12.524407]).addTo(map);
  ALD02.bindPopup("<b>Alt Daber 02</b><br>url#");
  var ALD01 = L.marker([53.799631, 12.924407]).addTo(map);
  ALD01.bindPopup("<b>Alt Daber 01</b><br>url#");

```

Abbildung 3.3.5: jQuery Leaflet Karte mit Pointer Beispielen

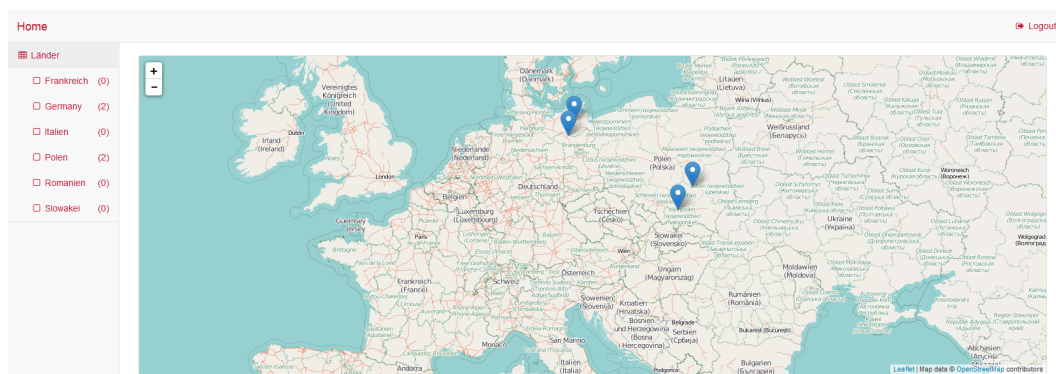


Abbildung 3.3.6: Die Karte im Seite mit neuen Layout

Nach der Kraftwerksauswahl erfolgt eine Weiterleitung zum Dashboard, wo sich alle wichtige Informationen bzw. Daten befinden. Die Dashboard soll so aussehen, dass der Benutzer die Statusmeldungen und paar Diagrammen auf den ersten Blick sieht. Das erfolgt mittels Panels, Statusleisten, Charts (Diagrammen) und andere Elementen. Einige HTML Elemente sind bereit mit sb-admin-2 CSS-stilisiert. Anfangs wurde mit Tabellen gearbeitet (siehe Abbildung 3.3.7), wo die wichtige Kenndaten dargestellt wurden. Es wurde ein neues Konzept entwickelt und nach der Einigung mit dem Betreuer wurde es in das Bootstrap umgewandelt (siehe Abbildung 3.3.8).

