

Bericht zum Praktischen Studiensemester Sommersemester 2021

Von
Rudolf Beck

Inhaltsverzeichnis

| | |
|---|-----------|
| 0. Abbildungsverzeichnis..... | 2 |
| 1. Einleitung..... | 3 |
| 1.1. Gründe für die Auswahl des Unternehmens und der Praxisstelle..... | 3 |
| 1.2. Unternehmensbeschreibung | 3 |
| 1.3. Erwartungen und Ziele an das Praktikum | 4 |
| 2. Vorstellung der Praktikumeinrichtung..... | 4 |
| 2.1. Art und Ziel der Einrichtung | 4 |
| 2.2. Vorbereitung auf die einzelnen Aufgaben im Praxissemester..... | 4 |
| 2.3. Inhalte Ausbildungsplan..... | 4 |
| 3. Produktbeschreibung | 5 |
| 3.1. EcgMove4..... | 5 |
| 3.2. Software für die Verwendung des EcgMove4..... | 6 |
| 3.3. Der DataAnalyzer | 8 |
| 3.3.1. Beschreibung des Software-Programms..... | 8 |
| 3.3.2. DataAnalyzer 1.0 vs. DataAnalyzer 2.0 | 10 |
| 3.3.2.1. DataAnalyzer 1.0 – Vorteile und Nachteile | 11 |
| 3.3.2.2. DataAnalyzer 2.0 | 11 |
| 4. Architektur und Wireframe der Anwendung | 13 |
| 4.1. Architektur | 13 |
| 4.1.1. die geplante Architektur der Anwendung | 13 |
| 4.1.2. die neue Architektur des DataAnalyzers..... | 13 |
| 4.2. Konzeption (Wireframe) & Ideen für DataAnalyzer 2.0..... | 15 |
| 5. Teilweise erledigte Aufgaben über einen längeren Zeitraum | 19 |
| 5.1. Datenmodell..... | 19 |
| 5.2. Das Programmieren der Anwendung – Beschreibung einzelner Schritte und Funktionen ... | 21 |
| 5.3. Endzustand der Anwendung am Ende des Praxissemesters | 22 |
| 6. Anhang..... | 29 |

0. Abbildungsverzeichnis

| | |
|---|----|
| Abb. 1: klassische Messung des EKG-Signals über die Elektroden | 5 |
| Abb. 2: Messung des EKG-Signals mit EcgMove4 | 5 |
| Abb. 3: UnisensViewer (Darstellung der Daten durch die ausgewählte Datei) | 6 |
| Abb. 4: PDF-Bericht mit Grafiken zu den ausgewählten Parametern..... | 7 |
| Abb. 5: DataAnalyzer Hauptmenü | 8 |
| Abb. 6: Teilnehmerdaten sowie Sensorposition einer Person in Notepad++..... | 9 |
| Abb. 7 : Auswahl Batch-Mode + Teilnehmerinfos + Auswahl Parameterwerte | 9 |
| Abb. 8: Auswahl Ausgabeparameter und Auswahl der Parameterwerte..... | 10 |
| Abb. 9: Ein typisches Nutzungsszenario für die Bewertung von Daten und die Offline-Analyse | 10 |
| Abb. 10: geplante Architektur DataAnalyzer 2.0 | 13 |
| Abb. 11: neu erstellte Architektur für DataAnalyzer 2.0 | 13 |
| Abb. 12: Schaubild JSON-Daten sowie Funktionen über den Transport zwischen Websockets | 14 |
| Abb. 13: "Input/Output" im Studienmodus | 15 |
| Abb. 14: "OutputParameter": Anzeige Ausgabeparameter sowie Info-Box..... | 16 |
| Abb. 15: "Advanced Settings" am Beispiel 'Algorithm config' | 17 |
| Abb. 16: "Editing DataSets": Anzeige Info über Datenbearbeitung im Excel-Modus (links) und in der Anwendung (rechts)..... | 17 |
| Abb. 17: (Check / Run) Zusammenfassung Konfigurationen (links); Analyse Messungen (einzeln und gesamt) (rechts) | 18 |
| Abb. 18: Erfolgreiche Analyse der Messungen sowie Buttons für weitere Schritte in der Anwendung | 18 |
| Abb. 19: UML-Klassendiagramm zum DataAnalyzer 2.0 | 19 |
| Abb. 20: Endzustand "Input/Output" | 23 |
| Abb. 21: Endzustand „Input/Output“ mit Angabe letzter Messungen + Modus..... | 24 |
| Abb. 22: Anzeige Anzahl der Messungen bei Auswahl des Pfades (wird in der Abbildung nicht richtig angezeigt)..... | 24 |
| Abb. 23: Endzustand "OutputParameter" | 25 |
| Abb. 24: Custom preview Image für Ausgabebetyp in "OutputParameter" (links) und Anzeige "Preview" durch Mouse-Hover (rechts)..... | 25 |
| Abb. 25: Endzustand "Advanced Settings" am Beispiel 'Algorithm config' | 26 |
| Abb. 26: Endzustand "Editing DataSets" (ohne Mock-Daten) | 27 |
| Abb. 27: geplante Anzeige der Mock-Daten in Form einer Tabelle..... | 27 |
| Abb. 28: Endzustand "Check/Run" (Zusammenfassung) | 28 |
| Abb. 29: Anzeige Benachrichtigung über erfolgreiche Analyse der Messung (oben rechts: Anzeige des Sensors)..... | 28 |

1. Einleitung

Der Studiengang Wirtschaftsinformatik ist eine Kombination aus Wirtschaftslehre und Informatik und befasst sich mit der Planung, Entwicklung, Anwendung und Wartung betrieblicher Informationssysteme. Mit diesem Studiengang sind die Absolventen in der Lage, „die Geschäftsprozesse in Unternehmen von morgen“ mitzubestimmen und diese „durch den Einsatz von Informationstechnologie (IT) entscheidend“ zu gestalten (HKA, Wirtschaftsinformatik – Profil, Z.1-4). Im Rahmen des Studiums ist man verpflichtet, ein Praxissemester bei einem Unternehmen zu durchlaufen, um die theoretischen Fähigkeiten, die an der Hochschule gelernt wurden, das erste Mal in die Praxis umsetzen zu können. Zudem können die Studierenden ihre Fähigkeiten ausbauen und Kontakte knüpfen, um z.B. bessere Chancen bei der Bewerbung um eine Stelle für die Bachelorthesis oder für den späteren Berufsweg zu entwickeln. Die Auswahl des Unternehmens und auch der ausgewählten Praxistätigkeit obliegt dem Studenten, muss jedoch zum Studium passen.

1.1. Gründe für die Auswahl des Unternehmens und der Praxisstelle

In der heutigen Zeit sind viele Unternehmen in der IT-Branche tätig oder kommen mit dieser in Berührung. Unsere moderne digitale Welt basiert auf der Arbeit dieses Berufszweiges. Sei es bei diversen Herstellungsprozessen (z.B. in der Mobilbranche), in der Medizin (Sensoren, Diagnostikprogramme, Dokumentationsprogramme), im Finanzwesen oder sei es die Anwendung von Alltagsgegenständen und Alltagsabläufen (z.B. Smart-Home, Smartphones, etc.). Dabei gibt es IT-Unternehmen, die Kunden durch verbesserte Webseiten unterstützen und dadurch ihre Präsenz auf dem Markt optimieren. Es gibt aber auch IT-Firmen, die eine kundenspezifische Software liefern, um betriebliche Arbeitsabläufe vieler Kunden zu verbessern.

Die Berufsfelder in der IT-Welt sind vielfältig: vom App-Entwickler über Datenbankentwickler, ERP-Manager, IT-Berater und Software-Entwickler bis hin zum User Experience Designer und Web Developer. Die Bereiche Softwareentwicklung und Programmieren stellen auch die gefragtesten Jobs in Deutschland dar.

Da ein Familienmitglied ein Studium im Bereich der Medizin abgeschlossen hatte, weckte die Firma „movisens GmbH“ Interesse, da es sich mit der Anwendung von IT im Bereich der Medizin beschäftigt.

Daher sollte ausprobiert werden, ob die Richtung Softwareentwicklung für den Praktikanten eine berufliche Zukunft hat und auch das Thema Programmieren an sich begeistert.

1.2. Unternehmensbeschreibung

Movisens GmbH ist ein Unternehmen mit Sitz in Karlsruhe und bietet "weltweit führende Lösungen für das ambulante Assessment und das mobile Monitoring an" (movisens GmbH, Unternehmen, Z.1-2). Das Unternehmen wurde im Jahr 2009 als Spin-Off der Universität KIT Karlsruhe gegründet. Das Unternehmen umfasst neben 2 Geschäftsführer 12 Mitarbeiter*innen.

Das Unternehmen bietet neben Produkten und der kundenspezifischen Anpassung dieser auch Workshops fürs ambulante Assessment. Das ambulante Assessment bezeichnet die Datenerhebung physiologischer Funktionen von Personen in deren Alltag. Mithilfe computergestützter Methoden können Auskünfte über das momentane Befinden und Verhalten (Selbstberichte), Verhaltensmaße, Bewegungsverhalten und physiologische Messwerte aufgezeichnet werden. (Wikipedia – Ambulantes Assessment – Definition, Z.1-2). Zur Anwendung kommt es bei Risikopatienten, v.a. mit Herz-Kreislauf-Erkrankungen, und z.B. im Bereich der Psychophysiologie, um zu untersuchen, wie sich verändernde Alltagssituationen auf die Psyche und die Gesundheit von Patienten auswirken.

1.3. Erwartungen und Ziele an das Praktikum

Das Praktikum soll einen Einblick in die Welt eines Softwareentwicklers und in den Bereich Medizintechnik gewähren. Kern dieses Praktikums ist es, erste Erfahrungen zu sammeln, um damit festlegen zu können, in welche Richtung und in welcher Branche man in der Zukunft arbeiten möchte. Dabei spielt es kaum eine Rolle, ob am Ende des Praktikums die Anwendung fertig erstellt ist oder nicht.

2. Vorstellung der Praktikumseinrichtung

2.1. Art und Ziel der Einrichtung

Vor Praktikumsbeginn war geplant, dass der Praktikant von Anfang an beim Unternehmen vor Ort im Team arbeiten sollte. Jedoch konnte dies aufgrund aktueller Corona-Auflagen nicht umgesetzt werden. Daher wurde die Möglichkeit erwogen, dass der Praktikant seine Tätigkeiten von Zuhause aus erledigen kann und nur bei Bedarf vor Ort erscheint bzw. sobald es die Corona-Verordnungen zugelassen hätten. Der Großteil der Kommunikation mit dem Praktikumsbetrieb lief über die Kommunikationsplattform „MS Teams“ oder via Email.

Der Praktikant sollte nach einer gewissen Einarbeitungszeit, sich selbstständig am Entwickeln der Software versuchen und die gestellten Aufgaben erfüllen. Gearbeitet wurde mit Programmierwerkzeugen, wie z.B. VS Code und mit Gestaltungswerkzeugen fürs Wireframe, wie z.B. Pencil Project.

2.2. Vorbereitung auf die einzelnen Aufgaben im Praxissemester

Am Anfang des Praxissemesters wurde ein Demo-Kit des Produktes EcgMove4 zur Verfügung gestellt, um das Produkt näher zu erforschen und sich mit der dazugehörigen Software, dem DataAnalyzer, genauer zu beschäftigen. In den ersten Wochen des Praxissemesters wurde begonnen, sich mit den Programmierwerkzeugen, Libraries sowie Skriptsprachen vertraut zu machen. Dank spezieller Literatur und Tutorials, die vom Betreuer zur Verfügung gestellt wurden, sollte der Praktikant in der Lage sein, die Inhalte dieser Themen besser verstehen und später für die Aufgaben anwenden zu können.

2.3. Inhalte Ausbildungsplan

Der Praktikumsbetreuer der Firma „movisens“ legte in Absprache mit dem Studenten folgenden Ausbildungsplan vor:

- Selbstständiges Kennenlernen des Produktportfolios (die Sensorsysteme und die Software zur Datenauswertung und –visualisierung)
- Auseinandersetzung mit den benötigten Entwicklungswerkzeugen (u.a. mit git, gitlab, SourceTree und IntelliJ sowie VS Code)
- Auseinandersetzung mit den verwendeten Programmiersprachen/Libraries (wie JS/TS, React, ANT Design, Java, JavaFx, WebView)
- Einarbeitung in die bestehende Codebasis
- Erstellung eines Wireframes
- Erstellung eines Datenmodells
- Programmierung der Anwendung (Endprodukt)

3. Produktbeschreibung

3.1. EcgMove4

Das Produkt EcgMove4 ist ein EKG- und Aktivitätssensor der 4. Generation. Mit dem Produkt soll die Ermittlung des EKG-Signals „in Kombination mit körperlicher Aktivität, der Langzeitmessungen ohne störende Kabel ermöglicht“ (movisens – Produkte – EcgMove4) werden. Damit „ermöglicht es in wenigen Schritten Rückschlüsse auf das Verhalten, die Aktivität, die Funktionsfähigkeit des Herzens sowie des vegetativen Nervensystems zu ziehen“. Ein besonderer Vorteil des Produkts ist es, dass die Rohdaten nicht über einen Zeitraum von bis zu 2 Wochen verloren gehen & beliebig für andere Messungen weiterverarbeitet werden können.

Das EKG (= Elektrokardiogramm) ist eine Untersuchungsmethode, bei der die Funktion des Herzens gemessen wird und somit Rückschlüsse über die Herzfunktion und die Gesundheit des Herzens zu ziehen. Das EKG-Signal kann über 2 Arten gemessen werden: Klassisch über Elektroden mit einem EKG-Gerät oder mit einem Sensor (EcgMove4).



Abb. 1: klassische Messung des EKG-Signals über die Elektroden

Wird das EKG-Signal mit den Elektroden (Abb.1) gemessen, dann werden insgesamt 10 Elektroden am Körper befestigt: 6 Elektroden auf der Brust und jeweils eine Elektrode am Arm und an den Waden. Dabei sind die Elektroden über einen Kabel mit dem EKG-Gerät verbunden. Mit den Elektroden können die Spannungsänderungen am Herzen, die jeweils durch das Zusammenziehen der Herzmuskeln ausgelöst werden, abgeleitet werden. Das EKG-Gerät verstärkt die sehr schwachen Signale und stellt sie als Kurve dar, auf einem Monitor oder ausgedruckt auf Papier. (apothekenumschau – EKG-kurz erklärt, Z. 1-11). Diese klassische Mess-Methode hat den Nachteil, dass bei Langzeitmessungen die diversen Kabel die untersuchte Person in ihren Bewegungen und im Alltag einschränkt.



