Praktische Evaluation des Frameworks Laravel Nova am Beispiel einer Anwendung zur Verwaltung und Auswertung von Radiosondenaufstiegen

Exposé im Rahmen des Praxisprojektseminars
an der Fakultät für Informatik und Ingenieurwissenschaften
der Technischen Hochschule Köln

vorgelegt von:	Niklas Canisius
Matrikelnummer:	11110023
Adresse:	Gelpestraße 96 51647 Gummersbach niklas@canisius.email
eingereicht bei:	Prof. Dr. Kohls

Gummersbach, 13. September 2022

1 Problemfeld & Kontext

Bisherige individuelle Projekte des Softwareunternehmens Kiwis & Brownies GbR zeigten einen anfänglich großen und repetitiven Aufwand bei der Erstellung unterschiedlichster Software auf. Fast alle Projekte benötigen einen Login, die Verwaltung von Entitäten in einem sogenannten CRUD Interface, verschiedene Auswertungen, sowie spezielle Abläufe für ein oder mehrere Entitäten.

2 Ziel

Im vorliegenden Projekt soll das Framework Laravel Nova verwendet werden, um den Aufwand der Entwicklung zu reduzieren, gleichzeitig aber keine Flexibilität bei der Umsetzung zu verlieren. Die Wahl des Frameworks ergab sich auf Basis von Erfahrungen im Unternehmen, sowie Kundenwünschen. Das System soll webbasiert und auf Basis von weit verbreiteten und möglichst quelloffenen Frameworks umgesetzt werden. Das Unternehmen hat in der Vergangenheit primär Kompetenzen in der Entwicklung mit PHP und dem Framework Laravel gesammelt, beide erfüllen die Projektanforderungen. Als First-Party Paket ist daher die Wahl auf Laravel Nova gefallen.

3 Aufgabenstellung

Die Hauptaufgabe liegt in der Implementation des Systems mit allen vorgegebenen Funktionalitäten (siehe Lastenheft und Leistungspaket im Anhang). Dabei soll sich zeigen, inwiefern Laravel Nova die Arbeit erleichtert und welche Einschränkungen sich ergeben. Außerdem könnte eine Literaturrecherche zeigen, ob es bisherige Arbeiten gibt, die eine ähnliche Problematik beleuchten.

3.1 Forschungsfrage

Welche Vor- und Nachteile bietet der Einsatz eines Administration-Panels am konkreten Beispiel von Laravel Nova? Verglichen wird mit einer Entwicklung auf Basis von Laravel, ohne Nova.

4 Lösungsansätze

Die Beantwortung der Forschungsfrage soll sich durch den praktischen Einsatz der Technologie an einem Projektbeispiel zeigen. Um das dabei gewonnene Fazit zu überprüfen, könnte in einer späteren Erweiterung, z.B. im Rahmen der Bachelorarbeit, ein weiteres Projekt oder Erweiterungen mit dem Framework umgesetzt werden.

5 Chancen & Risiken

Der Einsatz des zu überprüfenden Frameworks bietet die Chance einer schnelleren und stringenteren Entwicklung der Software mit dem Fokus auf die wirkliche Anwendungslogik und nicht auf grundlegende Funktionalitäten/Bausteine beziehungsweise das Programmiergerüst. Ein mögliches Risiko hingegen ist die Einschränkung in der Umsetzung durch die fehlende Anpassbarkeit der Software.

Problematisch ist der Projektkontext in einem Unternehmen. Hier kommt ein wichtiger Stakeholder mit ins Boot und daher gibt es möglicherweise im Laufe des Projektes Konflikte zwischen wirtschaftlichen und wissenschaftlichen Anforderungen.

Ein Problem liegt außerdem in der Erarbeitung des Fazits. Es ist zu klären, nach welchen Metriken die Vor- und Nachteile erhoben werden und mit welcher Gewichtung sie in das Gesamtfazit eingehen.

6 Ressourcen, Setup & Abhängigkeiten

Eine erste Literaturrecherche bei Google Scholar brachte nur wenige und nicht zufriedenstellende Ergebnisse.
https://gitlab.com/graw-radiosondes/sounding-console/-/wikis/04-Research

Die Technologie ist bereits recht konkret festgelegt, Änderungen daran sind unwahrscheinlich und ergeben sich nur evtl. im laufenden Projekt.