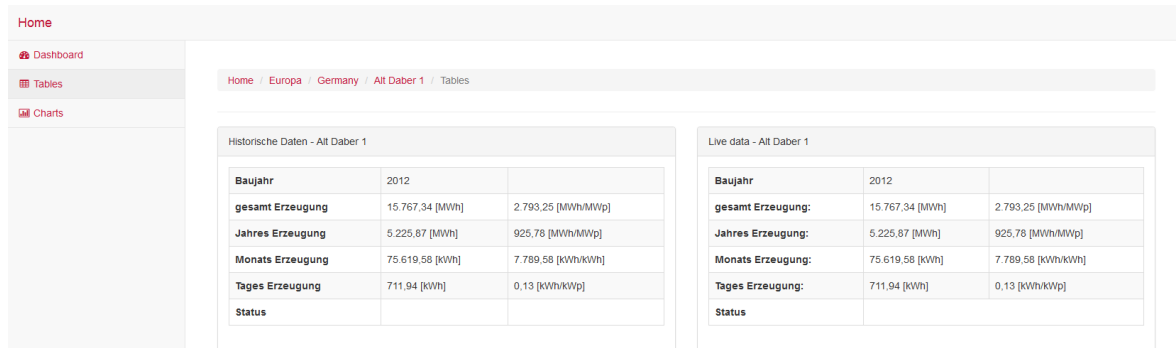
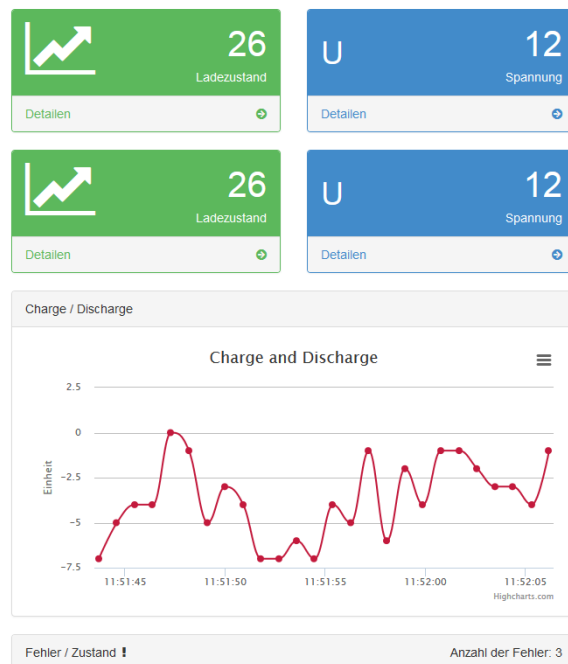


Abbildung 3.3.7: Die erste Umsetzung mit Tabellen als Beispiel

EBU master



EBU slave

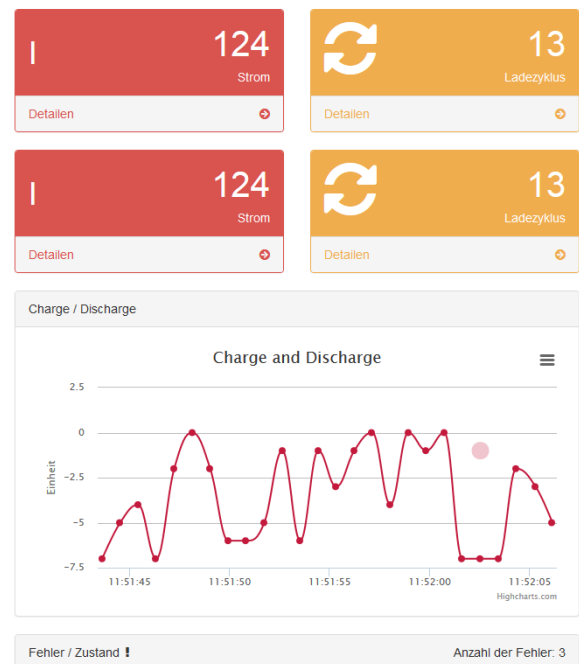


Abbildung 3.3.8: Ein Teil der Test-Dashboard ohne Navigationsleisten

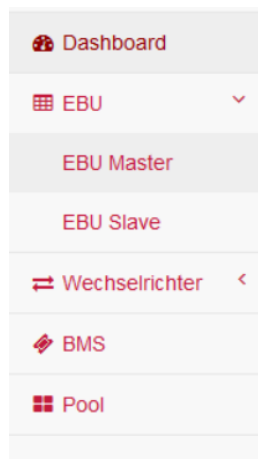
Im linken Menü sollen die jeweiligen Module des Kraftwerkes stehen. Beim Anklicken wird die Information, Status-Meldungen, Diagrammen bzw. Tabellen zu dem Modul angezeigt. Das Menü wird mit Bootstrap und mit HTML mit CSS Klassen aufgebaut. Es wurde noch die **collapse** Funktion eingebaut, das beim Anklicken einer zusammengesetzten Gruppe, zum Beispiel: EBU Master und EBU Slave als EBU, wird das nach unten rutschen und die zwei EBUs anzeigen. Damit es funktioniert, muss man zu den Menü-Elementen im HTML über die Klassen die Funktion einbinden (siehe Abbildung 3.3.9). Die Funktion selbst ist mit jQuery geschrieben und ist schon im sb-admin-2 eingeführt.

