**Telemetry Gateway Monitoring**



Maciej Dzialoszynski, Sebastian Grünewald, Jordi Rieder und David Jovanovic

|  |  |  |
| --- | --- | --- |
|  | **Name** | **Datum** |
| **erstellt** | Maciej Dzialoszynski | 06.03.2019 |
| **geprüft** | Sebastian Grünewald | 17.03.2019 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version** | **Autor** | **QS** | **Datum** | **Status** | **Kommentar** |
| 1.0 | Maciej Dzialoszynski | SG | 06.03.2019 | Init, Finish | Frontend, Backend |
| 1.1 | Sebastian Grünewald | MD | 17.03.2019 | Draft | Frontend |
| 1.2 | Jordi Rieder | DJ | 17.03.2019 | Draft, Finish | Backend |

**Inhaltsverzeichnis**

[**1** **Evaluationssprint** 3](#_Toc3732802)

[**1.1** **Evaluierung Frontend** 3](#_Toc3732803)

[1.1.1 Einleitung 3](#_Toc3732804)

[1.1.2 Kriterien 3](#_Toc3732805)

[Akzeptanz 3](#_Toc3732806)

[Dokumentation 3](#_Toc3732807)

[Umfang 3](#_Toc3732808)

[1.1.3 Optionen 3](#_Toc3732809)

[React 3](#_Toc3732810)

[Vue 3](#_Toc3732811)

[1.1.4 Nutzwertanalyse 4](#_Toc3732812)

[1.1.5 Fazit 4](#_Toc3732813)

[**1.2** Technologien und Getting Started 4](#_Toc3732814)

[1.2.1 React 4](#_Toc3732815)

[1.2.2 SASS und Koala 5](#_Toc3732816)

[**1.3** **Evaluierung Backend** 6](#_Toc3732817)

[1.3.1 Einleitung 6](#_Toc3732818)

[1.3.2 Kriterien 6](#_Toc3732819)

[Akzeptanz 6](#_Toc3732820)

[Dokumentation 6](#_Toc3732821)

[Performance 6](#_Toc3732822)

[1.3.3 Optionen 6](#_Toc3732823)

[NodeJS 6](#_Toc3732824)

[PHP 6](#_Toc3732825)

[Python 6](#_Toc3732826)

[1.3.4 Nutzwertanalyse 7](#_Toc3732827)

[1.3.5 Fazit 7](#_Toc3732828)

[**1.4** **Gesamtfazit** 8](#_Toc3732829)

[**1.5** **Abnahme** 8](#_Toc3732830)

# **Evaluationssprint**

## **Evaluierung Frontend**

### Einleitung

Das Ziel dieser Evaluierung ist es, die passende Frontend-Technologie fürs Projekt zu bestimmen. Unser Webinterface soll kartenmäßig (einzelne Module in Form von Karten) aufgebaut sein. Hierbei eignen sich die JavaScript – Frameworks React und VueJS sicherlich am besten. Zwischen diesen soll nun entschieden werden.

Weiters ist ein „Technlogien und Getting Started“ Kapitel vorhanden, welches einen schnellstmöglichen Einstieg ermöglichen soll und zusätzlich auch die verwendeten Technologien und Programmstrukturen erläutert.

### Kriterien

### Akzeptanz

Wie gut kommt das Team mit der Technologie zurecht? Hierunter fallen bereits gemachte Erfahrungen, die Struktur und Syntax des Frameworks und mögliche Hilfestellungen durch eine große und vor allem aktive Community und/oder durch die Ansprechpersonen auf Seiten des Auftraggebers.

### Dokumentation

Wie gut ist eine Technologie dokumentiert? Gibt es ausreichend Material, um sich mit jener zurecht zu finden und ist dieses zum einen ausführlich und zum anderen kompakt zusammengefasst verfügbar?

### Umfang

Das Framework soll ausreichend Funktionalität bieten, damit die kommenden Aufgabenstellungen gelöst werden können. Hierbei ist jedoch „ausreichend“ zu unterstreichen, da es weder zu umfangreich noch zu klein sein darf. Wäre es zu umfangreich, so würde die Einarbeitungszeit über die verfügbare und eingeplante Vorbereitungszeit hinausragen. Würde es zu wenig Funktionalität aufweisen, müssten zusätzliche Technologien verwendet werden, worauf wiederum viel Einarbeitungszeit folgen würde.

### Optionen

### React

React JS ist ein, vom Unternehmen Facebook entwickeltes JavaScript – Framework, welches durch seinen langjährigen Aufenthalt am Markt und weltbekannten Hersteller eine stets wachsende Community aufrechterhält. Umfangsmäßig ist es genau zwischen die anderen beiden bekanntesten JS-Frameworks AngularJS und VueJS einzuordnen. React verwendet eine eigene „Programmiersprache“ namens **JSX**, bei welcher es sich um eine Mischform von HTML und JavaScript handelt. Somit behandelt das Framework Struktur (HMTL) und Funktionalität (JS) als eine Einheit, was gute Voraussetzungen für komponentenbasierte Programmierung bietet.