Prüfer und Betreuer an der Technischen Hochschule ist *noch zu klären.*

Kooperationspartner des Projektes ist die Kiwis & Brownies GbR mit Sitz in Gummersbach. Verantwortlicher im Unternehmen ist Benjamin Braun und technischer Betreuer ist Keanu Buschbacher.

Der Hauptteil der Arbeit wird bei Kiwis & Brownies im Office oder Homeoffice absolviert.

7 Motivation

Persönlich motiviert mich die Herausforderung ein neues Framework kennenzulernen und einzusetzen. Zudem ergibt sich aus meiner bisherigen Arbeit im Unternehmen ein starkes Interesse an einem solchen Framework, da es unternehmensintern eine Eigenentwicklung gibt, die im Prinzip das gleiche Problemfeld bearbeitet. Daher könnte eine mögliche Folgearbeit, z.B. im Rahmen der Bachelorarbeit, der Vergleich der Beiden Lösungen Brezel und Laravel Nova sein.

Durch das Studium der Medieninformatik besitze ich die technischen Kompetenzen für die Entwicklung eines solchen Systems und bin interessiert diese einzusetzen. Vor allem die notwendigen Kompetenzen im Bereich Softwaretechnik und Webtechnologien, welche durch die entsprechenden Module im Schwerpunkt Web-Development vermittelt wurden.

8 Arbeitsergebnis

Das Ergebnis ist zum Hauptteil die Implementierung der Software sowie deren Dokumentation. Außerdem ist das Fazit über die Vor- und Nachteile des Frameworks ein zentrales Ergebnis der Arbeit.

9 Meilensteine

Die Meilensteine und ihre konkreten Aufgaben sind im privaten Repository bei GitLab definiert. Bei Bedarf wird Einsicht gewährt.
https://gitlab.com/graw-radiosondes/sounding-console/-/milestones
https://gitlab.com/graw-radiosondes/sounding-console/-/boards

10 Sperrvermerk

11 Anhang

11.1 Lastenheft

Lastenheft Sounding Console

Einführung

Das Unternehmen Graw Radiosondes ist einer der weltgrößten Anbieter von Radiosonden-Aufstiegssystemen. Radiosonden sind meteorologische Messinstrumente, die von Wetterballons getragen, Messwerte wie Temperatur (T), Luftfeuchtigkeit (U), Luftdruck(P)/Position/Wind abtasten und per Funk an eine Bodenstation übermitteln.

Ein Kundenstamm von Graw sind Wetterdienste, die innerhalb ihres Hoheitsgebiets eine Reihe von Aufstiegsstationen betreiben, an denen händisch ein- oder mehrmals am Tag zu international festgelegten Uhrzeiten Radiosondenaufstiege durchgeführt werden. Aufstiegsstationen sind regelmäßig an Orten eingerichtet, die infrastrukturell schlecht angebunden sind.

Zahlreiche Wetterdienste haben die händische Durchführung von Starts eingestellt, und starten Wetterballons automatisiert durch durch sog. Autolauncher. Dieses Marktsegment hat Graw derzeit nicht erschlossen, plant jedoch mittelfristig einen Markteinstieg.

Graw Radiosondes stellt derzeit z.B. die Hardware für den Betrieb des kanadischen, türkischen sowie eines großen Teils des US-amerikanischen und indischen Messnetzes.

Radiosonden übermitteln ein Telemetriesignal, was zeitlich üblicherweise mit einem Abtastintervall von 1 s elektronische Größen enthält, aus denen durch Formelzusammenhänge die physikalischen Parameter (PTU) der Sensoren der Sonde berechnet werden können (diese werden im folgenden als Rohdaten bezeichnet). Weiterhin sind im Telemetriesignal auch GPS/GNSS-Daten und weitere Statusdaten enthalten. Der Telemetriedatenstrom wird in der Bodenstation, die sich üblicherweise am gleichen Ort wie die Aufstiegsstation befindet, empfangen und dekodiert. Die berechneten Rohdaten werden mit einer instrumentenabhängigen Korrektur versehen, um den Zustand der Atmosphäre zum Zeitpunkt des Durchflugs anzunähern. Die dabei entstehenden Werte werden als Datenprodukt bezeichnet.