```

<!-- Left Menu -->
<div class="navbar-default sidebar" role="navigation">
  <div class="sidebar-nav navbar-collapse">
    <ul class="nav" id="side-menu">
      <li class="active">
        <a class="active" href="dashboard.html"><i class="fa fa-dashboard fa-fw"></i> Dashboard</a>
      </li>
      <li>
        <a href="#"><i class="fa fa-table fa-fw"></i> EBU<span class="fa arrow"></span></a>
        <ul class="nav nav-second-level">
          <li>
            <a href="EBUmaster.html">EBU Master</a>
          </li>
          <li>
            <a href="EBUslave.html">EBU Slave</a>
          </li>
        </ul>
      </li>
      <!-- /.nav-second-level -->
      <li>
        <a href="#"><i class="fa fa-exchange fa-fw"></i> Wechselrichter<span class="fa arrow"></span></a>
        <ul class="nav nav-second-level">
          <li>
            <a href="wr1.html">WR1</a>
          </li>
          <li>
            <a href="wr2.html">WR2</a>
          </li>
        </ul>
      </li>
      <li>
        <a href="bms.html"><i class="fa fa-ticket fa-fw"></i> BMS</a>
      </li>
      <!-- /.nav-second-level -->
      <li>
        <a href="pool.html"><i class="fa fa-th-large fa-fw"></i> Pool</a>
      </li>
    </ul>
  </div>
  <!-- /.sidebar-collapse -->
</div>
<!-- /.navbar-static-side -->

```

(a) HTML - Quellcode Linke Menü



(b) im Browser: EBU Master aktiv

Abbildung 3.3.9: Modulen-Menü links

3.4 Die SQL-Datenbank

Die PostgreSQL⁴ ist ein essenzieller Bestandteil des Projektes. In dieser Datenbank liegen die Messwerte und Kraftwerkskenndaten vor. Es gibt eine Tabelle für die Meta-Daten und dann eine Tabellen für die gesendeten Daten eines Kraftwerkes. Unser Team hat abgesprochen, wie die Meta-Datenbank⁵ aussehen wird. Die nächste Aufgabe war dann die Datenbank erstellen. Das kann auf der Kommandozeile oder einem Web-Frontend z.B. Adminer geschehen, in dem SQL-Abfragen abgesetzt werden können.(siehe Abbildung 3.4.1).

```
CREATE TABLE countries (
    country_id character varying(2) NOT NULL,
    country_name character varying(64) NOT NULL
);

ALTER TABLE public.countries OWNER TO php;

--
-- Name: continents; Type: TABLE; Schema: public; Owner: php; Tablespace:
--

CREATE TABLE continents (
    continent_id character varying(2) NOT NULL,
    continent_name character varying(64) NOT NULL
);

ALTER TABLE public.continents OWNER TO php;

--
-- Name: ebupowerplantsetups; Type: TABLE; Schema: public; Owner: php; Tablespace:
--

CREATE TABLE ebupowerplantsetups (
    setup_id integer NOT NULL,
    powerplant_id character varying(5) NOT NULL,
    module_id character varying(20) NOT NULL,
    sort_id integer NOT NULL
);