Abb. 2: Messung des EKG-Signals mit EcgMove4

Wird das EKG-Signal mithilfe eines Sensors (EcgMove4) gemessen, dann wird der Sensor auf den Klebeelektroden geklebt und der Brustgürtel mit dem geklebten Sensor wird um den Brustbereich

herum befestigt (Abb.2). Mit diesem Sensor können Langzeitmessungen in Kombination mit alltagsnahen Bedingungen ohne Kabel ermöglicht werden. Für den Einsatz in wissenschaftlichen Studien ist der Sensor besser geeignet als die Befestigung mehrerer Elektroden am Körper, da so ein hoher Tragekomfort ermöglicht wird und somit zu einer verbesserten Compliance bei den Probanden sowie zu einer höheren Datenqualität beiträgt. Dadurch führt es auch zu einem verringerten Aufwand bei der Durchführung einer Studie und auch zu geringeren Kosten (movisens – EcgMove4, Z. 28-30). Ein weiterer Vorteil dieses Sensors ist, dass die gemessenen Daten mithilfe der Software DataAnalyzer weiter gestaltet werden können, um unterschiedliche Ausgabeparameter wie Schritte, Herzfrequenz, etc. in Form eines Berichts darstellen zu können.

3.2. Software für die Verwendung des EcgMove4

Für die Anwendung des Produkts EcgMove4 müssen 3 Softwareprogramme heruntergeladen werden: SensorManager, UnisensViewer und DataAnalyzer.

Mit dem Software-Programm SensorManager werden die Sensordaten des EcgMoves erfasst. Mit dieser Software ist man in der Lage, „die fertigen Messungen vom Sensor über eine USB-Verbindung auf den PC“ (movisens Docs, DataAnalyzer – Handling, Z.5-7) zu übertragen. Die Speicherung solcher Sensor-Daten kann durch die Erfassung neuer Sensor-Daten ersetzt werden

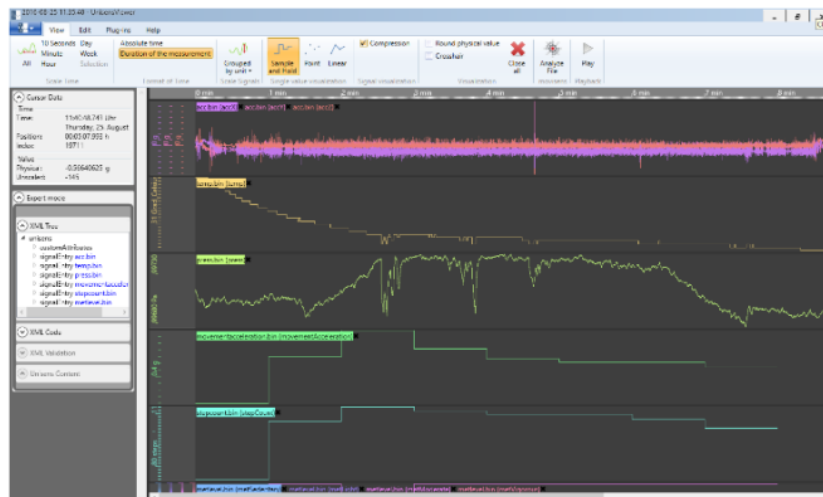


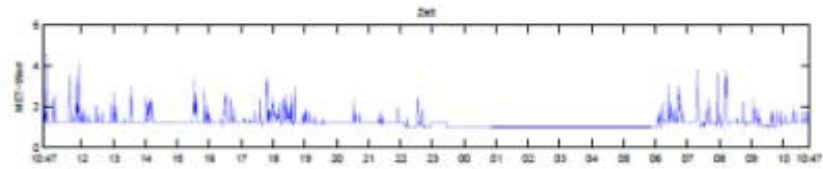
Abb. 3: UnisensViewer (Darstellung der Daten durch die ausgewählte Datei)

Mit dem Softwareprogramm UnisensViewer (Abb.3) können die vom SensorManager erfassten und gespeicherten Daten im Rohdatenformat (unisens) anhand verschiedener Grafiken angesehen und bearbeitet werden. Dabei können anhand erstellter Grafiken die Signalqualität geprüft, Messungen auf den gewünschten Zeitpunkt zugeschnitten werden, Marker gesetzt und eventuell aufgetretene Artefakte kommentiert oder entfernt werden. (movisens Docs, DataAnalyzer – Viewing Measurement Data, Z.5-7).

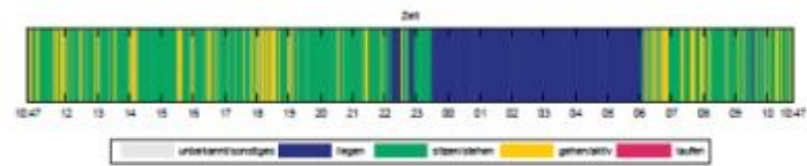
Mit der Analyse-Software DataAnalyzer, dem eigentlichen Schwerpunkt des Praxissemesters, sollen die erfassten Daten, ohne größeren Aufwand „Ausgabeparameter wie Herzfrequenz, Herzratenvariabilität, Aktivitätsklassen, Schritte, Energieumsatz und Metabolische Äquivalente (MET)“ berechnet werden und somit „aussagekräftige Berichte (PDF)“ (movisens – Produkte – EcgMove4 – Z.51-53) erzeugt werden können (Abb.4).

Mittwoch, 09.10.2013

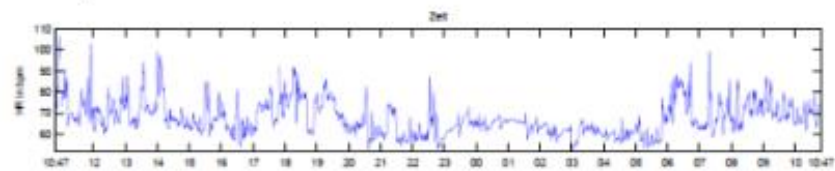
MET Level



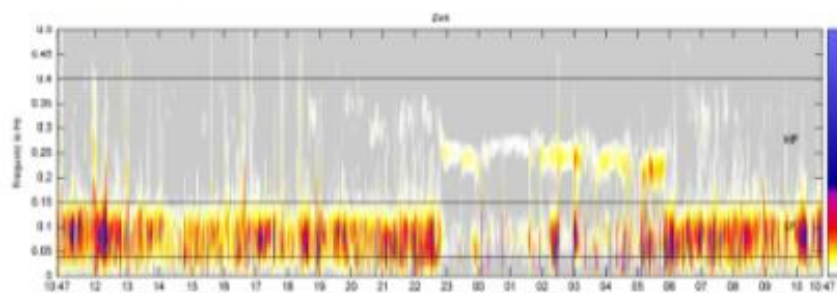
Aktivitätsklassen, Verlauf



Herzfrequenz



HRV-Spektrogramm



movisens GmbH - www.movisens.com

Abb. 4: PDF-Bericht mit Grafiken zu den ausgewählten Parametern

3.3. Der DataAnalyzer

3.3.1. Beschreibung des Software-Programms

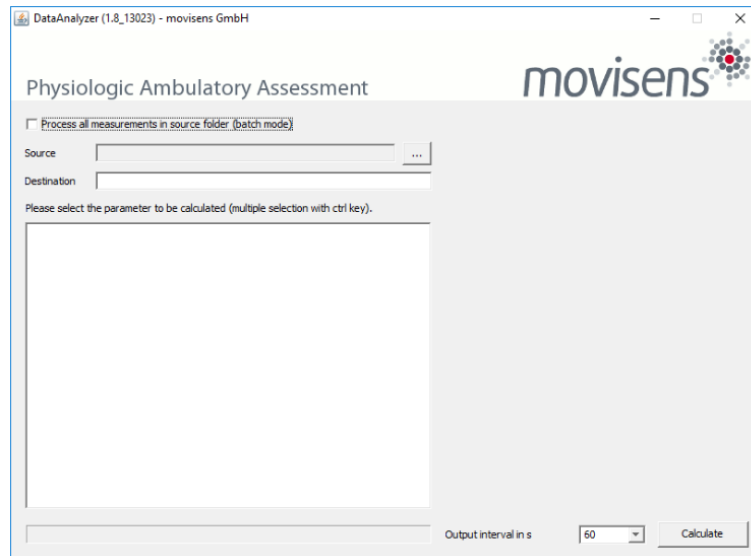
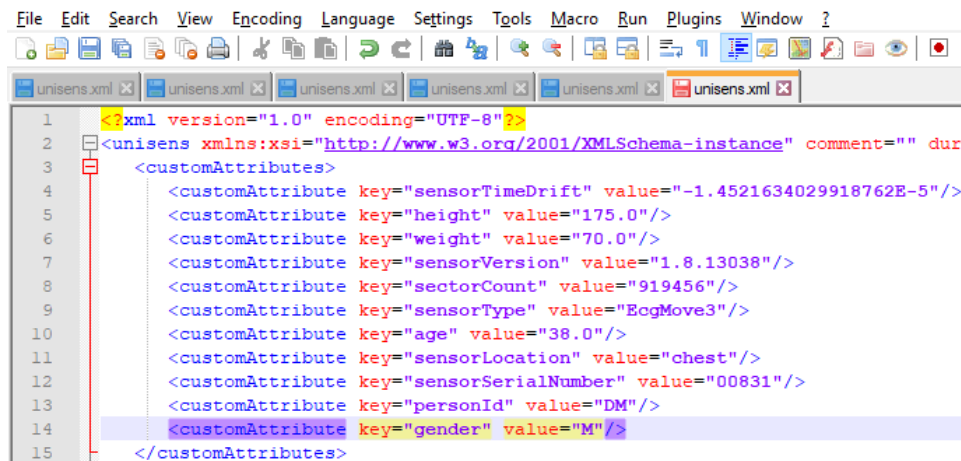


Abb. 5: DataAnalyzer Hauptmenü

Der DataAnalyzer hat eine sehr einfache Benutzeroberfläche (Abb.5). Im Input-Feld (Quelle) kann ausgewählt werden, welche Messung für die Analyse benötigt wird und im öffnenden Dialog wird die betreffende unisens.xml-Datei für die Messung ausgewählt. Nach der Eingabe in das Input-Feld wird automatisch ein Ordner erstellt, wo alle berechneten Ergebnisse gespeichert werden. In der untenstehenden Tabelle kann aus mindestens einem Ausgabeparameter und der dazugehörigen Darstellungsform (z.B. PDF, Excel, TimeSeries) ausgewählt werden. Welche Ausgabeparameter angegeben werden können, hängt von den „lizenzierten Modulen“ und in den „Messungen enthaltenen Daten“ (movisens Docs, Sensor Software, DataAnalyzer-Handling-Select output parameters, Z.2-3) ab. Sollen mehrere Ausgabeparameter ausgewählt werden, wird dies durch Betätigen der Strg-Taste erreicht. Oberhalb des Input-Feldes gibt es auch noch die Möglichkeit, den Studienmodus auszuwählen, d.h. es werden alle Daten, die mit dem Sensorwerkzeug aufgezeichnet wurden, mit den ausgewählten Parametern zusammengerechnet.

Wenn bestimmte Parameter ausgewählt werden, erscheint auf der rechten Seite eine Liste mit den Teilnehmerdaten und die Position des Sensors. In der Liste könnte dabei die Aufforderung stehen, die Parameter zu dem ausgewählten Ausgabeparameter zu überprüfen. Sollten solche Daten bearbeitet werden, gibt es 2 Möglichkeiten: entweder durch Änderung der Datenwerte in Notepad++ oder durch Editierung der Datenwerte in einer Dropdown-Box in der Anwendung.



```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <unisens xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" comment="" dur
3  <customAttributes>
4      <customAttribute key="sensorTimeDrift" value="-1.4521634029918762E-5"/>
5      <customAttribute key="height" value="175.0"/>
6      <customAttribute key="weight" value="70.0"/>
7      <customAttribute key="sensorVersion" value="1.8.13038"/>
8      <customAttribute key="sectorCount" value="919456"/>
9      <customAttribute key="sensorType" value="EcgMove3"/>
10     <customAttribute key="age" value="38.0"/>
11     <customAttribute key="sensorLocation" value="chest"/>
12     <customAttribute key="sensorSerialNumber" value="00831"/>
13     <customAttribute key="personId" value="DM"/>
14     <customAttribute key="gender" value="M"/>
15 </customAttributes>
  
```

Abb. 6: Teilnehmerdaten sowie Sensorposition einer Person in Notepad++

Wird die unisens.xml-Datei in Notepad++ geöffnet (Abb.6), dann wird eine Auflistung aller Teilnehmerinformationen wie Alter (age), Gewicht (weight), Höhe (height), usw. sowie die Sensorposition angezeigt. Sollen bestimmte Teilnehmerdaten geändert werden, dann werden bei der Property „value“ die Werte zwischen den Anführungszeichen geändert. Sind die Werte bspw. vom Gewicht oder der Größe falsch, gibt es die Möglichkeit, jeweils die richtigen Werte anzugeben und danach die Datei zu speichern. Im DataAnalyzer werden die korrigierten Angaben angezeigt, wenn dieselbe Unisens-Datei geöffnet wird. Sollen die Teilnehmerdaten über die Dropdown-Box geöffnet werden (Abb.7), dann erscheint ein zusätzliches Fenster, wo alle Teilnehmerinfos eingetragen und überprüft werden können, das ist nur dann der Fall, wenn einige Informationen dazu fehlen sollten.

