**QuizMakerAPI – Tehnička dokumentacija**

**Arhitektura projekta:**

* QuizMaker.API – Glavni projekt zadužen za obradu api poziva
* QuizMaker.Core – Library koji sadrži sve entitete, DTO-e i interfejse
* QuizMaker.Data – Library za rad sa bazom podatak – EF
* QuizMaker.UseCases – Library za poslovnu logiku aplikacije
* QuizMaker.API.Tests – Projekt za pisanje testova

**QuizMaker.API**

* U sebi ima referencirano Core, Data i UseCases
* Koristi AutoMapper za promjene iz entiteta u DTO
* Ima BaseAPIController u koji su sa DI injectani fieldovi za rad sa bazom (\_quizMakerDb), sa mapperom(\_mapper) i servis za pitanja (\_questionService)
* QuestionController nasljeđuje od BaseAPIControllera i tu su svi api endpointi koji su vezani za pitanja
* QuizController nasljeđuje od BaseAPIControllera i tu su svi api endpointi koji su vezani za kvizove. U njega je injectan i exporter servis (\_exportService)

**QuizMaker.Core**

* U ovom projektu se nalaze folderi DTOs (svi DTO-i od aplikacije), Entities (svi entiteti od aplikacje) i Interfaces (svi interfejsi od aplikacije)

**QuizMaker.Data**

* U sebi ime referenciran Core
* Ima instaliran nuget EntityFramework za rad sa bazom podataka
* U njemu se nalaze
  + Folder Contexts u kojem je QuizMakerContext u kojem je napravljena cijela baza podataka i svi indeksi
  + folder Repositories u kojem se nalazi glavni Repozitorij i repozitoriji za glavne entitete koji nasljeđuju od njega
  + folder Migrations u kojoj su sve migracije koje su izvršene
  + EFTransacion – klasa za korištenje sql transakcija
  + UnitOfWork – glavna klasa za rad sa bazom
* **VAŽNA NAPOMENE**
  + Da bi se moglo pokrenuti kreiranje baze potrebno je napraviti praznu bazu i njezin connection string staviti u web config od API aplikacije i App config od Data aplikacije
  + Prije nego se pokrene update na serveru je potrebno instalirati podršku za FullText indekse i nakon toga na bazi treba izvršiti naredbu CREATE FULLTEXT CATALOG QuizMakerFullTextCatalog AS DEFAULT; - Ako samo želite isprobati aplikaciju i nemate ovo instalirano onda je potrebno otvoriti 202409282108489\_AddFullTextIndex.cs iz Migrations foldera i komentirati sve u Up() i Down() metodama.
  + Kreiranje baze se pokreće iz Package Manager Console sa naredbom : Update-Database -ProjectName QuizMaker.Data -StartUpProjectName QuizMaker.API

**QuizMaker.UseCases**

* U sebi ima referencirano Core
* Ima instalirano System.ComponentModel.Composition za rad sa MEF
* U folderu Exporters su eksporteri, trenutno samo jedan
  + CSVExporter – služi za eksportanje kviza u csv
* U folder Services su servisi
  + QuestionService – radi hash od teksta pitanja tako da sa njim osiguramo da se ne mogu spremati dupla pitanja (kada bi se samo gledalo text pitanja to bi previše opterećivalo sustav jer je napravljeno da može imati 1000 znakova i hash je puno kraći i bolje rješenje za to)
  + ExportService – služi za rad sa exporterima, dohvaća sve one koji imaju interfejs IExporter. Koristi MEF pa se mogu kao dll-ovi ubacivati novi exporteri bez novog kompajliranja programa
  + AuthService – služi za prijavu u aplikaciju i dohvaćanje JWT tokena

**Zaštita**

API je zaštićen sa JWT (Java Web Token). Trenutno imamo u configu podatke za ulazak u aplikaciju (username i password).

<add key="AdminUsername" value="your\_username" />

<add key="AdminPassword" value="your\_password" />

Kada se aplikacija postavi na server tamo se ti podaci mogu promijeniti u nešto drugo tako da osjetljivi podaci neće biti vidljivi na serveru. Ovako to izgleda na Azure serveru:

A screenshot of a computer

Description automatically generated

**Kontrola grešaka**

Za zapisivanje grešaka koristi se NLog paket. Konfiguriran je tako da se za svaki dan (u kojem se dogodi neka greška) otvori drugi txt dokument u koji se zapisuju sve greške za taj dan.

Postoji GlobalExceptionFilter koji zapisuje sve greške koje nisu uhvaćene sa try/catch.

U kontrolerima za api pozive se greške hvataju sa try/catch pa tamo u BaseApiControlleru imamo posebnu metodu LogError koja se poziva u Catch bloku i u log file zapisuje naziv kontrolera i naziv metode u kojem se greška dogodila i tekst od te greške. Kako se može vidjeti na gornjoj slici na serveru se isto može promijeniti lokacija gdje će se txt dokumenti sa greškama spremati.