Test av Musicsite

Krav för adminComment.php

Förväntat resultat:

När användaren trycker på ”Delete” skall hon/han få en bekräftelsefråga (OK/Cancel) om hon/han verkligen vill ta bort posten.

Resultat: Bekräftelsefråga OK/Avbryt poppar upp.

Samtliga kommentarer skall visas i en jQuery UI Accordion.

Resultat: Samtliga kommentarer visas i en jQuery UI Accordion.

Krav för adminSong.php

Förväntat resultat:

När användaren trycker på ”Delete” skall hon/han få en bekräftelsefråga (OK/Cancel) om hon/han verkligen vill ta bort posten.

Resultat: Bekräftelsefråga Ok/Avbryt poppar upp.

Om användaren trycker på ”OK” skall formulärdata skickas (submittas) till servern.

Resultat: När man klickar på ”OK” så skickas formuläret.

Om användaren istället trycker på ”Cancel” skall default-beteendet (en submit) avbrytas och likaså att klickhändelsen ”bubblar” till elementets förälder.

Resultat: Default-beteendet avbryts och ingenting händer.

När användaren trycker på ”Edit” skall data från aktuellt formulär kopieras till det övre formuläret vilket möjliggör att användaren kan redigera en post.

Resultat: När man trycker på ”Edit” så fylls det övre formuläret med det aktiva formulärets data.

När användaren trycker på ”Reset” skall formulärdata rensas.

Resultat: När man trycker på ”Reset” så töms formuläret.

När användaren trycker på ”Save” skall formulärdata valideras och om inmatad data uppfyller givna villkor skall formulärdata skickas (submittas) till servern.

Resultat: När giltig data är ifylld så skickas formuläret till servern.

Om data inte uppfyller givna villkor skall default-beteendet (en submit) avbrytas och likaså att klickhändelsen ”bubblar” till elementets förälder.

Resultat: När data inte är giltig så avbryts formuläret från att skickas.

Samtliga sånger skall visas i en jQuery UI Accordion

Resultat: Alla sånger visas i en Accordion.

Krav för adminArtist.php

Förväntat resultat:

När användaren trycker på ”Delete” skall hon/han få en bekräftelsefråga (OK/Cancel) om hon/han verkligen vill ta bort posten.

Resultat: Bekräftelsefråga Ok/Avbryt poppar upp.

När användaren trycker på ”Edit” skall data från aktuellt formulär kopieras till det övre formuläret vilket möjliggör att användaren kan redigera en post.

Resultat: När man trycker på ”Edit” så fylls det övre formuläret med det aktiva formulärets data.

När användaren trycker på ”Reset” skall formulärdata rensas.

Resultat: När man trycker på ”Reset” så rensas formuläret.

När användaren trycker på ”Save” skall formulärdata valideras och om inmatad data uppfyller givna villkor skall formulärdata skickas (submittas) till servern.

Resultat: När man trycker på ”Save” så valideras formuläret och skickas sedan till servern.

Om däremot inte data uppfyller givna villkor skall default-beteendet (en submit) avbrytas och likaså att klickhändelsen ”bubblar” till elementets förälder

Resultat: När man trycker på ”Save” och formuläret inte upp fyller alla villkor så avbryts det ifrån att skickas till servern.

Samtliga artister skall visas i en jQuery UI Accordion.

Resultat: Samtliga artister visas i en Accordion.

Krav för search.php

Förväntat resultat:

När användaren ser den färdigladdade sidan skall hon/han se tre länkar för respektive framsökt sång: ”Show all comments”, ”Comment [filnamn.ogg]” och ”Like [filnamn.ogg]” (existerande kommentarer och formuläret för att lägga till en ny kommentar skall alltså inte vara synliga).

Resultat: När sidan är färdigladdad så syns tre länkar ”Show all comments”, ”Comment” och ”Like”.

När användaren klickar på länken ”Show all comments” och inga kommentarer är synliga skall samtliga kommentarer göras synliga med någon form av animering.