### Vue

Vue JS wird vom ehemaligen Google-Mitarbeiter und AngularJS-Anwender Evan You entwickelt. Es wurden somit gute Aspekte des zuvor genannten Frameworks übernommen und weitgehend vereinfacht. Vue gilt als ein sehr einsteigerfreundliches Framework und punktet durch seine Einfachheit. Durch sein junges Alter besitzt es jedoch eine vergleichsweise kleine bis mittelgroße Community. Vue trennt JS, HTML und CSS voneinander, weshalb eine gute Übersicht eingehalten wird. Dies hindert Vue jedoch nicht daran, eines der besten Frameworks für komponentenbasierte Programmierung zu sein.

### Nutzwertanalyse

Mittels der Nutzwertanalyse wird bestimmt, welches Framework sich letztendlich als passend herausstellt. Je weniger Punkte in der Spalte „R\*G“ enthalten sind, desto besser.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | **React** | | **Vue** | |
| **Kriterium** | **G** | **R** | **R\*G** | **R** | **R\*G** |
| Akzeptanz | 45% | 1 | 45 | 2 | 90 |
| Dokumentation | 35% | 2 | 70 | 1 | 35 |
| Umfang | 20% | 1 | 20 | 2 | 40 |
| **Gesamt** | |  | 135 |  | 165 |

G ... Gewichtung

R ... Rang

### Fazit

Die Entscheidung fällt auf das Framework React JS, da hier zum einen durch bereits bestehende Erfahrung weniger Einarbeitungszeit benötigt wird, zum anderen eine starke Community dahintersteckt, was eine Vielzahl von Hilfestellungen im Internet bietet. Umfangsmäßig entspricht dieses Framework den Anforderungen, da keine zusätzlichen Technologien benötigt werden.

## Technologien und Getting Started

### React

Um eine neue React-Anwendung zu erstellen, benötigt man Node.js (Version >= 6) und npm (Version >= 5.2). Nun muss man folgende Befehle in seiner Shell ausführen:

npx create-react-app applicationName

cd applicationName

npm start

Es wird ein lokaler Development-Server gestartet und die Anwendung kann nun unter <http://localhost:3000> eingesehen werden.

Da jedoch das Backend-Team sowieso mit dem Einsatz ihrer SDKs for Node.js Cloud-Foundry-App einen eigenen Webserver startet, ist es das Ziel des Frontend-Teams die React-Anwendungen auf diesen Server auszulagern.

