Opdracht Web 4 – bouwen van een REST service

Er wordt gevraagd om een REST service te bouwen voor het beheren van reservaties voor een samenwerking van verschillende restaurants. Enerzijds hebben we *gebruikers* die zich moeten kunnen registreren, een reservatie aanmaken en ook op zoek moeten kunnen gaan naar een restaurant. Anderzijds is er ook een *beheerder* die restaurants kan toevoegen, verwijderen en updaten, maar die ook een statistiek kan geven aangaande de reservaties van een restaurant.

Wat moet de gebruiker kunnen ?

- De gebruiker moet zich kunnen registreren en moet daarvoor de volgende info meegeven :
  - Naam (mag niet leeg zijn)
  - Email (basis check uitvoeren)
  - Telefoonnummer (controleer vorm)
  - Locatie (zie verder)
  - Als de gebruiker zich heeft geregistreerd dan ontvangt hij een klantnummer (numeriek en groter dan 0)
- De gebruiker moet ook in staat zijn om zijn gegevens aan te passen.
- De gebruiker moet zich ook kunnen uitschrijven uit het systeem.
- Een gebruiker moet ook in staat zijn om een restaurant op te zoeken. Daarbij moet hij kunnen zoeken op locatie (postcode) en/of keuken (vb italiaans, turks, …).
- Een gebruiker moet ook in staat zijn om een overzicht te krijgen van de restaurants die nog een tafel vrij hebben voor een bepaalde datum (extra : in combinatie met locatie en keuken). Je geeft hierbij ook op hoeveel plaatsen er nodig zijn.
- De gebruiker moet in staat zijn om een reservatie te maken. Een reservatie omvat :
  - Reservatienummer (numeriek en groter dan 0)
  - Restaurantinfo (alle info over het restaurant)
  - Contactpersoon (= diegene die de reservatie heeft gemaakt)
  - Contactgegevens (email/telefoonnummer)
  - Aantal plaatsen
  - Datum
  - Uur
  - Tafelnummer
- De gebruiker moet ook in staat zijn om een reservatie aan te passen (enkel datum/uur/aantal plaatsen).
- De gebruiker moet ook in staat zijn om een reservatie te annuleren (enkel reservaties die nog moeten komen natuurlijk)
- De gebruiker moet ook in staat zijn om zijn reservaties op te zoeken (vb met een begen en/of einddatum)

Wat moet de beheerder kunnen ?

- De beheerder moet een restaurant kunnen registreren. Voor een restaurant moet de volgende info worden opgeslagen :
  - Naam (mag niet leeg zijn)

    - Locatie (zie verder)
    - Keuken (vb italiaans, turks, …)
    - Contactgegevens (telefoon en email)
- De beheerder moet een restaurant kunnen verwijderen.
- De beheerder moet de informatie over het restaurant kunnen aanpassen.
    - Naam, locatie, keuken, contactgegevens
    - Tafels moeten kunnen worden toegevoegd, verwijderd of aangepast
- De beheerder moet voor een restaurant een overzicht kunnen geven van de reservaties (en dit zowel voor een specifieke dag als voor een periode)

Een locatie omvat :

- Postcode (4 cijfers)
- Gemeentenaam (verplicht en niet leeg)
- Straatnaam (optioneel)
- Huisnummerlabel (optioneel – bevat eventueel ook busnr ….)

Opmerkingen :

- Wanneer een gebruiker zich uitschrijft, moeten de reservaties van die gebruiker nog wel steeds opvraagbaar zijn.
- Wanneer een restaurant wordt verwijderd dan moeten alle gegevens wel behouden blijven.

Business rules :

- Voor een reservatie voorzien we steeds dat de tafel dan bezet is voor anderhalf uur (1.5h).
- Het reservatietijdstip kan enkel een exact uur zijn of een half uur (dus bijvoorbeeld om 19h, 19h30, 20h ,…)
- De applicatie kiest de meest geschikte vrije tafel (een tafel van 2 als het aantal personen 2 is). Als er geen tafel van 2 meer vrij is voor een aanvraag voor 2 personen, dan pas wordt een tafel van 3 toegewezen …)

Te gebruiken technologie :

- Maak gebruik van Visual Studio voor de ontwikkeling en gebruik daarbij een blank solution voor al je projecten in onder te brengen.
- Maak gebruik van een ASP.NET Core Web API project voor de REST service (https is niet nodig, maak wel gebruik van Swagger)
- Je moet geen reking houden met logins (voorzie dus enkel een aparte API voor de gebruiker als voor de beheerder).
- Maak gebruik van het 3-lagen model.
- De databank is SQL server (indien dat een probleem zou opleveren dan moet je dat persoonlijk komen melden en een alternatief voorstellen)
- Wens je gebruik te maken van bepaalde NuGet packages dan moet je dat ook eerst vragen (standaardpaketten zoals Newtonsoft, SQL Client, Entity Framework, … kan je natuurlijk vrij

gebruiken – alle paketten die we in de les hebben gezien (web 4 + specialist) zijn zeker toegestaan).

- Voor de communicatie met de databank mag je kiezen (Entity Framework of ADO.NET)
- Unit testen moeten enkel voorzien worden voor de REST-service (niet business- of datalaag)
- Zorg er ook voor dat alle request worden gelogd in een logbestand (wel onderscheid maken voor de beheerder en de gebruiker).

Wat moet je opleveren ?

- Het ontwerp van je service (kan eenvoudig door een screenshot(s) van Swagger)
- Een architectuuroverzicht, een schets met de verschillende lagen, de relatie tussen deze lagen en de klassen die in deze lagen zitten (met hun relaties). Je hoeft van de klassen niet al de properties te tonen, want dat zal het overzicht niet ten goede komen.
- Je moet in staat zijn om een demo te geven (zorg er dus voor dat je reeds gegevens in de databank hebt en dat je weet welke gegevens daar in zitten zodat de demo vlot kan verlopen)

Veel succes !

Tom