

Solution Structure

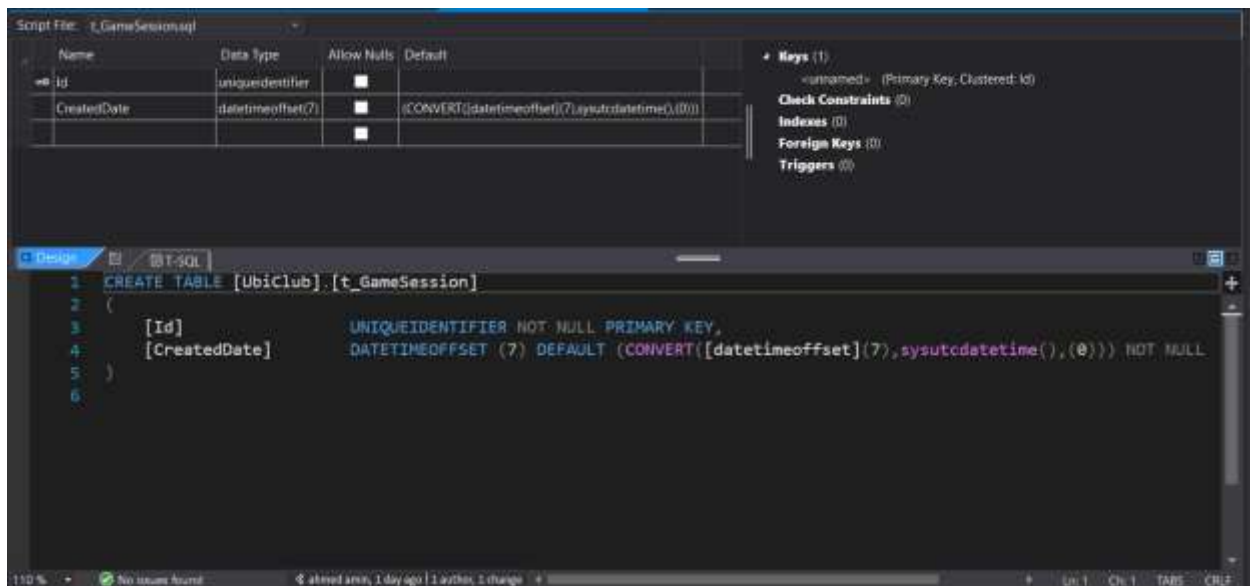
- Solution is built using .NET core 3.1 and Visual Studio 2019.
- HTTP endpoints are implemented as Azure Functions.
- Database Schema is maintained in Database project.
- Solution consists of following projects
 - **“UbiClub.DB”**: SQL-Server Database project; to maintains Db schema and publish database changes.
 - **“UbiClub.Feedback.Core”**: dot NET core class library used to define Model classes used by multiple projects in solution.
 - **“UbiClub.Feedback.Entities”**: dot NET core class library defines database entities and database context.
 - It uses EF core 3.1 as ORM framework.
 - **“UbiClub.Feedback.Data”**: dot NET class library defines generic repository and data-access services.
 - **“UbiClub.Feedback.Api”**: Azure Function 3.0 project. It defines Http endpoints as azure functions.