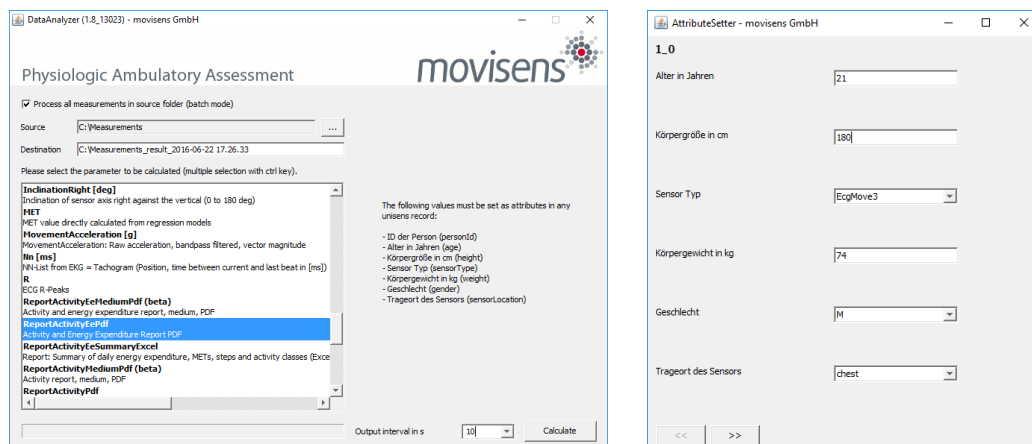


Abb. 7 : Auswahl Batch-Mode + Teilnehmerinfos + Auswahl Parameterwerte

Mit dem „Output Interval“ kann festgelegt werden, in welchem Zeitintervall die Ergebnisse angezeigt werden sollen (also in 30-Sekunden-, 60-Sekunden-, 120-Sekunden- oder in 600-Sekunden-Schritten).

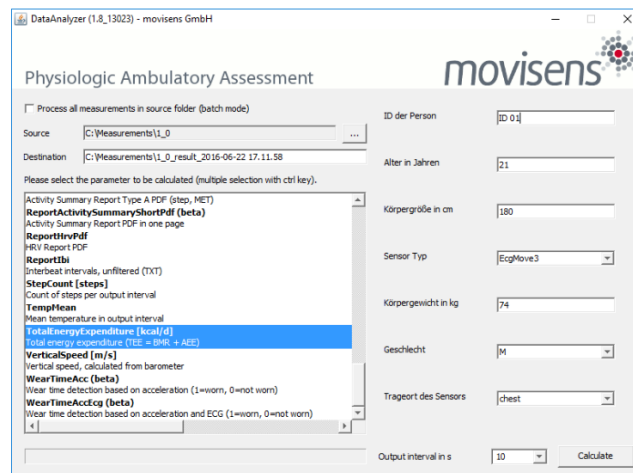


Abb. 8: Auswahl Ausgabeparameter und Auswahl der Parameterwerte

Eine Möglichkeit, nicht nur einen Datensatz zu messen, sondern mehr Daten in die Analyse einzubeziehen (z. B. die Daten einer ganzen Studie), ist oben links den Haken für den Batch-Modus zu setzen. Dabei wird der Ordner für die Quelle ausgewählt, der alle Messungen enthält.

Wurden alle Konfigurationen in der Anwendung gemacht, dann wird der Button “Berechnen” betätigt und in der Leiste (unten links) werden die Daten und die dazugehörigen Konfigurationen analysiert. Sobald die Analyse mit Hilfe der Software abgeschlossen ist, wird im separaten Fenster angezeigt, welche Schritte als Nächstes gemacht werden können: Zum Zielpfad gehen, die Ergebnisse betrachten oder zum Hauptmenü der Anwendung zurückkehren. Hier unten wird nochmal ein Nutzungsszenario vom Vorbereiten des Sensors bis hin zur Analyse der Messdaten mit dem DataAnalyzer angezeigt (Abb.9):

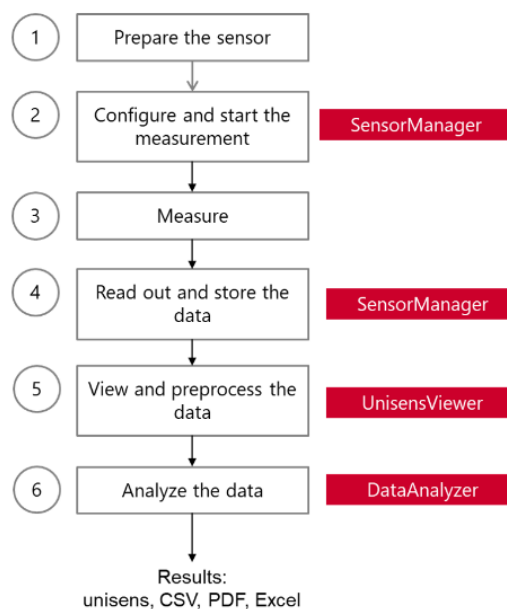


Abb. 9: Ein typisches Nutzungsszenario für die Bewertung von Daten und die Offline-Analyse

3.3.2. DataAnalyzer 1.0 vs. DataAnalyzer 2.0

Die aktuell bestehende Anwendung (DataAnalyzer 1.0) gibt es seit mehreren Jahren auf dem Markt. Jedoch haben sich die Kundenanforderungen verändert und es haben sich neue Technologien sowie Algorithmen etabliert. Daher gab es von der Unternehmensseite und der Kundenseite die

Entscheidung, den DataAnalyzer weiterzuentwickeln, um die neuen Anforderungen zu erfüllen sowie den Einbau neuer Algorithmen zu ermöglichen. Außerdem soll die neue Version des DataAnalyzers mit einer neuen Benutzeroberfläche punkten und die einzelnen Konfigurationen auf mehrere Seiten aufgeteilt werden, um die Benutzerfreundlichkeit sowie die Kreativität bei der Auswahl unterschiedlicher Konfigurationen zu erhöhen.

3.3.2.1. DataAnalyzer 1.0 – Vorteile und Nachteile

Vorteile:

Die aktuelle Version des DataAnalyzers hat eine einfache und schlichte Benutzeroberfläche, und kommt mit einer geringen Anzahl an benötigten Programmseiten aus. Es gibt eine Tabelle, wo die Ausgabeparameter ausgewählt werden können, wenn der Input mit den Rohdaten schon ausgewählt wurde. Mit dem eingeschalteten Expertenmodus gibt es die Möglichkeit, den Output-Pfad für die Input-Daten selbst zu generieren. Im Batch-Mode können alle Input-Daten in einem Ordner zusammengelegt werden. In der Dropdown-Box können die Teilnehmerdaten sowie die Sensorposition eingestellt werden, wodurch in Sachen Konfigurationsmöglichkeiten kaum Grenzen gesetzt sind.

Nachteile:

Jedoch beinhaltet der aktuelle DataAnalyzer auch zahlreiche Nachteile: die Benutzeroberfläche des DataAnalyzers wirkt veraltet. Damals lag das Augenmerk mehr auf die Funktionalität als auf der Gestaltung. Ein weiterer unschöner Aspekt am DataAnalyzer ist es auch, dass die einzelnen Konfigurationsschritte (wie z.B. die Auswahl des Input-Feldes, der Studienmodus, die Auswahl der Ausgabeparameter, ...) auf einer Programmseite komprimiert sind. Zwar können im "Batch Mode" (Studienmodus) die Parameter auf der anderen Seite angegeben werden, jedoch fehlt es an weiteren Informationen (z.B. die Übersicht der Daten, die Zusammenfassung der Auswahl bestimmter Ausgabeparameter, weitere Konfigurationen, ...), um damit etwas anfangen zu können. Außerdem ist die Progress-Bar zu ungenau und liefert keine zuverlässige Angabe über die Dauer der Analyse.

3.3.2.2. DataAnalyzer 2.0

Die neue Version des DataAnalyzers soll die von den Kunden und den Forschern angegebenen Anforderungen erfüllen:

1. Gefordert wird eine Beschreibung zu jedem Ausgabeparameter, um zu wissen, welche Funktionen die einzelnen Ausgabeparameter haben. Wenn bei bestimmten Ausgabeparametern Probandenparameter eingetragen werden, dann sollte es auch noch die Möglichkeit geben, dass die einzelnen Parametern eine Validierung zurückgeben, um einen falschen Eintrag zu einem Probandenparameter erkennen zu können.
2. Eine weitere Anforderung der Kunden und Forscher ist es, dass die Anwendung eine Hilfe-Funktion besitzen soll. Über einen Link (<https://docs.movisens.com/DataAnalyzer>) erfolgt eine Überleitung ins Internet, um Antworten auf vorhandene Wissenslücken oder allgemeine Informationen zu der Anwendung zu erhalten.
Diese Hilfefunktion war beim DataAnalyzer 1.0 bisher nicht vorhanden.
3. Im Vergleich zu den anderen Produkten besitzen heutzutage viele Software-Anwendungen eine Benutzerführung. DataAnalyzer 2.0 soll als Einstieg eine Schritt-für-Schritt-Einweisung besitzen, damit die Kunden lernen, wie die Anwendung aufgebaut ist.
Auch dies war in der vorherigen Version nicht vorhanden.

4. Bei der Einzelmessung soll es eine Übersicht der Measurement-Parameter geben, beim Batch-Mode sollen diese Daten als Tabelle dargestellt werden. (Measurement-Parameter: Start, Dauer, Sensortyp, ...).
 - Im Einzelmodus soll es für die Measurement-Parameter einen Editor geben, um bestimmte Werte ändern zu können (z.B. Alter, Gewicht, Körpergröße, Geschlecht, ...).
 - Zwar gibt es im DataAnalyzer 1.0 auch einen Editor, dieser ist entweder auf der rechten Seite im Hauptmenü abgebildet oder in der Dropdown-Box zu finden. Dies sieht von der Gestaltung eher unübersichtlich aus.
5. Des Weiteren braucht die Anwendung einen neuen Schritt namens „Analyse-Config“. Durch dieses neue Feature sollen die Konfigurationsmöglichkeiten für die Kunden erhöht werden. Diese ist nur im Expertenmodus vorhanden und kann daher über die Einstellungen aktiviert werden. Beim Analyse-Config sollen folgende Konfigurationen möglich sein: Konfiguration der Datenausgabe, Selektive Auswertung (nur bestimmten Zeitbereich, nur ganze Tage, ...), Zeitzone, Quelle/Ziel und die Option „Alles analysieren / nur nicht analysierte Daten. Die Einstellungen im Analyse-Config sollen vor dem Weitergehen gespeichert werden.
6. Die Lizenzierung soll so einfach wie möglich sein und es soll eine Möglichkeit geben, weitere Lizenzen hinzufügen zu können.
 - Man habe außerdem noch vorgeschlagen, ein Subscription Modell bzw. „Pay per use“ einzuführen.
 - Die lizenzierten Module werden bei der Tabelle mit den Ausgabeparameter angezeigt. Damit können die Ausgabeparameter je nach Modul sortiert werden.
 - Die Algorithmen, die nicht zu der gekauften Lizenz gehören, sollen ausgegraut bzw. „disabled“ dargestellt werden.
7. Beim DataAnalyzer 1.0 wird zwar eine Fortschrittsanzeige während der Analyse der Daten angezeigt, jedoch ist diese zu ungenau.
 - Daher benötigt der neue DataAnalyzer eine neue Progress-Bar, die aber in zwei Progress-Bars aufgeteilt werden sollen: die erste Progress-Bar zeigt die Prozentzahl der analysierten Rohdaten pro Messung an, die zweite Progress-Bar zeigt die analysierten Rohdaten insgesamt an.
 - Es soll auch noch die Möglichkeit geben, fehlgeschlagene Analysen fortsetzen zu können, sobald die angezeigten Fehler (bspw. bei der Tabelle-Editierung) beseitigt wurden.
 - Die Geschwindigkeit der Analyse soll beschleunigt werden.
8. Der DataAnalyzer 2.0 soll im Vergleich zum aktuellen DataAnalyzer mit einer moderneren Benutzeroberfläche sowie einer besseren Strukturierung und Übersicht punkten.

4. Architektur und Wireframe der Anwendung

4.1. Architektur

4.1.1. die geplante Architektur der Anwendung

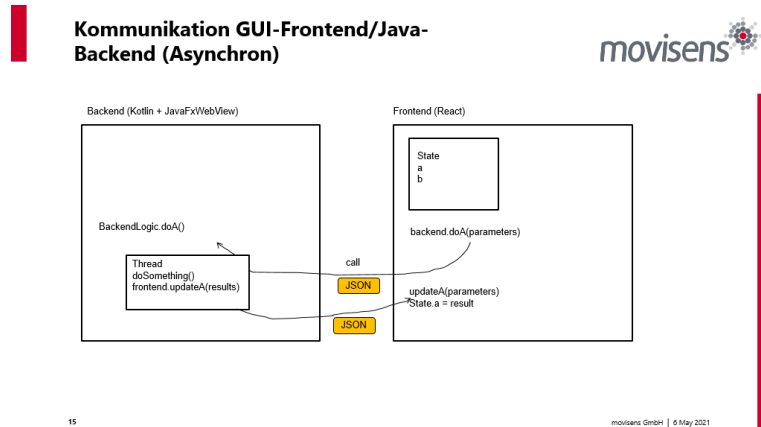


Abb. 10: geplante Architektur DataAnalyzer 2.0

Die vorherige Architektur bestand aus einem Backend und einem Frontend. Das Backend soll mit den Technologien Kotlin, JavaFx und WebView aufgesetzt werden, während das Frontend nur mit dem Webframework React gebaut werden soll (Abb.10).

Im Frontend gibt es bspw. eine Klasse mit dem Namen "state" und diese Klasse hat zwei Properties: a und b. Das Frontend möchte gerne das Ergebnis von der Property "a" haben und ruft die Funktion doA(parameters) auf. Nach dem Aufruf dieser Funktion wird der Request über die Call-Funktion als JSON-Datensatz an das Backend geschickt. Am Backend angekommen, schaut sich die Backend-Logik an, was das Frontend vom Backend möchte und sucht über die Funktion doA() nach den gesuchten Parametern, die das Frontend benötigt. Hat das Backend die gewünschten Informationen gefunden, werden die Informationen zu einem Thread gepackt und dieser Thread wird in der Form eines JSON-Datensatzes über die Call-Funktion wieder zurück an den Frontend geschickt. Somit kann mit der Funktion updateA(), was vom Backend mitgeschickt wurde, die Property a der Klasse State aktualisiert werden. Damit kann im Frontend angezeigt werden, welches Ergebnis die Property a von der Klasse State hat.

