

Labb 3

E-tjänster och webbprogrammering 7.5 hp VT-14

Introduktion

I denna laboration kommer vi att arbeta i två pass. I det första kommer vi att refaktorera vår applikation (alltså resultatet av tidigare labbar). För att öka nivån av “kontroll”, struktur och läsbarhet. Vi kommer att refaktorera vår `HTML`, `CSS`, `PHP`, `JavaScript` såväl som vår `SQL`. Med andra ord all kod vi skrivit :)

I det andra passet av denna laboration kommer vi att arbeta med att bygga om vår sida till en **Single Page Application**. Vi kommer alltså använda det tillvägagångssätt som ofta benämns `AJAX` för att se till att användaren kan interagera med sidan som vanligt, men utan att någon sida någonsin fullständigt “laddas om”.

Föreslagen struktur

Denna laboration består av 2 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, 2 och 3. 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.

- labb3-fornamn-etternamn (mapp)
 - index.php
 - partials (mapp)
 - * login.php
 - * register.php
 - * posts-new.php
 - * posts-list.php
 - include (mapp)
 - * db-connect.php
 - * db-queries.php
 - * process-register.php
 - * process-login.php
 - * process-post-comment.php
 - assets (mapp)
 - * img (mapp med bilder)
 - * js (mapp med javascriptfiler)
 - * css (mapp med stylesheets)

Hur många filer ni har i varje mapp beror förstås på vilken strategi ni väljer att lösa problemet med. Samt vilka refaktoreringar ni faktiskt genomför. Läs vidare i instruktionerna för klarare förståelse.

Uppgifter

Nedan följer uppgifterna som resulterar i inlämningen ovan.

Uppgift 1 Refaktorering

Refaktorerar din kod så att vi **åtminstone** uppfyller nedan.

- Organisera dina filer i en tydlig och konsistent mappstruktur.
- Eliminera duplikation av liknande kod **inom ett dokument** (i.e. generalisera).
- Eliminera duplikation av liknande kod **emellan dokument** (i.e. generalisera).
- Bryt ut långa block av procedurell kod till **metoder**.
- Låt alla **variabler**, **metoder** och **klasser** ha beskrivande och tydliga namn.
- Bryt ut varje databas-query till en egen **metod**. Använd metodargument för att skicka värden. Exempel: `createPost($author, $message);`
- Om flera databas-query-metoder är liknande. Försök slå ihop dem.
- Separera “printing” och “processing”. Låt **endast** de “yttersta” filerna hantera utskrivning av `html`. Låt inte “process-filerna” hantera `html`. Låt de istället hantera data-processning och business logic. Se förslagen inlämningsstruktur (högst upp) för ett förslag på denna indelning. I förslaget så har man valt att låta alla “visuella” filer bo i mappen `partials`. Sedan inkluderar `index.php` “rätt fil” beroende på vad som ska göras vid det givna tillfället. Vyerna (alltså mappen `partials`) samt `index.php` använder i sin tur filer ifrån `include` när komplexare operationer (såsom att skriva till databasen eller validera input) behöver utföras.
- Blanda `html` och `php` på ett tydligt och konsistent sätt. Du måste inte följa strukturen som är föreslagen i detta dokument. Men det är viktigt att du **har** en struktur.

Uppgift 2 Single-page application

När vi ifrån en webbsida, navigerar till en annan sida, så behöver webbläsaren “ladda om”. Detta gäller alltså förstås då även när vi t.ex. postar formulär. I vårt fall – när användaren postar kommentarer.

Den teknikfamilj som ofta benämns **AJAX** används alltså för att kunna skicka data fram och tillbaka emellan en klient och servern utan att användarens webbläsare (klient) ska behöva ladda om sidan.

Din uppgift är nu att se till att eliminera alla sidomladdningar i din applikation. Om användaren med andra ord postar en kommentar, så ska denna kommentar postas utan att hela sidan laddas om. Ett anrop till servern, och ett hanterande av ett response måste alltså göras i bakgrunden med hjälp av **JavaScript**.

Om du utfört alla refaktoreringar i ovan övning kommer du märka att denna föflyttning till AJAX faktiskt inte är särskilt krånglig. Om du däremot har valt helt andra refaktoreringar kan det hända att detta visar sig svårare. Beroende på vilken fil-/mappstruktur du valt kan det som sagt hända att detta visar sig mycket svårt. Försök fundera på vad det är som gör detta svårt i ditt fall och hur du hade kunnat göra det bättre nästa gång. Refaktorerera gärna mera innan du går vidare.

Den som vill kan även välja att endast låta postning och visning av kommentarer göras utan att ladda om någon sida. Med andra ord, låta login och registrering göras som vanligt. Men när användaren postar en kommentar så uppdateras sidan direkt utan att laddas om.

Om du finner uppgiften svår, kan detta vara ett bra ställe att börja på.