

Course: WEB322 - Web Programming Tools and Frameworks	Assignment 3-5 of 5	Contribution: 30% of final grade
Instructor: Wei Song	Date Given: October 30	Date Due: First submission: Thursday, Nov 26 Final submission: Thursday, Dec 10 In-class demo: Dec 08 or Dec 11

Notes for the Student: This Project corresponds to Assignment 3, 4, 5 and is a continuation of the work you have already completed in Assignments 1 and 2.

Background: You will need to have access to an IDE or text editor and have a thorough understanding of Node.js, Express, Git, GitHub (or Bitbucket), Heroku, MongoDB and Mongoose.

Assignment Regulations

- This project must be done individually.
- **An in-class (online) demonstration of this project is required. Failure to demonstrate your project would result in 0 marks.**
- Please review Seneca's policies on Academic Integrity, specifically:

*"Each student should be aware of the College's policy regarding Cheating and Plagiarism. Seneca's Academic Policy will be strictly enforced. To support academic honesty at Seneca College, all work submitted by students may be reviewed for authenticity and originality, utilizing software tools and third-party services. Please visit the Academic Honesty site on <http://library.senecacollege.ca> for further information regarding cheating and plagiarism policies and procedures." **Thus, ensure that your code or any part of it is not duplicated by another student(s). This will result in a percentage of zero (0%) assigned to all parties involved.***

Technical Requirements

- All back-end functionality **MUST** be done using Node JS and Express.
- Your views **MUST** be created with Express-Handlebars
- You are required to use a CSS Framework Bootstrap to make your website responsive and aesthetically pleasing.
- You can use any animation library (JavaScript or CSS) that you want. This is not mandatory; it is up to you.
- You are **NOT** allowed to use any Front-End JavaScript Frameworks, specifically, **No React, No Vue, No Angular.**
- You must use MongoDB as your database engine

Detailed App Specification

This assignment is a continuation of Assignment 1 and 2, thus all the requirements for this assignment is to be made “on top” of your Assignment 1 and 2.

Assignment 3 (10%)

Application Architecture

1. Your application **MUST** be structured **FULLY** according to the MVC Design pattern. Thus, your views, models and controllers must be separated as per the design pattern requirements.
2. **All** sensitive credential information **must** be stored in **environment variables**. Examples include: your sendgrid access token, MongoDB connection string, etc. Implement environment variables locally using the [dotenv](#) 3rd party package.

User Registration Module

You are required to implement database functionality for your registration page that was previously implemented in Assignment 1 and 2. Thus, when a user fills out the registration form and then hits the submit button, provided that all the validation criteria were not violated, your website must then create a user account in your database.

Once the user account is created, your web application must then redirect the user to a dashboard page.

Regarding your database functionality, the following rules must be followed:

1. Setup and configure a MongoDB cloud service using MongoDB Atlas <https://www.mongodb.com/cloud/atlas>.
2. Connect your web application to your MongoDB database using an ODM called **Mongoose**.
3. Name your database and collections appropriately.
4. Ensure that the email field in your registration form **is unique**, thus your application must prohibit different users from having the same email in the database.
5. Passwords **must not** be stored in plain text in the database; thus, your application must store passwords in an encrypted format. You can use a 3rd party package called [bcryptjs](#) to do the aforementioned.

Note:

- For MongoDB, see week 8 lecture Notes: <https://web322.ca/notes/week08>
- For bcryptjs, see week 11 lecture Notes : <https://web322.ca/notes/week11>

Authentication Module

You are required to implement a fully functional authentication module with the following features:

- Your application must allow a Data Entry Clerk and customers who want to purchase meal packages, to **log-in via the login form created in Assignment 1 and 2.**
- Upon a successful authentication (entering an email and password pair that exists in the database) **a session must be created to maintain the user state until they have logged out of the application.**
- Upon an unsuccessful authentication, the application **must** display an appropriate message (Example: **Sorry, you entered the wrong email and/or password**)
- Also, after successfully authenticating, the application must determine if the person logging in is a data entry clerk or a regular user and will be redirected to their respective dashboard.
- A customer will be directed to a user dashboard and a data entry clerk will be directed to a data entry clerk dashboard.
- Both dashboards, must show the user's name (first name and last name) and a logout link
- The logout link must destroy the session created when the user initially authenticated.
- Specific routes can only be accessed when users are logged-in, thus those routes must be protected.