Für die weitere Verwendung der Aufstiegssdaten ist jedoch das Datenprodukt nur selten geeignet. Historisch entstanden sind verschiedene Reports/Messages die durch Organisationen (z.b. WMO, NATO) standardisiert sind, und die zum Teil noch aus der Zeit stammen, als Radiosondenaufstiege von einer Gruppe Meteorologen händisch in Echtzeit ausgewertet, und die Messergebnisse über Funk oder Fernschreiber weitergegeben wurden. Diese Datenformate haben üblicherweise die

Form von Textdateien, binären Dateien oder gesetzten Dokumenten und werden über verschiedene Schnittstellen (z.b. FTP) für weitere Nachnutzung bereitgestellt.

Neben der Bereitstellung für andere Nationen im WMO Messnetz werden die Daten als Grundlage für nationale Wettermodelle genutzt. Die Berichte verschiedener Stationen werden hierfür eingelesen, validiert und mit Daten aus weiteren Quellen (synoptische Messstationen, Satelliten) in ein für das Modell geeignetes Ausgangsformat assimiliert. Aus den Ergebnissen verschiedener Modellläufe werden dann Wettervorhersagen gebildet.

Auch Funkamateure empfangen die Telemetriedaten von Radisosondenaufstiegen, vor allem um ein freizeitmäßiges Wiederauffinden der Aufstiegsgespanne nach der Landung zu ermöglichen. Hierfür existieren Open-Source Softwarelösungen zum Empfang, sowie zum Darstellen der Telemetriedaten über Webportale (sondehub.org, radiosondy.info).

Beschreibung und Bewertung des Ist-Zustandes

Bodenstationen von Graw sind softwaretechnisch (GrawMet) als C# WPF Anwendung realisiert, die unter Windows laufen. Innerhalb einer einzelnen Anwendung werden alle relevanten Prozesse der Bodenstation (Datenempfang, Berechnung der Telemetriedaten aus Rohdaten, Visualisierung, Berichterstellung) abgebildet. Die Möglichkeit, einen vergangenen Aufstieg erneut zu simulieren ist vorhanden (vor allem zu Diagnosezwecken), ebenso wie rudimentäre Statistikfunktionen.

Bodenstationen sind mit einer Möglichkeit ausgestattet, Telemetriedaten von Aufsteigen über eine TCP/IP Schnittstelle an eine Google Firebase Anwendung (GrawGo) weiterzuleiten, die eine Fernüberwachung von Flügen einzelner Stationen über Smartphones mithilfe einer dedizierten App ermöglicht. Die GrawGo Anwendung befindet sich derzeit im Experimentalbetrieb und hat keine produktiven Nutzer im betrachteten Marktsegment von Messnetzen.

Der gewählte Ansatz, alle Aspekte eines Sondenaufstiegs in einer Software zu vereinen, die zudem auf Anwendungs-PCs der Nutzer laufen, ist historisch entstanden und kritisch zu bewerten. Im betrachteten Marktsegment der Messnetze ergeben sich vor allem folgende Probleme:

- wenig qualifizierte Anwender an einzelnen Stationen sind mit kritischen Tätigkeiten bei der Erstellung des Datenprodukts und der Erstellung der Berichte betraut
- Überwachung mehrerer Stationen ist kaum möglich

- Die Topologie der Software ist kaum für den autonomen Einsatz in fernüberwachten Autolaunchern geeignet

Vor allem die ersten beiden Punkte führen zu einer hohen Rate von zurückgewiesenen Berichte von Graw Stationen bei Wetterdiensten. (citation needed, vermutlich von Bruce Ingleby)

Beschreibung der Angebote anderer Marktteilnehmer

Für alle anderen Marktteilnehmer, außer Vaisala, stellt sich die Situation vergleichbar zu der von Graw da.

Über den technische Realisation der Steuerung der Autolauncher von zwei Marktteilnehmern, Modem und Meisei, sind keine Informationen bekannt.

Vaisala hat einen vergleichbaren Aufbau der Bodenstation, stellt die UI in der neuesten Revision (MW51) jedoch über Web anstelle WPF dar. Weiterhin hat Vaisala ein Webportal NM10 im Angebot, welches die Fernüberwachung aller meteorologischer Produkte (darunter manuelle Radiosondenstationen und Autolauncher) ermöglicht. Über die genaue Architektur dieses Systems und die Schnittstellen zu der Bodenstation sind keine Informationen bekannt. NM10 nutzt eine UI, die Vaisala von ca. 2010-2018 verwendet hat und hat daher seit dieser Zeit verm. keine größeren Redesigns erhalten.

