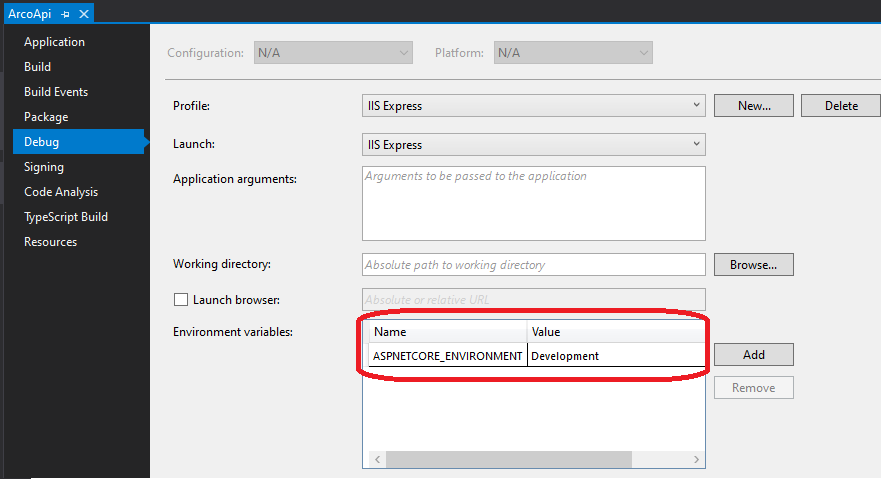
ArcoAPI

ArcoAPI è una **Web API** scritta in C# con il framework **ASP .NET Core 3.1**, che si appoggia ad **Entity Framework Core** per la comunicazione con il database SQL Server da cui estrae i dati. Elabora richieste in sola lettura, ovvero non fa altro che restituire dati in base a dei parametri forniti dal richiedente. Per il logging viene usato il framework **Serilog**, configurato per scrivere su file nella cartella dedicata *Logs*.

La soluzione è composta dall’API vera e propria, il progetto ArcoAPI, e dal progetto **IdentityProviderMock**, il cui utilizzo è dedicato allo sviluppo locale. Esso è un’API che mima e fornisce ad ArcoAPI i dati di configurazione, che altrimenti verrebbero consegnati da INAIL, riguardanti la validazione degli *access token*, di cui si discute più avanti. L’indirizzo da cui reperire tali dati è inserito nel file **appsettings.json** come “WellKnownEndpoint”, dunque varia in base all’ambiente e non bisogna preoccuparsi di cambiarlo manualmente nel codice nel caso non si sviluppi più in locale ma, ad esempio, in ambiente di collaudo. Bisogna invece specificare la variabile d’ambiente nelle proprietà del progetto. Di seguito un esempio con la variabile per l’ambiente di **Sviluppo**:



Modelli e viste

Al momento non sono mappati modelli veri e propri, soltanto le seguenti viste:

ViewQlikAuditOperativoAccesso

ViewQlikDatiPraticaAudit

ViewQlikDomandaValore

ViewQlikPraticaGruppo

ViewQlikRilievo

ViewQlikSede

ViewQlikTeam

Controller, logica di business ed interfacce

L’applicazione è stata predisposta per futuri miglioramenti del codice ed espansione dei modelli, nonché dell’imposizione di eventuale logica di business vera e propria (attualmente assente), ma al momento fa uso di una sola classe, **QlikBusiness**, che viene passata al controller, **QlikController**, tramite dependency injection. **QlikBusiness** implementa un’unica interfaccia, **IQlikBusiness**, coi metodi (la cui logica è identica) dedicati ad ogni vista: uno per ottenere il totale degli elementi, uno per restituire un certo numero di record da una certa pagina. **QlikController** accetta solo request di tipo **POST** ed implementa l’interfaccia corrispondente **IQlikController** ed espone dunque un endpoint per ogni metodo riguardante gli elementi di una pagina ed uno soltanto che gestisce, tramite il nome della vista ottenuto come parametro, la chiamata interna al vero metodo che restituisce il numero di elementi nella vista.

Endpoint esposti

Tutti gli endpoint dell’API restituiscono dati in formato **JSON**, con un eventuale messaggio di errore, un codice identificativo (ad esempio: 500 per errori interni del server) e il risultato della richiesta.