Die Architektur sieht zwar relativ einfach aus, birgt jedoch ein großes Problem: Mit WebView ist es schlichtweg schlecht zu testen, da für jede Funktion eine Test-Funktion definiert werden muss.

4.1.2. die neue Architektur des DataAnalyzers

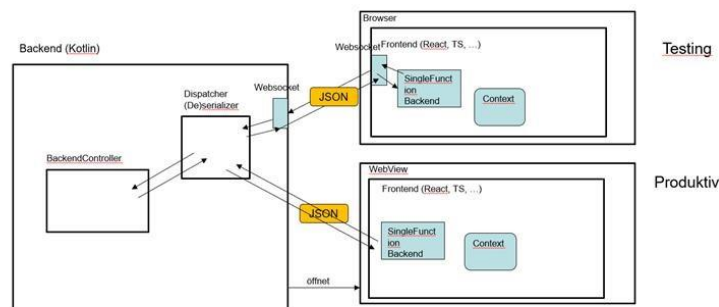


Abb. 11: neu erstellte Architektur für DataAnalyzer 2.0

Nach einer internen Besprechung entstand ein neues Konzept für die Architektur, um somit das Testen zu vereinfachen. Dabei bestand die Architektur nun aus 3 Teilen: dem Backend, welches mit Kotlin implementiert werden soll; dem Frontend mit einem Browser fürs Testen und ein Frontend mit dem WebView für den Produktivbetrieb (Abb.11).

Der Browser möchte gewisse Informationen haben und will diese Anfrage ans Backend schicken. Dabei schickt der Browser die Anfrage an ein WebSocket. Der WebSocket hat die Funktion, eine Session zwischen dem Server (Backend) und der Anwendung vom Benutzer (Frontend) herzustellen. Mit so einer Schnittstelle können Nachrichten ans Backend gesendet und ereignisorientierte Antworten erhalten werden können, ohne den Server nach Informationen abzufragen. Damit findet ein bidirektionaler Datenaustausch statt. Ein Vorteil von dieser Schnittstelle ist, dass die Daten im Frontend in Echtzeit angezeigt werden können. In anderen Worten: Der Kommunikationskanal bleibt nach dem Handshake quasi geöffnet. Der Server kann selbstständig aktiv werden und dem Client alle Informationen zur Verfügung stellen, ohne dass diese vom Client angefragt wurden.

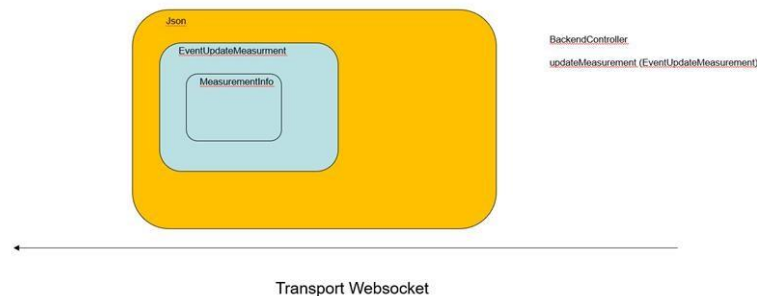


Abb. 12: Schaubild JSON-Daten sowie Funktionen über den Transport zwischen Websockets

Die JSON-Nachrichten können näher aufgefasst werden: Das Backend möchte die vom Client gewünschten Informationen zurück zum Frontend schicken. Dabei hat der BackendController die Funktion `updateMeasurement()` gefunden, dass den Parameter `MeasurementInfo` aktualisieren soll. Diese Funktion wird für den Parameter `MeasurementInfo` als Ereignis zusammengefasst und diese als JSON-Nachricht verpackt. Danach kann diese JSON-Nachricht über die Websockets transportiert werden (siehe Abb.12).

Zusammengefasst wird der Request im JSON-Format über den WebSocket vom Frontend aus an das WebSocket vom Backend gesendet. Sobald dieser Request am WebSocket des Servers angekommen ist, wird dieser Request dann an den Dispatcher geschickt, der dafür zuständig ist, die JSON-Nachrichten auszupacken und die richtigen Funktionen aufzurufen. Diese Funktionen werden danach an den Backend-Controller geschickt, der dann die gewünschte Aktion ausführen soll. Hat der Backend-Controller die Aktion ausgeführt, werden dann die Informationen zurück an den Dispatcher geschickt. Beim Dispatcher angekommen werden die Informationen im JSON-Format vom Backend zum Produktiv-Modus des Frontends geschickt (WebView). Sollen die Informationen anhand bestimmter Test-Funktionen gut funktionieren, dann werden die JSON-Daten zurück zum Backend geschickt und an den Dispatcher weitergeleitet. Dann werden die JSON-Nachrichten vom Dispatcher an den WebSocket im Backend geschickt und über einen Kanal zum anderen WebSocket im Frontend geleitet. Der Vorteil dieser neuen Architektur ist es, dass das Testen bestimmter Daten nicht mehr so kompliziert ist als bei der vorherigen Architektur.

4.2. Konzeption (Wireframe) & Ideen für DataAnalyzer 2.0

Am Anfang des Praxissemesters gab es die Aufgabe, für die neuen Anforderungen an die Software, ein Wireframe zu der Anwendung zu erstellen. Da bereits ein Wireframe vom Unternehmen existierte, sollte der Praktikant seine Ideen bei der Gestaltung der Anwendung miteinfließen lassen.

Im Vergleich zum alten Wireframe, das vom Unternehmen entwickelt wurde, sollte dieses Wireframe zusätzlich um eine weitere Funktion ergänzt werden: der Expertenmodus. Mit diesem Modus kann man nicht nur weitere Einstellungen in den einzelnen Schritten der Anwendung vornehmen, sondern es wird einen weiteren Schritt namens "Advanced Settings" freigeschaltet, wo weitere Konfigurationen vorgenommen werden können. Ist der Expertenmodus in den Einstellungen (Settings) ausgeschaltet, beschränkt sich die Anzahl der Schritte in der Anwendung auf 4 Schritte.

Alle Schritte der Anwendung beinhalten eine Schritt-für-Schritt-Leiste, die anzeigen soll, bei welchem Schritt man sich befindet und wie viele Schritte die Anwendung insgesamt beinhaltet. Unterhalb der Schritt-Namen gibt es noch eine kurze Beschreibung als Erläuterung. Außerdem soll jede Seite der Anwendung über die Information beinhalten, wie lange die ausgewählte Lizenz noch verfügbar ist. Dies geschieht mit Hilfe eines farblich markierten Kreises in der linken unteren Ecke. Dabei soll diese Anzeige aus folgenden Möglichkeiten bestehen: ist der Kreis grün gefärbt, wird angezeigt wie lange die Lizenz noch zur Verfügung steht. Ist die ausgewählte Lizenz noch eine Woche gültig, dann erhält der Kreis die Farbe Gelb. Ist die Lizenz ab einem bestimmten Datum abgelaufen, dann wird der Kreis rot. Mit der Signalisierung des roten Kreises bedeutet es auch, dass die Auswahl der Ausgabeparameter (siehe „OutputParameter“) unter der vorgegebenen Lizenz nicht mehr möglich ist. Oberhalb der Schritt-Leiste gibt es einen Icon, das auf den sogenannten „Drawer“ (Seitenleiste) hinweist und in diesem Drawer sind sämtliche Buttons drin, die auf die anderen Seiten der Anwendung verweisen. Neben der Möglichkeit, die Tabelle mit den Lizenzen anzuzeigen und den Verweis auf die Dokumentation des DataAnalyzers, kann man noch neben dem Ablauf eines Schritt-für-Schritt Tutorials auch in den Einstellungen den Expertenmodus aktivieren.

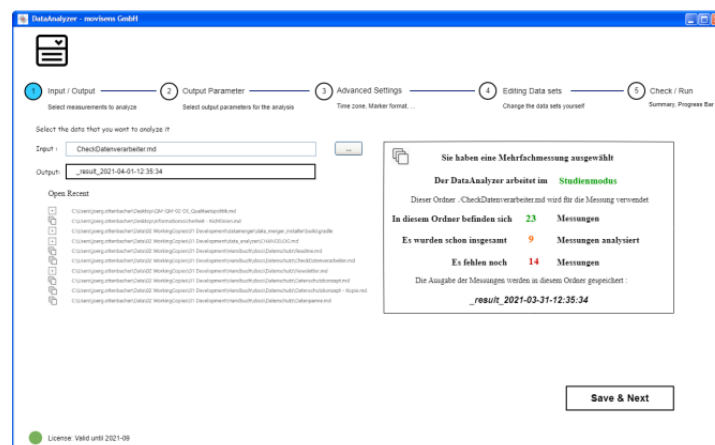


Abb. 13: "Input/Output" im Studienmodus

In der ersten Stufe "Input / Output" (Abb.13) kann man den Ordner mit den Messungen für den Input auswählen und je nach Modus wird auf der rechten Seite in der Info-Box angezeigt, um welche Messung es sich dabei handelt. Wenn eine Einzelmessung ausgewählt wird, d.h. es gibt im Ordner eine Messung mit nur einem Datensatz, dann wählt die Anwendung automatisch den Einzelmodus aus. Handelt es sich bei dem Ordner um eine Mehrfachmessung, d.h. der Ordner beinhaltet mehrere Messungen bzw. es ist ein Ordner mit mehreren Unterverzeichnissen, dann wählt die Anwendung automatisch den Studienmodus aus. Zusätzlich zum Studienmodus wird in der Info-Box noch

angezeigt, wie viele Messungen in dem Ordner sich befinden, wie viele Messungen schon analysiert wurden und welche Messungen noch analysiert werden sollen. Sollte der Expertenmodus durch die Einstellungen aktiviert werden, dann kann zusätzlich der Zielordner (Output) ausgewählt/generiert werden. Ist der Expertenmodus nicht aktiviert, dann wird der Zielordner automatisch generiert und in der Info-Box ganz unten angezeigt. Beim Einzelmodus gibt es keine Informationen über die Anzahl der Messungen, die Anzahl der analysierten Messungen und die Anzahl der noch fehlenden Messungen. Daher steht in der Info-Box nur, welcher Ordner für die Messung verwendet werden soll und in welchem Ordner die analysierten Messungen gespeichert werden.

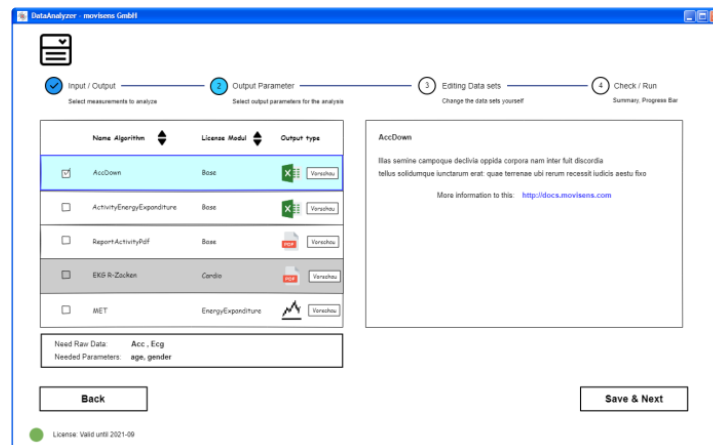


Abb. 14: "OutputParameter": Anzeige Ausgabeparameter sowie Info-Box

In der nächsten Stufe kann aus mehreren Ausgabeparameter (= Output-Parameter) gewählt werden (Abb.14). Dabei können nur die Output-Parameter ausgesucht werden, die zu einer bestimmten, gekauften Lizenz zugeordnet sind. Die restlichen Output-Parameter, die diese Lizenz nicht beinhaltet, werden ausgegraut und können nicht ausgewählt werden. Dennoch kann man bei den ausgegrauten und nicht ausgegrauten Parametern die Information dazu erhalten und jede Beschreibung verfügt über einen Link, der einen direkt zum Handbuch des Produktes führt. Bei jedem Output-Parameter zeigt eine Vorschau an, wie die Parameter je nach Output (z.B. Excel, PDF, ...) angezeigt werden sollen. Nach der Analyse der Messungen wird der Zielordner mit den Enddaten generiert und im Zielordner ist dementsprechend die Output-Datei aus der Vorschau der Output-Parameter zu finden. Jeder Output-Parameter kann auf der rechten Seite mit weiteren Informationen versehen werden. Welche und wie viele Output-Parameter angezeigt werden, steht unter der Parameter-Tabelle in einer weiteren Info-Box. Dabei steht grundsätzlich, welche Parameter schon gesetzt wurden und welche Parameter benötigt werden, um diese optional einstellen zu können.

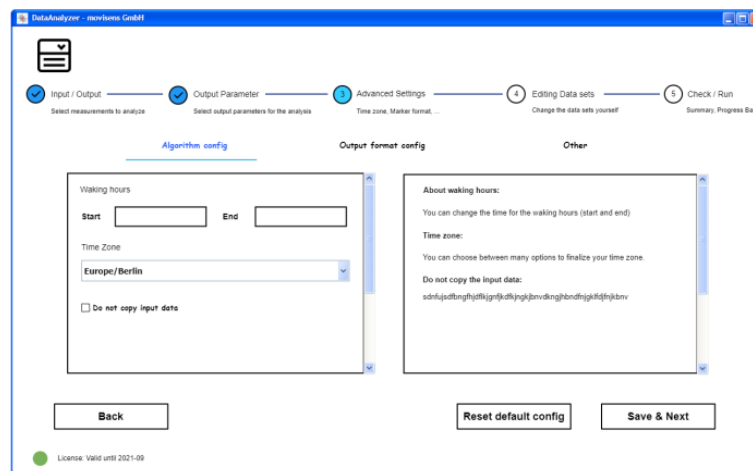


Abb. 15: "Advanced Settings" am Beispiel 'Algorithm config'

Sobald der Expertenmodus aktiviert ist, wird die Stufe „Advanced Settings“ freigeschaltet (Abb.15). In diesem Step kann aus 3 Kategorien (Algorithm config, Output Format Config & Other Configurations) gewählt und zusätzliche Anpassungen durchgeführt werden z.B. kann beim „Algorithm config“ der Start- und Endzeitpunkt der Wachzeit angegeben werden. Außerdem ist es noch möglich die Zeitzone zu bestimmen. Zu den einzelnen Funktionen steht auf der rechten Seite in der Info-Box die benötigte Beschreibung dazu. Sollen die einzelnen eingestellten Parameter wieder auf den Standardwert zurückgesetzt werden, dann kann dies mit dem Button „Reset default config“ erfolgen. Sind alle notwendigen Anpassungen durchgeführt, so gelangt man mit dem Button „Save & Next“ zur nächsten Stufe.

