Institutionen för
Tillämpad fysik och elektronik
Stefan Berglund och
Per Kvarnbrink

2019-03-20

Laboration:
Samverkande system

Målsättning

Syftet med denna laboration är att ge grundläggande kunskap och erfarenhet i att utveckla flerskiktade applikationer baserade på ASP.NET Core MVC som är byggda på ett sätt så att de kan samverka med varandra. Ytterligare ett syfte är att ge den studerande ett tillfälle att tolka och följa en lite mera bristfälligt skriven teknisk kravspecifikation.

Teori

Kursens föreläsningsfilmer och länkar.

Förberedelseuppgifter

Det är en stor fördel om du som student är helt klar med föregående laboration.

Material

Utvecklingsmiljön för denna laboration är Visual Studio och den version av SQL Server som följer med denna. Önskar ni genomföra laborationen i alternativ utvecklingsmiljö så förankrar ni detta hos handledande lärare.

Genomförande

Uppgiften och dess specifikation är omfattande så ta er tid att läsa igenom och försök förstå uppgiften innan ni sparkar igång med själva utvecklandet. En tillfredställande lösning på samtliga moment är ett måste för att bli godkänt på laborationen. De uppgifter som ingår i laborationen kan ni genomföra individuellt eller i grupper om två studenter. Om ni laborerat tillsammans med någon vill vi att detta framgår i er rapport.

I uppgiften erhåller ni en teknisk kravspecifikation som medvetet är vagt formulerad, det finns två bakomliggande tankar till detta. Den första är att det är vanligare än man tror att en beställare av ett system saknar erforderlig beställarkompetens (det är då vanligt att man som utvecklare får kliva in och stötta beställaren genom att förtydliga specifikationen), den andra tanken är att inte vingklippa er utan att ge er ett tillfälle att själva reflektera över vilken lösning som blir den bästa.

För att erhålla plus på denna laboration ska två av fyra fördjupningsuppgifter vara lösta.

Redovisning

Redovisning sker genom en individuellt skriven laborationsrapport och genom uppvisande av en väl fungerande applikation. Rapporten ska i möjligaste mån följa den rapportmall som återfinns på kurshemsidan och den muntliga redovisningen kan ske ute på lab, via Skype eller i form av inspelade Jing-filmer.

Siktar ni på högre betyg för denna laboration så ska detta framgå klart och tydligt i er rapport. Vad ni gjort och hur.

Uppgift

Uppgiften går ut på att så långt det går uppfylla de krav som ställs i den tekniska kravspecifikationen. Ni ska skapa två system som ska arbeta tillsammans, system som efter komplettering skulle kunna gå i drift på en tidning. Det ena systemet ska hantera annonsering och annonsörer och det andra ska hantera prenumeranter.

OBSERVERA!
Vi pratar om två system med var sin databas. I Visual Studio skapar ni med andra två projekt, ett för vardera system.

Eventuella oklarheter i kravspecifikationen eller önskemål om att ändra eller frångå vissa krav tar ni upp med handledande lärare.

Kravspecifikationen återfinns i sin helhet sist i detta dokument.

Fördjupningsuppgifter

Som fördjupning vill vi att ni kompletterar er applikation med minst två av fyra uppgifter.

1) Utveckla ett windowsprogram som agerar gränssnitt med CRUD-funktionalitet mot prenumerantsystemet.

2) Modifiera ditt annonssystem så att det konsumerar ett Web API som utvecklats av någon kurskamrat. Vid redovisning vill vi se både ursprungligt och modifierat annonssystem.

3) Utveckla en funktion där man kan exportera prenumeranter till en fil med prenumeranter lagrade i XML-format. Utveckla även en importfunktion där man kan läsa in nya prenumeranter genom att importera en XML-fil.

4) Lägg till funktionalitet i annonssystemet som hämtar växlingskurser via valfritt API och implementera så att användare kan välja i vilken valuta priset ska visas.

Kravspecifikation

Vi vill att ni ut utvecklar ett prenumerantsystem som hanterar och lagrar information om tidningens prenumeranter. Vi vill också att ni utvecklar ett separat annonssystem där annonsörer själva kan lägga in sina annonser.

Det finns två typer av annonsörer:

1. Prenumeranter som ska få en fördel av att vara prenumeranter
2. Företagsannonsörer — Behöver ingen db

För att slippa dubbellagrad information ska annonssystemet hämta uppgifter från prenumerantsystemet. Om annonsören finns med i prenumerationssystemet så ska personuppgifter hämtas därifrån. Kommunikationen mellan annonssystemet och prenumerantssystemet sker via ett Web API.

Funktionalitet hos annonssystemet

Detta system ska ha ett webbgränssnitt där man kan lämna in annonser. Det är en sida som gör skillnad mellan prenumeranter och företag. En prenumerant ska bara behöva ange sitt prenumerationsnummer för att kunna lägga till en annons. Systemet hämtar då uppgifter som namn, telefonnummer, utdelningsadress, postnummer samt ort från prenumerationssystemet.

