A Simple Trader Service

Intro

The service provides a very simple handling (base CRUD operations) of traders list. It is based on GraphQL technology. GraphQL allows to retrieve and submit stored data in a very ordered and at the same time flexible manner. It provides schema acting as a contract between client and server. The schema also defines retrieval procedure in the server.

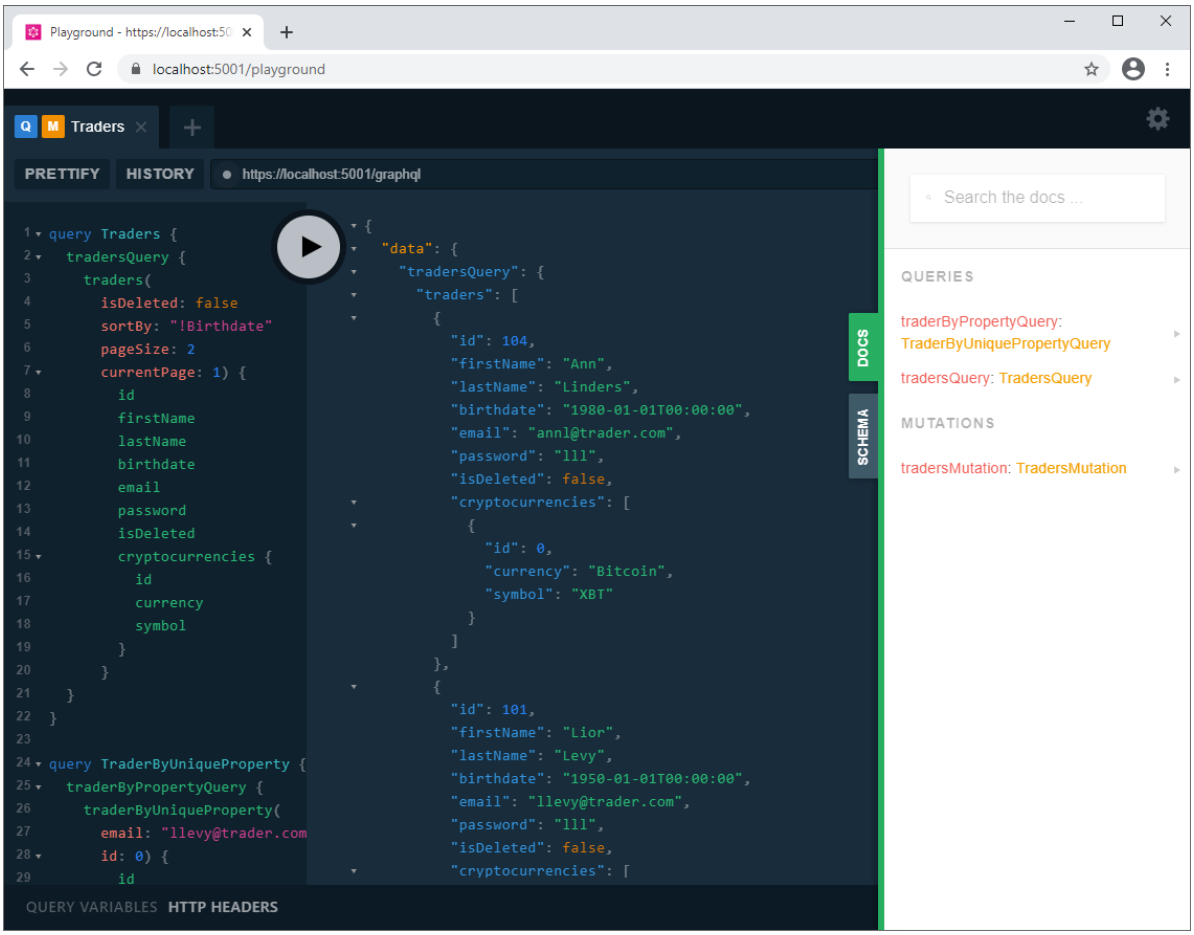
How to Run?

Prerequisites (for Windows):

