Express Views

Templating with Pug

Lesson Plan

- express config - set template engine
- Hello world template with pug
- pass variables to template
- template loop/condition
- partials
- template inheritance

Lesson summary

https://www.nerdeez.com/articles/express/express-views-pug

Template engine

- convert a template to html
- template syntax depends on the engine
- express works with all the popular template engines among them are
  - Pug
  - Mustache
  - Dust
- to use a template engine you need to install it and also set the following configurations in express
  - **views**
  - **view engine**
- to tell express to turn template to html we use the response object **render** method

Pug

- Template engine
- thing of HTML tags like functions
  - **h1(class=”my-class” id=”stam”)**
- the function arguments are the attributes separated by space
- the text in the node is separated by space from the function
  - **h1(class=”my-class” id=”stam”) this is the text in h1**
- Equipped with this knowledge lets create an a view that displays a todo list

pass data to template

- lets make the todo list get the data from the express route.
- the second argument of the variable is an object of key and value
- The keys of the object will represent variable names that will be available in the template
- lets send in the render and array of items
- to access the variables we can use javascript in our template by simply adding the **-**
    - **- for(let i of item)**
- notice you don’t need curly brace instead we will use indentation to mark the scope of the loop
- you can also drop lines to write multiple code by indentation
- using the sign **=** will also add code but will print it

Condition in template

- you can use javascript if the same way we learned to put js code
- you can indent to mark the block of the if
- lets pass a variable from express marking if the user is admin or not

partials

- its common to have a snippet of html that will be repeating in our app
- lets say we have a footer that will repeat in many cases
- lets create a file in the views folder called **footer.pug**
- this file will contain some text and an image that will repeat in many pages
- to include the file we use the keyword **include footer**
- the path will be relative to the current template
- notice that the partial can use variables from the father

template inheritance

- template inheritance gives us the power to create different layouts for our app
- we can layouts with blocks
- those block we can override on the pages we create
- we can place defaults in those block as well
- we define a block in the layout by using the keyword **block** <name of block>
- we can indent after defining the block to give a default value
- we using the inheritance we have to place on the first line: **extends <name of file>**
- in the child template we then define **block <block name>** and indent with the block content
- we can also add **append/prepend** to include the default content

Summary

- rather you choose to use pug as a templating engine or another engine makes sure you can do the following:
  - inheritance
  - partials
  - js code
- those are the main important and commonly used feature that you will use often

EX.

- in your login application create a layout template which the login page, welcome page, and error page will inherit
- the layout template will define an html tag, a head tag, a body with a block called content, and a footer.
- The other pages will have to put data in the content block
- Make the footer as a partial template
- pass an array to the login page and print that array in a ul li
- make sure you are able to serve an image in your app by using express.static