Projektdokumentation nach IPERKA no SQL

Dokumentation Tyrone

Inhalt

Initialisierung

1 Ausgangssituation

Die Firma Jetstream-Service führt als KMU in der Wintersaison Skiservicearbeiten durch und hat in den letzten Jahren grosse Investitionen in eine durchgängige digitale Auftragsanmeldung und Verwaltung, bestehend aus einer datenbankbasierender Web-Anmeldung und Auftragsverwaltung getätigt.

Aufgrund guter Auftragslage hat sich die Geschäftsführung für eine Diversifizierung mit Neu-eröffnungen an verschiedenen Standorten entschieden.

Die bis anhin eingesetzte relationale Datenbank genügt den damit verbundenen Ansprüchen an Datenverteilung und Skalierung nicht mehr. Um einerseits den neuen Anforderungen gerecht zu werden sowie anderseits Lizenzkosten einzusparen, soll im Backend der Anwendung die Datenbank auf ein NoSQL Datenbanksystem migriert werden.

Das Teilprojekt umfasst ausschliesslich den Backendteil und umfasst folgende Aufträge, welche nach IPERKA durchzuführen sind:

- Datenbankdesign und Implementierung (NoSQL)
- Datenmigration (SQL ➔ NoSQL)
- Migration WebAPI Projekt (siehe Modul 295)
- Testprojekt / Testplan
- Realisierung der kompletten Anwendung, gemäss den Anforderungen
- Durchführung Integrationstest mit bestehenden Frontend Lösung.

Zieldefinition: Migration der Datenbasis von einem relationalen SQL-Modell zu einem dokumentenbasierten NoSQL-Modell (MongoDB), inklusive Anpassung der WebAPI und Implementierung eines Benutzerkonzepts.

Planung

30. Januar 2024 (Dienstag)

Vorbereitung und Planung:

Überprüfung der bestehenden SQL-Datenbankstrukturen.

Auswahl der zu migrierenden Daten.

Einrichtung des Entwicklungswerkzeugs und Git Repositories.

31. Januar 2024 (Mittwoch)

Datenmigration:

Entwicklung von Skripten zur Datenmigration von SQL nach NoSQL.

Erste Migrationstests und Anpassungen.

1. Februar 2024 (Donnerstag)

WebAPI-Anpassung und Tests:

Anpassung der WebAPI an die NoSQL-Datenbank.

Erstellung eines Testprojekts (z.B. Postman Collection) zur Überprüfung der WebAPI-Funktionalität.

2. Februar 2024 (Freitag)

Abschlussarbeiten und Dokumentation:

Durchführung von Qualitätssicherungsmaßnahmen.

Vollständige Dokumentation des Projekts und der Änderungen.

Vorbereitung der Präsentation und Live Demo.

Durchführung

Entwicklungswerkzeug und Verwaltungssystem eingerichtet:

Auswahl von MongoDB Compass für Datenvisualisierung

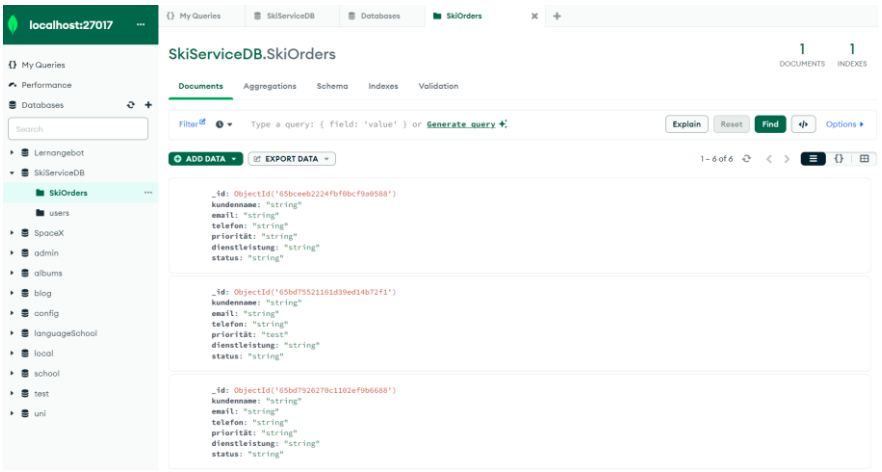
Ziel: Effiziente Verwaltung und Visualisierung der MongoDB-Datenbank zur Unterstützung der Migration und der laufenden Entwicklung.