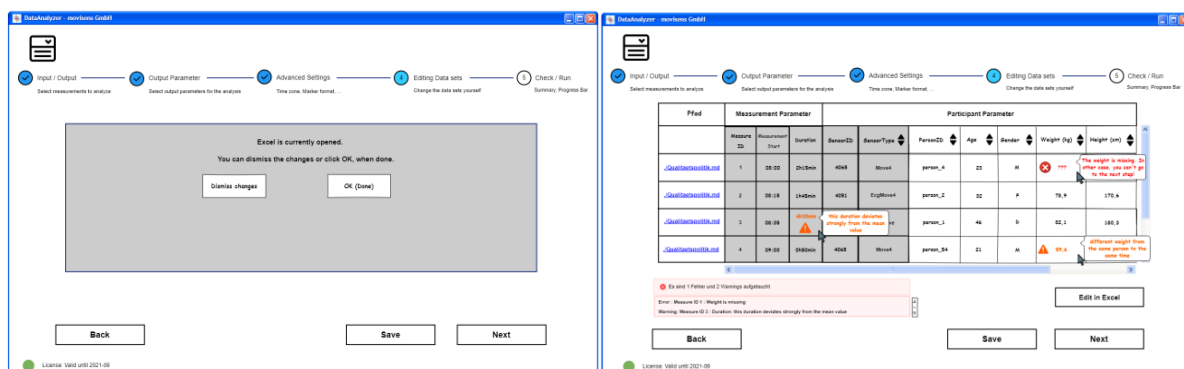


Abb. 16: "Editing DataSets": Anzeige Info über Datenbearbeitung im Excel-Modus (links) und in der Anwendung (rechts)

Im vorletzten Step gibt es die Möglichkeit, bestehende Datensätze aus den Messungen zu bearbeiten (Abb.16). Dabei kann man im DataAnalyzer Parameterdaten ändern und speichern lassen. Soll Excel verwendet werden, dann drückt man auf dem Button „Edit in Excel“ und man wird zu der Excel-Datei geleitet. Im DataAnalyzer erscheint ein Fenster mit dem Hinweis, dass Excel geöffnet ist. Sind sämtliche Änderungen durchgeführt worden, dann werden die Änderungen über den Button „OK(done)“ gespeichert. Sollen die Änderungen, die mit Excel durchgeführt wurden, rückgängig gemacht werden, dann werden die ganzen Änderungen über den Button „Dismiss Changes“ zum Normalzustand zurückgesetzt. Sobald man wieder in die Anwendung gelangt, werden die Daten anhand verschiedener Validierungsparameter geprüft und sobald irgendwelche Fehler erscheinen, werden diese Fehler unterhalb der Tabelle angezeigt. Außerdem kann die Zeile angeklickt werden (z.B. Measure ID 2 - Weight) und direkt zu dem Punkt gelangen, wenn der Fehler nicht schnell entdeckt wird. Diese Funktion gibt es auch beim VS Code, wenn beim Starten der

Anwendung unerwartete Fehler auftauchen und über die Anzeige in der unteren Leiste kann geprüft werden, welche Fehler entstanden sind und wo diese Fehler zu finden sind. Man gelangt erst dann zum letzten Step, wenn alle Fehler / Warnungen aus der Tabelle korrigiert wurden. Sind alle Änderungen gemacht, dann wird der Button „Save“ aktiviert, um die Daten in der Tabelle zu speichern und erst dann der Button „Next“, um somit zum letzten Step zu gelangen.

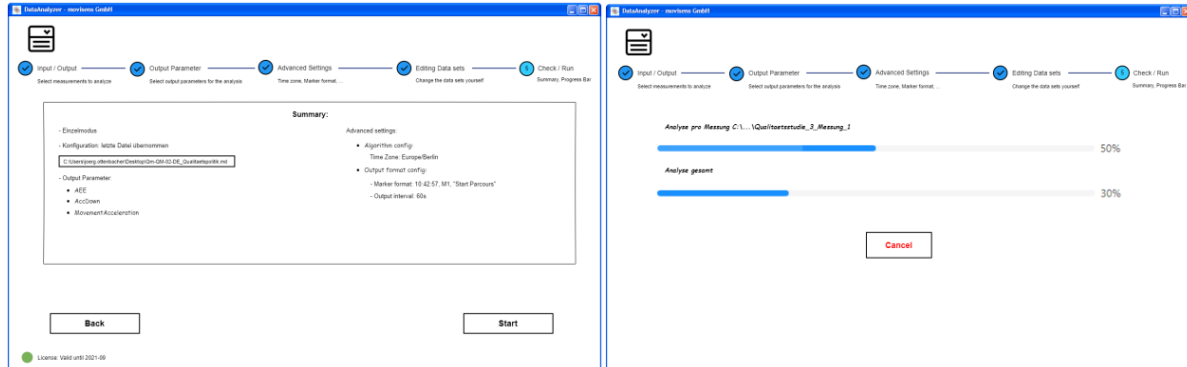


Abb. 17: (Check / Run) Zusammenfassung Konfigurationen (links); Analyse Messungen (einzeln und gesamt) (rechts)

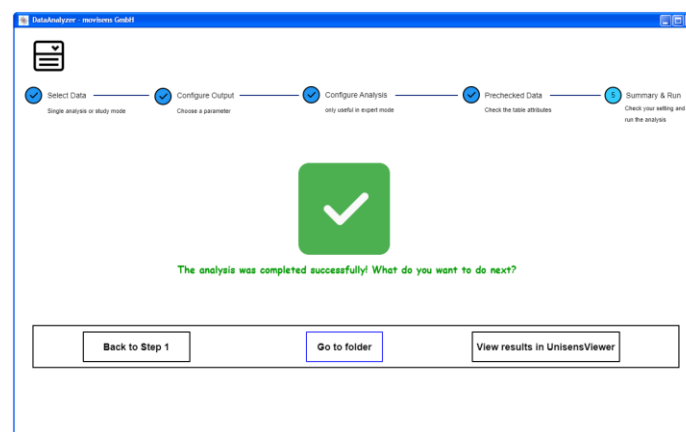


Abb. 18: Erfolgreiche Analyse der Messungen sowie Buttons für weitere Schritte in der Anwendung

Beim letzten Schritt kann anhand der Info-Box die Zusammenfassung aller ausgewählten Konfigurationen wie Ordner, Parameter, Einstellungen, usw. eingesehen werden. Sollte man mit den Einstellungen, die man bei den vorherigen Steps gemacht hat, nicht zufrieden sein, dann kann man immer noch zu den einzelnen Steps zurückgehen und bestimmte Anpassungen durchführen. Sobald die Anpassungen abgeschlossen sind, wird der Button „Start“ betätigt und somit werden die Messungen analysiert. Diese analysierten Messungen werden in der ersten Progress Bar angezeigt. Dabei wird im ersten Progress Bar angezeigt, um welche Dateien es sich handelt. Im zweiten Progress Bar sieht man die Gesamtprozentzahl der gesamten Messungen (siehe Abb.17). Sind alle Messungen analysiert worden, dann gibt es die Möglichkeit, aus 3 unterschiedlichen Buttons auszuwählen: „View in Folder“, „Back to app“ und „Back to Step 1“ (Abb.18). Sollte dennoch die Analyse der Messungen scheitern, dann erscheint eine Fehlermeldung.

5. Teilweise erledigte Aufgaben über einen längeren Zeitraum

5.1. Datenmodell

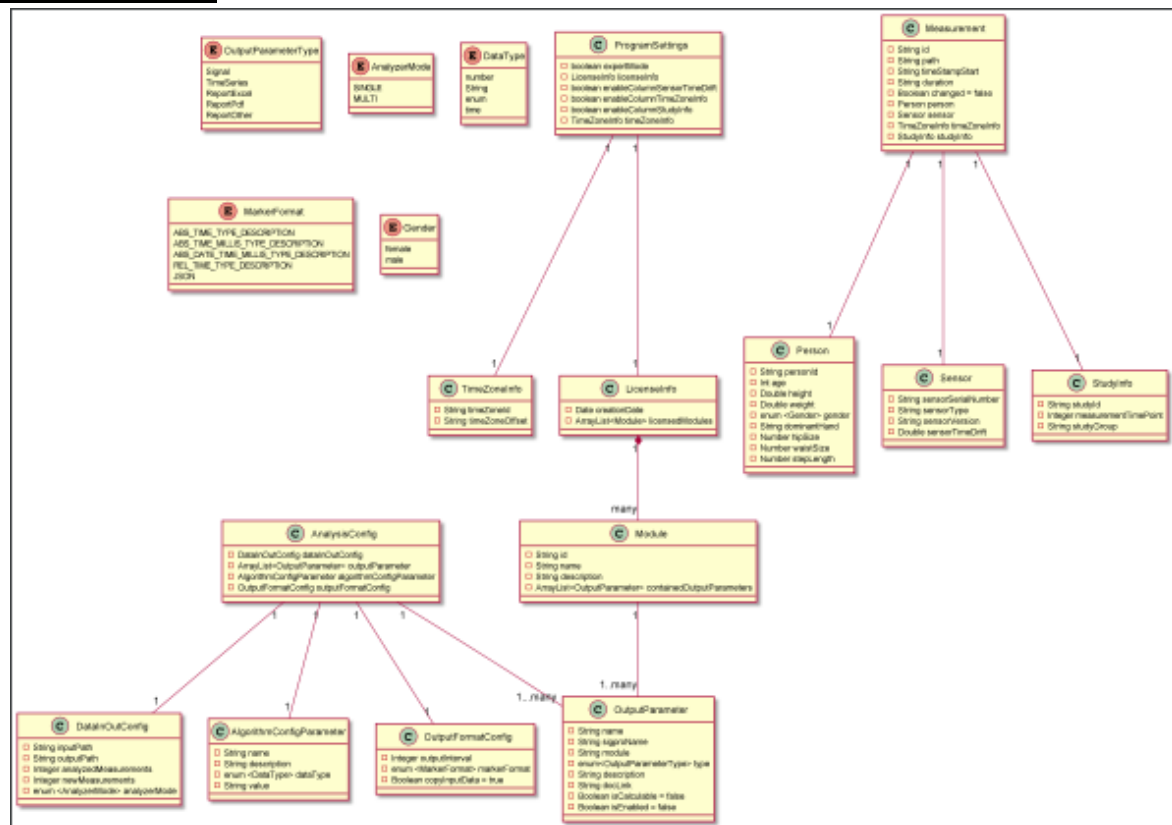


Abb. 19: UML-Klassendiagramm zum DataAnalyzer 2.0

Ein Datenmodell dient dazu, „die Struktur für die in den Systemen zu verarbeitenden (im Besonderen für die zu speichernden) Daten zu finden und festzulegen.“ (Wikipedia – Datenmodell)

Die Klasse ProgramSettings soll die Einstellungen in der Anwendung darstellen und aus mehreren Attributen bestehen: die Attribute EnableColumnTimeDrift (Spalte für Anzeige der TimeDrift eines Sensors), EnableColumnTimeZoneInfo (Spalte für Anzeige der Zeitzone) und EnableColumnStudyInfo (Spalte für Anzeige der Informationen im Study-Mode (hier Batch-Mode)) sollen dazu dienen, in Schritt „Editing DataSets“ weitere Spalten in der Tabelle hinzufügen zu können. Außerdem soll neben der Aktivierung des Expertenmodus (expertMode) auch noch die Möglichkeit bestehen, eine Seite mit den vorhandenen Lizenzen anzuzeigen (LicenseInfo).

In der LicenseInfo wird angezeigt, welche Lizenzen zur Verfügung stehen. Dabei soll die LicenseInfo aus einem Erstellungsdatum (creationDate) und aus einer Liste mehrerer Module bestehen (ArrayList<Module> licensedModules). Diese Lizenzmodule beinhalten neben einer Id für die Identifizierung der einzelnen Module auch der Name der Lizenz (name), die Beschreibung (description) und eine Liste der benötigten Outputparameter (ArrayList<OutputParameter> containedOutputParameters), die für den Schritt „OutputParameter“ von entscheidender Bedeutung ist. Die OutputParameter (dt. Ausgabeparameter), die wiederum auch mit der Klasse AnalysisConfig in Zusammenhang stehen, besitzen neben einem Namen für den Ausgabeparameter eine Beschreibung (description), die dazugehörige Lizenz (wobei ein Ausgabeparameter immer zu einer bestimmten Lizenz gehören muss) (license) und einen Link, die im Browser zu der Dokumentationsseite des DataAnalyzers geleitet wird (docLink). Außerdem beinhaltet die Klasse OutputParameter über eine Enum-Klasse von Ausgabeparametertypen (Enum

<OutputParameterType> type), das die unterschiedlichen Ausgabeformate (PDF, Excel, TimeSeries, usw.) ausgeben soll. In der Klasse Output-Parameter kann man zwischen den Parameter isCalculable (Kalkulierbarkeit der Ausgabeparameter) und isEnabled (Auswahl bestimmter Ausgabeparameter) entscheiden. Beide Parameter sind auf den Standardwert „false“ gesetzt.

Die Klasse TimeZoneInfo, die mit der Klasse AnalysisConfig in Zusammenhang steht, hat neben der Property Id für die Identifizierung einzelner Zeitzonen auch die Property timeZoneOffset, dass die Funktion hat, einzelne Abstände zwischen den Zeitzonen berechnen zu können (z.B. Kanada : UTC-8).

Die Klasse AnalysisConfig beschreibt sämtliche Schritte, in denen Einstellungen bzw. Konfigurationen in den einzelnen Schritten der Anwendung gemacht werden können. Diese Klasse besteht wiederum aus mehreren Unterklassen: die Unterklasse „DataInOutConfig“, das den Schritt „Input/Output“ widerspiegeln soll, die „OutputParameter“ für den Schritt ‚OutputParameter‘ und die Klassen „AlgorithmConfigParameter“ und „OutputFormatConfig“ für den Schritt „Advanced Settings“.