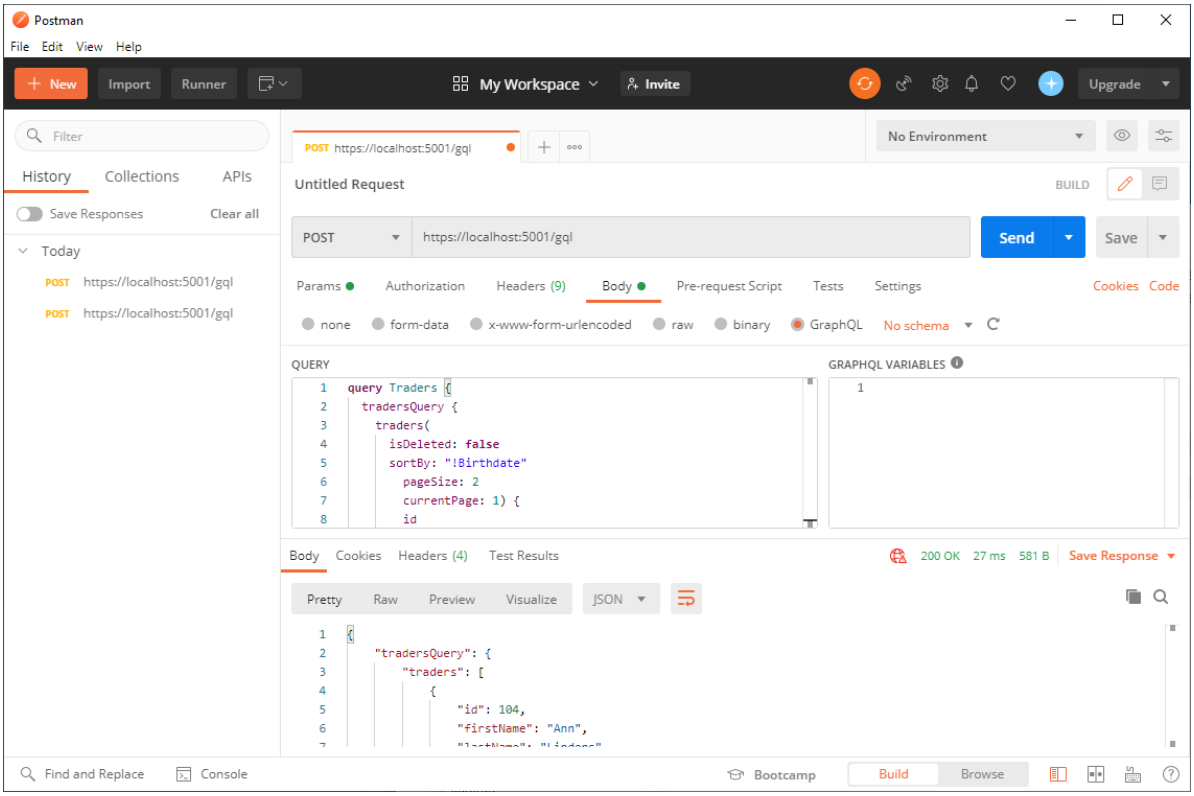
- Local SQL Server (please see connection string in file *appsettings.json* of the service),
- Visual Studio (VS) 2019 with .NET 5 support.

Sequence of Actions

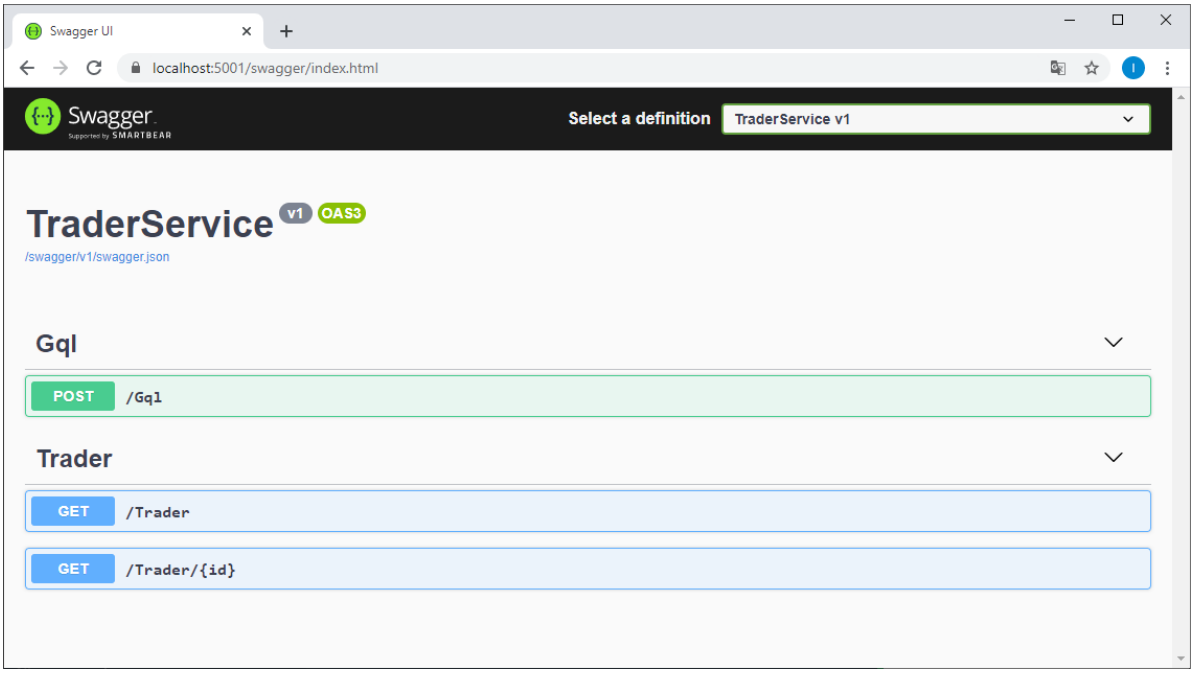
1. Open solution *TraderService.sln* with Visual Studio 2019 (VS) that supports .NET 5 and build it.
2. SQL Server is used. For the sake of simplicity, Code First paradigm is adopted. Database *TradersDb* is automatically created when either the service or its integration tests run. Please adjust connection string (if required) in *.\appsettings.json* service configuration file.
3. Run *TraderService*.
   3.1 It may be carried out from VS either as a service or under IIS Express.
   Browser with *Playground* Web application for GraphQL starts automatically.
   3.2 Alternatively, the service may be started by activating
   .\TraderService\TraderService\bin\Debug {or Release}\net5.0\TraderService.exe .
   In this case browser should be started manually browsing on https://localhost:5001/playground when the service is already running.
   In *Playground* Web page you may see GraphQL schema and play with different queries and mutations
   3some predefined query and mutation may be copied from *.\QueriesSample.txt* file).



