

# EgzaminyAPI

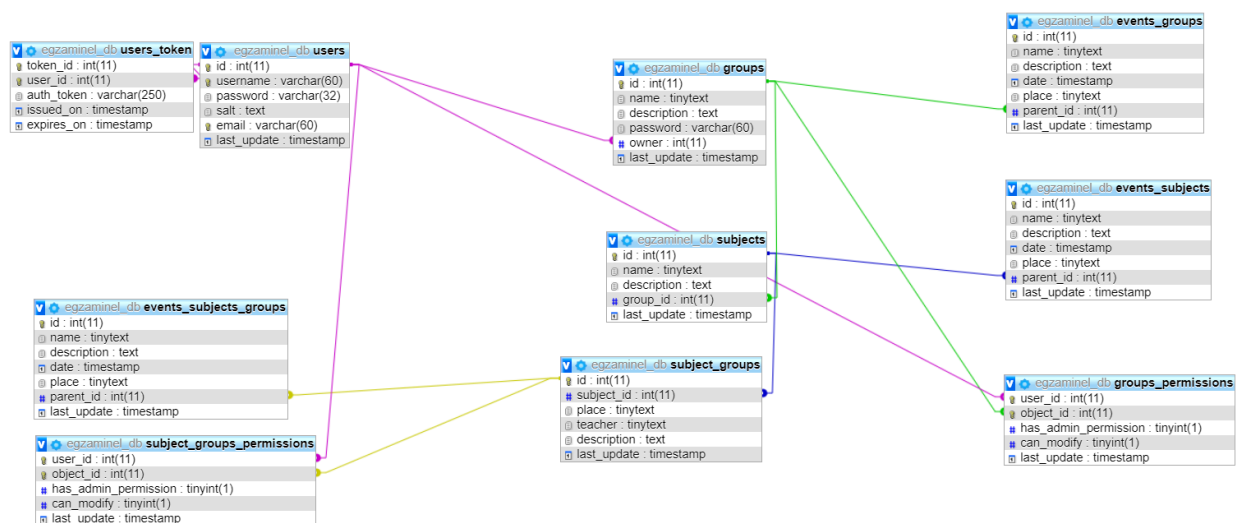
Aplikacja „EgzaminyAPI” ma służyć do obsługi bazy danych, poprzez zapytania RESTowe klientów.

Aplikacja została napisana przy użyciu technologii **ASP.NET Core 2**.

**Bitbucket:** <https://bitbucket.org/Cililing/v2-egzaminelapi/overview>

## Baza danych

Baza danych (MySQL) ma następujący schemat:



Tabele przechowują odpowiednio:

- groups – informacje na temat grup
- events\_groups – informacje na temat wydarzeń w grupie
- subjects – informacje na temat kursów
- events\_subject – informacje na temat wydarzeń w kursie
- subject\_groups – informacje na temat poszczególnych grup kursów
- events\_subjects\_group – wydarzenia dot. grup kursów
- users – dane dotyczące użytkowników
- groups\_permissions – pozwolenia do edycji grupy dla poszczególnych userów
- subject\_groups\_permissions – pozwolenia do edycji grupy przedmiotu dla poszczególnych userów.

Aby edytować tabele events\_[typ] należy posiadać uprawnienia do [typ].

Uprawnienia do „subjects” są równoznacznie z uprawnieniami do grupy.

Walidacji wymagają wszystkie operacje poza czytaniem zawartości tabel groups/subjects/subject\_groups/events\_\*.

Struktura bazy danych znajduje się w [master]/\_Info/egzaminel\_db.sql

**Aby zmienić w projekcie connectionString należy zrobić to w pliku appsettigins.json. Aby zmienić sposób dostarczania connectionStringa można zrobić to w klasie IConfig.**

## Aplikacja

### Walidacja użytkownika

Walidacja użytkownika odbywa się przez metodę **api/users/login**. Należy podać id i hasło, po walidacji API odeśle w headerze token ważny przez 15 min (odnawia się po każdym użyciu). Należy podawać go w zapytaniu restowym w headerze „Authorization”.

W przypadku negatywnej walidacji API wyrzuca odpowiedni wyjątek.