Note:

- For implementing session and protecting routes in Express app, see week 11 lecture notes : <https://web322.ca/notes/week11>

Assignment 4 (10%)

Data Entry Clerk Module

You are required to implement a Data Entry Clerk module that allow a data entry clerk to do the following:

1. **Create Meal Packages:** The Data Entry Clerk must be able to add new meal packages to the database. See the following data that must be added when a product is created:
 - a. Meal Package name,
 - b. Meal Package price,
 - c. Meal Package description or details,
 - d. Meal Package food category
 - e. The number of meals inside the package
 - f. Set Meal package as a top package to be shown on the home page (or not).
 - g. Upload a photo of the meal package (to keep the project simple, only upload one image per meal package).
2. Ensure that all created meal packages that were entered into the database are populated on the front-end of the web application, specifically on the meal package listing page that was created in Assignment 1 and 2. Additionally, only meal packages that were set as "Top Meal Packages" should be populated in the "Top Meal Package" section of the home page. **Please note, a visitor to the web application does not need to be logged in to view the meal packages that were created by the data entry clerk.**
3. Ensure that a user can only upload an image, i.e. jpgs, gifs, or pngs, for the meal package photo.
4. View a list of all created meal packages.
5. Edit and change meal package details for a selected meal package. Example: price, title etc

Please note, a Data Entry Clerk must be logged-in to the web application to be able to do the aforementioned (create meal packages, view all created meal packages, edit and change meal package details). To facilitate the logging in of the Data Entry clerk, create a regular user in the Front-End and then log into your MongoDB Atlas account and manually change a field in the document to make allow the application to be able to make the distinction between customer vs data entry clerk - **I will further expound on this in class.**

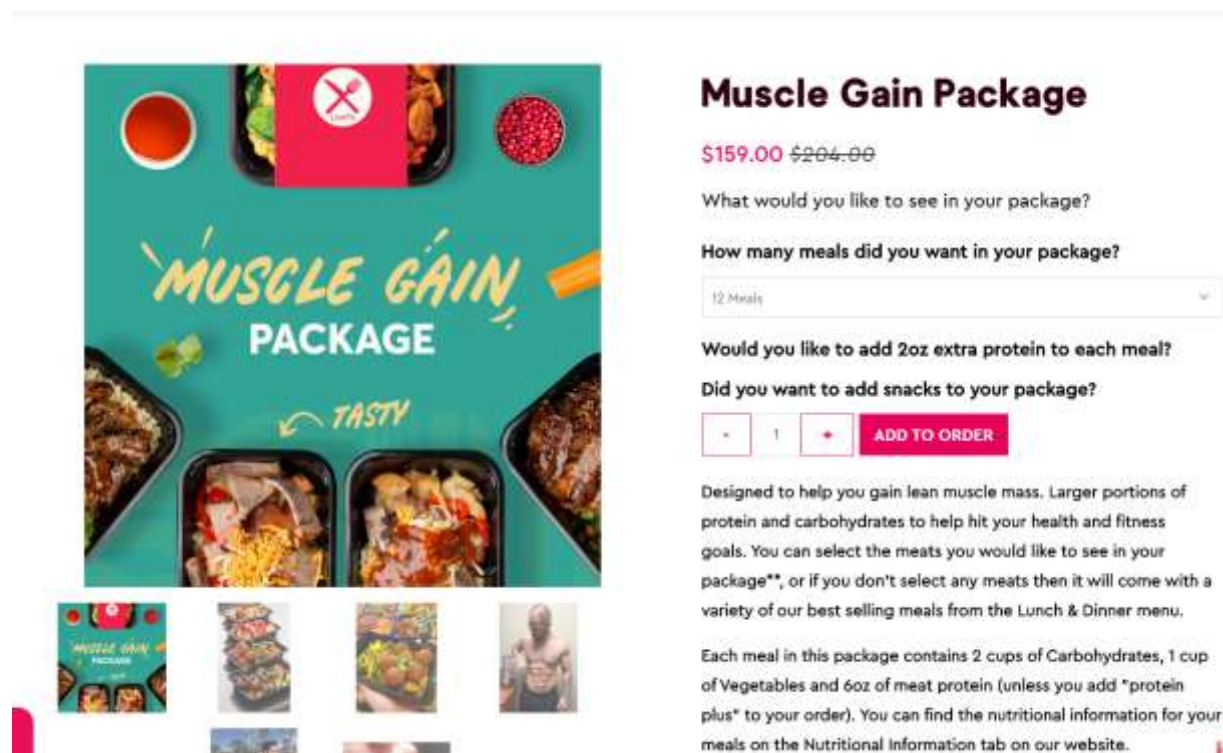
Assignment 5 (10%)

Meal Package Description Page

Only logged in users should be able to “purchase” meal packages by adding selected meal packages to their shopping cart.

From the meal package listing page, when a user clicks on a particular meal package, they should be navigated to the **Meal Package Description page** of the clicked meal package and from that page they can add the meal package to their shopping cart.

Sample of a Product Description Page Example:



The Meal Package Description page of any meal package should list the following:

- A. Meal Package Image
- B. Meal Package title
- C. Meal Package description
- D. Meal Package price
- E. No of meals within the package
- F. Add to order

When the user clicks the “**Add to order**” button, the given package will be added to the user's shopping Cart.

