13.03.2021

# FEEDBACK MICROSERVICE

## Backend Technical Test @UbisoftClub

Betül BOYLU
Senior Software Engineer
betulboylu@yahoo.com

# Index

DEFINITION....2
TECHNOLOGIES....2
ENDPOINTS....2
1. GET....2
2. POST....3
RUNNING INSTRUCTIONS....4
RESULTS....5
DATABASE....6

## DEFINITION

Feedback Microservice is a REST API that records user feedbacks for game sessions and lists the last 15 feedbacks as requested in the technical test by Ubisoft.

## TECHNOLOGIES

The microservice is developed by using .NET Core 5.0, Entity Framework, Repository, Dependency Injection and MS SQL Server 2019 with code first approach.

## ENDPOINTS

The API has two endpoints.

### 1. GET

**Endpoint** : https://localhost:44381/api/Feedback?Filter=max

The method receives **Filter** parameter from QueryString and returns to the last 15 feedback records by filtering according to the parameter.

| Parameter | Type | Receiving Location |
|---|---|---|
| **Filter** | String | from QueryString |

Table 1: Parameters for GET method

The records are listed in descending order if the Filter parameter is set to “max”, and in ascending order in any other case as seen on Figure1.

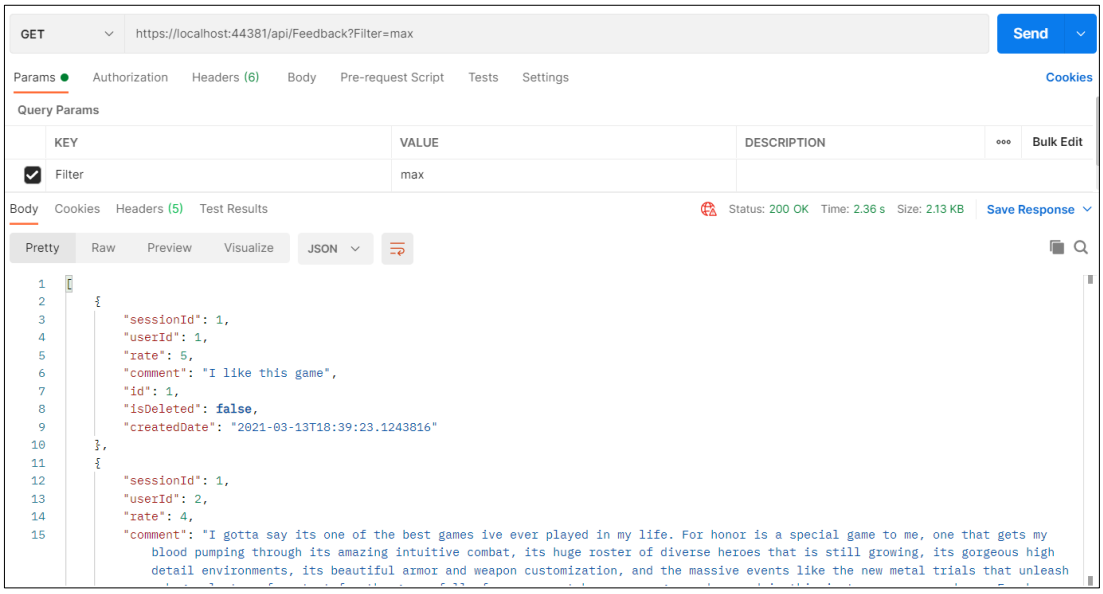


Figure 1: GET method result retrieved by using Postman application.

## 2. POST

**Endpoint :** https://localhost:44381/api/Feedback/1

The method records a user feedback for a game session. It allows a single feedback for each session for the same user. It checks if the user and session exists and then records the feedback. It receives parameters from 3 different locations as seen on Table 2.

| Parameter | Type | Receiving Location |
|---|---|---|
| **SessionId** | Int | from request URL |
| **Ubi-UserId** | Int | from request header |
| **{Rate, Comment}** | JSON object | from request body |

Table 2: Paramaters for POST method

Figure 2 shows SessionId parameter send via request URL and Ubi-UserId parameter send in the Header. The method returned recorded feedback as seen on the Response section.