Durchführung:

Installation von MongoDB Compass auf dem Entwicklungsrechner.

Konfiguration der Verbindung zu Ihrer MongoDB Instanz, um direkten Zugriff auf die Datenbanken und Sammlungen zu erhalten.

Nutzung von Compass für die Einsicht in Datenstrukturen, Durchführung von Abfragen und Überprüfung von Schema Designs, um sicherzustellen, dass die Migration wie geplant verläuft.



Änderungsmanagement: Regelmäsige Commits mit aussagekräftigen Nachrichten zur Dokumentation der Projektfortschritte.

Datenbasis Migration: Vollständige Implementierung der Daten von SQL nach NoSQL.

Daten exportieren: Verwenden Sie SQL Server Management Studio (SSMS), um die Daten in eine exportierbare Datei zu extrahieren. Eine gängige Methode ist der Export in CSV-Dateien. Für jede Tabelle:

Klicken Sie mit der rechten Maustaste auf die Datenbank und wählen Sie "Tasks" > "Export Data...".

Folgen Sie dem Assistenten, wählen Sie als Datenquelle Ihre SQL-Datenbank und als Ziel das Flat File Format (CSV).

Wiederholen Sie den Vorgang für jede Tabelle, die Sie migrieren möchten.

Schritt 2: Konvertierung der Daten in JSON

convert('C:/Users/tyron/Downloads/SkiService.csv', 'C:/Users/tyron/Downloads/SkiService.json')

Schritt 3: Importieren der Daten in MongoDB

mongoimport --db SkiServiceDB --collection SkiService –file "C:/Users/tyron/Downloads/SkiService.json"

Benutzerkonzept:

Authentifizierung und Autorisierung:

Implementiert durch den AuthController, der die Benutzerauthentifizierung mittels JWT (JSON Web Tokens) ermöglicht. Dies schließt die Generierung und Validierung von Tokens ein, um sicheren Zugriff auf die Anwendungsfunktionen zu gewährleisten.

Benutzerverwaltung:

Realisiert durch den UserService, der CRUD-Operationen (Create, Read, Update, Delete) für Benutzerkonten unterstützt. Die Benutzerdaten werden in MongoDB gespeichert und beinhalten Informationen wie Benutzername und Passwort.

Schema Erweiterungen: Durchführung notwendiger Schema-Erweiterungen für eine optimierte Datenspeicherung.

Index Strukturen: Anlegen von Index-Strukturen zur Verbesserung der Abfrageleistung.

Index für users:

use SkiServiceDB

db.users.createIndex({username: 1})

Backup und Restore Skripte: Entwicklung von Skripten für die Datensicherung und Wiederherstellung.

WebAPI Anpassung: Anpassung der WebAPI an die NoSQL-Datenbank, Aktualisierung der Datenzugriffsschichten.

