Web API – Projektuppgift

BED-20 projektuppgift våren 2021

Beskrivning av uppgiften

Ett API till en webbapplikation

Du har fått uppdraget att implementera webbtjänster för fastighetsannonser med hjälp av ASP.NET 5. En del ändpunkter är publikt åtkomliga, medan andra kräver inloggning. **Användare registrerar sig med användarnamn, e-post och lösenord. Sedan kan de begära en nyckel (token) för autentisering och åtkomst.** Användare kan skapa annonser för fastigheter som de vill sälja eller hyra ut. Annonser kan läsas och kommenteras. Dessutom kan varje användare få ett betyg från 1 till 5 av andra besökare.

Innan du lämnar in uppgiften glöm inte att göra Build -> Clean Solution.

Uppgifter för data

Uppgift 1: Datalager

Skapa en databas som kan användas med webbservern. Du skall använda MS SQL server och Entity Framework. Välj själva om ni vill använda code first eller database first som angreppssätt.

Valideringar:

- **Fastighetstitel** består av 5 till 50 tecken
- **Fastighetsbeskrivning** består av 10 till 1000 tecken
- **Fastighetstyp** består av en av följande: *Lägenhet, Hus, Kontor* eller *Lagerlokal*
- **Byggnadsår** är inte före år 1600
- **Fastighetstitel, -beskrivning, adress** och **kontakt** är **obligatoriska.**
- Fastigheten kan vara till **endast försäljning, endast för uthyrning** eller **båda**. Dock måste försäljning eller uthyrning anges.
- **Kommentarer**

Uppgift 2: Repositories + Services

Skapa de nödvändiga repositories med hjälp av *Repository pattern* för att abstrahera användningen av datalagret. Det är inte nödvändigt att använda Services, utan det går lika bra att skapa *Unit of Work*.

Uppgifter för Webb-API

Alla tjänster i Webb-API beskrivs I dokumentet "API beskrivning".

Uppgift 3: Tjänster för inloggning och registrering av användare

Skapa REST-tjänster för inloggning och registrering enligt följande format:

HTTP Metod	Url endpoint	Beskrivning

POST	/api/account/register	Skapar ett nytt användarkonto med **användarnamn, e**-post och **lösenord**
POST	/token	Loggar in en befintlig användare

Uppgift 4: Tjänster för hantering av fastigheter

Skapa funktioner för att **lista/skapa/visa detaljer för** fastigheter.

Icke-inloggade användare kan se alla fastigheter i systemet, men kan inte se detaljerna för dem.

Inloggade användare kan:

- Se detaljer för fastigheter, så som kommentarer och kontaktuppgifter
- **Skapa nya fastighter/annonser**

HTTP Metod	Url endpoint	Beskrivning
GET	/api/RealEstates	**Kräver inte inloggning** Hämtar grundläggande information om alla fastigheter. Returnerar bara topp-tio fastigheter, efter sortering på datum och tid när de skapades, i **fallande** ordning
GET	/api/RealEstates*?skip=S&take=T*	**Kräver inte inloggning** **Samma som /api/RealEstates**, men hoppar över S annonser och hämtar bara fram T annonser. S och T behöver inte båda anges. Standardvärdet för överhoppning är 0, standardvärdet för hämtning är 10. T kan inte vara större än 100
GET	/api/RealEstates/*ID*	Hämtar information om fastigheten med det angivna id-numret. **Icke-inloggade användare kan inte se kommentarer eller kontaktinformation. Se API-beskrivningen för mer detaljer**
POST	/api/RealEstates	**Kräver inloggning** Skapar en ny fastighetsannons med all nödvändig information för att spara den. **Se API-beskrivningen för mer detaljer**

Uppgift 5: Kommentarsfunktion

Implementera en kommentarsfunktion. Alla dessa funktioner kräver att användaren är inloggad.

HTTP Metod	Url endpoint	Beskrivning
GET	/api/Comments/*ID*	**Kräver inloggning**. Hämtar alla kommentarer för en fastighet med ett givet **ID** Returnerar bara topp-tio kommentarer, efter sortering på datum och tid när de skapades, i **fallande** ordning
GET	/api/Comments/*ID?skip=S&take=T*	**Kräver inloggning**.

