Labb 4

E-tjänster och webbprogrammering 7.5 hp VT-14

Introduktion

I denna laboration kommer vi att uttöka vårt kommentarsfält så att användare kan posta kommentarer som svar på andra kommentarer.

Detta betyder att varje kommentar bör ha en länk (eller liknande), som användaren kan klicka på för att ges möjligheten att besvara just den kommentaren.

Detta kommer även innebära att det kommer krävas en algoritm (i `PHP`), för att kunna sortera kommentaren så att de visas i en "rimlig" ordning för användaren. Hur ni väljer att sedan visuellt presentera detta väljer ni själva. Under denna länk finner ni en del inspiration :)

Föreslagen struktur

Denna laboration består av 3 faser. Dessa är inte separata uppgifter, utan föreslagna steg för att bygga ovan nämnd applikation. Som alltså är en vidarutveckling av den applikation vi byggt i labb 1 och 2. Denna laboration kräver ingen inlämning. Men för att få en uppfattning om hur ett färdigt projekt efter denna labb skulle kunna se ut – beakta nedan filstruktur.

- `labb4-fornamn-efternamn` (mapp)
  - `index.php`
  - `login.php`
  - `register.php`
  - `assets` (mapp)
    - `img` (mapp med bilder)
    - `js` (mapp med javscriptfiler)
    - `css` (mapp med stylesheets)

Uppgifter

Nedan följer uppgifterna som resulterar i inlämningen ovan.

Uppgift 1 Designa databasen

Först och främst måste vi uppdatera vår databasdesign. Låt oss fundera över det. Varje kommentar ska alltså kunna vara ett svar till en annan kommentar. Men eftersom varje svar är en kommentar så kan även varje svar till en kommentar ha svar. Och så vidare. Detta låter ovanligt mycket som rekursion. En kommentar kan referera till en annan kommentar.

Notera ordvalet **kan**. Om en kommentar **inte** refererar till en annan kommentar så implicerar det att kommentaren inte är ett svar till en annan.

1. Lägg till en kolumn i kommentarstabellen i databasen. Kolumnen ska vara en `foreign key` som pekar på kommentarer.
2. Kolumnen måste vara `nullable`. Eftersom om en kommentar **inte** pekar på en annan kommentar så implicerar det att kommentaren inte är ett svar till en annan.

Uppgift 2 Hämta & Spara data

Låt oss nu faktiskt ge användaren en möjlighet att svara på andra kommentarer. Vi har behöver alltså förändra sidan som listar kommentarer så att det går att svara på dem. Och när vi sparar en kommentar i databasen, måste vi se till att den refererar till sin eventuella "förälder".

1. Börja med att se till att en `reply`-länk/knapp visas under varje kommentar.
2. När användaren klickar på reply, behöver ni alltså skicka denna till kommentarsformuläret.
3. Glöm inte att skicka med "förälderns" (alltså kommentaren användaren vill svara på) `ID` till kommentarsformulärssidan.
4. Glöm inte att spara förälderns `ID` i `foreign key`-fältet när svarskommentaren sparas.

Skapa nu en funktion för att kunna besvara kommentarer. Antingen skapar ni en ny sida eller att man skickas till det formulär ni redan har och sedan lägger in det i databasen. Det senare är rimligen att föredra.

Uppgift 3 Smyckning

Nu är det dags att designa listningen av kommentarer. Börja med att visuellt föreställa er (med fördel: rita på ett papper) hur ni skulle vilja att er listningen av kommentarer fungerar. I vilken ordning kommer de? Indenteras de? Finns det andra visuella hjälpmedel för att förstå vem som svarar på vems kommentar?

När ni har en idé nedritad på papper är det bara att sätta igång med att försöka exekvera er design. Tänk på att vi antagligen inte bara kan lösa detta problem med `css`. Det kan även hända att det underlättar om vi skriver en annan `sql`-query eller helt enkelt sorterar kommentarerna med hjälp av `php`. Fundera och använd er kreativitet! :)