Database Schema

Tables

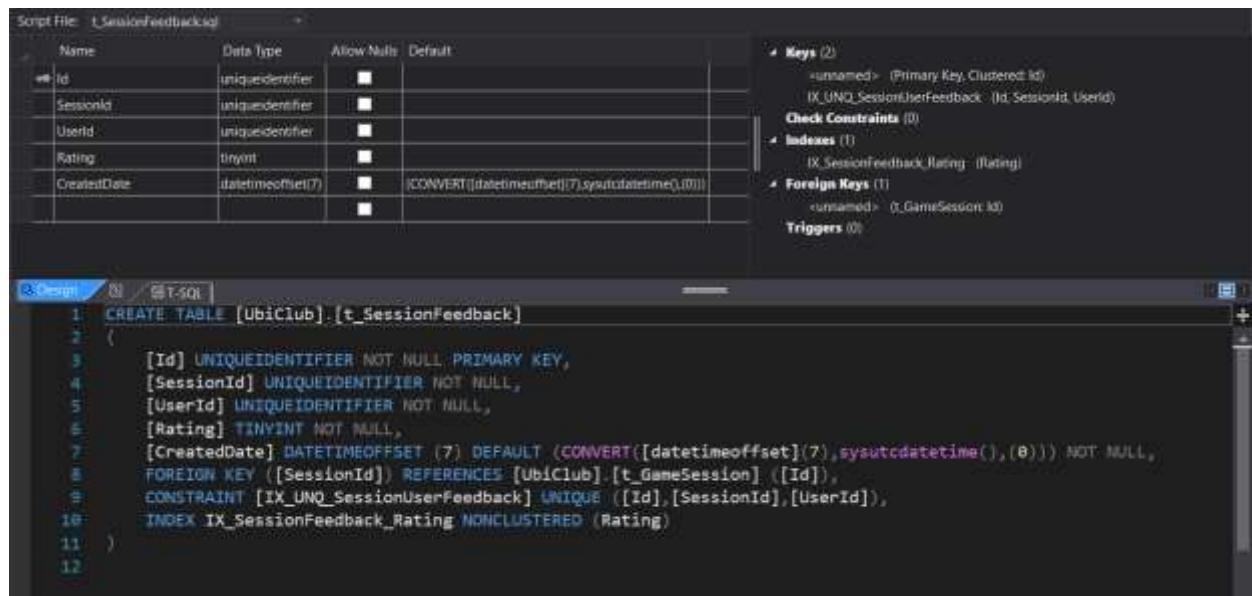
“t_GameSession” table

- Stores information about game sessions.



“t_SessionFeedback” table

- Stores users' rating about specific game session



Api Documentation

Post feedback endpoint

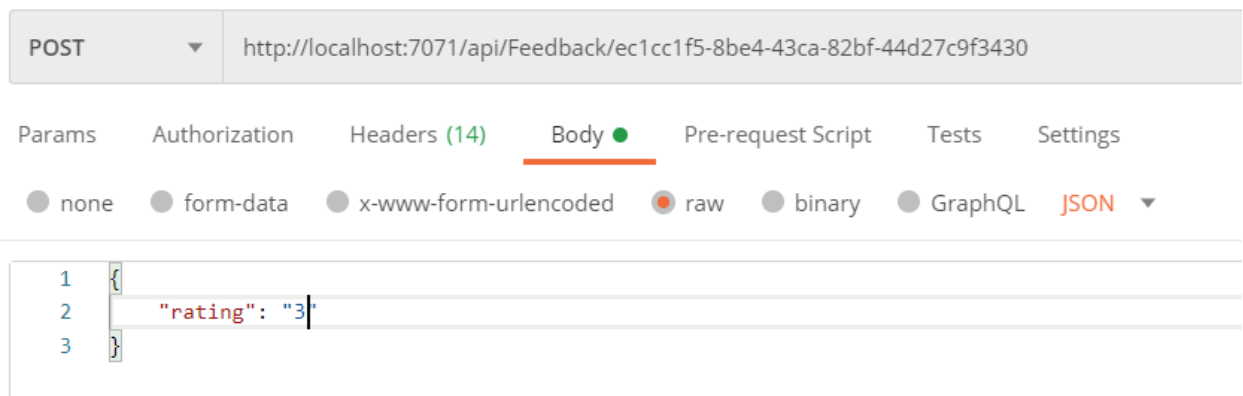
Route

- Http verb: POST
- Endpoint URL: /api/feedback/<session-id>

Input parameters

name	type	required	source
Session-id	GUID	yes	Endpoint URL path
User-id	GUID	yes	"Ubi-UserId" header in http request
rating	byte	yes	Request body

Request example



	KEY	VALUE
<input checked="" type="checkbox"/>	Content-Type	application/json
<input checked="" type="checkbox"/>	Accept-Encoding	gzip, deflate, br
<input checked="" type="checkbox"/>	Accept	application/json
<input checked="" type="checkbox"/>	Connection	keep-alive
<input checked="" type="checkbox"/>	Ubi-UserId	E0E04A7C-E836-4A57-93B9-028294B55C70

Response status codes

- 201: used when feedback is created successfully.
- 400: used when there are validation errors in request.
- 500: internal server error.

