Hotel Database – Programmering

Ide:

Du skal programmere en REST service til en tabel i Hotel databasen, samt programmere en REST Konsumer.

Baggrund:

A) Databasen

En række opgaver i faget SWD om at lave en Hotel Database.

Bl.a. med følgende 4 tabeller:

HOTEL: (Hotel_No, Name, Address)
ROOM: (Room_No, Hotel_No, Types, Price)
BOOKING: (Booking_Id, Hotel_No, Guest_No, Date_From, Date_To, Room_No)
GUEST: (Guest_No, Name, Address)

(Primary keys are underlined).

Det forventes du har lavet disse 4 tabeller i en lokal database, samt initialiseret dem med nogle værdier, samt kender til SQL sætninger som SELECT, INSERT, UPDATE og DELETE.

Se evt. https://www.w3schools.com/sql/

B) REST servicen

Note om REST Service http://pele-easj.dk/2020f-swc2/materiale/NetHttpNote.pdf

Sammenhæng mellem SQL og HTTP https://www.restapitutorial.com/lessons/httpmethods.html

Navngivning af resurser i API https://www.restapitutorial.com/lessons/restfulresourcenaming.html

C) C# Database tilgang

Ikke fuldstændig men nogenlunde beskrivelse af design og implementering en REST servise
https://www.restapitutorial.com/lessons/restfulresourcenaming.html

Konsumer af en REST service
https://blog.jayway.com/2012/03/13/httpclient-makes-get-and-post-very-simple/

C# Referencer:

- SQL Connection: https://docs.microsoft.com/en-us/dotnet/api/system.data.sqlclient.sqlconnection?view=netframework-4.8
- SQL Command: https://docs.microsoft.com/en-us/dotnet/api/system.data.sqlclient.sqlcommand?view=netframework-4.8

Aflevering senest Mandag d. 2 marts kl 16 i Wiseflow - Husk opgaven er individuel

Opgaven:

NU er det tid til at lave en egentlig REST-Service. Til det skal du have model-klasser, en controller samt selve REST-servicen.

Trin 1 Model klasser

Du kommer til at arbejde med modelklasserne både på REST-Provider siden, såvel som på REST-konsumer siden, derfor skal du lave dem i et 'Library' (dll-fil), der kan deles.

Du skal lave et projekt i VS2019 **'Class Library (.Net Standard)' – version 2.0**.

Du skal implementere de 4 model klasser:

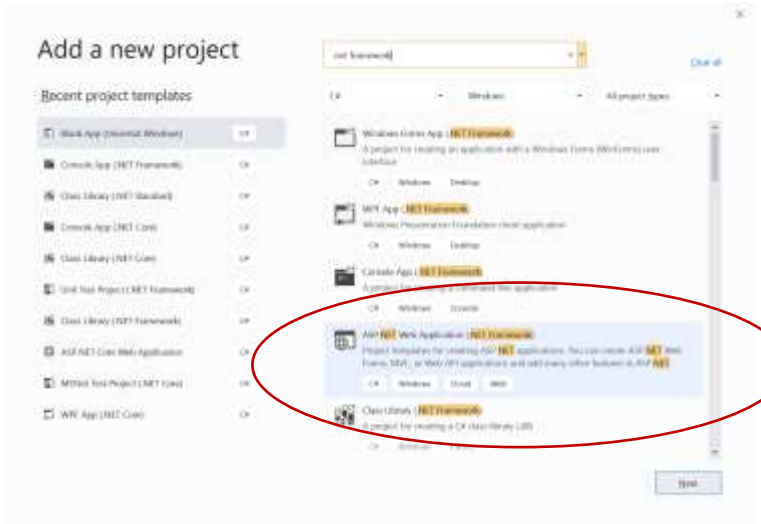
- `Hotel` `properties: HotelNr, Navn, Adresse`
- `Room` `properties: RoomNr, RoomType, Pris`
- `Guest` `properties: GuestNr, Navn, Adresse`
- `Booking` `properties: BookingId, HotelNr, Room, Guest, FraDato, TilDato`

Husk at alle klassene skal være public, samt de skal have en **default konstruktør**.
Du må gerne tilføje (overload) endnu en konstruktør til initialisering. Typerne skal følge typerne i databasen.