- DataInOutConfig: Diese Klasse beinhaltet einen „inputPath“ für die Auswahl des Ordners im Input-Feld. Unter dem Input-Feld gibt es auch noch den „outputPath“ für die Ablage der analysierten Daten in einem Ordner. Ist der Expertenmodus nicht aktiviert, dann legt die Anwendung automatisch den Output-Ordner fest, wo die analysierten Daten enthalten sind, ist der Expertenmodus aktiviert, dann kann der Output-Ordner festgelegt werden. In der Info-Box wird bei der Auswahl des InputPaths berechnet, wie viele Messungen schon analysiert wurden(analyzeMeasurements) und wie viele Messungen noch analysiert werden müssen (newMeasurements). Zuletzt wird anhand der Bestimmung des InputPaths festgelegt, in welchem Modus die Daten angegeben werden: beinhaltet das Verzeichnis nur einen einzigen Datensatz, dann handelt es sich um den Einzelmodus (SINGLE) und die Properties „analyzeMeasurements“ und „newMeasurements“ werden ausgelassen. Handelt es sich bei dem Ordner um mehrere Unterverzeichnisse mit mehreren Datensätzen, dann handelt es sich um den Studienmodus (MULTI).
- AlgorithmConfigParameter: In den „Advanced Settings“ gibt es die Möglichkeit, zu den Ausgabeparametern weitere Konfigurationen zu erstellen. Bspw. legt man für den Algorithmus „Wakinghours_start“ (name) den Zeitpunkt (DataType: time) mit dem Wert „08:40:00“ (value) fest. Zu diesem Algorithmus gibt es in der Info-Box auf der rechten Seite die Beschreibung, die erklärt, bis zu welcher Uhrzeit diesen Algorithmus einstellen kann.
- OutputFormatConfig: Hier wird nicht nur festgelegt, in welchem Abstand die Mess-Daten angezeigt werden (outputInterval), sondern es kann aus 5 unterschiedlichen Marker-Formaten gewählt werden (markerFormat) (z.B. ABS_TIME_TYPE_DESCRIPTION, JSON, usw.). Werden keine Einstellungen gemacht, dann wird automatisch der Standard-Wert des Marker-Formats festgelegt, das Gleiche gilt auch für den Output-Interval.

Neben den ProgramSettings und der AnalyseConfig gibt es noch eine weitere Klasse, die eine große Rolle bei dem Schritt „EditingDataSets“ spielt: Measurements. Die Klasse Measurements besteht aus einem Pfad, der anzeigt, woher diese Datensätze kommen (path), einen Zeitstempel (timestampStart), der Durchlauf mit dem Sensor (duration), den Daten zu der Person (person_id, age, height, weight, Gender (=> female oder male), dominantHand, hipSize, waistSize, stepLength), den Daten zu dem Sensor (sensorTimeDrift, sensorLocation, sensorType und sensorSerialNumber) sowie der StudyInfo (StudyId, measurementTimePoint und studyGroup). Alle Angaben der Spalten sind davon abhängig, welcher Sensor ausgewählt wird. Daher können unterschiedliche Sensoren nicht alle Parameter abbilden (siehe Abb.19).

5.2. Das Programmieren der Anwendung – Beschreibung einzelner Schritte und Funktionen

Das Programmieren der Anwendung hatte die meiste Zeit des Praxissemesters in Anspruch genommen und dauerte von Mitte Mai bis Ende Juli. Man sollte mit den Skriptsprachen JavaScript/TypeScript sowie mit dem ANT-Design-Framework die Anwendung bauen. Die Definition einzelner Funktionen und der Daten für das Backend sollte mit der Programmiersprache Kotlin erfolgen.

Als Erstes begann man mit dem Schritt „OutputParameter“. Die Aufgabe erschien dem Betreuer als „machbar“, da man bereits wusste, mit welchen ANT-Design-Komponenten man diese Seite erstellen konnte. Die ersten Aufgaben zu diesem Schritt waren, eine Tabelle sowie die Beschreibung zu den einzelnen Ausgabeparametern einzubauen. Die Tabelle sollte ursprünglich 4-spaltig sein. Da das Anklicken der Checkbox aufgrund der ANT-Design-Komponenten nur in Kombination mit dem Namen des Ausgabeparameters möglich war, bestand letztendlich die Tabelle nur noch aus 3 Spalten (Algorithm Name, License Modul und Output Type). Sowohl die Spalten „Algorithm Name“ als auch „License Modul“ waren sortierbar und beim Output Type konnte durch einen entsprechenden Button eine Vorschau über das z.B. im PDF formatierten Dokument bzgl. dem ausgewählten Parameter ermöglicht werden. Dabei besaß die Tabelle noch über ein weiteres Feature: wenn man mit der Maus über die Zeile des Ausgabeparameters fährt, dann erscheint dynamisch neben der Überschrift zu den Ausgabeparametern noch die dazugehörige Beschreibung und ein Link, der über den Browser zu der Dokumentation der Anwendung hinführt. Dabei sollen die Daten zu der Tabelle, der Beschreibung, etc. vom Backend zum Frontend geholt werden und die ausgewählten Parameter sollen über einen Button zum Backend geschickt werden. Diese Funktion sei auch notwendig für den letzten Step „Check / Run“ notwendig, um die ausgewählten Algorithmen-Parameter in der Zusammenfassung anzeigen zu können. Jedoch sollen nicht nur in der Zusammenfassung die ausgewählten Parameter angezeigt werden, sondern es sollen der Modus (Einzel- / Studienmodus), die ausgewählten Parameter in den Advanced Settings sowie andere Informationen angezeigt werden.

Die Entwicklung dieser Seite „OutputParameter“ hatte insgesamt 1 Monat gedauert, da es an einigen Stellen zu Leichtsinnsfehlern kam, die erst durch den Betreuer sichtbar gemacht wurden. Eine weitere Aufgabe war es, die einzelnen Komponenten in diesem Framework anzuschauen und zu überlegen, für welche Seiten die Komponenten passen könnten. Weiterhin kann für die weitere Entwicklung der Anwendung auch noch überlegt werden, welche Komponenten man, z.B. für die Buttons, in der Anwendung eingebaut bzw. ersetzt werden können.

Nachdem die Seite „OutputParameter“ schon größtenteils von der Funktionalität her fertig entwickelt war, ging es als Nächstes mit dem Komponenten-Einbau in „Editing DataSets“ weiter. In der Seite „Editing Data Sets“ sollten bei der Tabelle nicht nur die Daten sortierbar sein, sondern auch über einen Button editierbar sein, sodass man dann zum Programm Excel weitergeleitet wird. Die Editierbarkeit der Daten kann durch 2 Arten erfolgen: Entweder werden die Daten direkt in der Anwendung überarbeitet oder als Alternativmöglichkeit wird das Programm Excel ausgewählt. Sollte die Bearbeitung der Daten in Excel abgeschlossen sein, werden die einzelnen Daten anhand der Validierung überprüft. Dennoch soll es auch Validierungen geben, die auch in Beziehung mit anderen Daten in der Tabelle stehen könnten. Die Fehler / Warnungen sollten unterhalb der Tabelle angezeigt werden. Ähnlich wie bei VS Code oder anderen Programmierwerkzeugen sollte es die Möglichkeit geben, durch Anklicken eines bestimmten Fehlers oder Warnung auf die entsprechende Datenzeile hingeführt werden, um somit Zeit zu sparen und die Arbeit der Fehlersuche erheblich zu erleichtern.

Neben der Entwicklung beim Step „Editing DataSets“ gab es auch noch den Komponenten-Einbau in den weiteren Schritten der Anwendung.

- Beim ersten Schritt der Anwendung „Input/Output“ sollte es eine Funktion geben, die es ermöglichte, beim Klicken des Input-Buttons eine Nachricht ans Backend zu schicken. Neben den Buttons für den Input und den Output der Pfade sollte auch noch auf der rechten Seite die Info-Box dynamisch angezeigt werden, wenn der Modus (Einzelmodus oder Studienmodus) für die ausgewählten Pfade geändert wird. Wenn die Datei nur einen Datensatz enthielt, dann ist der Modus auf Einzelmodus eingestellt, handelte es sich bei dem Pfad um mehrere Verzeichnisse mit mehreren Messungen, dann war es der Studienmodus (Batchmode). Über den Button „Open Recent“ sollte die Tabelle Folgendes beinhalten: Der Pfad für den Input sowie den Modus für den Einzelmodus oder den Studienmodus. Dabei sollte beim Auswählen des Pfades nur eine Einfach-Auswahl möglich sein und keine Mehrfachauswahl. Die Anzeige in der Info-Box änderte sich dynamisch je nach Art des Modus abhängig von der Auswahl des Pfades. Die Buttons für den Input und den Output waren ANT-Design-Komponenten.
- Der zweite Schritt der Anwendung wurde bereits vorher beschrieben.
- Beim dritten Schritt der Anwendung in „Advanced Settings“ sollten der Aufbau der Konfigurationen sowie die inhaltliche Beschreibung der einzelnen Konfigurationen wie bei dem zweiten Schritt der Anwendung „OutputParameter“ entwickelt werden. Die Auswahl der einzelnen Kategorien „Algorithm config“, „Output format config“ sowie „Other configurations“ erfolgte durch das Anklicken der Komponente „Tabs“. Jedoch sollte die Seite im Vergleich zu der Seite „OutputParameter“ über kein Mouse-Hover-Feature besitzen. Die Entwicklung dieser Seite beinhaltete nur die anschauliche Darstellung der Seite, an dieser Seite wurde keine einzige Funktionalität eingebaut. Selbst der Button „Reset default config“ wurde nur im Frontend angezeigt und beinhaltete keine Funktionalität. Die Freischaltung des Steps sollte eigentlich über die Aktivierung des Expertenmodus in den Settings geschehen. Da man sich auf die Fokussierung der Steps konzentriert hatte, wurde die Programmierung der Aktivierung des Expertenmodus ausgelassen und man sei davon ausgegangen, dass der Step „Advanced Settings“ von Anfang an aktiviert war.

Die Seitenleiste, indem die Settings aufgerufen werden, wurde über einen sogenannten Drawer erstellt. Diese wurde über einen Button entsprechend aufgerufen und es erschien auf der linken Seite eine Seitenleiste mit den dazugehörigen Buttons wie „Run Quick Tutorial“ oder „Licensing“. Da die Entwicklung des Drawers durch den Fokus anderer Inhalte und die Programmierung weiterer Funktionen für die Kommunikation zwischen dem Frontend und dem Backend ins Stocken geraten war, wurden nur die Buttons mit der dazugehörigen Überschrift eingebaut.

5.3. Endzustand der Anwendung am Ende des Praxissemesters

Die folgenden Bilder und Inhalte dazu zeigen den Zustand der Anwendung, der bis zum Ende des Praxissemesters (Ende Juli) fertig programmiert wurde. Eine Anmerkung zu den gezeigten Bildern: an vielen Bildern sieht man die untenstehenden Buttons „Zurück“ und „Weiter“ nicht, da die Seite etwas großflächig gestaltet war und der Button für den Drawer zusätzlichen Platz benötigt hatte. Im Praktikum ging es, laut dem Betreuer, weniger um die Gestaltung der Anwendung, sondern mehr um die Funktionalität. Zu den einzelnen Steps der Anwendung gibt es auch noch eine Beschreibung, welche Inhalte der einzelnen Steps fehlen und wie sie angeordnet werden sollten.

- **Anwendung allgemein:**

Bei der Anwendungsentwicklung wurden nur die einzelnen Schritte (1-5) implementiert. Die zusätzliche Seitenerstellung für die Lizenzen, Tutorials sowie die Darstellung der Progress-Bars für den Step „Check/Run“ wurden ausgelassen. Der Einbau der Information über die Verfügbarkeit der Lizenz wurde aufgrund der geringen Entwicklungszeit und der geringen Erfahrung in der Programmierung weggelassen. Neben der Entwicklung des Frontends wurden noch einige Funktionen für die Kommunikation zwischen dem Frontend und dem Backend implementiert. Sämtliche Validierungsaspekte sowie Tests zu der Anwendung, die im Ausbildungsplan beinhaltet waren, wurden aus zeitlichen Gründen nicht mehr berücksichtigt.

Für den Test wurde bei der Präsentation der Anwendung auf Serenity gesetzt. Damit soll bspw. das Ändern der Daten in „Editing Data Sets“ überprüft werden, aber auch auf weitere Kleinigkeiten gesetzt werden. Diese Aufgabe konnte aber aus Zeitgründen leider nicht angefangen werden.

Bei jeder Seite gab es oben ein Stufenliste, die anzeigt, auf welcher Stufe man sich befindet und welche Inhalte in dieser Stufe drin sind. Geplant war außerdem noch mit dem Programm namens „MasterCalculator“, die eigentliche Datenanalyse der Messdaten durchzuführen. Der MasterCalculator sollte nur mit dem Backend und nicht mit dem Frontend verbunden werden.

