






Category	Comments	Actions Taken by Reviewed Group
Build	The instructions were very clear, and by reading the README I was able to successfully clone the entire repository and build the project with ease.	
Legibility	There is a good amount of comments and explanations present in the code, and the variable and function names allow me to understand what is going on. The structuring of the code files is also very well done, with the overall organization/logical-pathing of the code base allowing for easy perusing of the code itself. With that said, there are some files such as App.js under components that have no comments or documentation whatsoever, so it is hard to follow in these cases.	Added more comments to the front end components.
Implementation	Almost all of the code written in this codebase is extremely concise and well thought out; I am impressed to say the least. Most of the code has also been modularized, such as the API functions being placed in a models folder, and the individual routes being placed in their own folders, so the code is very clean and concise. The group did a great job of putting certain helper functions as well into modular structures and files, since this will improve scalability and succinctness of code without sacrificing performance.	
Maintainability	There are unit tests. The tests are pretty expansive, and cover most of the functionality seen throughout the application. However, one thing I would change is using the same inputs for most of the test cases. Instead, several different batches of inputs should be used within the functions to truly gain insight into the nature of the functions under load of differing real world conditions as well as expose any underlying bugs or issues.	We feel that our unit tests have good coverage for the functions that they cover.
Requirements	Yes, this code fulfills the requirements listed out in requirements doc. It appears that 95% commands even fulfill the time constraint of three seconds, and 95% of content is on a single page. I am curious to see how the website can handle 300 users as that is a requirement as well, as well as how the 'maximum pool size' of 25 connections handles this high network load especially with the requirements doc specifying an estimated API request every 5 seconds per user.	Just to be safe we have increased the pool size to 100. However our application can easily handle 300+ users.
Other	No other comments, as I think the group did a fantastic job creating high performing, scalable, and well documented code.	

CS72





Category	Description	Reviewers Comment	Action Taken by Reviewed Group
Build	Could you clone from Git and build using the README file?	Yes. I had some issues on windows connecting, but I figured out that I had to use git bash or WSL.	We have added suggested Shells for each major OS to our README.
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	The code is well documented as uses the latest standards of React. I appreciated the comments and the project structure because it was easy to navigate and find things.	
Implementation	Is it shorter/easier/faster/c leaner/safer to write functionally equivalent code? Do you see useful abstractions?	Is there a reason why you didn't write individual CSS or SCSS files for components instead of defining the CSS inline? I feel like having the CSS separate might make readability easier and might make the program load faster.	<p>Yes. Our client was adamant about us using emotion for styling.</p> <p>Our client believes that to conform to the modular conventions of react the styling must also be done on a per component basis.</p>
Maintainability	Are there unit tests? Should there be? Are the tests covering interesting cases? Are they readable?	There are unit tests for the back end to test plan creation and edge cases. There should be for this type of application because there are so many caveats in applied plans. These tests are very readable.	



Requirements	Does the code fulfill the requirements?	Yes it does	
Other	Are there other things that stand out that can be improved?	You could include tests for the front end. I also think that you should separate the CSS from the components in React.	<p>There are some basic snapshot tests for the front end.</p> <p>See the above answer for why we are keeping the CSS.</p>

**Group
72**


Out of 4 I will give this group 3.5

Not sure why it isn't a 4 out of 4
as there wasn't any suggestions of things we
should change

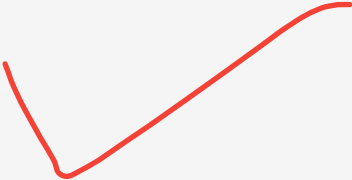

Category	Description	Reviewers Comment	Action taken by reviewed group
Build	Could you clone from Git and build using the README file?	Yes I was able to clone from Git and build it using README file. The steps were quite clear and easy to follow.	
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	Yes the variable names and method gave me a gist of what the program is trying to accomplish. The code also adhere to general guidelines and code style.	
Implementation	Is it shorter/easier/faster/cleaner/ safer to write functionally equivalent code? Do you see useful abstraction?	Yes there are several different ways to write code but don't know if they would be faster and shorter than their current working code. But I found code refactor was done as they weren't repeating steps.	
Maintainability	Are there unit tests? Should there be? Are the test covering interesting cases? Are they readable?	Yes there are unit test and these test were covering interesting unit test.	

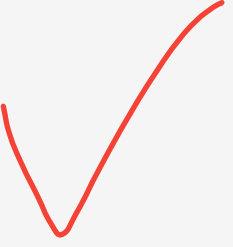
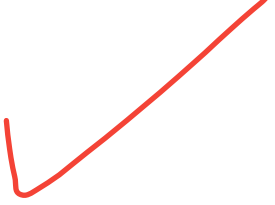
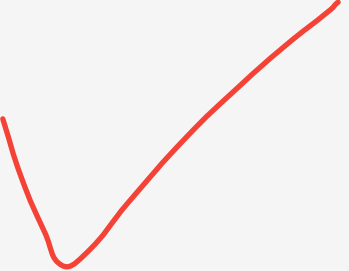
Require ments	Does the code fulfill the requirements?	Yes the code fulfill the requirements.	
Other	Are there other things that stand out that can be improved?	No felt confident with their layout and front end interface of their application.	

