

Lab 6

COMP 4711
FALL 2019

What makes internet applications stand the test of time is the foundation they were built on. If the foundation is weak, things will fall apart soon. As our application grows being organized and structured become increasingly important.

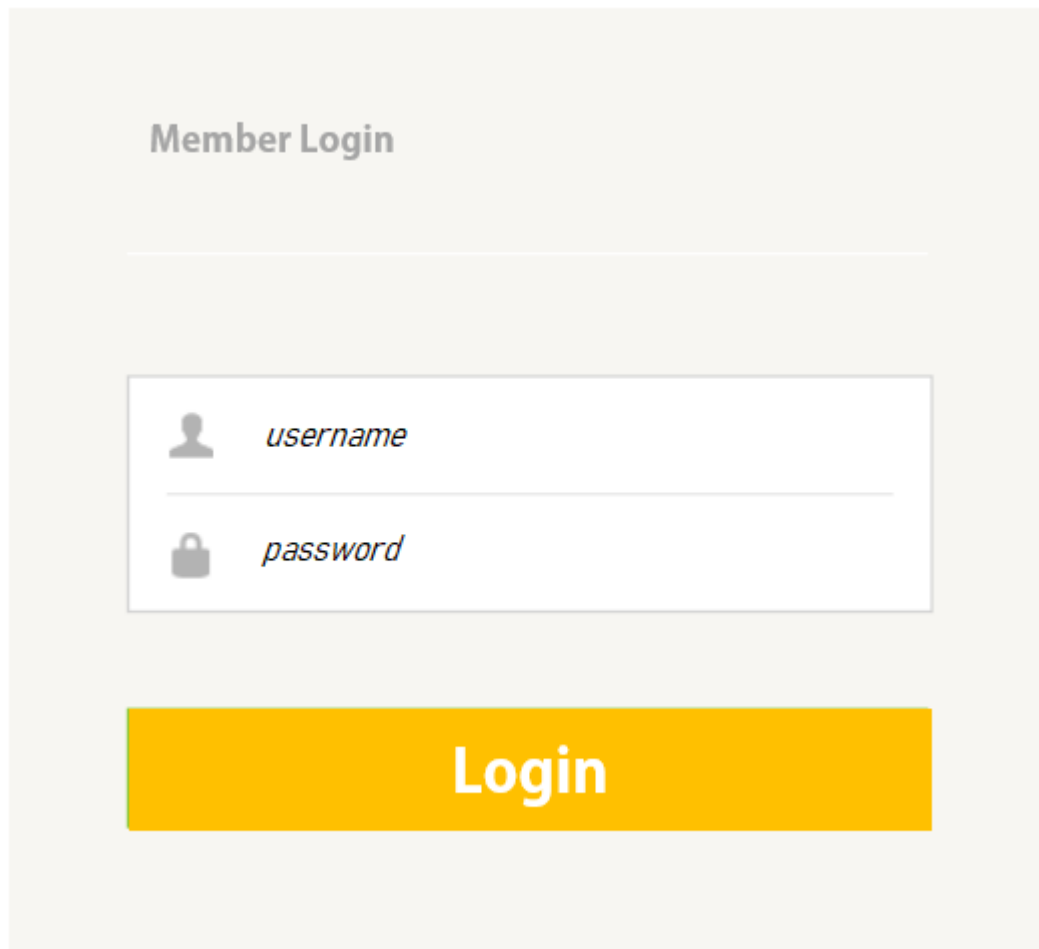
MVC is a design pattern which brings separation of concerns to internet applications. Our task in this lab is to structure our artist directory using MVC and connect to database to store and retrieve our artist data.

MODEL

The application needs a single model (artist), create a Model which will be responsible for adding new artist(s), deleting artist(s), getting all artists and searching.

VIEW

Add a new view, called **login**, this view will need to have two input fields (username, password)
The username should be your A0 number and password = password




The image shows a 'Member Login' form. At the top, the title 'Member Login' is displayed in a bold, dark grey font. Below the title is a horizontal line. The form consists of two input fields stacked vertically. The first field has a user icon on the left and the placeholder text 'username'. The second field has a lock icon on the left and the placeholder text 'password'. Below these fields is a large, solid yellow button with the word 'Login' in white, bold, sans-serif font. The entire form is set against a light grey background.

The artist view, the only new addition is the logout button at the top

Logout

Artist Directory


AddSearch



Kevin Sigma

Best in town

Delete



Katie Holmes

Gotta listen to me

Delete

CONTROLLER

We will have two controllers

LoginController :

Handle any actions on the login view (in our case successful login and redirect to the artist view)

ArtistController:

Retrieval of artist data (database), handle actions : addition, deletion, searching and logout (redirect to login view)

Note all views should be created using a templating engine (pug, handlebars etc) make use of partials, looping mechanism to iterate over a list of artists, provided by the engine. Setup routes and link them to the appropriate controller methods.

Marking guideline		Note (If some of the below are not followed marks will be deducted)
Providing a url for lab	2 Marks	<ul style="list-style-type: none"> Cannot use local or file Storage to retrieve artist data
Connecting to database and executing queries	2.5 Marks	<ul style="list-style-type: none"> Provide a zip of all your code Begin from a empty list of artists
Following MVC structure as described	5.5 Marks	<ul style="list-style-type: none"> Make use of templating engines (pug, ejs, handlebars etc) If you are not providing a link, provide the SQL you used to create your database/table Provide your A0 number while submitting the Lab (comment box) Week 7 Material : node-sql-in-class-exercise-solved can be a good starting point

It's expected that students spend some time looking into nodeJs hosting (aws/azure/heroku etc) and learn how to deploy. They come with a free plan, so you don't need to spend any dollars.