Resultat: När man klickar på ”Show all comments” så glider alla kommentarer fram.

När användaren klickar på länken ”Show all comments” och samtliga kommentarer är synliga skall dessa döljas med någon form av animering.

Resultat: När kommentarerna är synliga och man klickar på ”Show all comments” så glider alla kommentarer undan.

Givetvis skall också default-beteendet vid klick på ”Show all comments” avbrytas och likaså att klickhändelsen ”bubblar” till elementets förälder.

Resultat: När man klickar på ”Show all comments” så hindras länken från att gå vidare.

När användaren klickar på länken ”Comment [filnamn.ogg]” och formuläret inte är synligt skall formuläret visas med någon form av animering.

Resultat: När man klickar på ”Comment” så glider ett kommentarsfält fram.

När användaren klickar på länken ”Comment [filnamn.ogg]” och formuläret är synligt skall detta döljas med någon form av animering.

Resultat: När kommentarsfältet är synligt och man klickar på ”Comment” så glider kommentarsfältet undan.

Givetvis skall också default-beteendet vid klick på ”Comment [filnamn.ogg]” avbrytas och likaså att klickhändelsen ”bubblar” till elementets förälder.

Resultat: När man klickar på ”Comment” så hindras länken från att gå vidare.

När användaren klickar på länken ”Like [filnamn.ogg]” skall default-beteendet för länken avbrytas och likaså att klickhändelsen ”bubblar” till elementets förälder. Därefter skall filen ajax/likesong.php anropas med AJAX. Datatypen skall vara JSON och det är med POST metoden AJAX skall kommunicerar med servern på. Data som skall skickas till servern är id:et för den låt som användaren gillar. Om något går fel skall detta meddelas användaren på valfritt sätt.

Resultat: När man klickar på ”Like” så ökas räknaren med 1.

När användaren klickar på bilden för en utsökt artist skall bilden vissas som fullstorlekt med SlimBox 2.

Resultat: När man klickar på en bild så öppnas den i fullstorlek (SlimBox2).

Funktionalitet med PHP för att på servern…

Förväntat resultat:

… Söka ut och lista data om artister och sånger.

Resultat: Det går att söka på artister och sånger och de listas.

… Hantera jpg- och ogg- filer från klient till servern.

Resultat: Kontroller görs för att försäkra att det är jpg- och ogg- filer som skickas till servern.

Påloggningsfunktionalitet för adminArtist.php, adminSong.php och adminComment.php.

Resultat: adminartist, adminsong och admincomment kräver inloggning och fungerar med angivet användarnamn och lösenord.

I deleteSong() skall en post tas bort från databasen (tblsong).

Resultat: Den valda posten tas bort ifrån databasen.

I insertSong skall en ny post läggas till databasen (tblSong).

Resultat: En ny post skapas i tlbsong.

I updateSong() uppdatera en befintlig post i databasen (tblSong).

Resultat: Posten uppdateras korrekt.

I InsertSong() skall en ogg-fil laddas upp till servern.

Resultat: ogg-filen laddas upp och lagras på angiven plats.

I updateSong() kan en ogg-fil bytas ut.

Resultat: Ogg-filen byts ut mot den nya ogg-filen.

Varje artist skall listas med namn och bild enligt search.php.

Resultat:

Varje sång skall listas med titel, filnamn, antal gilla och filnamn i ett audio element.

Resultat: Sångerna listas inom ett audio element med titels, filnamn och antal gilla.

I ajax/savecomment.php spara en ny kommentar till databasen (tblcomment) och skriva ut datum samt kommentaren som JSON för vidare användning i jQuery.

Resultat: Kommentaren sparas i tblcomment, datum och kommentar skrivs ut.

I startSession() startar ni en session, i endSession() avslutar ni en session och avslutningsvis i checkSession() kontrollerar ni om det finns en session igång.

Resultat: Fungerar som det ska.