Beschreibung des Soll-Konzepts

Es soll ein Webportal entwickelt werden, welches die Roh- oder Telemetriedaten teilweise in Echtzeit von einzelnen Stationen erhält, verarbeitet, speichert, und visualisiert. Das Webportal soll sowohl in der Cloud als auch On-Premises gehostet werden können. Die Erstellung, Validierung, und Weiterleitung von Berichten soll über das Webportal möglich sein.

Den oben benannten Probleme soll durch Einführung eines zentralen Administrationsportals, ähnlich wie Vaisalas NM10, begegnet werden. Hierzu sollen die Daten von Bodenstationen in Echtzeit an eine Sounding Console (SC) gesendet werden. In der SC können alle relevanten Aufgaben zentralisiert erledigt werden.

Die Funktionalitäten der SC lassen sich dabei in folgende Kategorien einteilen:

- Erzeugung von Datenprodukten aus Rohdaten

- Erzeugung von Berichten aus Datenprodukten
- Parametrierung von Autolaunchern
- Automatisierung (der bis jetzt genannten Punkte)
- **Echtzeit-Überwachung von Bodenstationen und Autolaunchern**
- **Statistik und Performance-Monitoring von Bodenstationen und Autolaunchern**

Als Prototyp (und ggf. Minimum Viable Product) sollen zunächst die **letzten beiden Punkte**, unter Verwendung des auf der Bodenstation generierten Datenproduktes, genutzt werden.

Die nachfolgenden Angaben beziehen sich ausschließlich auf diesen Prototyp.

