# Justin Cabral : Team Bioluminescence

# Purpose

The purpose of this document is to give me information that I can use to appropriately assign grade-bearing credit for the project work. I am an external viewer of the work that was done, and this information will be useful in understanding how the teams worked.

# Introduction

Team Bioluminescence consisted of myself, Jonathan Duran, Humberto Martinez, and Miles Gregg. We worked incredibly well together, and it was really a great experience all around. I feel that if we had more time, we really could have fleshed out the user interface to a level of those seen on websites in production. However, since our grade wasn’t reflective of UI polish, we put ample time into making sure the front end and back end were in sync during the entire development process.

My role and responsibilities for this project were to work on the front end using React JS and manage the GitHub repository for the front-end portion of the project. I was responsible for merging branches and making sure there were no conflicts between them. I was also tasked with getting the initial project set up and started since I had previous experience with setting up multi-page websites using the React Router Dom package.

# Your accomplishments

My accomplishments on this project include starting the initial project, creating/maintaining the GitHub repository, merging branches, and ensuring the final build worked before uploading to S3 bucket.My accomplishments also included programming the front-end alongside Humberto using ReactJS where we both split the work evenly.

The part I’m especially proud of is the ease of which I made it, so we easily add more URL targets to our website when we wanted to create new pages for the use cases. I had previous experience with multi-page web sites before, so I knew of the react-router-dom and how to take advantage of it when it came to managing the flow. An example can be seen below:

Text

Description automatically generated

A simple thing like this makes it much easier to continue scaling the project, since each page is just another ReactJS component. This means that most code can be reused and cuts down on the amount of work needed to be done. When it comes to software, I think the best code you can write is no code at all, and that’s always what I try to do when I approach software development.

Another accomplishment I’m really proud of is the uniformity of all of our calls to the server, and transforming the values from input fields to the correct format. Having worked on previous pages before that did similar actions, I knew a few ReactJS useState() that could help us. For an example in the photo below:

Text

Description automatically generated

As you can see, you can capture all form values from TextFields inside of a JavaScript object using React useState(), and name their parameters. Then when I want to send a body to the API endpoint, I can simply parse the JavaScript Object with JSON.stringify(). This help simplified the entire process when user input was required, and its what we used cross all of our front-end pages when taking values and sending them to the server.

# Lessons learned

I think the most important lesson learned for me personally was how to communicate with REST API’s and how those REST API’s are constructed on the server. Although I have been programming for quite some time, I never really got into server or network programing. However, this class gave me a great opportunity to understand at a deep level how the parts are working together, which is something I really appreciate. I can definitely see myself taking these skills into the future for any work I may do.

This class was as expected for a master’s level course, I only wish we had more time to flesh out the project to a production level website. I would say that the entity, boundary, controller for ReactJS was a bit weird with the PlanarPuzzle but it was still fun!

If I had to do something different, it would probably have been to choose a different UI library other than Material UI by Google. It led to some headaches along the way, but we made it work.

# Advice to future teams

Since our team was able to effectively meet all deadlines, I would say the advice to future teams should be to decide early on which people are working on the front-end, and which people are working on the back-end. However, that does not mean you work separately. There was never a time where the front-end developers and the back-end developers were not programming at the same time. We applied a new industry standard that is swarm programming. This helped reduce all conflict when issues arose to why certain API endpoints weren’t working. We also used Trello to keep track of tasks to be completed, which helped the team stay very organized.

# Cheers and jeers

I truly believe that everyone on this project put in the same amount of effort to put forth a truly great showing of a project. There were never any arguments or anyone not putting their best foot forward. This was one of the few times in college where the team had **all A** players, and I think our final result showed that.