Success response example

Body
Cookies
Headers (5)
Test Results
Status: 201 Created

Pretty
Raw
Preview
Visualize
JSON

```

1 {
2   "sessionId": "ec1cc1f5-8be4-43ca-82bf-44d27c9f3430",
3   "userId": "e0e04a7c-e836-4a57-93b9-028294b55c70",
4   "rating": 3,
5   "id": "f24dead6-fa13-4dac-0ca8-08d8ba39816e",
6   "createdDate": "2021-01-16T16:12:22.0799774+00:00"
7 }

```

Error Response example

Body
Cookies
Headers (4)
Test Results
Status: 400 Bad Request
Time: 49 ms

Pretty
Raw
Preview
Visualize
JSON

```

1 {
2   "code": "BadArgument",
3   "message": "Feedback Create Request Data contains invalid/missing arguments",
4   "details": [
5     {
6       "target": "SessionId",
7       "message": "'Session Id' must not be empty."
8     }
9   ]

```

Get Feedback endpoint

Route

- Http verb: GET
- Endpoint URL: /api/feedback?rating=5

Input parameters

name	type	required	source
rating	byte	optional	Query string

Request example

GET

▼

http://localhost:7071/api/Feedback?rating=4

Params ●

Authorization

Headers (14)

Body ●

Pre-requests

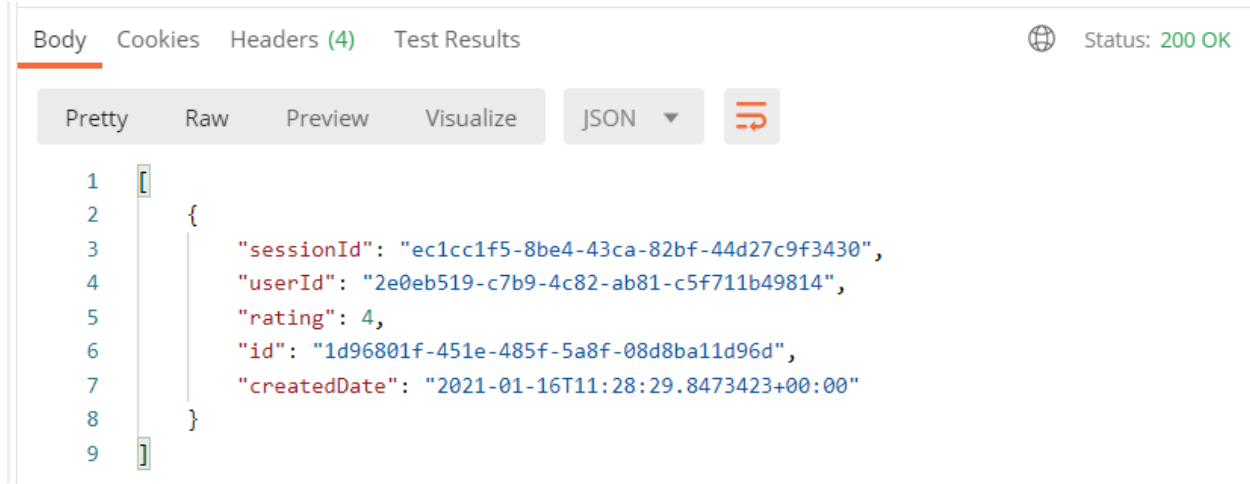
Query Params

	KEY	VALUE
<input checked="" type="checkbox"/>	rating	4
	Key	Value

Response status codes

- 200: data is retrieved successfully.
- 400: used when there are validation errors in request.
- 500: internal server error.

