Dokumentation

Bilfinger Wartungscenter

Florian Hansen

30. März 2020

1 Einleitung

Das *Bilfinger Wartungscenter* ist eine Web-Anwendung, die eine Verwaltung von Wartungsarbeiten ermöglicht. Hierfür stehen dem Benutzer verschiedene Funktionen zur Verfügung, um Wartungen einzusehen, zu bearbeiten und Berichte erstellen zu lassen. Das Ziel ist es, dem Benutzer einen guten Überblick über anstehende und bereits getätigte Wartungsarbeiten zu geben und ihm damit einen Großteil der Verwaltungsarbeit abzunehmen.

2 Anforderungen

In diesem Kapitel sollen die Anforderungen an das zu entwickelnde System analysiert und definiert werden. Hierbei wird zwischen allgemeinen, funktionalen und nicht-funktionalen Anforderungen unterschieden.

2.1 Allgemeine Anforderungen

1. Software-Architektur

1.1. Die Anwendung soll mehrere Microservices erhalten, die komplett unabhängig voneinander arbeiten.
1.2. Die Microservices sollen mithilfe von NodeJS / Express entwickelt werden.
1.3. Die Microservices sollen mithilfe von Schnittstellen (APIs) kommunizieren.
1.4. Die APIs sollen im RESTful-Design implementiert werden.
1.5. Die Clientanwendung soll mithilfe von JavaScript / React entwickelt werden.
1.6. Die Clientanwendung greift mithilfe der definierten RESTful-Schnittstellen auf die verschiedenen Microservices zu.

2. Systemanforderungen

2.1. Die Andwendung soll plattformunabhängig sein.
2.2. NodeJS soll als serverseitige Sprache verwendet werden, d.h., dass dies auf dem ausführenden System installiert werden muss.
2.3. Auf dem ausführenden System (serverseitig) soll Docker installiert werden, um Microservices zu installieren bzw. zu laden.

2.2 Funktionale Anforderungen

1. Benutzerverwaltung

1.1. Benutzer sollen sich anmelden können.
1.2. Benutzer sollen über eine Rolle verfügen.
1.3. Mögliche Rollen sollen sein: Verwalter, Wartungskraft.
1.4. Benutzer sollen einen Benutzernamen, einen Vornamen, einen Nachnamen, eine eindeutige Identifikationsnummer (ID) und eine E-Mail-Adresse besitzen.
1.5. Benutzer sollen nur manuell registriert werden können.
1.6. Benutzer sollen ihre Daten, mit Ausnahme der ID, anpassen können.

2. Microservice: Authentifizierung

2.1. Es soll möglich sein, sich mithilfe seiner Benutzerkennung und seines Passworts anzumelden.
2.2. Nach einer erfolgreichen Anmeldung soll es möglich sein, sich mithilfe eines Tokens zu authentifizieren.
2.3. Die Tokens sollen als Json Web Token (JWT)[1] implementiert werden.
2.4. Ein Token soll nur für eine bestimmte Zeit verwendet werden können.
2.5. Nach Ablauf der Lebenszeit eines Tokens, muss sich der Benutzer erneut im System anmelden und einen neuen anfordern, um sich authentifizieren zu können.

3. Microservice: Wartungen

3.1. Dieser Dienst soll nur authentifizierten Benutzern zur Verfügung stehen.
3.2. Benutzer sollen Wartungen abrufen können. Dabei werden ihnen nur diejenigen zur Verfügung gestellt, die ihnen zugeteilt wurden.
3.3. Nur Benutzer der Rolle *Verwalter* sollen neue Wartungen erstellen können.
3.4. Es soll als Verwalter möglich sein, Wartungen zu bearbeiten und zu verändern.
3.5. Es soll als Verwalter möglich sein, Wartungen Wartungskräften zuzuornden.
3.6. Es soll als Verwalter möglich sein, Kunden anzulegen.

[1] https://jwt.io/

3.7. Es soll als Verwalter möglich sein, Kunden zu bearbeiten.

3.8. Es soll als Verwalter möglich sein, Kunden zu löschen.

3.9. Kunden sollen von Verwaltern Wartungen zugewiesen werden können.

3.10. Ein Kunde besitzt einen Namen, eine Ansprechsperson (Vor- und Nachname, Telefonnummer, E-Mail, ...) und eine Anschrift (Ort, PLZ, Straße)

2.3 Nicht-funktionale Anforderungen