Echoes

Damien Flury, Tim Hess

30. November 2018

Inhaltsverzeichnis

1 Auftrag

Unser Auftrag besteht darin, eine Applikation mit Datenbankanbindung zu entwickeln. Technologien können frei gewählt werden.

2 Projektbeschreibung

Unsere Idee ist die Entwicklung einer Webapplikation für die Planung von Hausaufgaben und Prüfungen. All diese Daten werden in einer Datenbank gespeichert. Wir planen eine Multi-User-Applikation, das heisst, Benutzer werden einen Account erstellen und sich dann damit anmelden können.

3 Anwendungsfälle

Unsere Applikation bietet die perfekte Plattform für Schüler, ihre Aufträge und Prüfungen zentral zu verwalten. Sie soll die Verwaltung verschiedener Klassen mit Schülern ermöglichen und vereinfachen. Man soll einen Account erstellen, Hausaufgaben und Prüfungen zu einer Klasse hinzufügen und diese auch als erledigt markieren können. Auf die jeweiligen Hausaufgaben einer Klasse haben alle zugehörigen Schüler Zugriff und können diese lesen. Ob sie Aufträge auch bearbeiten können, ist noch nicht endgültig entschieden. Momentan planen wir, die Bearbeitung für den Ersteller des Auftrags zu ermöglichen, aber nicht für die anderen Schüler. Man soll Aufträge kommentieren können.

4 Technologien

Für die Webapplikation verwenden wir verschiedene Frameworks. Serverseitig bieten wir eine REST API mit ASP.NET Core an, Clientseitig konsumieren wir diese API mit dem Single Page Application (SPA) Framework Angular.

4.1 Backend

- ASP.NET Core 2.1 [7]
- C# 7.3
- MySQL (MariaDB) [6]
- Entity Framework Core

4.2 Frontend

- Angular 7 [1]
- Typescript [8]
- Bootstrap 4 [2]

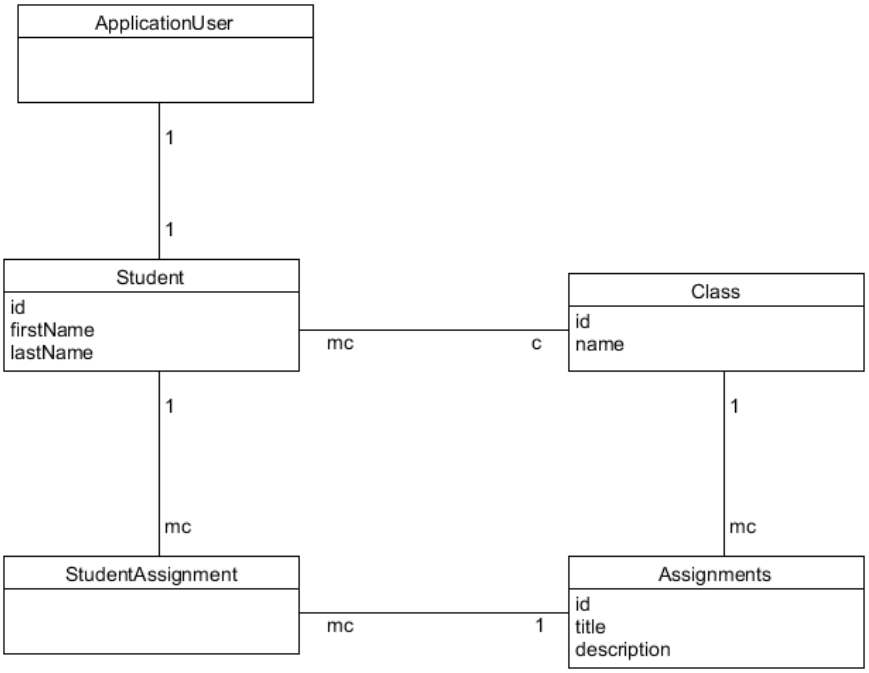

Abbildung 1: Entity Relationship Diagram (ERD)

4.3 Dokumentation und Allgemeines

- LaTeX [5]
- Visual Studio Code [10]
- Visual Studio [9]
- Jetbrains Rider [4]
- Git und GitHub [3]

