Web Application Development Project Submission 2023

Name: Ronan Connor

Student ID: G00296371

Date:

Home page/index page/start page (eg., page user should open first):.....node index.js Then load http://localhost:3000/ to get to the homepage

Using the site - information:

Use the Navigation header at the top to move around or search for products

Each page links to another to allow for easy movement.

Logging in with “user” and “pass” will redirect you to the Shop page.

Project Requirements Implementation

ITEM 1	Reference
Allow the customer to enter their login details:	Go to index.html. Option to enter details is presented to the user. If the incorrect details are entered the user will be redirected to failed.ejs where they can re-attempt login
Login details validated (via a login screen) before receiving a summary of the order:	Validation performed by JS
Username set to “user”	user
Password set to “pass”	pass

Brief description of implementation details:	Used JavaScript to validate the details entered. They need to match the ones hardcoded in index.js. auth.js is used to authenticate the details

ITEM 2	Reference
Perform form validation through JavaScript or HTML to ensure that text fields are not empty, and a valid email address is entered	contactForm.html and servicesContact.html both require details before the form can be submitted.
Brief description of implementation details:	Validation is completed through HTML. The user needs to enter valid details, eg. Email, select an option, before they are able to submit the form.

ITEM 3	Reference
Include a slideshow or carousel which displays a different image each time the page is loaded;	Used Boostraps carousel feature to show a carousel on the gallery.html page

Brief description of implementation details:	Followed https://getbootstrap.com/docs/4.0/components/carousel/ to add the carousel. Used images from the images folder in the carousel. The images will be cycled through or the user can manually move through them using the onscreen arrows.

ITEM 4	**Reference**
Allow the user to 'purchase' items from the site;	checkout.html page shows what products have been added to the cart and the prices
Brief description of implementation details:	Items can be added to the cart from the product pages. When the user presses "Add to Cart" they will get a notice letting them know their cart has been updated. Checkout.html holds the logic of getting the product information through JSON parser. This info is collated and displayed to the user. This allows them to review the order before confirming. When the place rode button is pressed a notice will appear letting the user know their order has been placed

ITEM 4	**Reference**
Allow the user to 'purchase' items from the site;	checkout.html page shows what products have been added to the cart and the prices
Brief description of implementation details:	Items can be added to the cart from the product pages. When the user presses "Add to Cart" they will get a notice letting them know their cart has been updated. Checkout.html holds the logic of getting the product information through JSON parser. This info is collated and displayed to the user. This allows them to review the order before confirming. When the place rode button is pressed a notice will appear letting the user know their order has been placed

ITEM 5	**Reference**

Use an object or an array in JavaScript;	auth.js uses an array to store valid user details productView.ejs uses an object in price_quant to store the price and quantity of products the user adds to the card
Brief description of implementation details:	The details are manually hard coded in index.js with auth.createUser("user", “pass"); Any created users will be able to successfully login through the login forms on the site When addToCart() is called productView.ejs uses local storage to store the price and quantity of items added to the cart by the user.

ITEM 6	**Reference**
Use at least one custom module in node;	auth.js has two custom modules; createUser and authUser
Brief description of implementation details:	CreateUser and authUser are used in index.js to create users which are then stored in the users array. authUser is used to validate the details entered on log in forms using JavaScript

ITEM 7	**Reference**
Include capability for handling post and get requests;	index.js has examples of GET and POST requests.

Brief description of implementation details:	Get requests to the /shop endpoint will direct the user to the shop page app.get(“/shop", function (req, res) {
	  res.render("shop.ejs");
	})
	Post requests to /shop endpoint are used to authenticate the users login form input. If successful the user is directed to the shop, other wise they are directed to the failed.ejs page
	pp.post("/shop", function (req, res) {
	[…]}

ITEM 8	**Reference**
Include both static and dynamic content;	index.js serves static webpages and images to the user. Dynamic content is used to show product information from the connected database
Brief description of implementation details:	app.use(express.static(“pages")); handles the static content for the site Line 68 of index.js connects to the database and dynamically serves product information depending on what ID is entered by the user

ITEM 9	**Reference**
Include the use of templates in Node;	The “Views” folder holds all of the templates used
Brief description of implementation details:	The Views folder holds the failed, productView and shop templates. This is most useful for the productView page which allows for the pages to be dynamically updated with product information depending on the product id

ITEM 10	Reference
Include error messages to provide feedback to user sin case of issues or errors;	Error messages are included when connecting to the database in the index.js file. Rudimentary error handling with the failed.ejs
Brief description of implementation details:	The error is extracted and output to the console for debugging should it occur when connecting to the database or getting products form the database Manual error handling by directing the user to the failed.ejs page if their log in details fail when authUser is called

ITEM 11	Reference
Connect to a database that contains relevant site information (eg., product info, prices) using NODE (your database name should be your ATU ID);	Used mySQL to connect to a local database and access product information
Brief description of implementation details:	Used mySQL to connect to a locally hosted database called G0029637 . The user and password details are “root” . This will allow the site to access the product information like, name, price, description, etc.

ITEM 12	Reference
Use Bootstrap version 5 via CDN	All HTML pages use Bootstrap to serve CCS content
Brief description of implementation details:	Using Bootstrap via the CDN by including it in the <head> tag of pages. Eg. index.html, gallery.html.