ALTER TABLE public.ebupowerplantsetups OWNER TO php;
```

Abbildung 3.4.1: Ein Teil der implementiertes SQL Quellcode

⁴PostgreSQL [...] ist ein freies, objektrelationales Datenbankmanagementsystem (ORDBMS).

⁵Meta-Datenbank dient in unseren Fall als Datenbank mit Kenndaten und allgemeine Information, wie zum Beispiel: Name, Lage, Baujahr usw. eines Kraftwerks

Dort werden die Tabellen der Datenbank mit allen Spalten und Datentypen definiert. Im unseren Fall wurden in der Datenbank Tabelle: continents, countries, countrycoordinates für die Karte, ebupowerplants mit Namen und Lagen, ebupowerplantsetups, wo die Modulen definiert werden, locations, modules, modulesname mit Module-Namen, temperature und users (siehe Abbildung 3.4.2). Die Tabellen wurden teilweise mit Daten gefüllt. Die Ausgabe von Daten an die Diagrammen wird später mit Hilfe von PHP-Funktion realisiert.

<input type="checkbox"/>	Tabelle	Motor	Collation	Datengröße	Indexgröße	Freier Bereich	Auto-Inkrement	Datensätze	Kommentar
<input type="checkbox"/>	continents	table		8 192	16 384	?	?	7	
<input type="checkbox"/>	countries	table		16 384	40 960	?	?	263	
<input type="checkbox"/>	countrycoordinates	table		8 192	16 384	?	?	6	Countries and their coordinates
<input type="checkbox"/>	database	table		0	8 192	?	?	0	Path to powerplants
<input type="checkbox"/>	ebupowerplants	table		8 192	0	?	?	0	Location and Name of powerplants
<input type="checkbox"/>	ebupowerplantsetups	table		8 192	0	?	?	0	
<input type="checkbox"/>	locations	table		8 192	16 384	?	?	2	
<input type="checkbox"/>	modules	table		8 192	0	?	?	0	
<input type="checkbox"/>	modulesname	table		8 192	0	?	?	0	Table used for sidebar
<input type="checkbox"/>	temperature	table		24 576	24 576	?	?	365	
<input type="checkbox"/>	users	table		8 192	16 384	?	?	2	
	11 insgesamt		en_US.UTF-8	114 688	139 264	0			

Abbildung 3.4.2: Ansicht der Metadatenbank im Adminer

3.5 Datenvisualisierung - Charts

Der nächste Schwerpunkt war eine Javascript oder jQuery Bibliothek zu finden, welche die Daten in geeigneten Diagrammen visualisieren und dynamisch Inhalte nachladen kann. Die Möglichkeiten wie 'Export Diagramm als Bild', Timeline⁶, direktes Datum-Auswahl, Zoom, gleichzeitige Vergleichen von mehreren Daten waren die Vergleichspunkte. Nach dem Untersuchen von mehreren freien oder auch kostenpflichtigen Bibliotheken war mit dem Betreuer