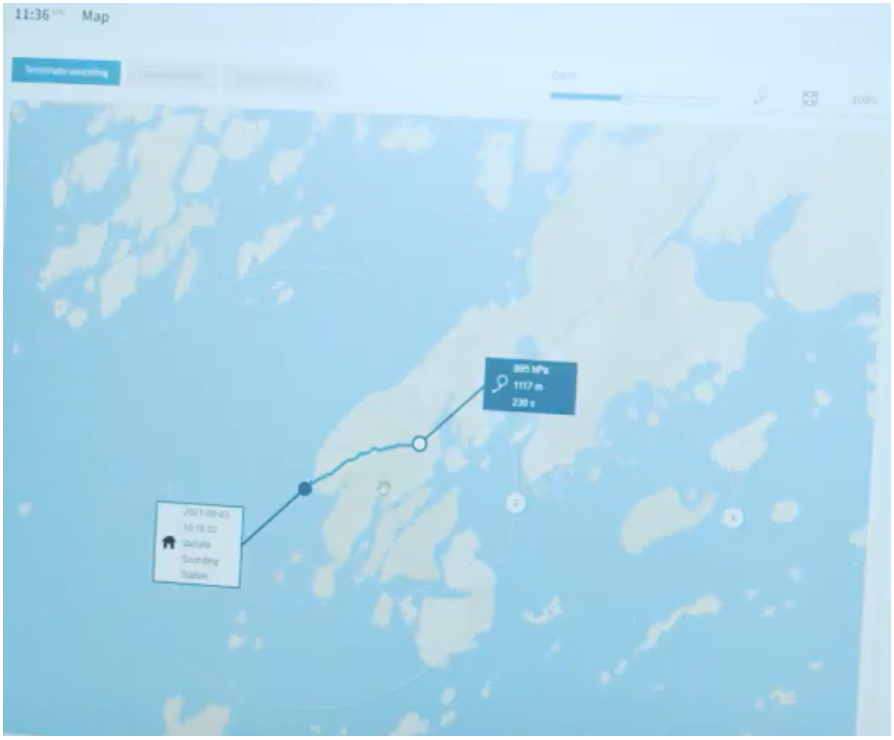
Beschreibung von Schnittstellen

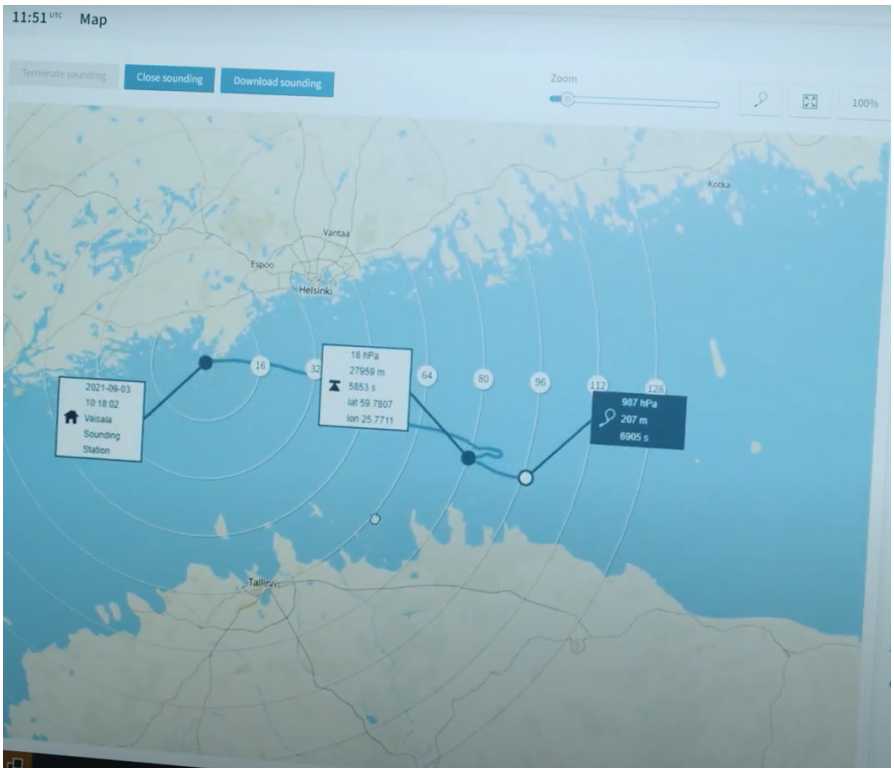
Für die Echtzeit-Übertragung zwischen GrawMet und SC ist eine Nutzung der bisherigen Schnittstelle zwischen der Bodenstation (GrawMet) und Firebase für den Prototyp zu prüfen. Firebase selbst kann als Datenbank nicht verwendet werden, da dies das Soll-Konzept, konkreter ein mögliches On-Premise Hosting, nicht erfüllt.

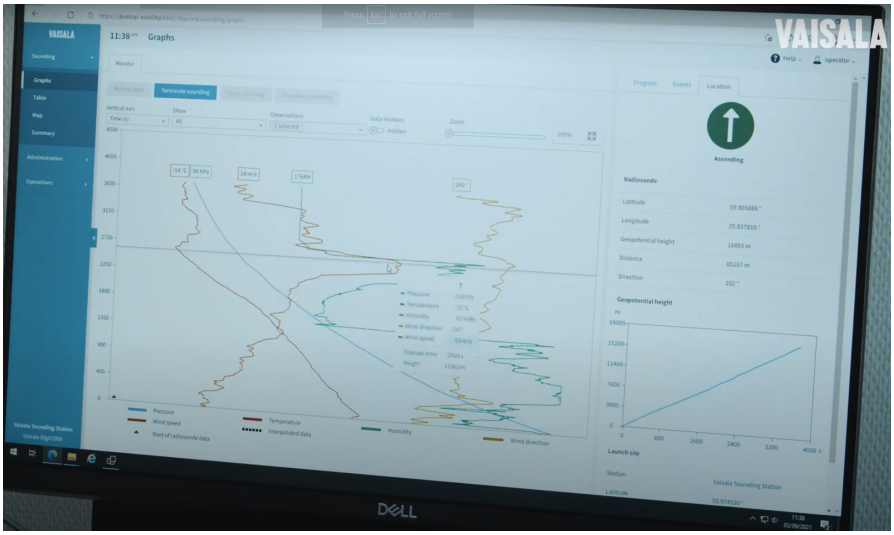
Weiterhin ist zu prüfen, ob ein bereits vorhandenes (bspw. CSV- oder XML-basiertes), oder einfach in GrawMet zu implementierendes Dateiformat für den Import gesamter Flüge in SC nutzbar ist; neben der Verwendung als Software-Feature wäre eine solche Funktionalität auch für Software-Testing (Seeding der Datenbank) wünschenswert.