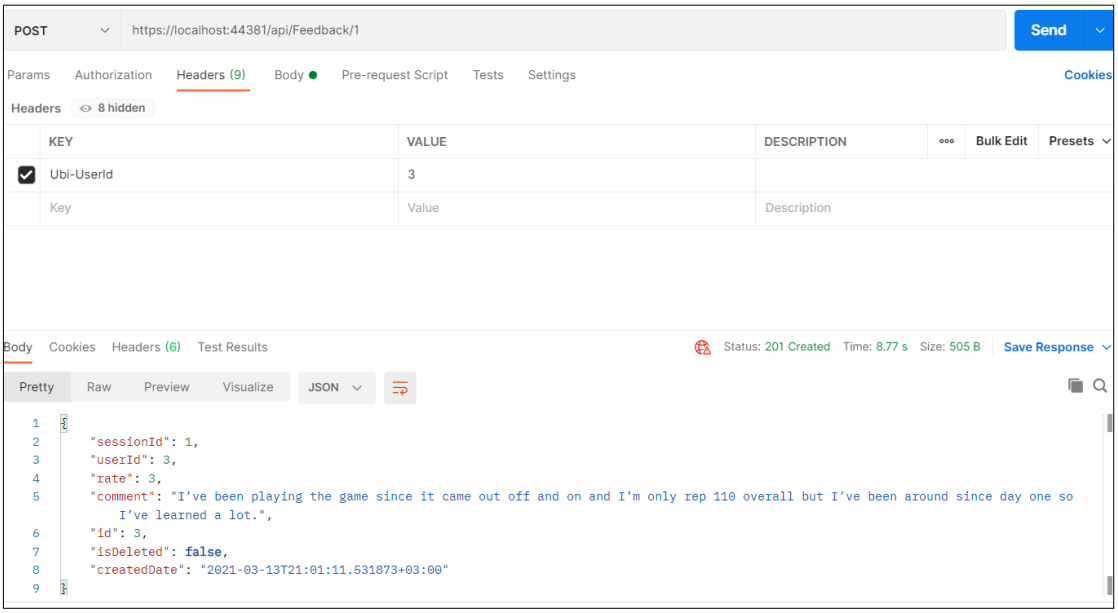


Figure 2: POST method result retrieved from Postman application

Figure 3 shows JSON object parameter in the Body section.


Figure 3: POST method with JSON parameters received from request body

## RUNNING INSTRUCTIONS

1. In order to run the API, please download the repository from GitHub link.
2. After opening the project in Visual Studio 2019, please follow the steps below to create a database and insert initial records with InitialMigration.
   - **2.1.** Please go to Tools > NuGet Package Manager > Package Manager Console
   - **2.2.** Type update-database and hit Enter. You have the database tables now.
3. Hit F5 to run the API on Visual Studio.
4. In order to test the API, please install Postman from https://www.postman.com/downloads/
5. After launching Postman, go to New > Request and name the request. Click on Create Collection below and type Ubisoft, hit confirm button and Add Ubisoft button.
6. Once the new request window is open, please follow the steps below;
   - **6.1.** Select POST as the method.
   - **6.2.** Type https://localhost:44381/api/Feedback/1 to the URL area. (Please change 44381 with the port provided on your browser when you run the microservice.)
   - **6.3.** Go to Headers tab and enter Ubi-UserId key with a value (1,2 or 3) as seen below.

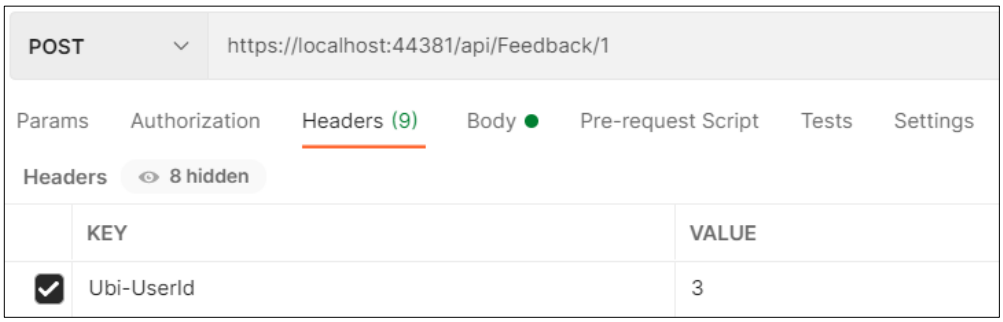


Figure 4: Header parameter

**6.4.** Go to Body tab, enter the data below (with your own choice) and select JSON as the type.

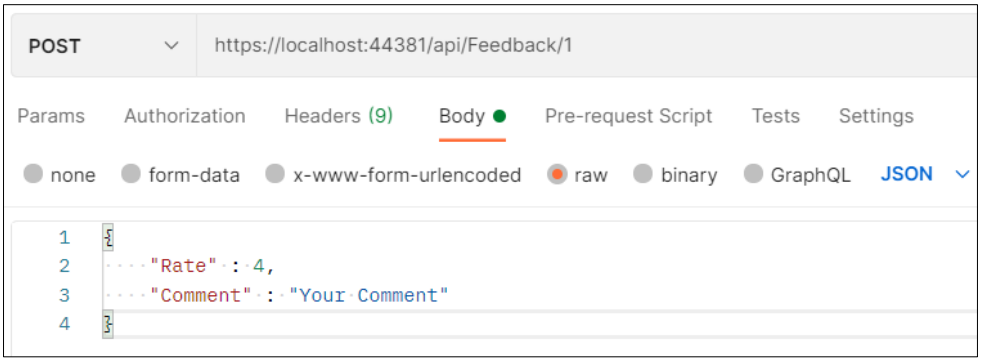


Figure 5: JSON Parameters

**6.5.** Hit Send button, once you see inserted record at the Response section, change the UserId and SessionId to insert different feedbacks. However, the API allows feedbacks from only recorded users and for only recorded sessions. So, initially the UserId can be 1, 2, 3 and SessionId can be 1, 2, 3.