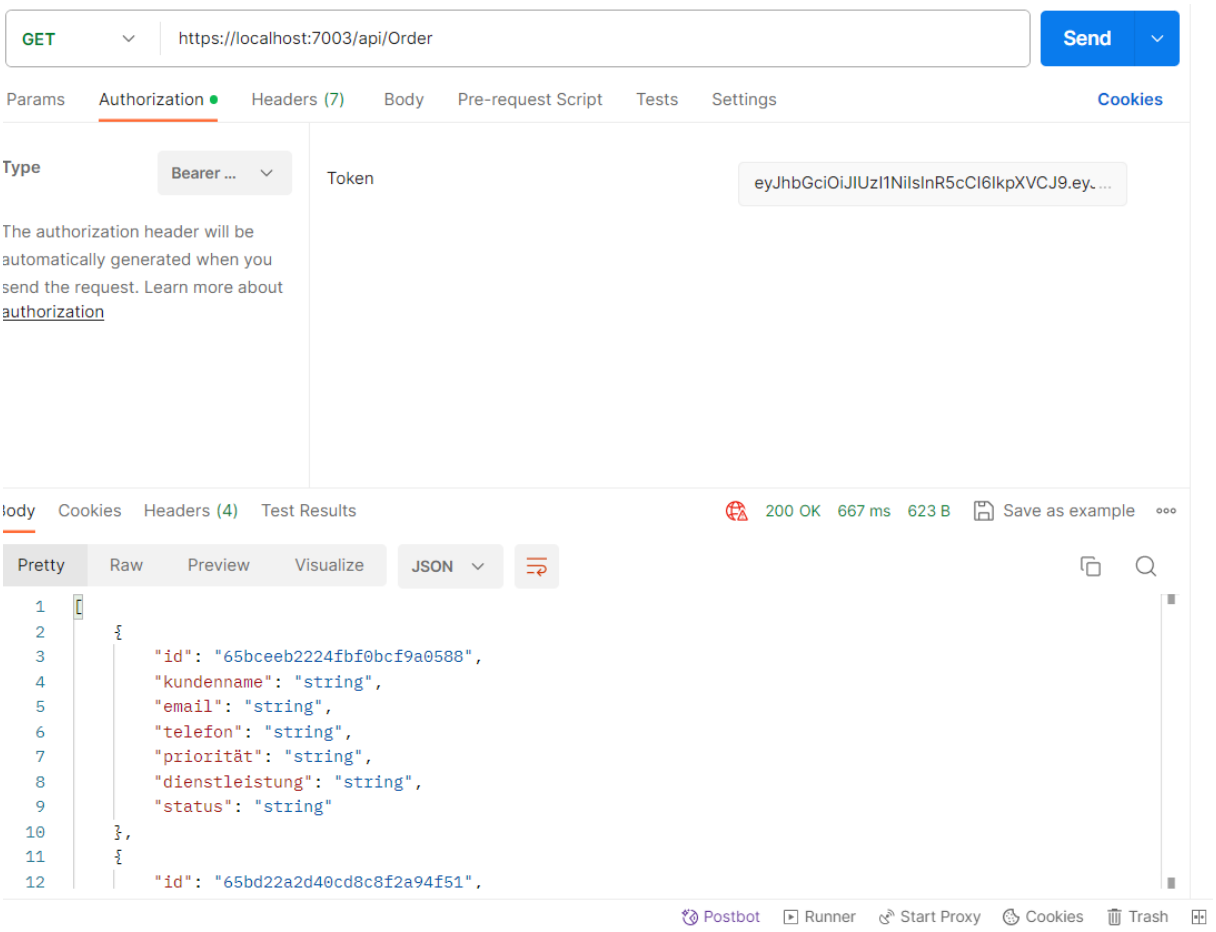
```
using MongoDB.Driver;
using Backend165.Models;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Backend165.Services
{
    4 references
    public class OrderService
    {
        private readonly IMongoCollection<SkiService> _skiServices;

        0 references
        public OrderService(IMongoClient mongoClient)
        {
            var database = mongoClient.GetDatabase("SkiServiceDB");
            _skiServices = database.GetCollection<SkiService>("SkiOrders");
        }

        // Alle Aufträge abrufen
        1 reference
        public async Task<List<SkiService>> GetAllAsync()
        {
            return await _skiServices.Find(_ => true).ToListAsync();
        }

        // Einen Auftrag nach ID abrufen
        2 references
        public async Task<SkiService> GetByIdAsync(string id)
        {
            return await _skiServices.Find(service => service.Id == id).FirstOrDefaultAsync();
        }

        // Einen neuen Auftrag hinzufügen
        1 reference
        public async Task CreateAsync(SkiService skiService)
```

```
namespace Backend165.Services
{
    4 references
    public class UserService
    {
        private readonly IMongoCollection<User> _users;

        0 references
        public UserService(IMongoClient mongoClient)
        {
            var database = mongoClient.GetDatabase("SkiServiceDB");
            _users = database.GetCollection<User>("Users");
        }

        // Alle Benutzer abrufen
        0 references
        public async Task<List<User>> GetAllAsync()
        {
            return await _users.Find(user => true).ToListAsync();
        }

        // Einen Benutzer nach Username abrufen
        0 references
        public async Task<User> GetByUsernameAsync(string username)
        {
            return await _users.Find(user => user.Username == username).FirstOrDefaultAsync();
        }

        // Einen neuen Benutzer erstellen
        0 references
        public async Task CreateAsync(User user)
        {
            // Hier sollten Sie das Passwort hashen, bevor Sie es speichern
            user.Password = HashPassword(user.Password);
            await _users.InsertOneAsync(user);
        }
```