- **Step 1: Input/Output**

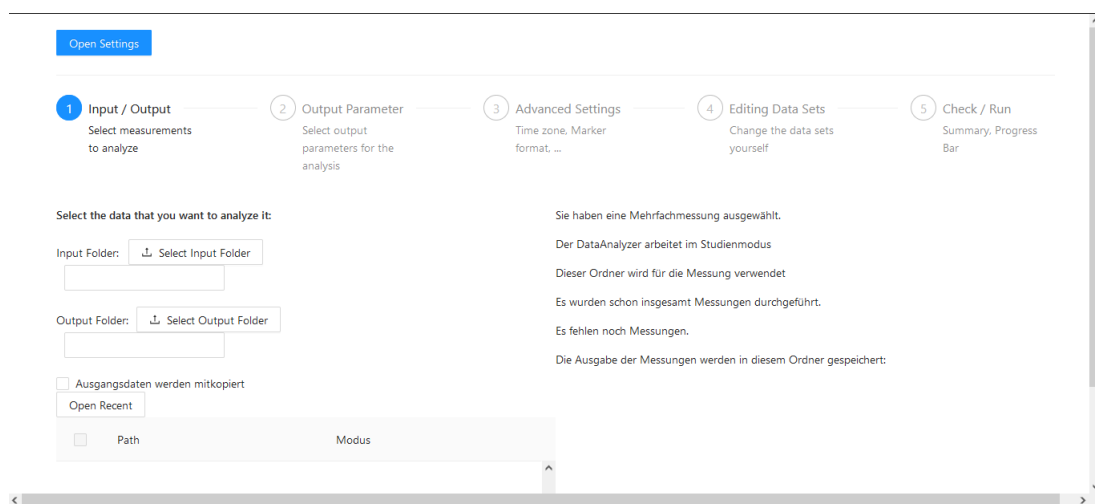
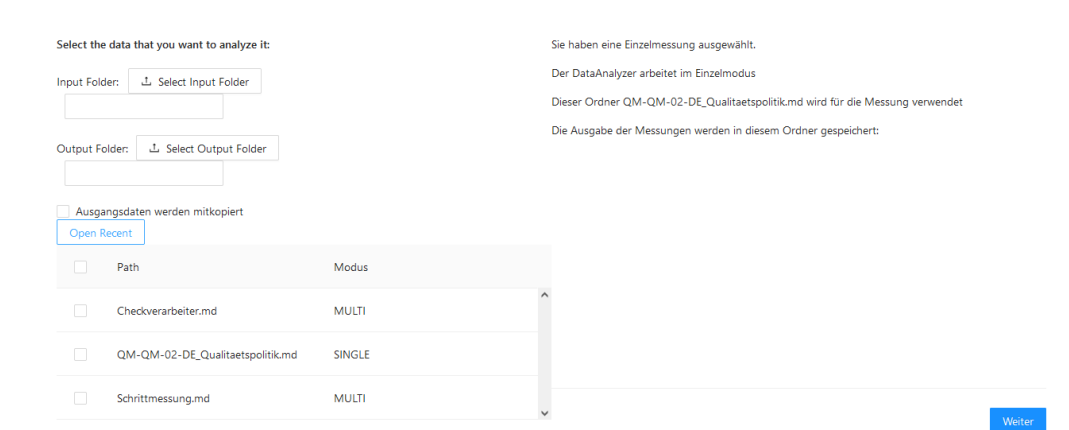


Abb. 20: Endzustand "Input/Output"

Dieser Schritt wird sich in 2 Bereiche aufgeteilt: im linken Bereich gibt es 2 Buttons für den Input und den Output, mit denen die Pfade für die Messung ausgewählt werden können. Nachdem die Pfade ausgewählt sind, werden diese in der Info-Box auf der rechten Seite angezeigt. In welchem Ordner die Ausgabe der Messungen gespeichert werden soll, wird nicht dargestellt. Die Funktion „Ausgangsdaten werden mitkopiert“ erscheint zwar im Frontend, jedoch ist keine Funktionalität dazu eingebaut worden. Diese Funktion sollte eigentlich bei Step 3 „Advanced Settings“ in der Rubrik „Algorithm config“ eingebaut werden und sollte in Step 1 „Input/Output“ ausgelassen werden (siehe Abb.20).



Select the data that you want to analyze it:

Input Folder:

Output Folder:

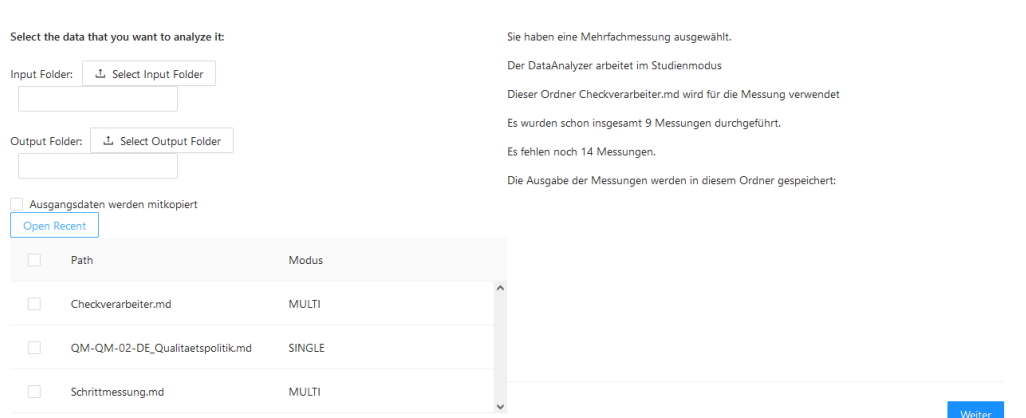
☐ Ausgangsdaten werden mitkopiert

| <input type="checkbox"/> Path | Modus |
|---|--------|
| <input type="checkbox"/> Checkverarbeiter.md | MULTI |
| <input type="checkbox"/> QM-QM-02-DE_Qualitaetspolitik.md | SINGLE |
| <input type="checkbox"/> Schrittmessung.md | MULTI |

Sie haben eine Einzelmessung ausgewählt.
 Der DataAnalyzer arbeitet im Einzelmodus
 Dieser Ordner QM-QM-02-DE_Qualitaetspolitik.md wird für die Messung verwendet
 Die Ausgabe der Messungen werden in diesem Ordner gespeichert:

Abb. 21: Endzustand „Input/Output“ mit Angabe letzter Messungen + Modus

In der Info-Box steht, abhängig vom Dateipfad, welche Messung ausgewählt wurde. Die Inhalte erscheinen nur, wenn mit der Maus über die entsprechenden Datenzeilen gefahren wird (Mouse-Hover-Feature). Wird durch die Auswahl des Dateipfades festgestellt, dass hier eine Einzelmessung ausgewählt wurde, dann wird dies angezeigt und dass der DataAnalyzer im Einzelmodus arbeitet. Außerdem werden in der Info-Box die Ordernamen für die Verwendung der Messung und für die Ausgabe der Messungen angezeigt. Handelt es sich bei der Auswahl des Dateipfades um eine Mehrfachmessung, dann arbeitet die Anwendung im Studienmodus. Die Inhalte aus der Info-Box sind die Gleichen wie zuvor, jedoch werden 2 weitere Inhalte hinzugefügt: Wie viele Messungen schon analysiert wurden und wie viele Messungen noch fehlen (siehe Abb.21). Der Pfad für den Input kann anhand von 2 Möglichkeiten ausgewählt werden: Entweder durch das Aktivieren des Input-Buttons neben dem Input-Feld und durch die Auswahl der Messung im Ordner oder durchs Aktivieren des „Open Recent“-Buttons, die schon mal für die Messung ausgewählt waren. Dabei steht in der Tabelle unten zusätzlich, in welchem Modus die Anwendung bei der Auswahl eines Pfades arbeiten wird. Sämtliche Pfade in der Tabelle sowie die Anzahl der analysierten und fehlenden Messungen werden als Mock-Daten angegeben (siehe Abb.22).



Select the data that you want to analyze it:

Input Folder:

Output Folder:

☐ Ausgangsdaten werden mitkopiert

| <input type="checkbox"/> Path | Modus |
|---|--------|
| <input type="checkbox"/> Checkverarbeiter.md | MULTI |
| <input type="checkbox"/> QM-QM-02-DE_Qualitaetspolitik.md | SINGLE |
| <input type="checkbox"/> Schrittmessung.md | MULTI |

Sie haben eine Mehrfachmessung ausgewählt.
 Der DataAnalyzer arbeitet im Studienmodus
 Dieser Ordner Checkverarbeiter.md wird für die Messung verwendet
 Es wurden schon insgesamt 9 Messungen durchgeführt.
 Es fehlen noch 14 Messungen.
 Die Ausgabe der Messungen werden in diesem Ordner gespeichert:

Abb. 22: Anzeige Anzahl der Messungen bei Auswahl des Pfades (wird in der Abbildung nicht richtig angezeigt)

Bei der weiteren Entwicklung der Seite sollte die Anzeige der Info-Box scrollbar sein und nicht weiter nach unten gezogen werden müssen, sonst wird die Anzeige der kompletten Anwendung scrollbar und das ist nicht erwünscht. Dabei sollen die Pfade für den Input und den Output nicht unterhalb, sondern innerhalb des Input- und Output-Feldes stehen. Die Validierung, wenn bspw. der Pfad für den Input-Feld nicht ausgewählt wurde, sowie die entsprechende Warnung in der Info-Box wurden aus zeitlichen Gründen weggelassen.

- **Step 2: OutputParameter**

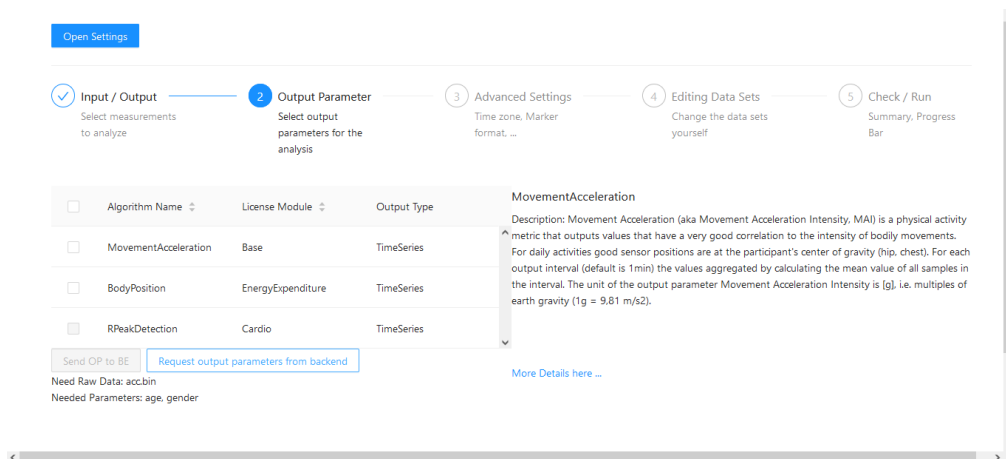


Abb. 23: Endzustand "OutputParameter"

In der Stufe „OutputParameter“ werden links in der Tabelle die einzelnen Ausgabeparameter mit der jeweils dazugehörigen Lizenz und den jeweiligen Ausgabebetyp angezeigt (Abb.23). Hier haben alle angegebenen Mock-Ausgabeparameter den Output-Typ „TimeSeries“ zugewiesen bekommen. Wenn man mit der Maus über die einzelnen Datensätze fährt (z.B. über den Ausgabeparameter „MovementAcceleration“), dann wird auf der rechten Seite die Beschreibung zu dem Algorithmus angezeigt. Mit dem Link „More Details here“ wird man automatisch zu der Dokumentationsseite des Produkts DataAnalyzer geleitet. Wird mind. 1 Algorithmus ausgewählt, dann wird der unten links stehende Button „Send OP to BE“ (Send OutputParameters to Backend) aktiviert und mit dem Button werden die ausgewählten Namen ans Backend geschickt. Über den Button „Request output parameters to backend“ werden die ganzen Mock-Daten in der Tabelle dargestellt. Die Inhalte zu „Need Raw Data“ und „Needed Parameters“ sind als Platzhalter zu betrachten.

Die beiden Buttons unterhalb der Tabelle sollen bei der weiteren Entwicklung ausgelassen werden. Die Parameter unterhalb der Buttons sollen einen Bezug zu den Ausgabeparametern haben. Außerdem soll die Info-Box scrollbar sein und die Anwendung nicht unnötig wie bei dem Schritt „Input/Output“ nach unten gezogen werden, sodass die komplette Seite scrollbar ist und die Buttons „Back“ und „Next“ auf der Seite nicht mehr sichtbar sind. Bei der Spalte „OutputType“ soll neben der Anzeige des Ausgabetyps auch ein Button vorkommen, der eine Vorschau über den Inhalt des Ausgabetyps anzeigen soll. Während der Entwicklung der Anwendung gab es eine weitere Möglichkeit, den Button sowie die Anzeige des Ausgabetyps zu vereinfachen.

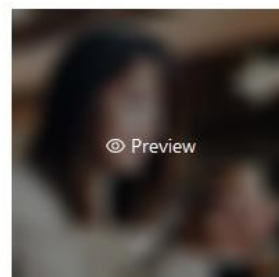


Abb. 24: Custom preview Image für Ausgabebetyp in "OutputParameter" (links) und Anzeige "Preview" durch Mouse-Hover (rechts)

Über die Komponente „Custom preview Image“ (Abb.24) kann man die Vorschau über den möglichen Output anhand entsprechender Dokumente wie PDF, Excel, usw. darstellen. Sobald man mit der Maus über das Bild fährt, wird das dazugehörige Bild leicht ausgegraut und in der Mitte des

Bildes erscheint der integrierte Button „Preview“. Dabei soll nach dem Drücken des „Preview“-Buttons nicht das Bild des Ausgabetyps, sondern ein Beispieldokument zum jeweiligen Ausgabetyp angezeigt werden.

- **Step 3 : Advanced Settings (am Beispiel „Algorithm config“)**

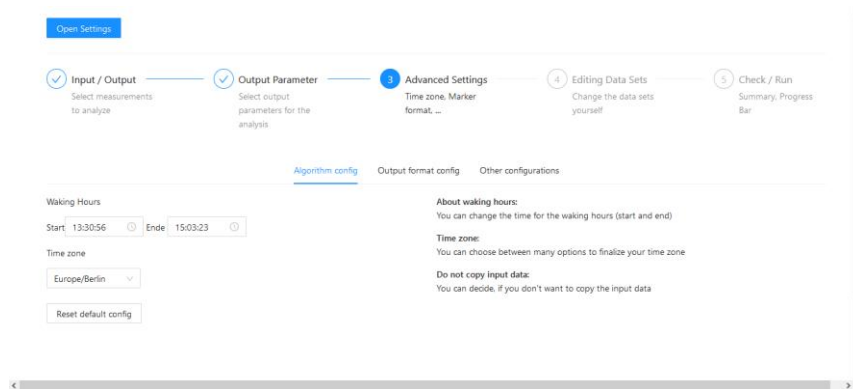


Abb. 25: Endzustand "Advanced Settings" am Beispiel 'Algorithm config'

Bei der Erstellung der Anwendung wird davon ausgegangen, dass der Expertenmodus über die Settings aktiviert ist. Bei der Seite „Advanced Settings“ können weitere Konfigurationen eingestellt werden, z.B. die Uhrzeit für den Wachzeitraum oder beispielweise die Zeitzone (Abb.25). Am Beispiel des „Algorithm config“ ist die Aufteilung ähnlich wie beim Step „Input / Output“: auf der linken Seite können sämtliche Einstellungen gemacht werden, auf der rechten Seite stehen weitere Informationen bezüglich der Konfigurationen in einer Info-Box drin. Die Funktion „Ausgangsdaten nicht mitkopieren“ muss, wie in Step „Input/Output“ beschrieben, hier eingebaut werden. Diese Funktion wurde bei der Überarbeitung leider vergessen. Mit dem Button „Reset default config“ sollen sämtliche Änderungen in „Advanced Settings“ auf die Standardeinstellungen zurückgesetzt werden.