abgesprochen, dass auf der Übersichtsseite und den Seiten der Module eine einfache Bibliothek: **dyGraphs** benutzt werden soll. Die **dyGraphs** sind einfach, responsive, zoom-fähig und können Daten im Format von CSV⁷ visualisieren (siehe Abbildung 3.5.1). Es war noch abgesprochen, dass es einen Link zur neuen Fenster mit Diagramm-Bibliothek **Amstockcharts** erstellen soll. Die **Amstockcharts** sind das Vergleichen mehrerer Datensätze gut geeignet. Man kann auch die Diagrammen exportieren und man hat viele Möglichkeiten, welchen Zeitabschnitt dargestellt sein soll (siehe Abbildung 3.5.2).

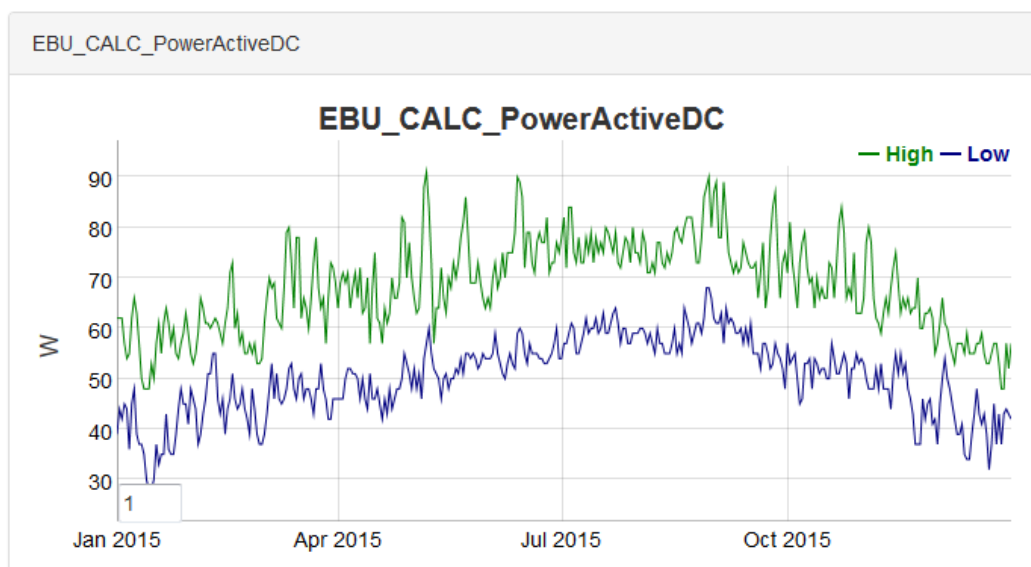


Abbildung 3.5.1: dyGraphs Beispiel mit Beispieledaten

Die Diagramme werden von Javascript-Bibliotheken bereitgestellt und müssen auch via Javascript initialisiert und konfiguriert werden. So ist erforderlich, dass die Einstel-

⁶ Auswählen einen Zeitabschnitt

⁷ Comma Separated Values (CSV) ist ein Textdateiformat, mit dem Sie Daten aus einem Datenbank- oder Tabellendokument anwendungsübergreifend austauschen können.[5]

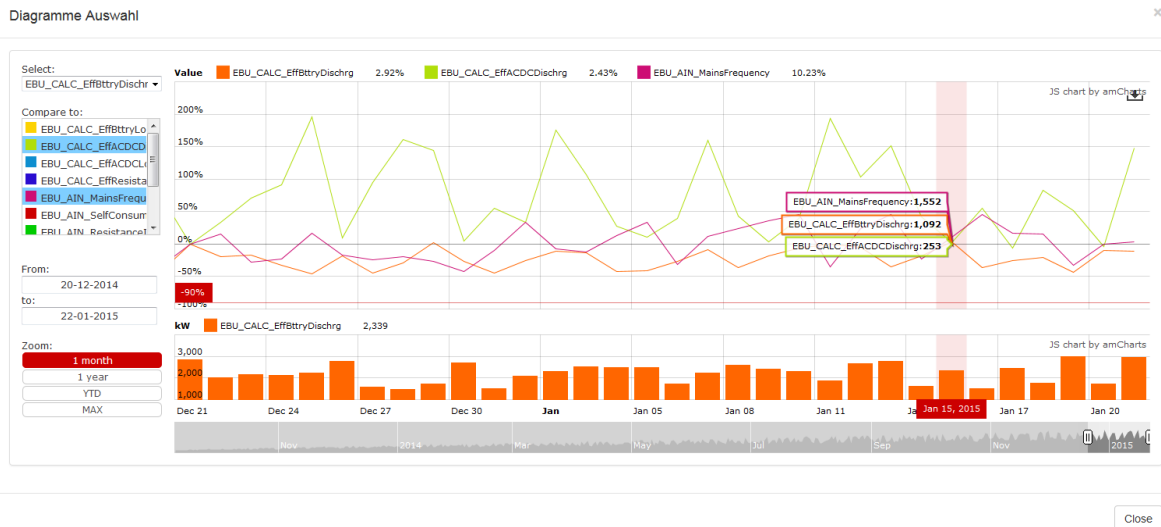


Abbildung 3.5.2: Amstockcharts Beispiel mit Beispelsdaten

lungen für Achsen, Beschriftungen und Export-Optionen mit Hilfe eines Skriptes im Quellcode definiert werden müssen. Verschiedene Bibliotheken erfordern verschiedene Syntax. Es wurden zwei unterschiedliche Bibliotheken benutzt, dass heißt es wurde mit zwei unterschiedlichen Syntaxen die Scripte geschrieben.

Bei der **dyGraphs** ist zu beachten, dass die Daten im Form von CSV sind oder in reines HTML gespeichert sind. In den Optionen kann man Titel, Achsenbeschreibung, Legende, Position der Beschriftung definieren und ob man dann im Browser gleitende Mittelwert der Diagramm einstellen kann (siehe Abbildung 3.5.3). Das Diagramm wird in einen definierten **div** angezeigt. Der **div** soll eine konkrete Größe haben. Wenn man der Diagramm responsive machen will, muss man in den CSS-Style die Größe statt in Pixels in Prozent eingeben.

Die Amcharts sind komplexere, größere und reine Javascript Charts-Bibliothek, die man unter bestimmten Bedingungen auch frei benutzen kann. Es gibt auch mehrere Möglichkeiten, wie der Diagramm dargestellt werden können. Man muss, wie beim dyGraphs, die Quelle der Daten, Beschriftungen, Achseneinstellungen und andere einstellen (siehe Abbildung 3.5.4).