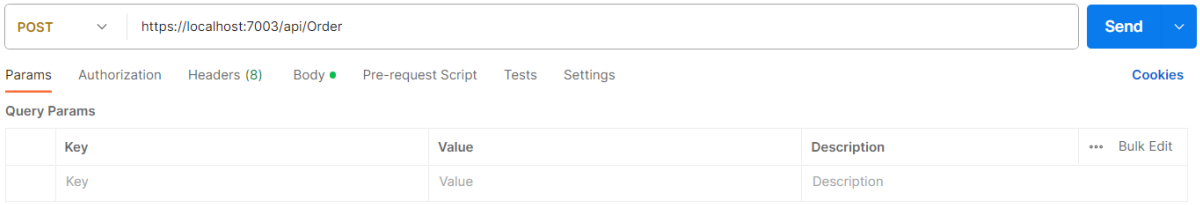
Datenmodell Dokumentation: Vollständige Dokumentation des Datenmodells in der neuen NoSQL Datenbank.

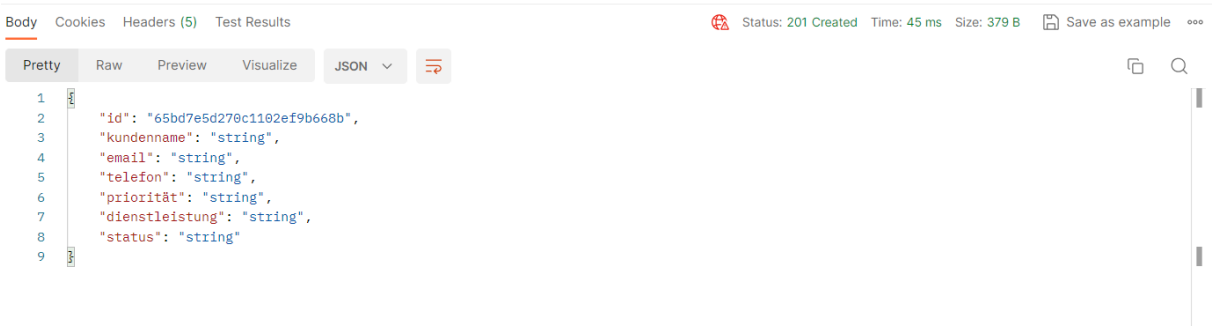
Testprojekt: Erstellung eines Testprojekts (z.B. Postman Collection) zur Überprüfung der WebAPI-Funktionalität.