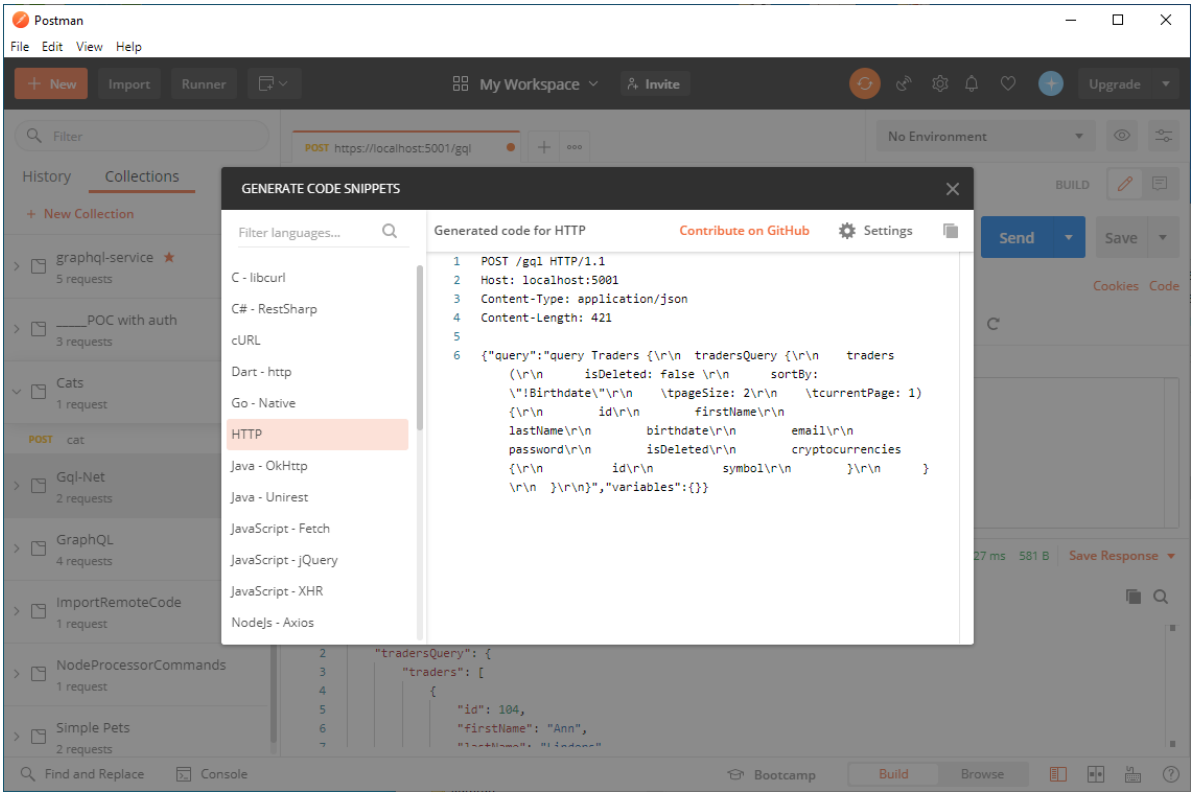
4. *Playground* application uses middleware to run (it is mostly used during development, but in this project available in all versions). It does not call *GqlController* that is used by clients in production. To work with *GqlController* you may use Postman application.
   From Postman make a POST to https://localhost:5001/gql
   with Body -> GraphQL providing in QUERY textbox your actual GraphQL query / mutation.



5. You may also use *OpenApi* (a.k.a. *Swagger*): browse to https://localhost:5001/swagger and activate POST /Gql .



In Postman press *Code* link in the upper-right corner, copy query to Swagger's *Request body* textbox and execute method.



6. In all cases you may use unsafe call to http://localhost:5000 (allowed for illustration and debugging).
7. Integration tests may be found in project *TraderServiceTest* in directory .*Test* .