Shopping Cart Module

The customer's dashboard should have a link to the logged-in user shopping cart. This page must display all the meal packages that were added to the shopping cart and how much of each package was added for the given user. Also, the page should have a "Place your order" button.

See Example


LiveFit

[Cart](#) > [Information](#) > [Shipping](#) > [Payment](#)

Express checkout

OR

Contact information

 Kadeem Best (kadeembestteaches@gmail.com)
[Log out](#)

☒ Keep me up to date on news and exclusive offers

Shipping address

Saved addresses
Use a new address

First name Last name

Company (optional)

Address

Apartment, suite, etc. (optional)

City

Country/Region Province Postal code

Canada Alberta

Phone

[< Return to cart](#)

[Continue to shipping](#)

Weight Loss Package \$145.00
10 / No Thanks! / No Thanks!

Gift card or discount code [Apply](#)

Subtotal \$145.00

Shipping [?](#) Calculated at next step

Total CAD **\$145.00**

When the **"Place your order"** button is pressed, your web application should "clean" your shopping cart, send an email to the logged in user's email, indicating all the packages purchased and the quantity amount for each meal package purchased as well as the customer's order total.

Assignment 3 Rubric

Criteria	Not Implemented 0	Partially Implemented 1	Fully Implemented 2
User Registration <ul style="list-style-type: none"> • MongoDB cloud service is setup Database and collection have appropriate names • User's data is inserted into the database when the user fills out the form and hits the submit button. • User is redirected to a dashboard page when form is submitted • Email is unique • Password is stored in encrypted format 		2	4
Authentication <ul style="list-style-type: none"> • A Session is created to maintain the user state until the user logs out of the application. • Upon an unsuccessful authentication, the application displays an appropriate error message • The application directs a data clerk to their dashboard and a regular user to his or her dashboard. • Both dashboards, show the user's name (first name and last name) and a logout link • The logout link destroys the session. • Routes that can only be accessed when users are logged-in, must be protected. 			

Total: 24 MARKS

Assignment 4 Rubric

Criteria	Not Implemented 0	Partially Implemented 1.5	Fully Implemented 3
Data Clerk Module <ul style="list-style-type: none">• A data Clerk account is manually created in the database.• A data Clerk can create meal packages with all the necessary data• A data can upload a photo for a meal package• Only an Images can be uploaded as a photo.• All meal packages created are populated on the Front-End of the website, specifically the meal package listing Page• Meal packages that are set as top meal packages are rendered in the appropriate section on the home page.• The data Clerk can view a list of all created meal packages.• Data Clerk can edit product details			

Total: 24 MARKS

Assignment 5 Rubric

Criteria	Not Implemented 0	Partially Implemented 3	Fully Implemented 5
Booking Module <ul style="list-style-type: none"> Only logged in users can add meal packages to their shopping cart. User's must be able to view all their meal packages in their shopping cart and the shopping cart must list all meal packages added, the quantity amount of each product and their total amount. When the user clicks the "Place Order" button, the application must "clear" the shopping cart and send email with the entire order information 			
Look and Feel <ul style="list-style-type: none"> Overall site looks polished on all devices. 	0	4.5	9

Total: 24 MARKS

Deductions

Criteria	Not Implemented	Partially Implemented	Fully Implemented
Deploying to Heroku <ul style="list-style-type: none"> Your web app was successfully deployed to Heroku. 	-3	-2	2

Assignment Submission

- Add the following declaration at the top of your **server.js** file:

```
/******  
* WEB322 – Assignment 03-05  
* I declare that this assignment is my own work in accordance with Seneca Academic Policy. No part  
* of this assignment has been copied manually or electronically from any other source  
* (including 3rd party web sites) or distributed to other students.  
*  
* Name: _____ Student ID: _____ Date: _____  
*  
* Online (Heroku, https://...) Link: _____  
*  
******/
```

- Compress (.zip) your project folder and submit the .zip file to My.Seneca under **Assignments -> Assignment 3-5**
- **First submission** is due on **Thursday, Nov 26 @ 23:59**
 - Halfway progress checking: at least 50% of the assignment tasks (overall) must be completed.
 - Failure to make this submission will result in 40% deduction for your assignment grade.
- **Final submission** - Due date: **Thursday, Dec 10 @ 23:59**
- **In-class (online) demonstration:** You're requested to demonstrate/run your project to your professor on either **Dec 08 or Dec 11** during the class/lab sessions.

Important Note:

- **NO LATE SUBMISSIONS** for assignments. Late assignment submissions will not be accepted and will receive a **grade of zero (0)**.
- After the end (11:59PM) of the due date, the assignment submission link on My.Seneca will no longer be available.
- Submitted assignments must run locally, ie: start up errors causing the assignment/app to fail on startup will result in a **grade of zero (0)** for the assignment.
- Posting your assignment solutions anywhere online, e.g. GitHub/Bitbucket, as public project constitutes a **violation of Seneca College's Academic Integrity Policy**.

THE END