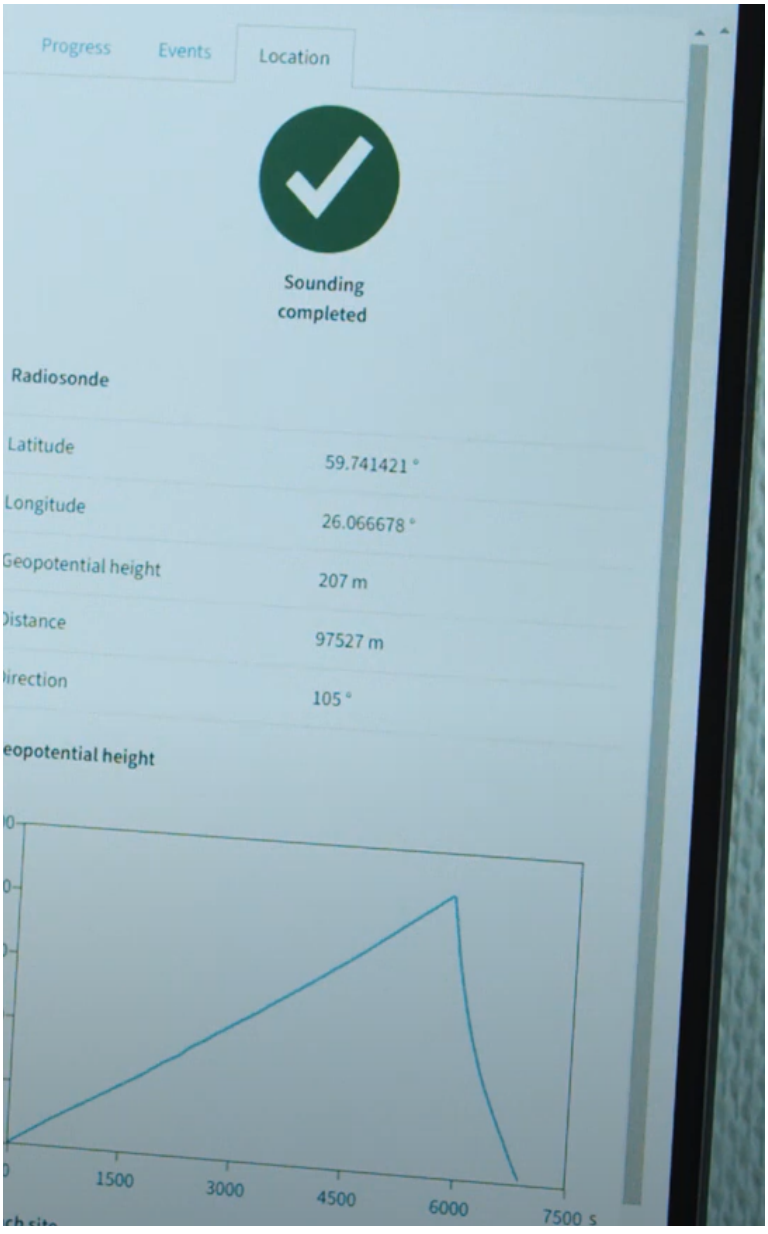
Funktionale Anforderungen

- Rechtesystem
    - welcher Nutzer kann welche Stationen sehen
    - Erweiterbar auf Rechteebenen pro Station
- Internationalisierung vorbereitet, im Auslieferungszustand nur Englisch
- Zuführung der Daten entweder
    - in Echtzeit über HTTP-API
    - oder durch Upload von Archivdaten (CSV/XML)
        - eine CSV Datei je Launch
- Liste von gespeicherten Aufstiegen von Stationen
- Berechnung eindimensionaler Performancekriterien (maximal 10, eher weniger) eines Aufstieges
    - Darstellung als zeitliche Statistik über mehrere Stationen hinweg

- Darstellung eines Aufstieges vergleichbar zu Vaisala MW51
  https://www.youtube.com/watch?v=W7hJPMUdmXA
  https://www.youtube.com/watch?v=_VT4YerjEmc
    - 4 Views
        - Map
            - Prüfen ob Code, insbesondere für Kartendarstellung, von der Open-Source Visualisierung https://github.com/projecthorus/sondehub-tracker nutzbar ist.
            - 

            - 

- Tabelle Datenproduktwerte
- Graphen Datenpoduktwerte vs Zeit/Höhe/Luftdruck
  - 

- Dashboard mit Status
  - 

Nichtfunktionale Anforderungen

- Neue Schnittstellen vorzugsweise RESTful
- Backend basierend auf Web Technologien: PHP + Laravel
- Berichterstellung und wissenschaftliche Berechnungen Python
- Installation über Docker/Kubernetes, inklusive relevanter Skripte und Anleitungen