### Schemat działania aplikacji

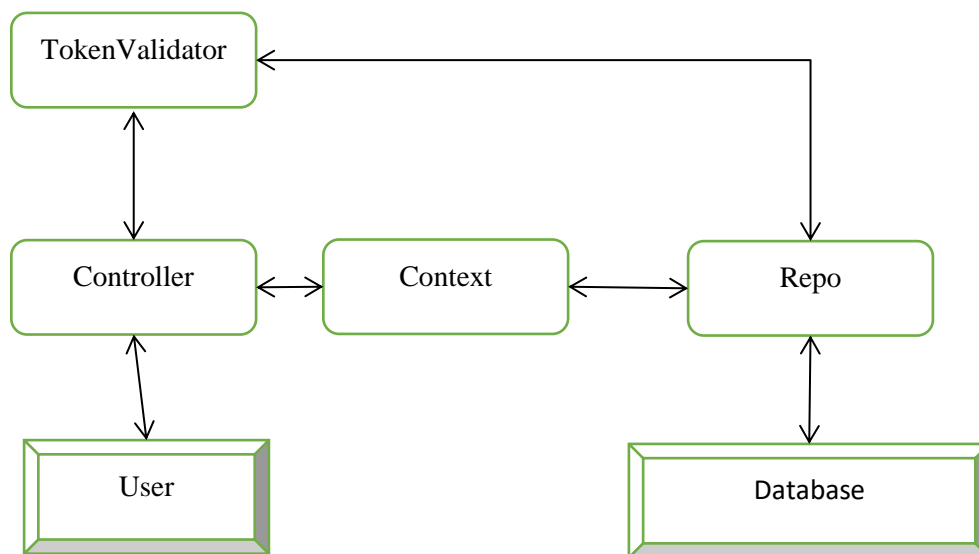
1. Walidacja ważności tokenu (dotyczy tylko niektórych metod). TokenValidator łączy się z repo w celu walidacji tokenu.
2. Controller łączy się z contextem przekazując odpowiednie obiekty DAO.
3. Context łączy się z repozytorium w celu walidacji uprawnień użytkownika (dotyczy tylko niektórych metod).
4. Jeżeli walidacja jest prawidłowa context wywołuje metody repozytorium, w przeciwnym wypadku wyrzuca wyjątek.
5. Repozytorium łączy się z bazą danych i przeprowadza odpowiednie operacje

### Logowanie użytkowników

1. Użytkownik podaje swoje dane, szyfrowane algorytmem RSA.
2. Context odszyfrowuje dane (poprzez ponowne generowania hasha z danych podanych przez użytkownika).
3. Context łączy się z repozytorium, jeżeli dane są poprawne baza zwraca unikatowy token.
4. Token zostaje wysłany w headerze jako pole „Authorization”
5. Token ważny jest 15 min od ostatniej akcji. Użytkownik podaje go w requestach HTTP w headerze w celu walidacji.





### Rejestracja użytkowników

1. Użytkownik podaje swoje dane, szyfrowane algorytmem RSA.
2. Context generuje salt oraz haszuje hasło.
3. Context łączy się z repozytorium wysyłając DAO użytkownika z wygenerowanymi danymi.
4. Repozytorium zapisuje dane w bazie.



## Spis pakietów i użytych frameworków

W projekcie zostały użyte następujące nugety:

	<b>Microsoft.AspNetCore.All</b> by Microsoft Microsoft.AspNetCore.All	v2.0.0 v2.0.3
	<b>Microsoft.NETCore.App</b> by Microsoft A set of .NET API's that are included in the default .NET Core application model. e8b8861ac7fa1042c87a5c2f9f2d04c98b69f28d	v2.0.0
	<b>Microsoft.VisualStudio.Web.CodeGeneration.Design</b> by Microsoft Code Generation tool for ASP.NET Core. Contains the dotnet-aspnet-codegenerator command used for generating controllers and views.	v2.0.0 v2.0.1
	<b>MySQL.Data</b> by Oracle MySQL.Data.MySqlClient .Net Core Class Library	v6.10.4 v6.10.5

Do Dependency Injection został użyty domyślny framework zawarty w ASP.NET Core 2.

Pakiety w projekcie:

- / - klasy odpowiadające za start aplikacji
- /Auth – klasy odpowiadające za walidację tokenu użytkownika
- /Context – warstwa odpowiadająca za logikę aplikacji
- /Controllers – warstwa odpowiadająca za komunikację z klientem
- /DataAccess – warstwa odpowiadająca za komunikację z bazą danych
- /Helpers – Extensions method, Utilsy etc.
- /Models – Obiekty DAO używane w projekcie

Listę dostępnych metod można zobaczyć importując kolekcję do Postmana.

Kolekcja znajduje się: **[master]/\_Info/Backup.Postman\_dump.sql**