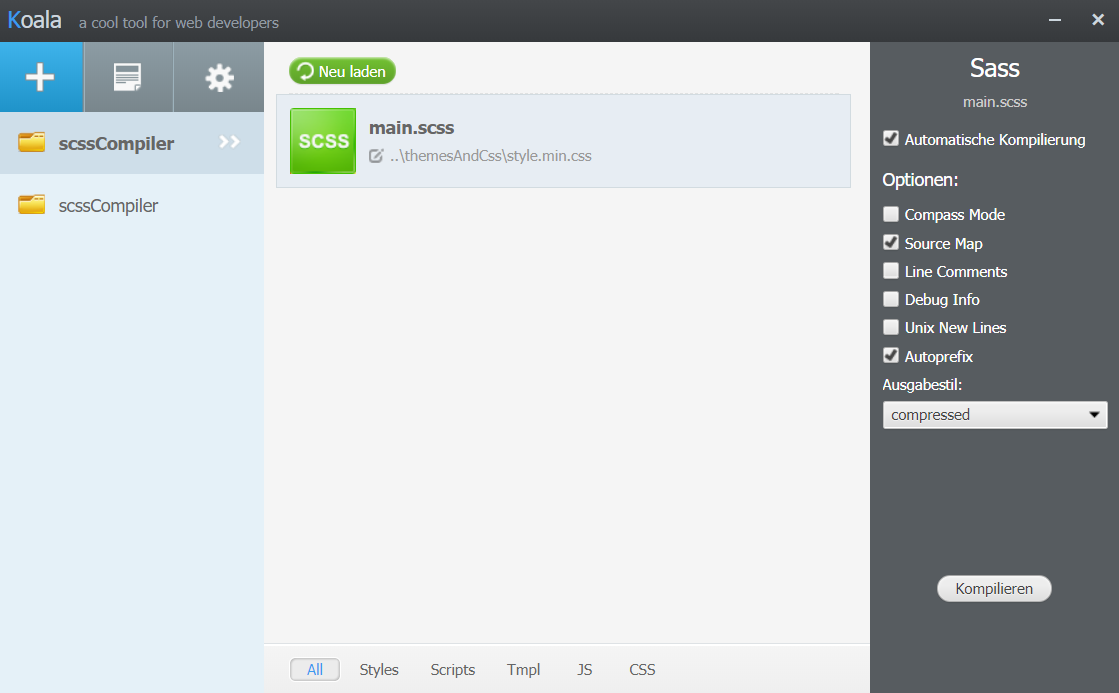
### SASS und Koala

Um die Arbeit mit CSS zu vereinfachen, benutzen wir SASS. SASS ist eine Stylesheet-Sprache, die als CSS-Präprozessor, mit Variablen, Schleifen und vielen anderen Funktionen, die Cascading Style Sheets (CSS) nicht mitbringen, die Erstellung von CSS vereinfacht und die Pflege großer Stylesheets erleichtert.

Um diese SASS-Files zu bearbeiten und zu kompilieren wird das Programm Koala seine Verwendung finden.

**Wichtig** ist, dass das Programm im Hintergrund gestartet sein muss, sodass sich Änderungen in den CSS Files auch bemerkbar machen. Ansonsten werden diese nicht übernommen.

Das main.scss File sollte diese Einstellungen besitzen:



## **Evaluierung Backend**

### Einleitung

Das Ziel dieser Evaluation ist es die, für das Backend zu verwendende, Sprache zu ermitteln. Hierbei wird zwischen den Scriptsprachen NodeJS, PHP und Python entschieden. Die Sprache soll bestmöglich mit der Arbeitsumgebung (IBM-Cloud) zusammenarbeiten, um viel Funktionalität bei möglichst wenig (kompliziertem) Code.

### Kriterien

### Akzeptanz

Wie gut kommt das Team mit der Technologie zurecht? Hierunter fallen bereits gemachte Erfahrungen, die Struktur und Syntax der Programmiersprache und mögliche Hilfestellungen durch eine große und vor allem aktive Community und/oder durch die Ansprechpersonen auf Seiten des Auftraggebers.

### Dokumentation

Wie gut ist eine Technologie dokumentiert? Gibt es ausreichend Material, um sich mit jener zurecht zu finden und ist dieses zum einen ausführlich und zum anderen kompakt zusammengefasst verfügbar?

### Performance

Wie viel Code muss für die Erfüllung einer konkreten Aufgabenstellung geschrieben werden? Wie schaut die Interaktion der Sprache mit Datenbanken aus, wie einfach lassen sich diese ansprechen? Ist mittels der Sprache asynchrone oder nebenläufige Programmierung möglich?

### Optionen

### NodeJS

Dies ist die Backend-Umgebung der Programmiersprache JS. Es wird oftmals mit asynchronen Callbacks gearbeitet. Die Zusammenarbeit zwischen server-seitigem NodeJS und client-seitigem JS ist fließend, weshalb Node für kleine Webapplikationen gut geeignet ist. Node besitzt eine kompakte und gut leserliche Dokumentation und wird laufend weiterentwickelt.

### PHP

PHP ist eine relativ alte, jedoch mächtige Sprache. Sie geht jedoch schon langsam in den Ruhestand und wird sicherlich im Laufe der Zeit von Node abgelöst. Hinter PHP steht immernoch eine sehr große Community. Diese Sprache wurde jahrelang für große Web – Projekte eingesetzt.

### Python

Python ist vor allem durch seine Einsteigerfreundlichkeit bekannt. Sowohl die Syntax, als auch diverse Datenbankzugriffe sind unkompliziert. Auch Python besitzt eine aktive und große Community. Diese Sprache wird derzeit oftmals in Bereichen wie „Machine Learning“ oder „Big Data“ verwendet.

### Nutzwertanalyse

Mittels der Nutzwertanalyse wird bestimmt, welche Technologie sich letztendlich als passend herausstellt. Je weniger Punkte in der Spalte „R\*G“ enthalten sind, desto besser.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **NodeJS** | | **PHP** | | **Python** | | |
| **Kriterium** | **G** | **R** | **R\*G** | **R** | **R\*G** | **R** | **R\*G** |
| Akzeptanz | 50% | 1 | 50 | 2 | 100 | 3 | 150 |
| Dokumentation | 30% | 1 | 30 | 3 | 90 | 2 | 60 |
| Performance | 20% | 2 | 40 | 3 | 60 | 1 | 20 |
| **Gesamt** | |  | 120 |  | 250 |  | 230 |

G ... Gewichtung

R ... Rang

### Fazit

Wie in obiger Nutzwertanalyse berechnet, fällt die Entscheidung auf NodeJS. Node punktet mit einer aktiven Community, guter Zusammenarbeit innerhalb der Sprache JS, welche sowieso beim Frontend verwendet wird und mit großer Erfahrung der Ansprechpersonen auf Seiten des Auftraggebers. Somit sollte die Einarbeitungszeit gering und ausreichende Performance geboten sein.