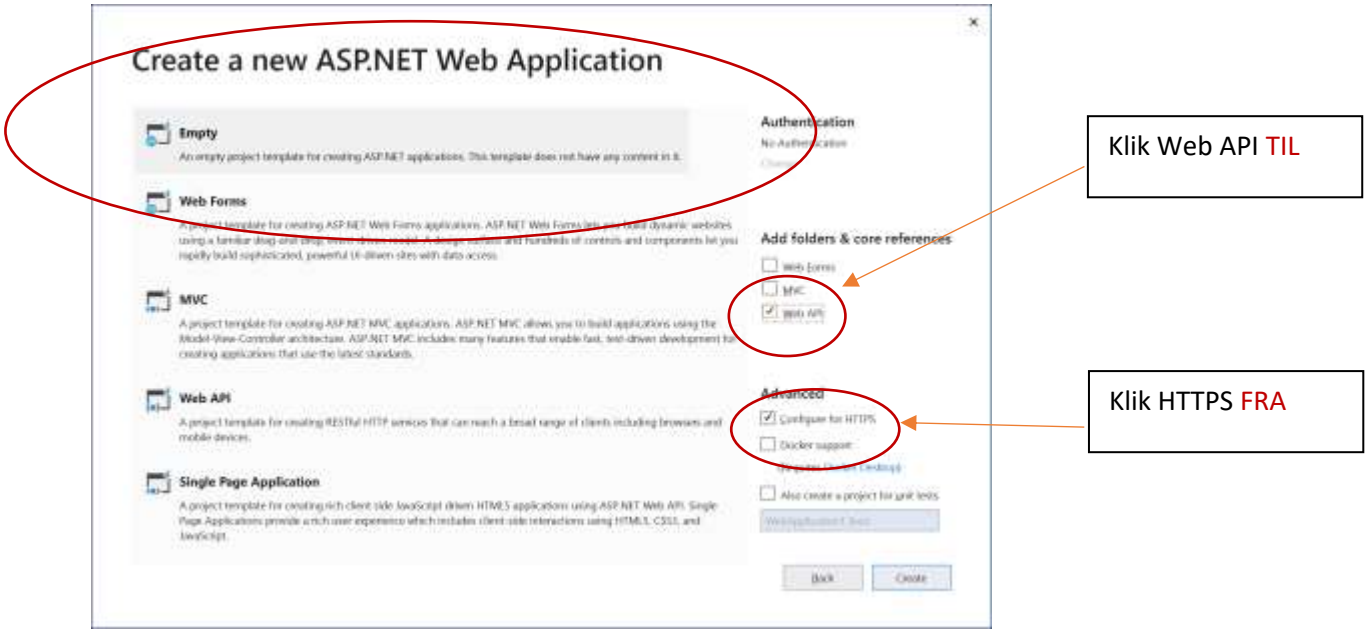
Trin 2 Opret REST Servicen

Inden du skal lave selve REST-servicen dvs. en controller klasse, så skal du isolere kommunikationen med databasen i en controller til tabellen 'Guest'.

Du skal lave et projekt i VS2019 under web **'ASP.NET Web Application (.NET Framework)'**.



Giv det et navn og næste bringer dig til dette valg:



☺ Nu har du oprettet dit projekt til at tilbyde (provide) en REST-Service.

OBS! INDEN du fortsætter: Du skal lige tilføje dit Library til dette projekt. (højre klik på references og tilføj library'et fra trin1.

Trin 3 Opret hjælpe klasser til håndtering af data til og fra databasen

Du skal i projektet lave en mappe 'DBUtil' til de hjælpe controllere.

Dernæst skal du oprette en klasse ManageGuest i mappen 'DBUtil'.

