HI1031, Distribuerade affärssystem Laboration 2, Community i ASP.NET

Inledning

Målet med denna laboration är att lära sig bygga ett distribuerat system i ASP.NET, med språket C# samt ramverken MVC 5 och Entity Framework.

Uppgiften löses i grupper om 2 personer; båda personerna ska kunna svara för alla delar av lösningen. Redovisningen sker muntligt tillsammans med en demonstration. Vid redovisningen ska klassdiagram som beskriver lösningen presenteras.

Krav på lösningen

- Lösningen ska vara skiktad, med uppdelning i datalager (m.h.a. Entity Framework), logiklager och presentationslager (t.ex. med hjälp av s.k. view models).
- Lösningen ska ha en genomtänkt uppdelning i ”name spaces” och klasser
- Filer, klasser, kontroller och metoder ska ha en genomtänkt namngivning
- Koden ska vara väl dokumenterad

Lösningen behöver inte driftsättas, d.v.s. ni kan demonstrera lösningen via Visual Studio (och flera separata browser-instanser).

Uppgiften

Er uppgift är att skapa en förenklad version av ett community. Användare ska kunna logga in till systemet och skicka meddelanden till andra användare och ev. till grupper av användare. Ett meddelande har alltid en avsändare samt en mottagare och kan vara oläst eller läst (eller borttaget). En användare ska kunna välja att se, och läsa, meddelanden som skickats till denne från andra användare. Användaren måste naturligtvis vara inloggad både för att skicka och för läsa meddelanden.
Det ska också gå att skapa grupper av användare där meddelanden går till, och ses av, alla gruppens användare (högre betyg).

Grundkrav, betyg 3

Användare ska kunna registrera sig och logga in (enklast är att använda den färdiga funktionalitet som finns i mallen för MVC-projekt i Visual Studio).

”Indexsidan”, som liksom alla sidor kräver inloggning, visar uppgifter som användarnamn, när användaren senast loggade in, antalet inloggningar senaste månaden och antalet olästa meddelanden. På sidan finns också en länk till en sida för läsning av meddelanden, ”lässidan”.

Alla sidor ska ha länkar till ”indexsidan”, ”lässidan” och ”skrivsidan” (lämpligen som en menyrad via en del-vy).

Nedanstående gäller för ”lässidan”.

1. En lista med namn på avsändare av meddelanden (som ännu inte tagits bort) visas.
2. När en avsändare väljs visas en lista med rubrik och tidsstämpel på meddelanden från just denna användare.
3. När en rad i denna lista väljs visas motsvarande hela meddelande. När ett meddelande på detta sätt öppnas ska det automatiskt markeras som läst. Man ska också kunna ta bort meddelanden; borttagna meddelande visas inte mer på denna sida. Man kan välja att läsa godtyckligt antal meddelanden.
4. Längst ner på ”lässidan” visas totala antalet meddelanden samt hur många meddelanden som lästs respektive tagits bort.

På ”skrivsidan” finns en lista (lämpligen en SelectList) med alla adressater i systemet samt en ett textfält och textruta för att skapa rubriken och själva meddelandet. När meddelandet är komponerat skall det kunna skickas. Det får antas att meddelandet bara har en adressat. När meddelande sänts svarar systemet med en bekräftelse enligt nedanstående samt tömmer textarean.

```
Meddelande nummer 1142 avsänt till Anders Lindström, 14:53 2014-10-24
```

Några krav på implementationen:

1. Object-Relational Mapping (ORM) med Entity Framework ska användas för modellklasserna.
2. Datat i modellklasserna ska, där det är lämpligt, märkas med attribut för t.ex. icke null (Required), maxlängd på textsträngar, namn som visas i vyn (DisplayName), format för t.ex. datum och tid et c.
3. Lösningen ska implementeras med hjälp av ViewModels som representerar det data som ska presenteras i vyerna. Vyerna ska alltså inte känna till modellklasserna i datalagret (EF-klasserna); däremot får controller-klasserna känna till modellklasserna från datalagret.
4. Om ni använder Visual Studios mallar för att skapa t.ex. controllers ska alla irrelevanta metoder tas bort från den genererade koden. Endast länkar som leder till existerande sidor får finnas i cshtml-sidorna.

Betyg 4

Lösningen för betyg 4 måste uppfylla alla kraven för betyg 3 samt det nedanstående.

1. Användare skall kunna skapa grupper som andra användare kan anmäla sig till. Meddelanden som skickas till en grupp ska kunna läsas av alla gruppens medlemmar.
2. Det skall också vara möjligt att adressera meddelanden till flera personer samtidigt även om dessa inte ingår i en grupp.

Betyg 5

Lösningen för betyg 5 måste uppfylla alla kraven för betyg 3 och 4 samt det nedanstående.

1. Effektivisera operationerna mot databasen så att webbservern bara hämtar relevant data från databasservern samt använder eager/lazy/explicit-loading på ett så effektivt sätt som möjligt. Notera att vad som är bästa lösningen här varierar från fall till fall – ni måste argumentera för era val i varje enskilt fall (varje enskild kontrollermetod).
2. Använd AJAX för att, där så är lämpligt, hämta data bara när användaren väljer att visa det.