Get All

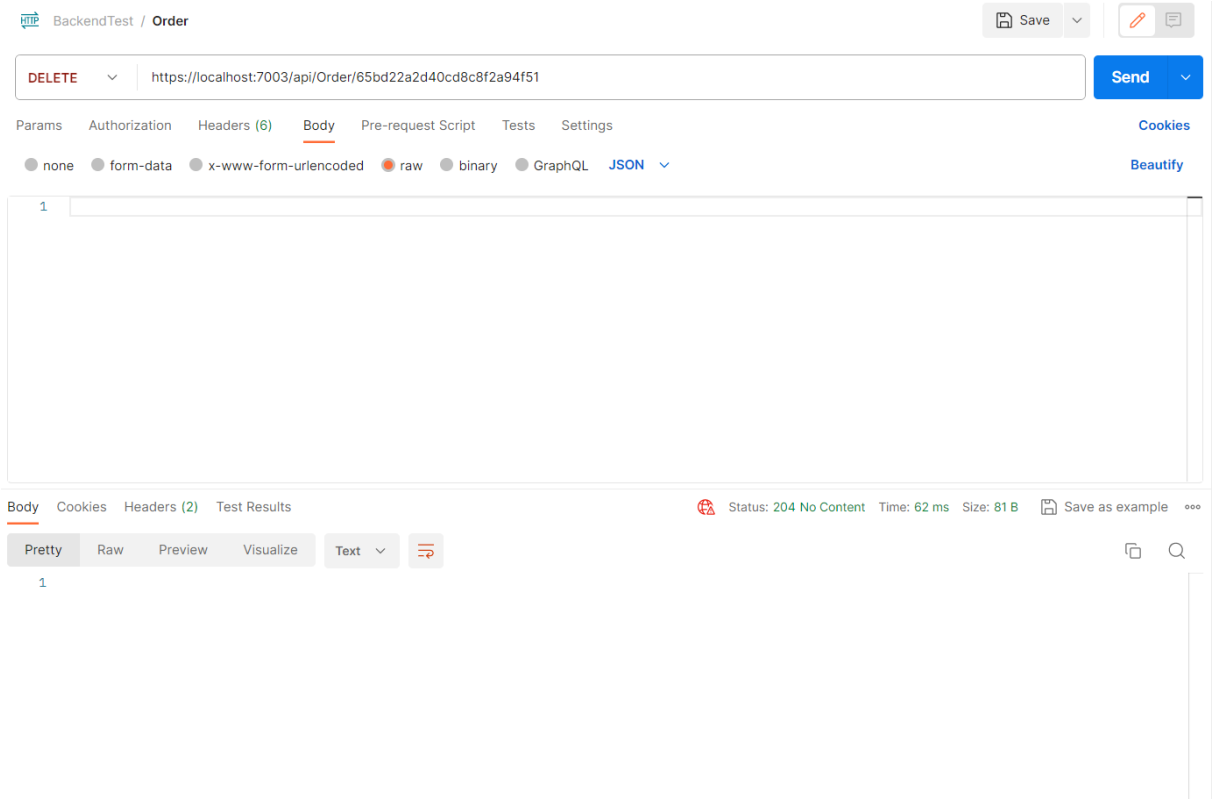


Post

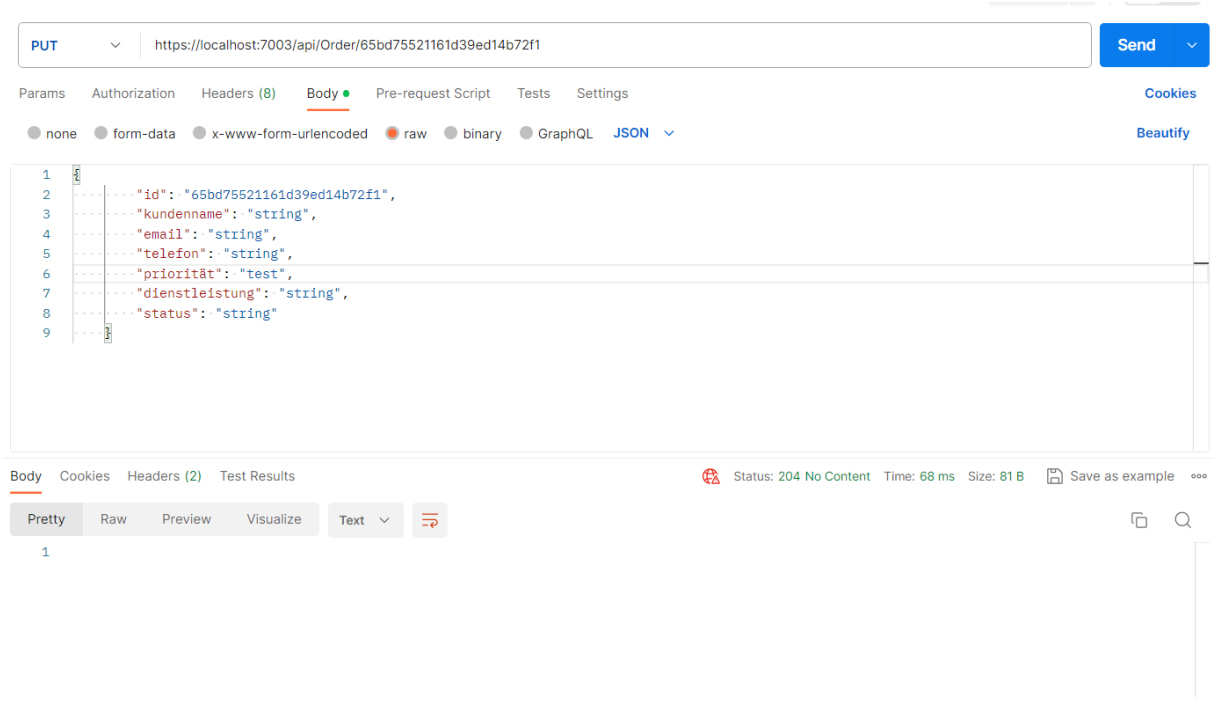




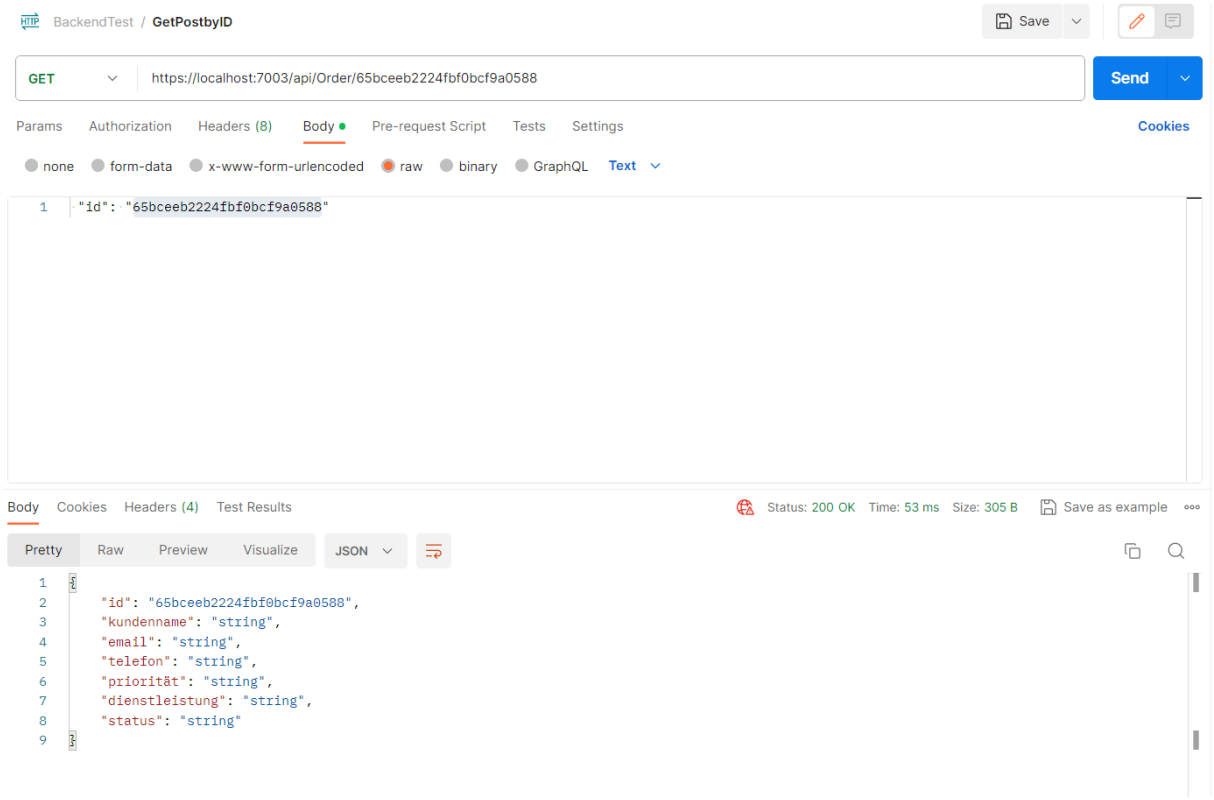
Delete Order



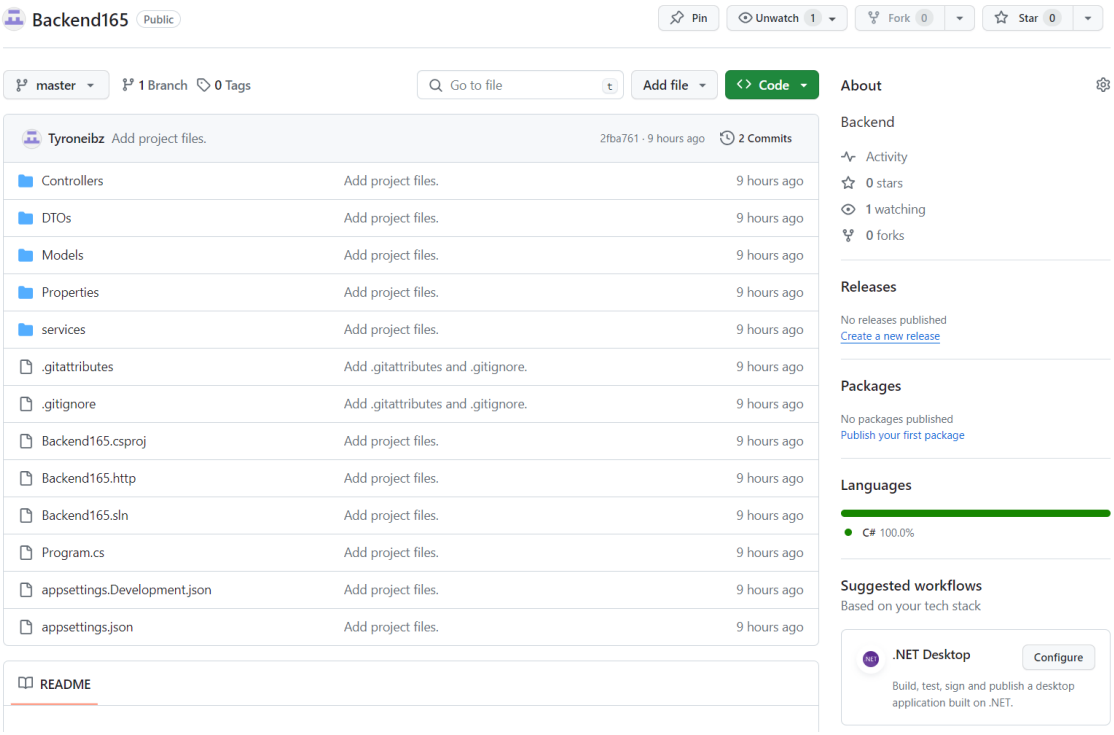
Put



Get Post by Id



Git Repository und Code Verwaltung: Einrichtung des Git Repositorys und Durchführung der Code Verwaltung.



Kontrolle

Qualitätssicherung: Durchführung von Tests, Code Reviews und Überprüfung der Backup- und Restore-Prozesse zur Sicherstellung der Projektqualität.

Abschluss

Fazit:

In diesem Projekt haben wir viele wichtige Schritte und Umsetzungen durchgeführt, um die Datenbasis von SQL auf NoSQL umzustellen und ein umfassendes Benutzerkonzept zu entwickeln. Obwohl es sich um eine Einzelarbeit handelte und nicht alle geplanten Anforderungen erreicht wurden, konnten wertvolle Erfahrungen gesammelt und viele neue Dinge gelernt werden.