Klassen skal have 5 metoder (mindst):

- **`List<Guest> GetAllGuest()`**
  `Henter alle gæster fra databasen, listen vil være tom hvis der ikke findes nogle gæster`

- **`Guest GetGuestFromId(int guestNr)`**
  `Henter en specifik gæst fra database eller null hvis gæsten ikke findes`

- **`bool CreateGuest(Guest guest)`**
  `Indsætter en ny gæst i databasen. Returnere sand hvis gæsten er indsat ellers falsk, evt. fordi gæsten allerede findes`

- **`bool UpdateGuest(Guest guest, int guestNr)`**
  `Opdaterer en gæst i databasen. Returnere sand hvis gæsten er opdateret ellers falsk, evt. fordi gæsten ikke findes`

- **`Guest DeleteGuest(int guestNr)`**
  `Sletter en gæst fra databasen. Returnere den gæst der er slettet fra databasen, returnere null hvis gæsten ikke findes`

Den skabelon som alle 5 metoder er opbygget af er som følger:

```
using (SqlConnection connection = new SqlConnection(connectionString))
{
    SqlCommand command = new SqlCommand(queryString, connection);
    command.Connection.Open();

    // for INSERT, UPDATE, DELETE
    int rows = command.ExecuteNonQuery();
    // eller for SELECT
    SqlDataReader reader = command.ExecuteReader ();

    … mere kode f.eks. læs alle rækker fra database tabellen
}
```

ConnectionString findes under properties for databasen.

Trin 4 Opret Controller til at tilbyde REST API til Guest

Nu kommer du til selve REST-servicen.

Du skal i din solution finde mappen controllers, her skal du added (dvs. højre klik og add) en ny controller.

Du skal vælge **'Web API 2 controller with read/write actions**' - kald din controller '**Guests**Controller'.

Nu skal du tilpasse din REST-Service controller, den har de samme 5 metoder som ManageGuest.

'public IEnumerable<string> Get()' skal ændres til public **'IEnumerable<Guest> Get()**' og du skal lave et object af din ManageGuest klasse og kalde **GetAllGuest**

'public string Get(int id)' skal ændres til '**public Guest Get(int id)**' og på objectet af ManageGuest-klassen kalde **GetGuestFromId**

'public void Post([FromBody]string value)' ændres til **'public bool Post([FromBody]Guest value)**' og benyt metoden **CreateGuest**

'public void Put(int id, [FromBody]string value)' ændres til '**public bool Put(int id, [FromBody]Guest value)**' og benyt metoden **UpdateGuest**

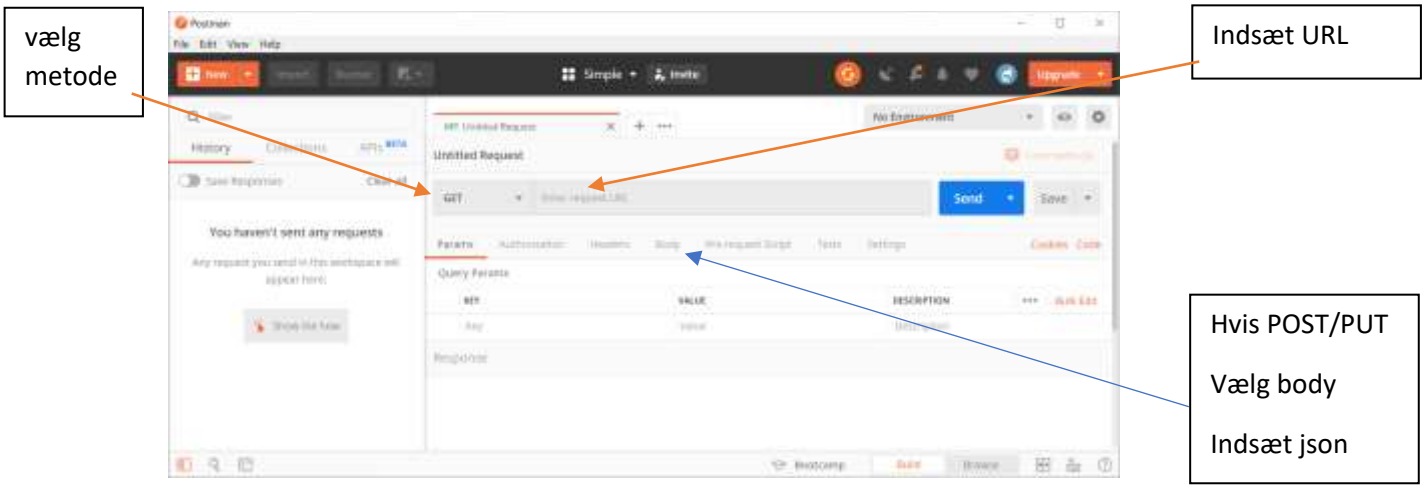
'public void Delete(int id)' ændres til '**public Guest Delete(int id)'** og benyt metoden **DeleteGuest**