**6.6.** In order to test GET method, please add another request on Postman by clicking on Ubisoft collection and Add Request link. This time, select GET method and type https://localhost:44381/api/Feedback?Filter=max in the URL. (Please change 44381 with the port provided on your browser when you run the API.) Hit the Send button, feedback records will be listed in the Response section in ascending order of Rates.

## RESULTS

The microservice returns 200 OK in case of success and custom errors shown in below, in case of invalid parameters usage.

| Parameters | Status | Message |
|---|---|---|
| | 200 OK | (Feedback record) |
| MissingParameter | 400 Bad Request | Parameter not found. |
| InvalidUser | 400 Bad Request | User not found. |
| InvalidSession | 400 Bad Request | Session not found. |
| FeedbackExist | 400 Bad Request | User feedback already exists. |
| InvalidRate | 400 Bad Request | Value for Rate must be between 1 and 5. |

Table 3: Feedback Microservice Results

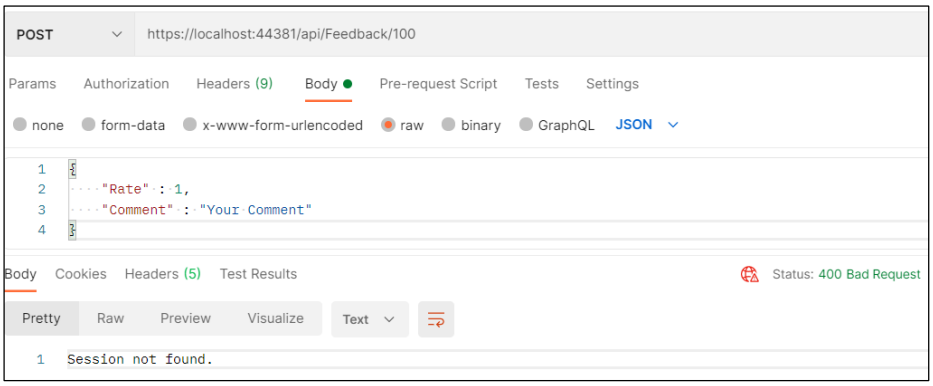


Figure 6 : A bad request example with a non existing session Id in the DB

## DATABASE

The database was created by using code first approach and Entity Framework migrations. As the concept suggests, the tables are automatically generated according to the entities.

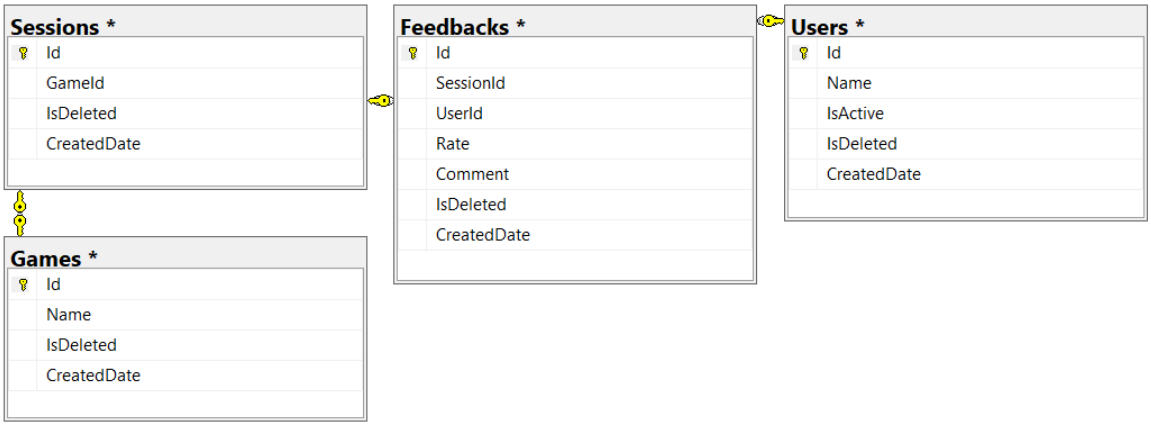


Figure 6: Database diagram for Feedback Microservice

The GET method filter the records by Rate. Therefore Rate column in the Feedbacks table has an index. SessionId and UserId also have indexes in order to provide a robust API.

That's it. If you have any questions, please do not hesitate to contact me.