```
g1 = new Dygraph(  
    document.getElementById("dyGraph1"),  
    "media/dygraphs/Leistung_AC.csv",  
    {  
        showRoller: true,  
  
        title: 'INV1_CIN_PowerActiveAC',  
        ylabel: '0,1 kW',  
        legend: 'always',  
        labelsDivStyles: { 'textAlign': 'right' }  
    }  
);
```

Abbildung 3.5.3: dyGraphs Beispiel-Diagramm

```
// create chart  
AmCharts.ready(function () {  
  
    // load some test data  
    var chartData = AmCharts.loadJSON('../amChartsImportData.php');  
  
    var chart = AmCharts.makeChart("amCharts1", {  
        type: "stock",  
        categoryAxesSettings: {  
            minPeriod: "mm"  
        },  
  
        dataSets: [{  
            color: "#547DAA",  
            fieldMappings: [{  
                fromField: "value1",  
                toField: "value1"  
            }, {  
                fromField: "value2",  
                toField: "value2"  
            }],  
            dataProvider: chartData,  
            categoryField: "category"  
        }],  
  
        panels: [{  
  
            showCategoryAxis: false,  
            title: "EBU_CALC_PowerActiveDC",  
            percentHeight: 70,
```

Abbildung 3.5.4: Amstockcharts Beispiel-Diagramm

3.6 PHP

Der nächste Schwerpunkt war die dynamische Umsetzung in PHP⁸. Es wurde benötigt, weil die Website dynamisch sein soll. Die bereits erstellte HTML Seite wurde mit PHP umgesetzt. Nach der Bearbeitung wird es als **.php** Datei gespeichert und auf den Server hingelegt. Der Server hat einen FTP-Zugang. Über den kann man die Dateien hochladen und dann die Seite über eine IP-Adresse besuchen. Die PHP Skript-Sprache ist eine serverseitige Sprache, das heißt, dass der Code auf den Server verarbeitet wird. In der PHP-Interpreter wird das Code verarbeitet und erzeugt eine Datei, welche an den Client zurückgegeben wird.

Über PHP kann man auf die Datenbank zugreifen und somit die Daten in die Diagramme einpflegen. Erstens muss man sich mit Datenbank verbinden. Das erfolgt mittels **connect** und in unseren Fall das **pg_connect**, weil wir die postgresQL benutzen. Nach der Verbindung mit SQL kann man mit den Befehlen **pg_query** die Daten aus der Datenbank auf eine Variable aufladen. Diese Variable wird dann benutzt, um den Diagramm richtigen Weg zur Daten konfigurieren.

Die Daten werden in eine Form gespeichert, welche die Diagramme nicht verarbeiten können. Das heißt, man muss die Umwandlung in die Form von JSON oder CSV durchführen. Diese sind von Diagrammen akzeptiert. Die Umwandlung in JSON kann man mit **encode_json** machen. Für die CSV Form wurde ein Parser⁹ geschrieben (siehe Abbildung 3.6.1).

Die **pg_query** wird nicht nur für die Diagrammen benutzt sondern auch für Menü mit Ländern, Menü mit Modulen eines Kraftwerkes und andere. Über diese **PHP** Funktionen kann man die Meta-Daten benutzen um dynamische Menüs bauen. Wenn zum Beispiel die Kraftwerk in Alt Daber zwei EBU, zwei Wechselrichter, zwei BMS und einen Pool hat, so wird das auch im Meta-Datenbank gespeichert und auf der Website wird die Menü in diese Konfiguration angezeigt (siehe Abbildung 3.6.2). So wird die Seite automatisiert und wenn eine neue Kraftwerk in die SQL hinzugefügt wird, dann

⁸ist eine Skriptsprache mit einer an C und Perl angelehnten Syntax, die hauptsächlich zur Erstellung dynamischer Webseiten oder Webanwendungen verwendet wird. [...] PHP zeichnet sich durch breite Datenbankunterstützung und Internet-Protokolleinbindung

⁹Ein Parser ist ein Computerprogramm, das in der Informatik für die Zerlegung und Umwandlung einer beliebigen Eingabe in ein für die Weiterverarbeitung brauchbares Format zuständig ist.

wird es auch an die Seite angezeigt.