5 Datenbank und konzeptionelles Datenmodell

Wie in Abbildung 1 ersichtlich, verwenden wir eine relationale Datenbank. Für die Anbindung an die Applikation verwenden wir Entity Framework Core. Die Tabelle *ApplicationUser* ist für das Login zuständig und besitzt eine 1-1 Beziehung mit *Student*. Jeder Student gehört einer oder keiner *Class* (Schulklasse) an. Eine Schulklasse besitzt beliebig viele *Assignments*, also Aufträge. Die Tabelle *StudentAssignment* bietet die Möglichkeit, die Hausaufgaben, welche bereits erledigt wurden, zu markieren.

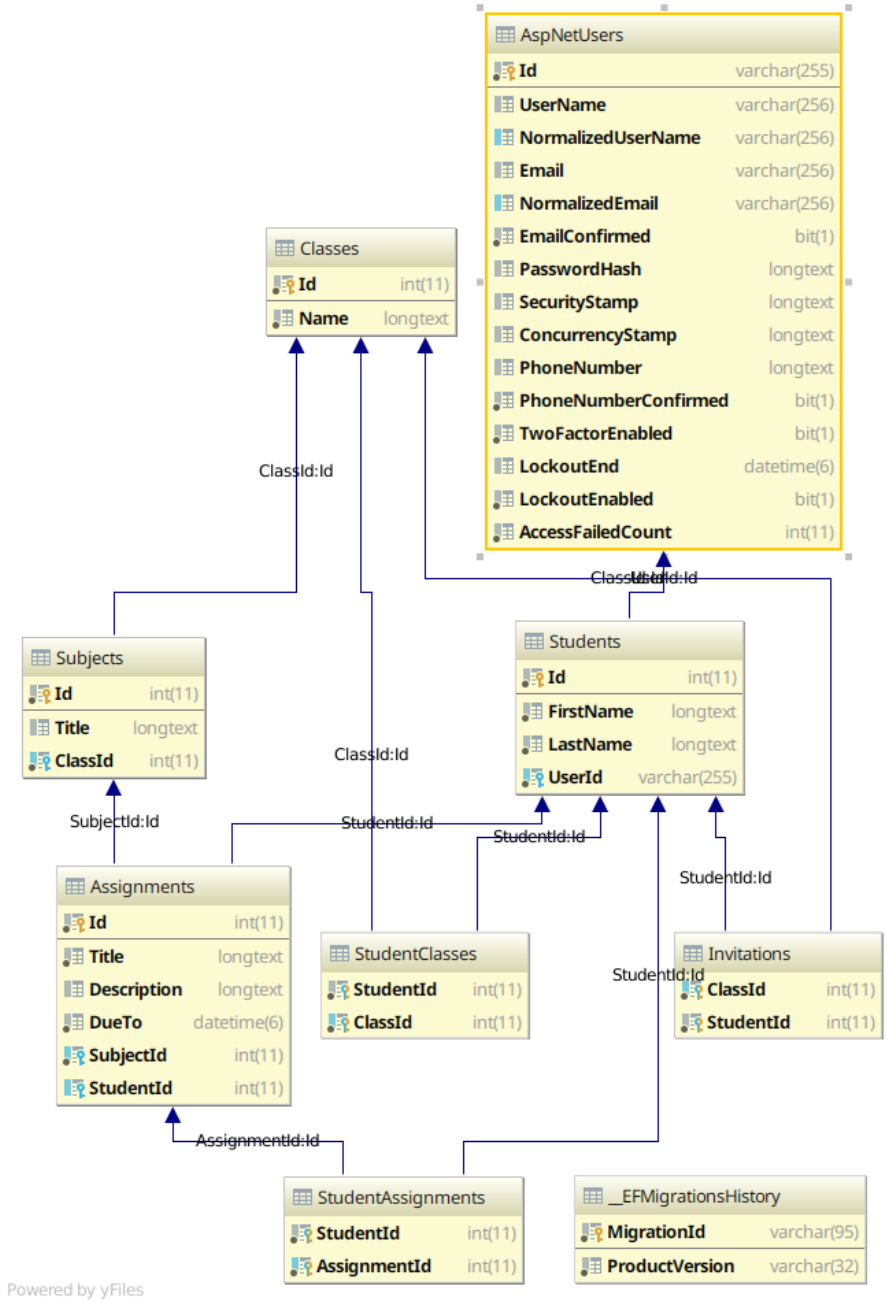

Abbildung 2: Entity Relationship Diagram (Generated)