		Samma som /api/Comments, men hoppar över S annonser och hämtar bara fram T annonser. S och T behöver inte båda anges. Standardvärdet för överhoppning är 0, standardvärdet för hämtning är 10. T kan inte vara större än 100
GET	/api/Comments/ByUser/*USERNAME*	**Kräver inloggning**. Hämtar alla kommentarer skrivna av **användren med angivet användarnamn** Returnerar bara topp-tio kommentarer, efter sortering på datum och tid när de skapades, i **fallande** ordning
GET	/api/Comments/ByUser/*USERNAME?skip=S&take=T*	**Kräver inloggning**. **Samma som /api/Comments/ByUser/*USERNAME***, men hoppar över S annonser och hämtar bara fram T annonser. S och T behöver inte båda anges. Standardvärdet för överhoppning är 0, standardvärdet för hämtning är 10. T kan inte vara större än 100
POST	/api/Comments	**Kräver inloggning**. Skapar en ny kommentar med all nödvändig information för att spara den. **Se API-beskrivningen för mer detaljer**

Uppgift 6: Tjänster för användare

Skapa funktioner för att hämta information om användare samt att betygsätta dem

HTTP Metod	Url endpoint	Beskrivning
GET	/api/Users/*USERNAME*	**Kräver EJ inloggning**. Hämtar användarens användarnamn, antal annonser, antal kommentarer och genomsnittligt betyg
POST	/api/Users/Rate	**Kräver inloggning**. Betygsätt en användare från 1 till 5

Uppgifter för hög kodkvalitet och publicering

Uppgift 7: Hög kodkvalitet

- Lägg in all nödvändig validering av data
- I händelse av fel, returnera adekvata HTTP statuskoder och annan felinformation
- Skriv kvalitativ abstrakt kod som är lätt att underhålla och bygga vidare på.

Uppgift 8: Publicera till molnet

Systemet skall publiceras till, och köras från, något lämpligt moln

- I första hand Microsoft Azure

- I andra hand från något annat

Uppgift 9: Extra uppgift: Integrera befintlig front-end applikation

Ni kommer att få utdelat tre olika frontend-applikationer. Dessa är utvecklade av studerande på en annan YH-utbildning. Ni får fritt använda de delar ni vill – de kan vara bra att testa och presentera sin lösning med.

Bedömning

För godkänt på projektet krävs:

- Kraven är korrekt och fullständigt uppfyllda
- God teknisk design och lämpliga tekniker
- Kod av hög kvalitet – korrekt, läsbar, underhållbar
- Prestanda – effektiv kod
- Det skall gå lätt att
    - Starta applikationen på en annan (dvs lärarens) dator
    - Sätta upp databasen med testdata

För väl godkänt krävs dessutom

- API är grundligt dokumenterat, dokumentationen visas med hjälp av Swagger.
- En fullständig uppsättning testfall finns definierade i Postman och medföljer inlämningen
- En detaljerad projektloggbok. Skriv varje vecka vad du har jobbat med. Vad som har gått bra, vad som har krånglat, hur du kom förbi problemen.
- Inlämning görs i tid.

Övrigt

- Det är fullt tillåtet att lämna in via Github. Dock är kravet att det skall gå lätt att få igång systemet på lärarens dator.
- Det är också tillåtet att ha en mer eller mindre omfattande webb-frontend till systemet, för underhåll av tabellerna i databasen.

Projektmetodik

Detta projektet bygger på gruppvisa inlämningar.

Vidare är det rekommenderat att ni arbetar agilt. Använd till exempel Trello för att hålla ordning på arbetet. Detta gäller även om ni väljer att genomföra projektet enskilt.

Inledande redovisning

Vi har en avstämning efter cirka halva projekttiden. Detta för att ni skall kunna redogöra för hur långt ni har kommit, vilka problem ni eventuellt har och vilken assistans ni behöver.

Betrakta avstämningen som en Sprint Review: berätta på 10–15 minuter var ni är och vart ni är på väg. Redovisningen blir intern inom klassen, så räkna med att alla åhörarna är väl insatta i projektet.

Slutredovisning

Projektets slutredovisning sker den sista dagen på kursen, dvs fredagen 2021-06-18.

- Tid till förfogande är 16–21 minuter, med fem minuters efterdiskussion
- Visa ert Trello/Jira board och berätta vad ni eventuellt inte hunnit med. Varför valde ni att prioritera just de punkterna lägre?
- Om ni använder en webbklient så demonstrera hur en användare kan använda systemet
- Det går bra att visa systemets funktionalitet genom en konsolapplikation eller Postman, om ni hellre vill det. *Detta är det tråkiga med att jobba med Back-end: vi får sällan något sexigt eller tjusigt att visa upp.*
- Berätta om de två största tekniska utmaningarna ni har haft under hela projektet, och hur ni löste dem
- Räkna med att det blir en del frågor
- Ta emot applåderna!