Marcus Doucette

2025-10-04

CS 499

M5 Milestone

Briefly describe the artifact. What is it? When was it created?

This project is a full stack web app. It has a user client and admin client each hosted by separate servers and there is also a data server that provides access to a database to both clients. This was made jus this past summer for my CS 465 class.

Justify the inclusion of the artifact in your ePortfolio. Why did you select this item? What specific components of the artifact showcase your skills and abilities in software development? How was the artifact improved?

Being one of the more recent artifacts that I’ve used that includes a database, I thought it would be good fit to meet the database requirements of this course. I don’t particularly like web dev, but I think that this also showcases my flexability being able to make complex additions to software types that I’m not as familiar or interested in.

Did you meet the course outcomes you planned to meet with this enhancement in Module One? Do you have any updates to your outcome-coverage plans?

I originally wanted an embedded systems project that would move my other projects around to fill this database slot; however that project requires real world components that I had already disassembled and this project uses a much more conventional database. I think that my other projects have met all the requirements apart from the database one, so as long as this project demonstrates good progress with databases as well as the associated security concerns then anything else would just be extra coverage.

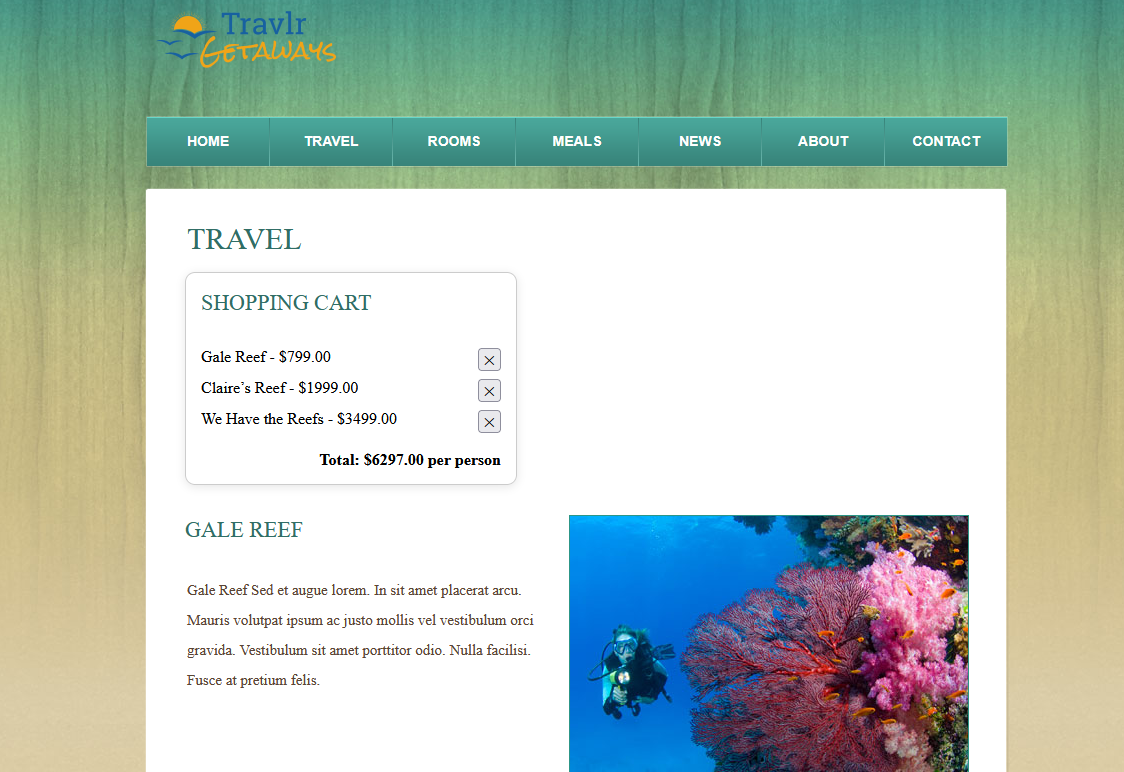
Reflect on the process of enhancing and modifying the artifact. What did you learn as you were creating it and improving it? What challenges did you face?

Adding the cart to the screen was fairly simple. I originally wanted it to be something like a popup but then decided it would be easier and clearer to have it just be a list of elements in the view and if I really want I can make it appear and disappear with a button later. Using css I also added styles to the div for the whole element as well as the div containing the listed items and the div for displaying the total price. Maybe in the future I could add a ‘buy’ button but since I don’t have anywhere for that to lead I decided to leave it out of this rendition.

Implementing the ability to have items be added to the cart was a bit more tricky because of the handlebars framework that I was unfamiliar with. Making a public script that stored the name price and ID in a list was pretty easy and implementing a render function that grabbed the <div>s based on their css class and populated them with appropriate html content dynamically was also fairly straightforward. Editing the handlebars to have an onclick listener for the <li>s was what I decided to do instead of trying to deal with the multiple hyperlinks I already had to reroute to different pages or data about the images; however the handlebars syntax caught me up a bit when trying to pass the appropriate arguments to my function.

I probably rely on local storage more than I should since it’s fairly easy and fits much better with the kinds of ways I like to program things. I discovered that this kind of cart feature is a good candidate for making use of the local storage though as having this data hosted on a server would just be more cumbersome for little to no security benefit. I did some research on the security of local storage and while the data stored there isn’t very secure, since the application only stores information that the server readily provides to all users, I don’t believe this data being stolen represents any significant risk to the user more than detecting they went to the page in the first place would.

I think of the 3 projects I’ve worked on for this course so far, this one has been the most cumbersome to update the git README for. It’s not that it’s anything particularly new but instead the fact that running a web app has so many more moving parts than something like android studio where I can basically just say plug in the url and hit play or vcpkg that requires the user to run one or two terminal commands in the right place and then the automation I wrote can do the rest. For this project I needed to remember and provide detailed instructions on several steps like starting the user and admin clients separately along with setting up a mongodb cluster and linking it to the app in one of a few ways as well as setting up admin access. While one of the benefits of working with this kind of tech stack is that most of this stuff only needs to be done once and then can be forgotten about. The consequence of this is that coming back to this after a while and having most of it just work led to a lot more work when trying to reverse engineer the instructions for someone with a clean machine. It also doesn’t even get into some of the tools that were used for this project like postman or angular that likely require separate explanations for how to setup and use them to their fullest to aid in debugging or development.



Features

* Add cart button and view/widget to show current items
* Change page interactions to add and remove items from cart instead of link to data responses
* Remember items in local storage between page loads
* Improve GitHub build instructions & presentation