Das ähnliche Prinzip wäre auch bei den anderen Rubriken wie „Output format config“ und „Other configurations“ gewesen. Dennoch besitzt dieser Button keine Funktionalität und wird nur im Frontend angezeigt. Im Tab „Other configurations“ sind bis zum Ende des Praxissemesters keine Konfigurationen eingebaut worden, da man bis zu dem Zeitpunkt keine Planung hatte, welche Inhalte in diesen Tab gehören. Die einzelnen Konfigurationen in den Tabs besitzen keine Funktionalität und werden nur im Frontend angezeigt. Zwar können die Uhrzeiten bei Start und Ende eingestellt werden, jedoch ohne den Bezug zu der Zeitzone und ohne ein bestimmtes Interval (z.B. von 05:00:00 bis 23:00:00). Die Beispiele zu den Zeitzonen sind Mock-Daten. Genauso sind die Inhalte für den Output-Interval und für die Zeitzone nur Mock-Daten. Generell können bei der weiteren Entwicklung der Anwendung in den weiteren Tabs „Algorithm config“ und „Output format config“ weitere Konfigurationen eingebaut werden.

- **Step 4 „Editing Data Sets**

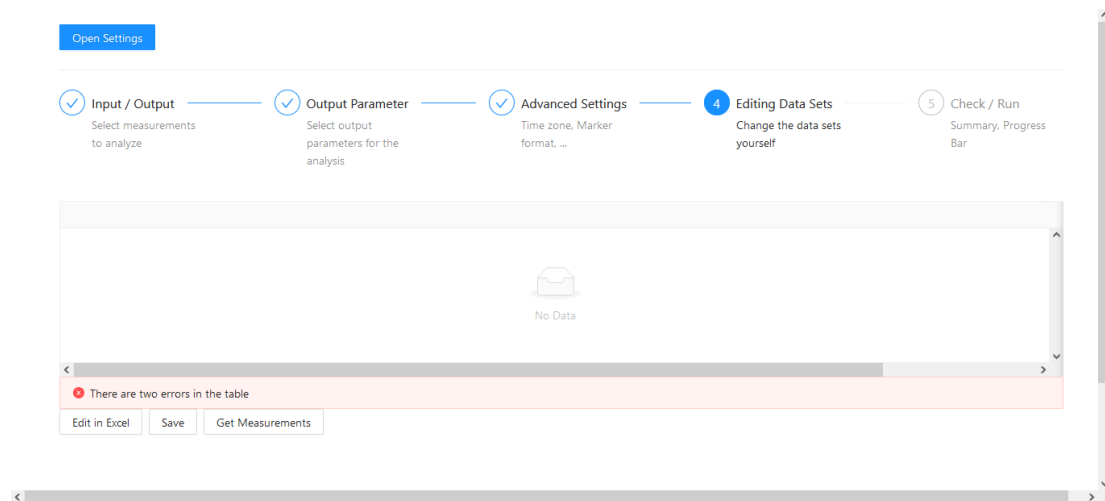


Abb. 26: Endzustand "Editing DataSets" (ohne Mock-Daten)

Bei dem Step „Editing Data Sets“ war ursprünglich geplant, die Mock-Daten, die im Backend erstellt wurden, in der Tabelle anzuzeigen. Durch einige Missverständnisse beim Datenmodell konnte der ursprüngliche Einbau der Mock-Daten nicht erfüllt werden und deshalb mussten die Inhalte aus der Tabelle entfernt werden (Abb.26).

Bei der Gestaltung der Tabelle wurde sich an dem Ant-Design-Feature „Editable Cells“ orientiert (siehe Abb.27). Dabei soll die Gliederung der Daten in 2 Bereiche erfolgen: unter der Rubrik „Measurement Parameter“ sollen der Startpunkt der Messung (MeasurementStart), die Id zu dem Datensatz (ID) sowie die Dauer der Messung (duration) abgebildet werden. Unter der Rubrik „Participant Parameter“ sollen sowohl die Sensor-Informationen (Sensornummer und Sensortyp) sowie alle benötigten Informationen zu der Person (wie Alter, Gewicht, Höhe und Geschlecht) abgebildet werden. Dabei dürfen die Spalten in „Measurement Parameter“ nicht abgeändert werden, bei „Participant Parameter“ darf nicht die Sensornummer des Sensors abgeändert werden.

| name | age | address | operation |
|----------|-----|-------------------|-------------|
| Edward 0 | 32 | London Park no. 0 | Save Cancel |
| Edward 1 | 32 | London Park no. 1 | Edit |

Abb. 27: geplante Anzeige der Mock-Daten in Form einer Tabelle

Bei allen Datenzeilen können in bestimmten Spalten die Werte der Daten abgeändert werden. Wenn die Ergebnisse anhand der Validierungen richtig sind, wird unterhalb der Tabelle eine Result-Zeile (mit grünem Hintergrund) angezeigt, die beschreibt, dass keine Fehler bei der Überprüfung der Daten aufgetreten sind. Sollten dennoch Fehler entstehen, dann erscheint die Result-Zeile im roten Hintergrund und beschreibt, wie viele Fehler es in der Tabelle gibt und wo sich die Fehler befinden. Unterhalb der Result-Zeile befinden sich 3 Buttons: mit dem Button „Edit in Excel“ soll die Überarbeitung der Datenwerte in Excel erfolgen, sofern die Überarbeitung der Daten nicht in der Anwendung erfolgen soll. Der Button „Save“ soll die überarbeiteten Daten anhand entsprechender Validierungen überprüfen und entweder die überarbeiteten Daten speichern, sofern keine Fehler aufgetreten sind oder Fehler anzeigen lassen und die Speicherung der überarbeiteten Daten abbrechen. Dabei soll der Button „Save“ die Spalte „operation“ überflüssig machen. Der Button „Get

Measurements“ soll die gleiche Funktion haben wie der Button „Request output parameters from backend“ aus dem Schritt „OutputParameter“, d.h. diese Funktion soll dazu dienen, die Daten vom Backend zu holen und die entsprechenden Inhalte im Frontend anzuzeigen.

- **Step 5 „Check/Run“**

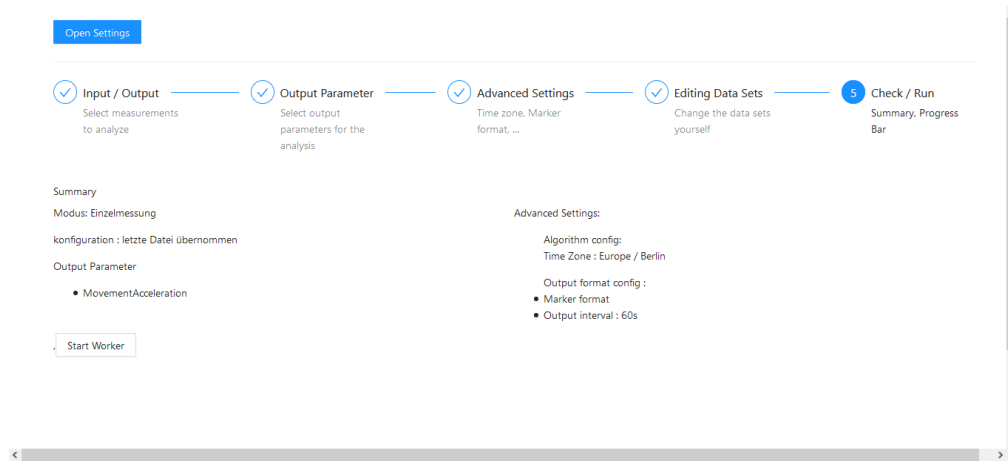


Abb. 28: Endzustand "Check/Run" (Zusammenfassung)

Beim letzten Schritt „Check/Run“ soll anhand der von oben dargestellten Info-Box die Zusammenfassung aller ausgewählten Konfigurationen wie Ordner, Parameter, Einstellungen, usw. dargestellt werden (Abb.28). Dabei sollte die Zusammenfassung bei Änderungen der vorherigen Schritte dynamisch reagieren. Die Daten für die Zusammenfassung sind Mock-Daten. Wenn man bei dem Schritt „Check/Run“ die Überprüfung einzelner Konfigurationen in den vorherigen Schritten abgeschlossen hat und versuchen will, die Analyse zu starten, dann drückt man auf den Button „Start Worker“. Sobald der Button gedrückt wurde, schaut sich das Backend die Daten an und versucht, die Konfigurationen, die in der Anwendung getätigt wurde, zu analysieren. Die Benutzeroberfläche ist nur dafür zuständig, dass die Analyse-Config (siehe „Datenmodell“) vollständig festgelegt wird. Während der Analyse der Messung erscheint unterhalb des Buttons „Zurück“ eine Progress-Bar, das von links nach rechts durchläuft und eine Prozentanzeige beinhaltet. Hat die Progress-Bar den Wert 100% erreicht, dann erscheint oben rechts nicht nur die Informationen zu einem „Sensor-Dummy“ (das auch nur aus Mock-Daten besteht), sondern es erscheint in der Mitte der Anwendung eine Notification-Box, das die erfolgreiche Analyse der Messung sowie weitere Buttons für die nächsten Schritte anzeigt.

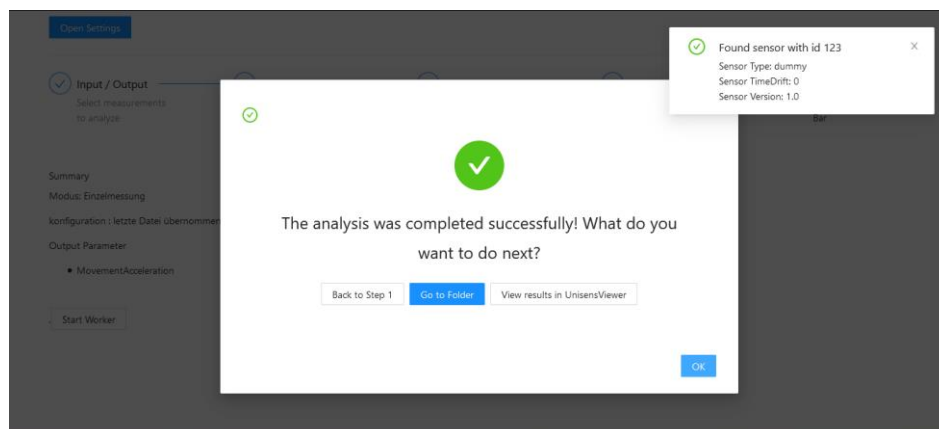


Abb. 29: Anzeige Benachrichtigung über erfolgreiche Analyse der Messung (oben rechts: Anzeige des Sensors)

Da die Anzeige der Info-Box laut Mockup auf der nächsten Seite geschehen soll, war zu Beginn unklar, wie anhand der Step-Anzeige die Notification eingebaut werden sollte, daher wurde aus Zeitgründen entschieden, die Info-Box als Notification zu verpacken (Abb.29). Bei der weiteren Entwicklung der Anwendung sollte neben der Gestaltung auch die dynamische Reaktion der geänderten Konfigurationen angezeigt werden. Bei mehreren Konfigurationen in „Advanced Settings“ sollte die Info-Box scrollbar sein, sodass die Buttons noch in der Seite der Anwendung gut zu sehen sind. Außerdem sollte die Notification-Feature durch eine separate Seite in der Anwendung ersetzt werden, da sie laut dem Wireframe so angegeben ist. Die Progress-Bar, die unterhalb der Seite „Check/Run“ dargestellt wird, sollte, wie im Wireframe angegeben, durch eine weitere Progress-Bar ausgebaut werden und wie bei der Notification-Box auf eine zusätzliche Seite der Anwendung ausgelagert werden. In dieser Seite fehlen außerdem sämtliche Validierungstests für die Analyse der Messung und ein Cancel-Button für den Abbruch der Analyse. Dabei sollte die Analyse auch selbstständig abgebrochen werden, wenn irgendwelche Fehler bei der Analyse auftauchen sollen.

6. Anhang

- Die einzelnen Abbildungen sind durch das Tool Pencil Project sowie einige Schaubilder aus MS Teams entstanden.
- Die Bilder aus „5.2. Endzustand der Anwendung am Ende des Praxissemesters“ sind durch Screenshots am Laptop entstanden und durch MS Paint überarbeitet.
- Bilder:
 - <https://docs.movisens.com/DataAnalyzer/#viewing-measurement-data>
 - <https://docs.movisens.com/DataAnalyzer/#checking-and-altering-participant-data>
 - <https://ant.design/components/table/#components-table-demo-edit-cell>
 - <https://ant.design/components/image> => Custom Preview Image
 - <https://www.hausarzt-armbruster.de/hausarzt-wAssets/img/untersuchungsbilder/weblication/wThumbnails/ekg-hausarzt-armbruster-6c2aac0f558562cg5406d372efc1d842@2x.jpg>
 - https://www.movisens.com/wp-content/uploads/product_ecgmove_worn.jpg
- Text :
 - Movisens DataAnalyzer Dokumentation ab dem Punkt „Physical activity“ => https://docs.movisens.com/Algorithms/physical_activity/
 - Movisens DataAnalyzer Doku => <https://docs.movisens.com/DataAnalyzer/>
 - Ausbildungsplan
 - Präsentation DataAnalyzer: [Planung DataAnalyzer2\(2\).pptx](#)
 - Serenity => <https://serenity-js.org/>
 - Info EcgMove4: <https://www.movisens.com/de/produkte/ekg-sensor/>
 - Datenmodell: <https://de.wikipedia.org/wiki/Datenmodell>
 - Ambulantes Assessment : https://de.wikipedia.org/wiki/Ambulantes_Assessment
 - EKG : <https://www.apotheken-umschau.de/diagnose/diagnoseverfahren/ekg-das-zeigt-die-elektrokardiografie-an-742927.html>