Lexicon LMS

Projektet ni skall arbeta med under den avslutande modulen är en lärplattform, ett så kallat LMS[1] *(Learning Managment System)*, anpassat för Lexicons påbyggnadsutbildningar. Ett LMS förenklar och centraliserar kommunikationen mellan lärare, lärosäte och elev genom att samla schema, kursmaterial, övrig information, övningsuppgifter och inlämningar på ett och samma ställe.

Ni skall från grunden producera systemet med databas, back-end funktionalitet och ett genomtänkt front-end. Detta kallas ett ”full-stack projekt” och syftar till att visa upp er förståelse för samtliga delar av en webbapplikation och nutida system. Projektet ämnar att testa bredden av er förståelse och att ni har en grund att stå på oavsett framtida inriktning inom .NET.

Produktbeskrivning

Systemet vi skall bygga har som främsta uppgift och mål att enkelt tillgängliggöra kursmaterial och schema för elever. Det skall även fungera som en samlingsplats för inlämningsuppgifter. För att detta skall vara möjligt behövs vi även smidig funktionalitet för lärare att enkelt kunna administrera dessa klasser, elever, scheman och dokument. För om det inte är enkelt för läraren att använda verktyget, så kommer eleverna aldrig få chansen att använda det.

Det färdiga systemet är ämnat att framför allt täcka grundläggande funktionalitet, men på ett genomtänkt och genomarbetat sätt. *Less is more* är ofta sant när det gäller denna typ av applikationer som skall användas dagligen. Tyvärr, för att nå en så bred marknad som möjligt är de flesta LMS som finns tillgängliga idag enormt tunga och överbelamrade av all tänkbar funktionalitet som man sällan har användning för - detta skall ni ändra på! Less is more behöver inte nödvändigtvis syfta till ren funktionalitet, utan snarare om upplevd komplexitet. Det får gärna finnas djup funktionalitet, men användaren skall inte behöva 14 alternativ i varje val den gör.

Ramverk och tekniker

Applikationen skall ha en back-end byggd med C# .NET och MVC 5. Databasen skall byggas med Entity Framework enligt *code first*-metoden. Front-end skall använda Bootstrap 3. Dynamisk funktionalitet i front-end skall skrivas med javascript eller javascriptramverk så som jQuery och AngularJS.

[1] Mer information: https://sv.wikipedia.org/wiki/Lärplattform

Entiteter, relationer och attribut (grundform)

Nedan beskrivna entiteter och attribut är ett minimum, inte en absolut beskrivning. Framför allt attributen kommer behöva byggas ut när ni i närmare detalj planerar systemet.

Användare

Applikationen skall hantera användare i rollerna av elever och lärare, dessa skall alla ha inloggningar och konton i applikationen. Användarna bör sparas med minst ett namn och en e-postadress.

Kurs

Alla elever tillhör en *kurs* som i sin tur har ett kursnamn, en beskrivning och ett startdatum. Exempel på kursnamn: ".NET Höst 2015".

Modul

Varje *kurs* läser en eller flera *moduler*, dessa har modulnamn, en beskrivning, startdatum och slutdatum. Exempel på moduler: "Databasdesign", "AngularJS"

Aktiviteter

modulerna i sin tur har *aktiviteter*, dessa aktiviteter kan vara e-learningpass, föreläsningar, övningstillfällen eller annat. Aktiviteterna har en typ, ett namn, en start/sluttid och en beskrivning.

Dokument

Alla entiteter ovan kan hålla dokument; inlämningsuppgifter från eleverna, moduldokument, generella informationsdokument för kursen, moduldokument, föreläsningsunderlag eller övningsuppgifter kopplade till aktiviteterna. Dessa *dokumententiteter* bör ha ett namn, en beskrivning, en tidsstämpel för uppladdningstillfället samt information om vilken användare som laddat upp filen. Om dokumentet är en inlämningsuppgift skall den även hålla tidsinformation om deadline.