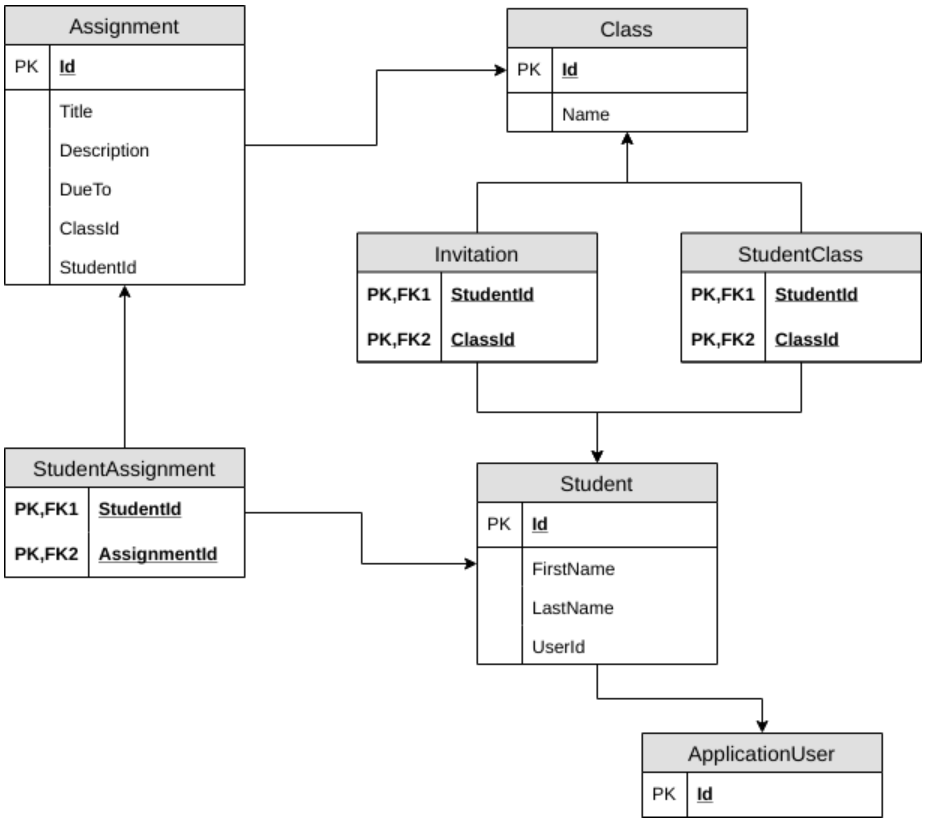

Abbildung 3: Logisches Datenmodell

```
[Route("api/[controller]")]
[ApiController]
[Authorize]
public class AssignmentsController : ControllerBase
{
    [HttpGet]
    public ActionResult<IEnumerable<Assignment>> Get()
        => Ok(GetAll());
}
```

Listing 1: Controller

```
var assignments = GetAll()
    .Where(assignment => assignment.DueTo < DateTime.Now)
    .OrderBy(assignment => assignment.DueTo);
```

Listing 2: Lambda-Syntax

```
var assignments = from assignment in GetAll()
    where assignment.DueTo < DateTime.Now
    orderby assignment.DueTo
    select assignment;
```

Listing 3: Query-Syntax

6 REST API

Wir verwenden das platformunabhängige Framework ASP.NET Core für die Erstellung einer REST-API. Der Datenbankzugriff läuft über Entity Framework Core. Die Tabellen werden automatisch aus dem Model generiert. Es wird zunächst eine Datenbankmigration erstellt, welche danach auf die Datenbank mit dem *update*-Command auf die Datenbank angewandt werden kann.

```
$ dotnet ef migrations add Initial
$ dotnet ef database update
```

ASP.NET Core verwendet standardmässig das MVC-Pattern. Somit haben wir Controller, welche die API-Schnittstellen darstellen (Siehe Listing 1).