```
{
    $db = $this->connect();
    $result = pg_query($db, $query);

    $prefix = '';
    echo "[\n";
    while ( $row = pg_fetch_assoc( $result ) ) {
        echo $prefix;
        echo '[ new Date("' . $row['date'] . '"),';
        echo $row['high'] . ',';
        echo $row['low'] . " ]" ;
        $prefix = ",\n";
    }
    echo "\n]";
}
```

Abbildung 3.6.1: CSV Parser für die Diagramme

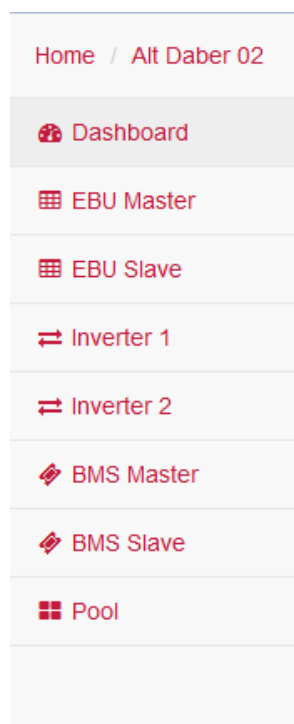


Abbildung 3.6.2: Menü über PHP

3.7 Test Durchführung

Der Test besteht daran, dass die Ladezeiten der Website-Diagrammen klein wie möglich bleiben sollen. Die ersten Ergebnisse zeigen lange Ladezeiten, die sich über 1 Sekun-

de laden. (siehe Abbildung 3.7.1). Bei mehrere Daten, wie zum Beispiel Frequenzen, Stromwerten und Spannungswerten eines Moduls, können die Ladezeiten der gesamte Website über 20 Sekunden gehen. Damit die Website sich schneller lädt, wurde die Datenmenge der eingeladenen Werten reduziert und damit die gesamte Ladezeit verringert. Die **pg_query** ermöglicht verschiedene Einstellungen, unter welchen sind auch Auswahl der Menge der Daten. Man kann einstellen, dass die **pg_query** nur jede zweite oder dritte Zahl sich nehmen soll. Damit wurden die Ladezeiten unter 1 Sekunde gebracht(siehe Abbildung 3.7.2).



Abbildung 3.7.1: Ladezeiten vor dem Reduzieren



Abbildung 3.7.2: Ladezeiten nach dem Reduzieren

3.8 Fazit

Die Website dient jetzt vor allem als Werkzeug zur Datenanalyse, welche an den Server gesendet werden. Die Benutzer können sich die Daten aus verschiedene Modulen des Kraftwerkes anzeigen lassen. Man kann auch größere Zeitspannen sowie kürzere Zeit-Intervallen einstellen und dann die Daten vergleichen.

Literaturverzeichnis

- [1] Visualisierung eines Containers. Adensis GmbH.
- [2] Otto, Mark: Bootstrap (Beschreibung). <http://holdirbootstrap.de> (Abruf am 15.03.2015).
- [3] Chaffer, Jonathan, Swedberg, Karl.: „jQuery lernen und einsetzen“. Verlag: Dpunkt.verlag, 2012.
- [4] GIMP Documentation. <http://docs.gimp.org/2.8/de/introduction.html> (Abruf am 10.03.2015).
- [5] Importieren und Exportieren von CSV-Dateien.
https://help.libreoffice.org/Calc/Importing_and_Exporting_CSV_Files/de (Abruf am 14.03.2015).

Anhang

Eidesstattliche Erklärung

Hiermit versichere ich, Oskar Engler, die vorliegende Arbeit selbständig, ohne fremde Hilfe und ohne Benutzung anderer als der von mir angegebenen Quellen angefertigt zu haben. Alle aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche gekennzeichnet. Die Arbeit wurde noch keiner Prüfungsbehörde in gleicher oder ähnlicher Form vorgelegt.

Dresden, DATUM

.....

Vorname und Name des Studenten