Entwickler Case – Backend Entwickler (m/w/d) für C# 8 WebAPI

1. Einleitung

Ziel des Entwickler Case ist die Entwicklung einer Prototypen Anwendung (PoC, Proof-of-Concept) auf Basis von C# 8 und ASP.NET Core WebAPI, welche alle funktionalen Anforderungen aus dem 3. Abschnitt beinhaltet.

2. Rahmenbedingungen

- Es ist alles zulässig, was nicht explizit untersagt ist.
- Zeitvorgabe: 1 Stunde
- Technologie: C# 8, ASP.NET Core WebAPI
- Es dürfen zusätzliche Third-Party-Komponenten eingebunden werden. Dabei muss der Bewerber erläutern, warum diese Komponenten eingesetzt wurden und nicht durch eigenen Code umgesetzt wurden.
- Bei funktionalen Anforderungen, die nicht umgesetzt werden können, muss eine Begründung gegeben werden, was eine Umsetzung verhindert hat.

3. Funktionale Anforderungen

Erstelle eine API, die grundlegende CRUD (Create, Read, ~~Update, Delete~~) Operationen ermöglicht:

3.1. GET-Endpunkt

Implementiere einen GET-Endpunkt, der eine Liste von Daten zurückgibt. Die Struktur der Daten soll die folgenden Felder enthalten: Anrede, Vorname, Nachname, Geburtsdatum, Adresse, PLZ, Ort und Land.

3.2. POST-Endpunkt

Implementiere einen POST-Endpunkt, der neue Daten zur bestehenden Liste hinzufügt. Die Struktur der Daten, die übergeben werden sollen, beinhaltet die gleichen Felder wie der GET-Endpunkt: Anrede, Vorname, Nachname, Geburtsdatum, Adresse, PLZ, Ort und Land.

3.3. Validierung des POST-Endpunktes

Füge Validierungsfunktionen hinzu, um sicherzustellen, dass die Pflichtfelder ausgefüllt sind. Die Pflichtfelder sind: Vorname, Nachname, Geburtsdatum und PLZ. Zudem muss das Geburtsdatum in der Vergangenheit liegen und die PLZ aus 5 numerischen Zeichen bestehen.
Die Angabe vom Land ist optional, wenn gefüllt müssen es aber genau 2 Zeichen sein.

COPiTOS GmbH | Großtobeler Straße 49 | 88276 Berg/Ravensburg | T +49 751 / 50 91 99 40 | E info@copitos.de | www.copitos.de

3.4. Beispiel

Ein Beispiel für einen POST Request:

```
{
  "anrede": "Herr",
  "vorname": "Max",
  "nachname": "Mustermann",
  "geburtsdatum": "1990-01-01",
  "adresse": "Musterstraße 1",
  "plz": "12345",
  "ort": "Musterstadt",
  "land": "DE"
}
```

Ein Beispiel für die Antwort des GET-Endpunkts:

```
[
  {
    "anrede": "Herr",
    "vorname": "Max",
    "nachname": "Mustermann",
    "geburtsdatum": "1990-01-01",
    "adresse": "Musterstraße 1",
    "plz": "12345",
    "ort": "Musterstadt",
    "land": "DE"
  }
]
```


COPiTOS GmbH | Großtobeler Straße 49 | 88276 Berg/Ravensburg | T +49 751 / 50 91 99 40 | E info@copitos.de | www.copitos.de

4. Nicht funktionale Anforderungen

Nicht-funktionale Anforderungen definieren die Qualitätsmerkmale und Attribute, die eine Anwendung erfüllen muss, abgesehen von ihrer funktionalen Logik. Hier sind einige nicht-funktionale Anforderungen, die für diese Proof-of-Concept Anwendung nicht relevant sind:

4.1. Laufzeitspeicherung

Die Daten müssen nur zur Laufzeit gespeichert werden. Eine persistente Speicherung in einer Datenbank ist nicht erforderlich.

4.2. Leistung:

Ladezeit: Die genaue Ladezeit der Seiten muss nicht optimiert werden, da es sich um eine Proof-of-Concept Anwendung handelt.
Reaktionszeit: Eine optimale Reaktionszeit für Benutzerinteraktionen ist für das Proof-of-Concept nicht erforderlich.

4.3. Sicherheit:

Datensicherheit: Da die Anwendung mit Mock-Daten arbeitet, muss die Datensicherheit nicht in vollem Umfang implementiert werden.
Kommunikationssicherheit: Die Verwendung von HTTPS und umfassende Sicherheitsprotokolle ist für das Proof-of-Concept nicht erforderlich.

4.4. Skalierbarkeit:

Anpassungsfähigkeit: Eine hohe Anpassungsfähigkeit für zukünftige Anforderungen muss nicht sichergestellt werden.
Performance-Skalierbarkeit: Die Leistung bei zunehmenden Datenmengen oder Benutzerzahlen muss im Rahmen des Proof-of-Concept nicht berücksichtigt werden.

4.5. Wartbarkeit und Erweiterbarkeit:

Codequalität: Eine umfassende Codequalität und -dokumentation müssen nicht priorisiert werden.
Modularität: Die Anwendung muss nicht zwingend in gut definierte Module unterteilt sein.

4.6. Logging und Überwachung:

Logging: Ein umfassendes Logging zur Identifizierung und Diagnose von Problemen muss nicht implementiert werden.
Überwachung: Die Überwachung von Leistung und Verfügbarkeit ist für das Proof-of-Concept nicht erforderlich.

COPiTOS GmbH | Großtobeler Straße 49 | 88276 Berg/Ravensburg | T +49 751 / 50 91 99 40 | E info@copitos.de | www.copitos.de