Rent praktiskt ska det gå till så här:

1. Man kryssar i en radioknapp för att markera att man är prenumerant.
2. Man skriver i sitt prenumerationsnummer.
3. Systemet konsumerar prenumerationssystemets Web API för att erhålla de uppgifter som behövs; Namn, Telefonnummer, Utdelningsadress, Postnummer samt Ort.
4. Hämtade uppgifter visas på webbsidan då de är hämtade.
5. Om någon uppgift är felaktig ska man kunna ändra denna.
6. Efter detta kan annonsen läggas in. Den innehåller då fem fält: id, varans pris, innehåll, rubrik och annonspris.
7. Annonspriset bestäms per automatik till 0 kr för prenumeranter och 40 kr för företag.

Om en företagare vill annonsera används inte Web API:et för att hämta information om företaget. Företagaren får istället fylla i alla uppgifter som behövs direkt i annonssystemet.

Då företagaren vill annonsera blir det enligt denna procedur:

1. Man kryssar i en radioknapp för att markera att det är en företagsannons.
2. Det dyker upp ett formulär där uppgifter om företaget ska matas in.
3. Det är följande uppgifter: namn, organisationsnummer, telefonnummer, utdelningsadress, postnummer samt ort plus fakturaadress som innehåller utdelningsadress, postnummer samt ort.

4. Efter detta kan annonsen läggas in. Den innehåller då fem fält: id, varans pris, innehåll, rubrik och annonspris.
5. Annonspriset bestäms per automatik till 0 kr för en prenumerant och 40 kr för ett företag.

För att visa annonserna gör du en sida som visar en lista med annonser. Det ska tydligt framgå om det är ett företag som annonserar.

Uppbyggnad och dokumentation av annonssystemet
System ska vara flerskiktat enligt MVC-struktur med en SQL Server-databas som datalager. Logiklagren bör bestå av ett dataaccesslager samt specialiserade klasser för annonser och annonsörer.

Systemet ska konsumera ett Web API som ska finns i prenumerantsystemet. Detta system bidrar med två saker. Först och främst vill man veta om det är en abonnent som vill annonsera men man vill även få uppgifter, som t.ex. adress om abonnenten. De som inte är abonnenter eller företag får inte annonsera.

All data som skickas mellan de olika lagren i systemet ska hanteras som objekt/modeller. Det kan vara ”rena” Entity Framework-objekt (EF-objekt), vy-modeller som utnyttjar EF eller modeller som har helt egenskriven kod som DAL, data-access-lager. Alla modeller ska dokumenteras med UML (eller annan typ av klassdiagram) samt en beskrivning som innehåller en funktionsbeskrivning, in- och ut-parametrar samt returvärden.

Vi är lite petiga då det gäller databasen. Databasen i detta system ska bestå av två tabeller, tbl_annonsorer och tbl_ads. tbl_ads ska innehålla annonserna. Vi använder engelska här för att det inte ska uppstå förväxlingar i namnet. De har en relation då en annonsör kan ha flera annonser. Vi vill att ni baserar er på en namngivningspolicy där samtliga tabellnamn börjar med tbl_. Tabellen med annonserna ska heta tbl_ads. De fält som ingår i denna tabell ska börja med ad_, som de två fösta bokstäverna i advertise. Det betyder att fältet rubrik ska heta ad_rubrik, fältet för innehåll, ad_innehall etc. Tecknen för å, ä och ö får inte förekomma. Alla namn ska skilja sig från varandra så att det aldrig uppstår risk för förväxlingar. Dokumentera databasen med ett ER-schema.

Om man genererar klasserna direkt med EF så blir det ganska dåliga namn på klasser, metoder och egenskaper. Kan man få det snyggt och läsligt på Svenska?

Uppbyggnad och dokumentation av prenumerantsystemet
Detta system ska som namnet antyder hantera tidningens prenumeranter. Det ska bestå av en SQL Server-databas med en tabell innehållande prenumerantens prenumerationsnummer, personnummer, förnamn, efternamn, utdelningsadress och postnummer.

För att spara värdefull tid hoppar vi över själva användargränssnittet i systemets grundutförande. Men systemet ska ha ett gränssnitt i form av ett Web API som svarar upp mot de behov som ställs i annonssystemet.

På samma sätt som för annonssystemet ska ni dela upp systemet i lager enligt MVC-struktur (men följaktligen då utan Views). Dokumentera på samma sätt som det andra systemet baserat på UML- och ER-scheman.

OBSERVERA!
Vi pratar fortfarande om två helt olika system med var sin databas.

Ni har säkert redan insett att det finns dubbellagrad information när man kombinerar systemen. Abonnentens namn, adress etc. finns både i annonssystemet och i abonnentsystemet. Byter abonnenten adress vet vi inte riktigt hur detta hanteras på bästa sätt. Här vill vi att ni utreder och implementerar det på ett sätt som ni tycker blir bäst.