Al momento ArcoAPI espone un solo endpoint per l’ottenimento del numero totale di elementi contenuti in una certa vista. Il numero restituito dipende dalla **vista** inviata come parametro, ad esempio:

<https://localhost:44382/viewqlikapi/totaleelementivista?vista=ViewQlikSede>

Dagli altri endpoint si ottiene il **numero di elementi** richiesti da una certa **pagina**, entrambi mandati come parametri (negli esempi successivi, 100 elementi dalla pagina 2):

<https://localhost:44382/viewqlikapi/domandavalore?numeroElementi=100&indicePagina=2>

<https://localhost:44382/viewqlikapi/team?numeroElementi=100&indicePagina=2>

<https://localhost:44382/viewqlikapi/sede?numeroElementi=100&indicePagina=2>

<https://localhost:44382/viewqlikapi/datipraticaaudit?numeroElementi=100&indicePagina=2>

<https://localhost:44382/viewqlikapi/auditoperativoaccesso?numeroElementi=100&indicePagina=2>

<https://localhost:44382/viewqlikapi/rilievo?numeroElementi=100&indicePagina=2>

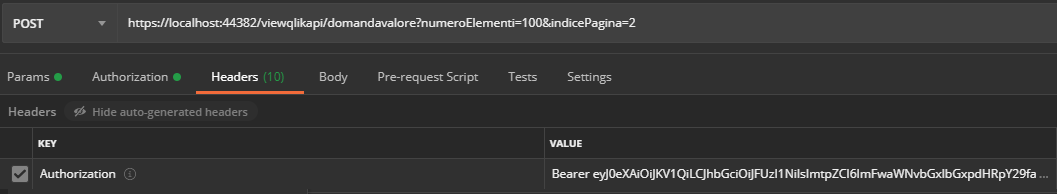
<https://localhost:44382/viewqlikapi/praticagruppo?numeroElementi=100&indicePagina=2>

Autenticazione e autorizzazione tramite Open ID Connect

L’API richiede l’autenticazione e l’autorizzazione ad ottenere i propri dati secondo il protocollo OIDC. Il client che effettua una richiesta ad un endpoint dell’API deve presentare un *access* *token* ottenuto dall’Identity Provider di INAIL. L’Identity Provider autentica il client e, in caso, ne autorizza l’accesso ad alcune funzioni ed operazioni offerte dall’AP, che deve comunque controllare che il token presentato sia valido. Solo in caso positivo, secondo quanto esposto nei documenti condivisi nel GitLab di INAIL, ArcoAPI concede l’accesso ai dati.

L’*access token*, o *bearer token*, è un’insieme codificato di dati in formati JSON. Esso contiene informazioni sul client, come le credenziali le autorizzazioni possedute e la firma codificata.

Viene passato nell’header della request nella chiave ***Authorization***, indicandolo come *Bearer*, come nel seguente esempio:



Per quanto riguarda il protocollo Open ID Connect (OIDC), che estende OAuth2, si rimanda a documenti sui siti ufficiali <https://oauth.net/2/> e <https://openid.net/connect/>.

Esecuzione di test

Attualmente non è possibile autogenerare un token, perciò si deve simulare un client da programmi come Postman, inviando come descritto il seguente token, ottenuto da INAIL:

eyJ0eXAiOiJKV1QiLCJhbGciOiJFUzI1NiIsImtpZCI6ImFwaWNvbGxlbGxpdHRpY29fa2V5In0.ew0KImlzcyI6ICJ3d3cuaW5haWwuaXQiLA0KImlhdCI6IDE1ODcxMzk2OTQsDQoianRpIjogIkFHQ0NkYjUxNDk4ZC1iM2Y5LTQxZGEtODJjZi1mMDU2YTljOWFjNWMiLA0KImF6cCI6ICJDbGllbnRUZXN0QVBQIiwNCiJleHAiOiAxNTg3MTUwNDk0LA0KImF1ZCI6IFsiSldUU0dQIiwiY29sbHBvcnRhbGUuaW5haWwuaXQiXSwNCiJzdWIiOiAieWVuZzU1MyIsDQoidW5pcXVlX25hbWUiOiAieWVuZzU1MyIsDQoidXNlcl9uYW1lIjogInllbmc1NTMiLA0KImFjciI6ICIiLA0KInJvbGVzIjogW10sDQoiYXV0aG9yaXRpZXMiOiBbXSwNCiJzY29wZSIgOiAiIg0KfQ.jB6LazkmHfDbLuZJnI1-xTfmtXMeUSPAMyjCTaKTJlBq3hFQTaXpppBWx5g9iUX04bKsbbLDbl3FrE0EPc5iPQ

Il framework permette di personalizzare i vari parametri di controllo del token. La validazione da parte dell’API è impostata nell’applicazione tramite la classe **AuthenticationService**. Di seguito una panoramica dell’*extension method* per la creazione del servizio, richiamato nella classe **Startup** (in evidenza la parte fondamentale per la validazione). Per un risultato sempre positivo, impostarla come nell’esempio.