### Umsetzung

**Aufsetzen der Arbeitsumgebung**

Diese Befehle sind im erstellten Directory durchzuführen:

npm init

npm install express

npm install bodyparser

**Module & Konzept**

Die einzelnen Module verwenden das Express.js Modul, um über eine REST-Schnittstelle zu kommunizieren. Mit dem Modul Bodyparser.js werden die vom Gateway Empfangen Daten verarbeitet. Die Verwendung dieser beiden Module ermöglicht eine enorme Zeitersparnis, da die benötigte Funktion nicht selbst implementiert werden müssen.

**Software-Architektur**

Die verwendete SW-Architektur basiert auf mehreren Modulen, welche über mehrere REST-Schnittstellen miteinander kommunizieren:

IOT-Plattform

Basiert auf der Watson-API, hier werden die Gateways registriert.

DB Handler

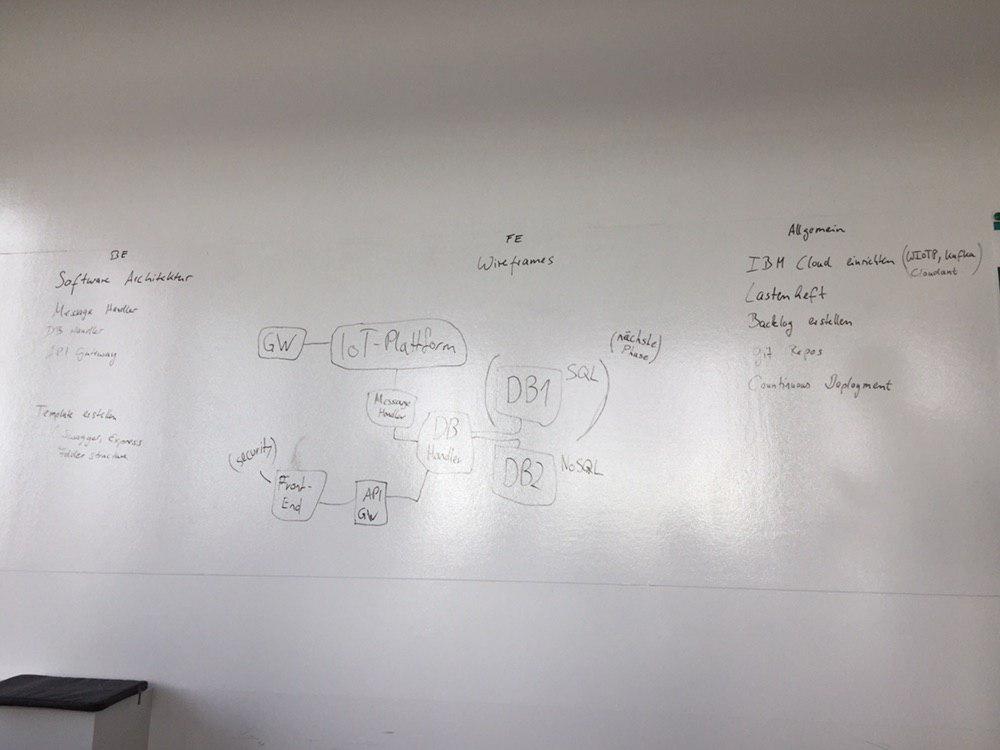
Dient zur Kommunikation mit der NoSQL-Datenbank

DB

NoSQL-Datenbank auf welcher die GW-Daten in Form eines JSON-Dokumentes abgespeichert werden.

API-Gateway

Bereitet die vom DB-Handler übermittelten Daten für das Frontend vor und leitet diese weiter.



**Deployment**

Die implementieren Node-Scripts werden anschließend in der IBM-Cloud als Cloud-Foundry-Application ausgeführt. Dies bieten Vorteil, dass die Scripts auf einer Zentralen Umgebung laufen und keine extra Hardware wie ein Server benötigt wird.

## **Gesamtfazit**

Zusammenfassend gilt: Frontend – ReactJS, Backend – NodeJS

Somit sollte ein fließender Übergang von Frontend über Datenbank zu Backend gesichert sein, da alles auf der Sprache JavaScript aufbaut.

## **Abnahme**

|  |  |  |
| --- | --- | --- |
| **Datum:** 17.03.19 | | |
| **Rolle** | **Name** | **Unterschrift** |
| Autor | Maciej Dzialoszynski |  |
| Qualitätssicherung | Sebastian Grünewald |  |
| Product Owner | Maciej Dzialoszynski |  |