Use-cases (krav)

Dessa use-cases är inte heltäckande; beroende på implementation måste mer detaljerade fall tas fram för att täcka in all praktisk funktionalitet.

En icke inloggad besökare skall kunna:

- Logga in

En elev skall kunna:

- Se vilken kurs denne läser och vilka de andra kursdeltagarna är
- Se vilka moduler denne läser
- Se aktiviteterna för en specifik modul (modulschema).
- Se om en specifik modul eller aktivitet har några dokument kopplade till sig och i sådant fall ladda ned dessa.
- Se vilka övningsuppgifter denne har fått; om den redan är inlämnad, när den senast skall lämnas in och om den är försenad.
- Kunna ladda upp filer som övningsinlämningar.

En lärare skall kunna:

- Se alla kurser
- Se alla moduler som ingår i en kurs
- Se alla aktiviteter och dokument en modul innehåller
- Skapa och redigera användare (lärare och elever)
- Skapa och redigera kurser
- Skapa och redigera moduler
- Skapa och redigera aktiviteter
- Ladda upp dokument för kurser/moduler/aktiviteter
- Ta emot inlämningsuppgifter

Use-cases (önskvärda)

En icke inloggad besökare skall kunna:

- Begära nytt lösenord

En elev skall kunna:

- Ta emot feedback på inlämningsuppgifter
- Dela dokument med sin kurs eller modul

En lärare skall kunna:

- Ge feedback på inlämningsuppgifter

Front-end

Önskemålet är att front-end visuellt förhåller sig till Lexicons grafiska profil, detta med logga och färger som kan ses på http://www.lexicon.se. Detta skall inte göras in i minsta detalj, utan kan ses som en riktlinje.

Utöver dessa rent estetiska önskemål skall resterande front-endfokus riktas mot användarupplevelsen och att minska användarens kognitiva friktion.
Applikationen skall vara responsiv. Bonuspoäng för en välfungerande mobilversion.

Arbetssätt

Scrum

Projektet skall utföras i grupp med ett *scrum*-baserat arbetssätt. Vi kommer att arbeta i 7-dagars sprintar från onsdag till onsdag. En ny sprint startar varje onsdag eftermiddag med en *sprintplanering* där ni sätter upp en sprint-backlog, fördelar arbetet och uppdaterar er *task-board*.

Varje dag inleds med en *standup (daily scrum)* där ni kort, en och en, avhandlar 1. vad ni gjort sedan förra *standup*, 2. vad ni planerar att göra fram till nästa och 3. om det är något som blockerar planerat arbete. Ni håller mötet framför er task-board, och uppdaterar vart efter.

Under efterföljande onsdagsförmiddag avslutar ni sprinten med en sprintdemo och ett retrospektiv tillsammans med lärare och beställare.

Versionhantering

Projektet skall versionshanteras med git och GitHub.

Avstämningpunkter och leverabler

Under projektets gång förväntas ni redovisa vissa moment innan ni fortsätter. Detta för att undvika återvändsgränder och maximera er effektiva utvecklingstid.

- *Produkt-backlog* skall godkännas innan ni startar implementation.
- ER-Diagram skall godkännas innan ni startar en implementation.
- Wireframes för några nyckelvyer skall godkännas innan ni startar en implementation.
- *Sprint-backlogg* skall godkännas innan ni startar en ny sprint.
- Vid avslutad sprint skall alla leveransklara ändringar demonstreras vid sprintdemo.

Redovisning

Detaljerad information om redovisningsmomentet kommer senare, men kommer att innehålla use-cases, frågor om kodlösningar, filhantering, detaljlösningar, felhantering, arbetssätt etc.

Systemet skall demonstreras vid projektorn genom en *round-trip* baserat på några use-cases. Sedan redovisas utvalda delar av kodbasen, några frågor om implementation och arbetssätt besvaras. Detta följs troligen av varma applåder och glada utrop kommentarer och konstruktiv kritik.

Lycka till!