Success response example



The screenshot shows a REST client interface with tabs for Body, Cookies, Headers (4), and Test Results. The Body tab is selected, and the response is displayed in JSON format. The status is 200 OK. The JSON response is an array containing a single object with the following properties:

```
1 [
2   {
3     "sessionId": "ec1cc1f5-8be4-43ca-82bf-44d27c9f3430",
4     "userId": "2e0eb519-c7b9-4c82-ab81-c5f711b49814",
5     "rating": 4,
6     "id": "1d96801f-451e-485f-5a8f-08d8ba11d96d",
7     "createdDate": "2021-01-16T11:28:29.8473423+00:00"
8   }
9 ]
```

Error response example



The screenshot shows a REST client interface with tabs for Body, Cookies, Headers (4), and Test Results. The Body tab is selected, and the response is displayed in JSON format. The status is 400 Bad Request. The JSON response is an object with the following properties:

```
1 {
2   "code": "BadArgument",
3   "message": "Feedback Get Request Data contains invalid/missing arguments",
4   "details": [
5     {
6       "target": "Rating",
7       "message": "'Rating' must be between 1 and 5. You entered 8."
8     }
9   ]
}
```

Run App locally steps

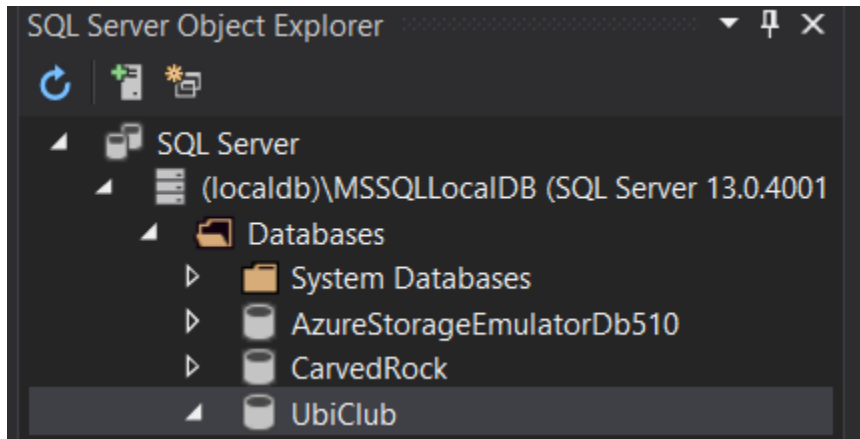
Prerequisites

- SQL Server Express LocalDB. More information about how to install and connect to LocalDB at <https://docs.microsoft.com/en-us/sql/database-engine/configure-windows/sql-server-express-localdb?view=sql-server-ver15#start-localdb-and-connect-to-localdb>
 - As mentioned in official documentation, the first time a user on a computer tries to connect to LocalDB, the automatic instance must be both created and started. The extra time for the instance to be created can cause the connection attempt to fail with a timeout message. When this happens, wait a few seconds to let the creation process complete, and then connect again.
- Azure functions tools. It is required to include the Azure development workload in Visual Studio installation (if it isn't installed)

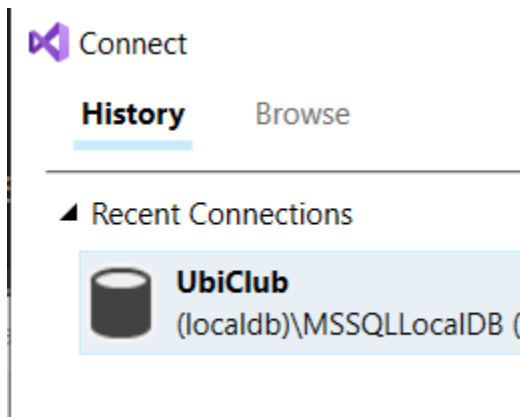
- more info about modifying VS workloads at <https://docs.microsoft.com/en-us/visualstudio/install/modify-visual-studio?view=vs-2019>

To Create database and insert seed data for testing purposes

- Open solution via “UbiClubFeedbackApp.sln”
- Open SQL Server Object explorer
- Connect to “(LocalDB)\MSSQLLocalDB”
- Create new database and name it “**UbiClub**”.