Lieferumfang und IP-Management

- Beispielinstallation auf einem Server von Graw
- Sourcecode als Git-Repository, einschließlich funktionaler Dokumentation als Wiki
    - ggf. teilweise Open Source (vorbehaltlich Kompatibilität mit Vermarktungsstrategie)
- IP übergeht soweit wie möglich auf Graw

11.2 Leistungspaket

Last edited by Niklas Canisius Jul 14, 2022, 5:06 PM

03 Leistungspaket

Arbeitstitel

> Graw Sounding Console

Generelles

Technisch

Anwendung

- Webbasiert
- PHP
- Laravel
- Laravel Nova - Unlimited license
- Vue.js
- Websockets

Hosting

- Cloud: containerbasiert (z.B. Docker, Kubernetes)
  - nach möglichkeit: komplett IaC via Terraform
  - wenn serverless, dann evtl. https://vapor.laravel.com/
- On-Premises
  - evtl. https://forge.laravel.com/

Inhaltlich

- Aufstiegsdatenempfang von Bodenstationen
- Echtzeitverarbeitung von Aufstiegen
- Archivierung von Aufstiegen
- Echtzeitüberwachung von Bodenstationen
  - Statistiken: z.B. wie viele erfolgreiche Aufstiege
  - Performance-Überwachung: z.B. Durchschnittliche Höhe aller Aufstiege

Sprache

- Umsetzung: einsprachig in englischer Sprache
- Vorbereitung: spätere Übersetzung

Models

User

name	type	special
id	unsignedBigInteger	autoIncrementing
created_at	timestamp	nullable
updated_at	timestamp	nullable
deleted_at	timestamp	nullable
created_by	unsignedBigInteger	nullable
updated_by	unsignedBigInteger	nullable
deleted_by	unsignedBigInteger	nullable
email_verified_at	timestamp	nullable

name	type	special
rememberToken	string	nullable; length(100)
email	string	unique
password	string	
role	string	
name	string	
timezone	string	

Datenquelle

- Erstellung im Backend durch Admin

Station

name	type	special
id	unsignedBigInteger	autoIncrementing
created_at	timestamp	nullable
updated_at	timestamp	nullable
deleted_at	timestamp	nullable
created_by	unsignedBigInteger	nullable
updated_by	unsignedBigInteger	nullable
deleted_by	unsignedBigInteger	nullable
wmo_id	unsignedBigInteger	nullable
publicly_readable	boolean	
name	string	
city	string	
country	string	
latitude	decimal	
longitude	decimal	
altitude	decimal	
weitere Stammdaten	various	

Datenquelle

- Erstellung im Backend durch Admin

Flight

name	type	special	comment
id	unsignedBigInteger	autoIncrementing	
created_at	timestamp	nullable	
updated_at	timestamp	nullable	

name	type	special	comment
deleted_at	timestamp	nullable	
created_by	unsignedBigInteger	nullable	
updated_by	unsignedBigInteger	nullable	
deleted_by	unsignedBigInteger	nullable	
station_id	unsignedBigInteger	relationTo(Station)	
sonde_serial	string		
set_frequency	unsignedDecimal	between 400 & 406	
sonde_firmware_version	string		
max_altitude	decimal	nullable	maximale Höhe
min_pressure	unsignedDecimal	nullable	minimaler Luftdruck
max_distance	unsignedInteger	nullable	größte Distanz zur Bodenstation
duration	unsignedInteger	nullable	Flugdauer
last_altitude	decimal	nullable	Höhe der Sonde beim letzten empfangenen Datenpaket
max_wind_speed	unsignedDecimal	nullable	maximale Windgeschwindigkeit
avg_wind_speed	unsignedDecimal	nullable	durchschnittliche Windgeschwindigkeit
avg_ascent_speed	decimal	nullable	Durchschnitt aller positiven vertical_speed
avg_descent_speed	decimal	nullable	Durchschnitt aller negativen vertical_speed

Datenquelle

- Erstellung im Backend durch Admin
- Erstellung im Backend durch StationAdmin
- Erstellung durch Bodenstation über HTTP Endpunkt
- Import aus einer oder mehreren XML/JSON
  - evtl innerhalb einer ZIP, z.B. mit https://gildas-lormeau.github.io/zip.js