Nu er din REST service klar og du kan køre din REST-Service.

Trin 5 Prøv din REST-Service

Du skal nu starte fx. Postman, se semester plan for download-url.

I Postman kan du opbygge og sende REST-Request til din REST-service.



Efter GET skriv din URL: fx. 'http://localhost:27481/API/Guests'. Du vil sikkert have et andet portnummer så erstat 27481 med DIT portnummer.

Prøv følgende REST-kald:

1) Get http://localhost:27481/API/Guests, se svaret

2) Get http://localhost:27481/API/Guests/4, se svaret (prøv med 444, som ikke findes)

3) Post http://localhost:27481/API/Guests -- OBS! Header Field indsæt 'Content-Type: application/json' + Boby noget der minder om '{"GuestNr":301,"Navn":"EtGodtNavn","Adresse":"EnFinVej"}'

4) Put http://localhost:27481/API/Guests/301 + Body noget der minder om '{"GuestNr":301,"Navn":"EtAndetNavn","Adresse":"EnNyVej"}'

5) Delete http://localhost:27481/API/Guests/301

Fint nu har du en REST service kørende for Guest

Trin 6 Gør det samme for tabellen Hotel i databasen

Du skal gøre trin 3 til trin 5 men for tabellen 'Hotel' .

Trin 7 Brug din REST service fra en Konsol Applikation

Du skal lave en ny 'solution' med et nyt projekt (alternativt lave nyt projekt inde i din solution fra ovenstående trin). Projektet skal være et '**Console app (.Net Framework)**' og giv (evt. solution) og projektet et navn.

HUSK at tilføje dit library fra trin 1 til projektet, samt NuGet pakken Newtonsoft (til Json)

Lav en klasse RestWorker med en Start()-metode. I Main-metoden lav et objekt af RestWorker samt kald metoden Start. Husk desuden at lave en Console.ReadLine(); (eller ReadKey).

I RestWorker klassen lav fem metoder:

- HentAlle
- HentEn
- Opret
- Slet
- Opdater

Når du skal hente (GET) kan du benytte følgende skabelon, hvor XXX er modelklassen og URI er stien til REST-Servicen:

```
public async Task<IList<XXX>> GetAllXXXAsync()
{
   using (HttpClient client = new HttpClient())
   {
      string content = await client.GetStringAsync(URI);
      IList<Item> cList = JsonConvert.DeserializeObject<IList<XXX>>(content);
      return cList;
   }
}
```

Når du skal sende data med dit kald (POST, PUT) kan du benytte følgende skabelon:

```
public async Task<bool> CreateXXXAsync(XXX newItem)
{
   using (HttpClient client = new HttpClient())
   {
      String jsonStr = JsonConvert.SerializeObject(newItem);
      StringContent content =
                  new StringContent(jsonStr, Encoding.UTF8, "application/json");

      HttpResponseMessage resultMessage =
                                    await client.PostAsync(GuestUri, content);

      if (resultMessage.IsSuccessStatusCode)
      {
         string jsonStr = await resultMessage.Content.ReadAsStringAsync();
         bool res = JsonConvert.DeserializeObject<bool>(jsonStr);
         return res;
      }
   }
   return false;
}
```

Selve opgaven, du skal lave en simple Konsol applikation, der kalder alle fem typer REST-kald på klassen 'hotel' og viser dem med Console.WriteLines.