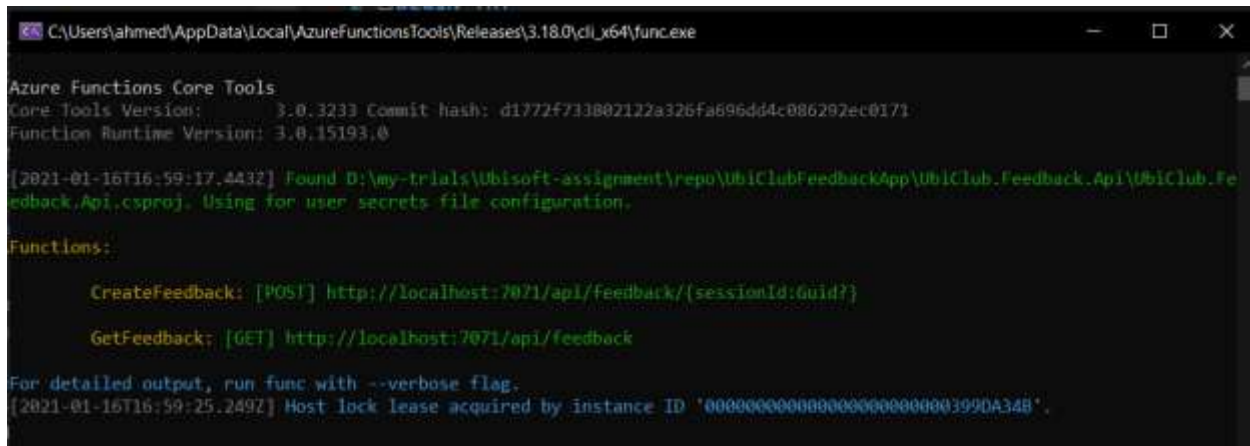
- Rebuild solution to restore nuget packages
- Right-click database project and click “**Publish**” from context menu
- Press “Edit” button in “Publish Database” dialog
- Select “UbiClub” database from “Connect” dialog and press “OK”



- Press “Publish” button.
- **Note:**
 - test data is defined in “**Scripts/ InsertGameSession_TestData.sql**” in database project.
 - Test script inserts test records in “**t_GameSession**” table.
 - **It is assumed** that game session data already exist in DB so when submitting POST requests to insert feedback, please use game session ids (defined in test data).

Run and Test Azure functions locally

- Select “**UbiClub.Feedback.Api**” as startup project
- Press F5
- Azure Functions Tools will run and console window will appear



```
C:\Users\ahmed\AppData\Local\AzureFunctionsTools\Releases\3.18.0\cli_x64\func.exe

Azure Functions Core Tools
Core Tools Version:      3.0.3233 Commit hash: d1772f733802122a326fa696dd4c086292ec0171
Function Runtime Version: 3.0.15193.0

[2021-01-16T16:59:17.443Z] Found D:\my-trials\Ubisoft-assignment\repo\UbiClubFeedbackApp\UbiClub.Feedback.Api\UbiClub.Feedback.Api.csproj. Using for user secrets file configuration.

Functions:

    CreateFeedback: [POST] http://localhost:7071/api/feedback/{sessionId:Guid?}

    GetFeedback: [GET] http://localhost:7071/api/feedback

For detailed output, run func with --verbose flag.
[2021-01-16T16:59:25.249Z] Host lock lease acquired by instance ID '0000000000000000000000000399DA348'.
```

- The application is now ready to receive any Http requests
- More information about running and testing azure functions locally at <https://docs.microsoft.com/en-us/azure/azure-functions/functions-develop-vs#testing-functions>