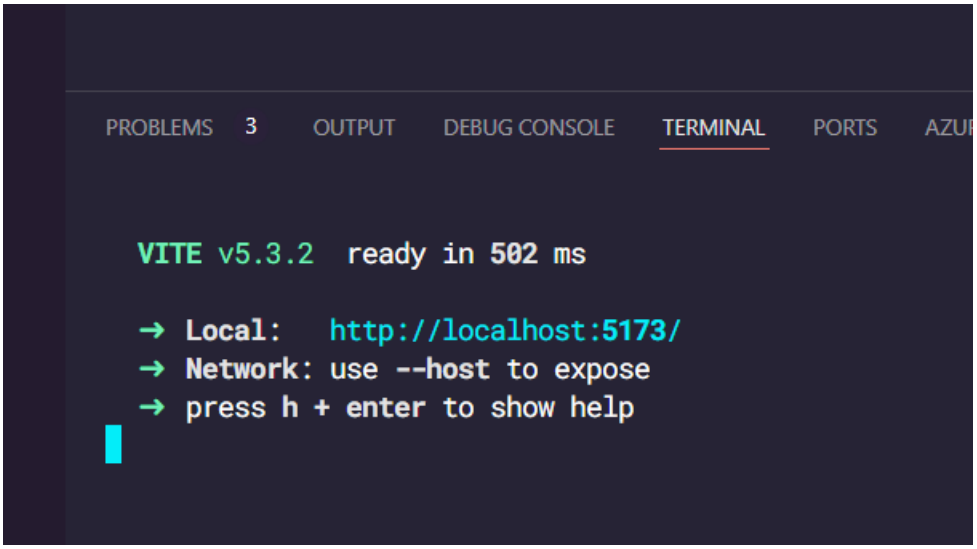
How to Run

1. You will need to run schema under SSMS '' which is found under TwitterDeluxeAuth2 > TwitterThrice.data (project) > Schema > create database and necessary tables v2.1.sql
2. You will need to have NodeJS installed on your machine and run NPM install under 'twitter-thrice-client' directory.
3. To allow communication between back-end and front-end, we need to ensure port number found under (program.cs of TwitterDeluxeAuth2 > TwitterDeluxeAuth2.api) CORS 'AllowSpecificOrigin' is correct

```
40    // Configure CORS
41    builder.Services.AddCors(options => {
42        options.AddPolicy("AllowSpecificOrigin",
43            builder => {
44                builder.WithOrigins("http://localhost:5173") // Specify the allowed origin
45                       .AllowAnyHeader()
46                       .AllowAnyMethod();
47            });
48    });
49
50
```

4. To run the client side you will need to run 'npm run dev'. The process will show you the local URL, which you can use to update CORS 'AllowSpecificOrigin' mentioned above.



5. On the client side, there is a .env file that holds information for the server-side URL. You will need to change the port number to that which is shown when you run the server-side.

```
VITE_BASE_URL=https://localhost:7206/
```

If you can't find .env in your file explorer then you will need to go to options and allow hidden files to be shown.

6. You can quickly populate the database by running ‘TwitterThrice.testKonsole’ project which can be found under TwitterDeluxeAuth2 solution.

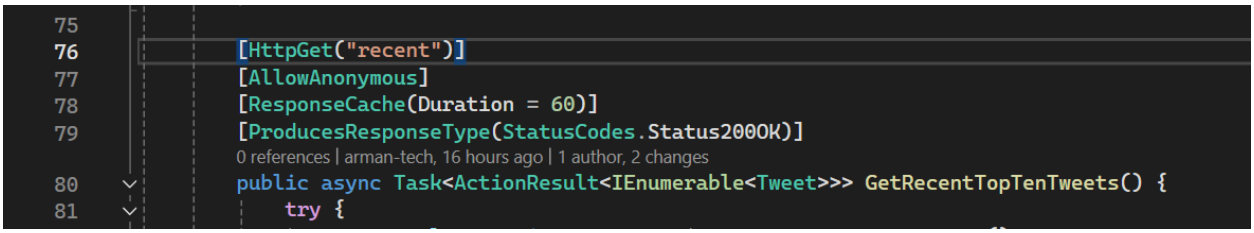
Optimization

- As part of the schema file, I have introduced two indexes that will allow for faster retrieval.

```
-- introduce indexing to make the query process faster when grabbing top 10 out of 25M records.
CREATE INDEX IX_Tweets_PostedDate ON dbo.Tweets(PostedDate DESC);
GO

-- introduce indexing to make the query process faster when searching by email
CREATE INDEX IX_Members_Email ON dbo.Members(Email);
GO
```

- When it comes to web API end points I have introduced response caching



Security

I have considered sensitive information to be a user’s email and password. Because of their sensitive nature, I have used a simple encryption process to hide the original contents. I should be noted that the encryption is simple and is meant to be for demonstration purposes.

Server side

TwitterDeluxeAuth2.api – holds relevant API content.

TwitterThrice.common – holds common code that are used by one more layer (api, domain (business layer), and data (database layer).

TwitterThrice.data – database layer.

TwitterThrice.domain – business layer.

TwitterThrice.testKonsole – used to quickly populate the database with random data for large set testing.

Client side

Important portion of the code is found under /src folder.

/components – holds any components that can be reused by any page (also called views)

/models – holds basic POCO classes that represent models.

/routes – holds necessary information on routing process

/stores – holds the store management that is used to communicate between the front-end and back-end via axios.

/views – these are the pages that are used to display to the user. These include Home page and Tweet page.