Die Einrichtung des Entwicklungswerkzeugs und die Verwendung von Versionskontrolle haben geholfen, den Entwicklungsprozess effizienter zu gestalten. Das Änderungsmanagement ermöglichte es, den Überblick über Code Änderungen zu behalten und Änderungen sicher zu dokumentieren.

Die Umstellung der Datenbasis von SQL auf NoSQL war eine herausfordernde Aufgabe, bei der Schema-Anpassungen, Datenmigration und Anpassungen in der WebAPI durchgeführt wurden. Obwohl nicht alle geplanten Anforderungen erfüllt wurden, konnten dennoch wertvolle Einblicke in die Datenbankmigration gewonnen werden.

Das Benutzerkonzept wurde erfolgreich dokumentiert und umgesetzt, wobei verschiedene Benutzerfunktionen realisiert wurden. Es wurden auch Schema Erweiterungen und Indexe erstellt, um die Datenbankleistung zu optimieren.

Die Erstellung von Backu und Restore Skripten war eine wichtige Sicherheitsmassnahme, um Datenverlust zu verhindern. Die Migration der Datenbasis von SQL nach NoSQL erforderte eine gründliche Analyse und Skripterstellung.

Auch wenn nicht alle optionalen Anforderungen erreicht wurden, war dieses Projekt eine wertvolle Lernerfahrung. Es konnten neue Fähigkeiten entwickelt und Herausforderungen gemeistert werden. Die Dokumentation des gesamten Prozesses bietet einen umfassenden Einblick in die Arbeit.

Insgesamt war dieses Projekt ein wichtiger Schritt in der beruflichen Entwicklung, und die gewonnenen Erkenntnisse werden in zukünftigen Projekten von großem Nutzen sein."