Die Datenbank-Aufrufe erfolgen über *LINQ* (Language Integrated Query), eine Abstraktion, um SQL-Statements in C# selbst zu schreiben. Dies erfolgt entweder über die Lambda-Syntax (Siehe Listing 2), oder über die Query-Syntax (Siehe Listing 3). Das Mapping auf die Objekte erfolgt automatisch.

7 App

Für das Frontend verwenden wir Angular, ein Single Page Application (SPA) Framework. Dies ist Component-Based und arbeitet viel mit Property-Binding. Für die API-Aufrufe verwenden wir die Klasse *HttpClient* (Siehe Listing 4).

```
export class AssignmentsService {
    constructor(private http: HttpClientService) {}

    getAssignments(): Observable<Assignment[]> {
        return this.http
            .get<Assignment[]>('/api/Assignments');
    }
}
```

Listing 4: HttpClient

8 Projektjournal

8.1 Erste Woche

Zunächst haben wir geplant, was die Aufgabe unserer Applikation sein wird, haben uns einen passenden Namen ausgedacht und haben die Technologien festgelegt, welche wir verwenden werden. Dann haben wir mit der API begonnen, GitHub-Projekt initialisiert und die Grundstruktur gelegt.

8.2 Zweite Woche

In der zweiten Woche haben wir das Model erstellt und dieses auf eine neue Datenbank migriert. Zunächst haben wir eine SQLite-Datenbank verwendet, um den Entwicklungsprozess zu vereinfachen. SQLite verwendet lediglich eine Datei und nicht ein komplettes Datenbanksystem. Ausserdem haben wir ein neues Angular-Projekt erstellt, erste Komponenten geschrieben und erste API-Abfragen erstellt.

8.3 Dritte Woche

In der dritten Woche haben wir sowohl Front- als auch Backend erweitert und optimiert. Wir haben ein Login mithilfe von JWT-Tokens und dem EF Core Identity Framework erstellt. Ausserdem haben wir das Design im Frontend mithilfe von Bootstrap verbessert.

8.4 Vierte Woche

In der vierten Woche haben wir das Projekt fertiggestellt und noch letzte Verbesserungen vorgenommen. Als Datenbank verwenden wir nun MySQL mit MariaDB.

9 Endprodukt

Bis jetzt hat Echoes einige nützliche Funktionen für Schüler und Lehrer. Sowohl Schüler als auch Lehrer können sogenannte Assignments in jeder Klasse erstellen und ein Datum dazusetzen. Diese Assignments werden danach auf dem Tab *My Assignments* angezeigt. Wenn das Datum des Assignments abgelaufen ist, wird es im Subtab *Inactive* angezeigt. Diese Assignments werden zusätzlich nur

angezeigt, wenn der User sich in der Klasse befindet, in der das Assignment erstellt wurde. Klassen kann man sehr einfach erstellen und gewisse Schüler mit ihrer eingegebenen Email-Adresse zur Klasse einladen. Wird ein Schüler eingeladen, kann er die Einladung im *Invitations* Tab finden und akzeptieren.

10 Zukünftige Erweiterungen und Updates

Wir haben definitiv noch weitere Ideen, die wir bei Echoes einbauen können. Da jetzt der grundlegende Baustein des Projektes gelegt ist, kann man noch viele zukünftige Updates darauf aufbauen und erweitern. Wir planten, dass Schüler abhaken können, ob sie ein Assignment schon fertiggestellt haben und dass diese Liste von Schülern für den Ersteller des Assignments sichtbar ist.

Literatur

[1] Angular. https://angular.io.

[2] Bootstrap. https://getbootstrap.com.

[3] GitHub. https://github.com.

[4] Jetbrains Rider. https://www.jetbrains.com/rider.

[5] LaTeX. https://www.latex-project.org.

[6] MariaDB. https://mariadb.org.

[7] .NET Core. https://dotnet.microsoft.com.

[8] TypeScript. https://www.typescriptlang.org.

[9] Visual Studio. https://visualstudio.microsoft.com.

[10] Visual Studio Code. https://code.visualstudio.com.