Measurement

Achtung: Unterscheidung Real-Time Daten und Profildaten: Feuchtigkeit ist bis zu 120 sek verspätet (humidity_raw > humidity_timelag)

name	type	special
id	unsignedBigInteger	autoIncrementing
created_at	timestamp	nullable
updated_at	timestamp	nullable
deleted_at	timestamp	nullable
created_by	unsignedBigInteger	nullable
updated_by	unsignedBigInteger	nullable
deleted_by	unsignedBigInteger	nullable
flight_id	unsignedBigInteger	relationTo(Flight)

name	type	special
time_after_launch	unsignedInteger	
utc_time	time	
pressure	unsignedDecimal	
temperature	decimal	
humidity_raw	unsignedInteger	
humidity_timelag	unsignedInteger	will be sent as an update each 60-120 sec
wind_speed	unsignedDecimal	
wind_direction	unsignedInteger	
latitude	decimal	
longitude	decimal	
altitude	decimal	
vertical_speed	decimal	either calculated from altitude or sent, not clear yet
geo_potential	unsignedInteger	
dew_point	decimal	
elevation	unsignedDecimal	
azimuth	unsignedInteger	
distance	unsignedInteger	

Datenquelle

- Import über einen Flight
- Erstellung durch Bodenstation über HTTP Endpunkt
- Update durch Bodenstation über HTTP Endpunkt

HTTP Endpoints

Unterstützung in GRAWMET muss noch gebaut werden, die Endpunkte werden aus Sicht des Web-backends definiert

Technisch

- HTTPS
- Auth (noch zu klären)
- Content- & Response-Type JSON

Inhaltlich

- create Flight
- create Measurement
- update humidity_timelag in Measurements

Roles & Permissions

if a Station is publicly_readable=true, every role can read the Station, its Flight & Measurements

Admin

- CRUD User
- CRUD Station
- CRUD Flight
- CRUD Measurement

StationAdmin

- C User (with Role User and min one Station)

Limited by assigned Station

- RUD User
- RU Station
- CRUD Flight
- CRUD Measurement

Limited to own User

- RU User

User

Limited by assigned Station

- R Station
- R Flight
- R Measurement

Limited to own User

- RU User

Visualisierung & Berechnung

- Flight
  - Dashboard
    - during flight: flight dashboard
      - Status der Sonde: noch im Aufstieg oder bereits geplatzt
      - Startzeit & bisherige Flugzeit
      - Frequenz
      - Seriennummer
      - Typ der Sonde (aus firmware version abgeleitet)
      - Station
      - Aktuelle PTU/Wind Messwerte
        - Pressure
        - Temperature
        - Humidity
        - Wind direction
        - Wind speed
      - Aktuelle Position
        - Lat
        - Long
        - Höhe (geopot)
        - Entfernung zur Bodenstation
    - after flight:
      - Status der Sonde: finished
      - Statistiken (eindimensional)
        - max altitude
        - max distance
        - *weiteres folgt*
  - Karte
    - Auf Basis von sondehub-tracker
    - Echtzeitdarstellung via *WebSockets oder Resource Polling (über Prototyp klären)*
    - *Zu Klären und Präzisieren: Durch die Echtzeitdaten werden selektierte Flights auch live auf der Karte getrackt. Neben der Kartendarstellung von Sondendaten wird innerhalb einer Station auch die Karte mit allen Measurements bereitgestellt. Ebenso kann mit der gleichen technischen Basis eine Karte für alle Flights einer Station oder alle Flights des gesamten Systems implementiert werden.*
  - Tabelle aller Measurements (Messwerte & Datenprodukte)
    - Echtzeit durch Resource Polling
  - Diagramme (mehrdimensional)
    - Liniendiagramme
      - alle Messwerte nach Zeit oder nach Höhe
      - filterbar nach Messwert
      - eventuell filterbar nach Zeit
    - eventuell auch Skew-T
- Station
  - Stammdaten
  - Zusammenfassung der statistischen Indikatoren der zugehörigen Flights in Durchschnittswerten, filterbar nach Zeit
    - eventuell auch in einigen *noch zu definierenden* Zeitabschnitten immer angezeigt

Weiteres

- Test-Umgebung auf Kiwis & Brownies Cloud-Architektur
- Finales Produkt wird einmalig unverschlüsselt auf einem passenden Server bei Graw eingerichtet
- Codeverwaltung in GitLab Organisation von Graw, andere Plattformen nur als Mirror
    - Dies ermöglicht auch eine Mischform aus Closed Source Entwicklung, bei der man manuell nach Prüfung von einem closed GitLab Repo in ein open GitHub Repo mirrored.