Review of Group 72

Category	Description	Reviewers Comment	Action
Build	Could you clone from Git and build using the README file?	Awaiting Access to the Documents needed to install	<p>Requested Access via Google Drive</p> <p>We always granted access to our database the same day it was requested.</p> <p>We also included a team email to contact in the README if there were any issues.</p>
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	Yes, the variables names and method were very self - explanatory. The comments throughout the code made it easy to understand and followed good guidelines.	Nothing that I can suggest. The directories are very nicely organized and named allowing outside users to easily find what they are looking for.
Implementation	is it shorter/easier/faster/cleaner/safer to write functionally equivalent code? Do you see useful abstractions?	The level of security required for the Applied Portal makes it difficult to write faster/shorter code. It may be cleaner to use static CSS files rather than emotion but I do know emotion is very useful when overriding other pre-existing CSS styles.	<p>Nothing I see that needs to evidently change.</p> 

Maintainability	<p>Are there unit tests? Should there be? Are the test covering interesting cases? Are they readable?</p>	<p>Yes there are unit tests and they are very readable as they all describe what they are validating.</p>	<p>Only suggestion is adding a couple more tests for the front end. It is challenging to add more unit tests to the front end that aren't just tests for the sake of writing tests.</p> <p>For now we have snapshot tests for the front end.</p>
Other	<p>Are there other things that stand out that can be improved?</p>	<p>Not sure who will be doing long term maintenance on this project but it may be beneficial to change some of the hard coded parts into methods for example allowing users to easily add a required class or view required classes without having to open the code and add it.</p>	<p>Create methods to allow authenticated users to add required courses, restrictions etc. from the command line or UI.</p> <p>This is beyond the scope of our project.</p> <p>Our client was happy with an internal array of required course codes.</p>

Category	Description	Reviewers Comment	Action
Build	Could you clone from Git and build using the README file?	Yes, I was able to clone the repository and run the app using the instructions from the README.	
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	The code was clean and easy to follow. Most files were well-documented, and nothing stood out as badly structured or confusing.	

Implementation	Is it shorter/easier/faster/cleaner/safer to write functionally equivalent code? Do you see useful abstractions?	Nothing stood out to me as inefficient or unsafe. Several functions are a bit long and could use extra modularization to make them easier to reuse and read.	
Maintainability	Are there unit tests? Should there be? Are the test covering interesting cases? Are they readable?	There are applicable unit tests that seem to cover all relevant cases. All tests are well-documented.	
Other	Are there other things that stand out that can be improved?	Nothing comes to mind.	

This review was labeled for CS 72, however,
this review is clearly not meant for our project. We did not use Python or implement a feed reader

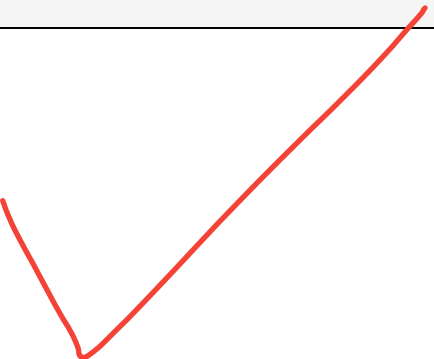
Category	Description	Reviewers Comment	Action
Build	Could you clone from Git and build using the README file?	Instructions were clear and had no issues compiling the repo.	n/a
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	All variable names and methods made sense and were easy to follow. There could be more comments on the front end to explain functionality, considering the project will be passed onto future code maintainers. Other than that, whitespace formatting looks great and allows the code to be very readable. Nice job!	n/a


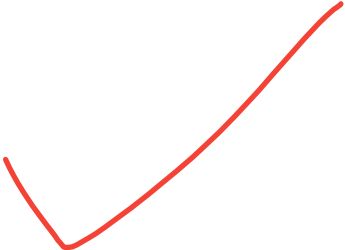
This review is clearly not meant for our project. We did not use Python or implement a feed reader

Implementation	is it shorter/easier/faster/cleaner/safer to write functionally equivalent code? Do you see useful abstractions?	<p>I see no outstanding issues concerning the implementation of the website. There are small bugs the team needs to reference, such as authentication processes.</p> <p>In reference to the Python script, rss_feed_reader.py, an additional class can be created to hold the feeds defined by main. Right now, the code references specific indexes of feed objects, which may confuse future developers. Main is a large function that could use more modularization.</p>	n/a
Maintainability	Are there unit tests? Should there be? Are the test covering interesting cases? Are they readable?	<p>Test coverage is provided when necessary.</p> <p>The group talked about performing manual testing on their authentication processes, which is the area that requires more attention.</p>	n/a
Other	Are there other things that stand out that can be improved?	n/a	n/a

This review is clearly not meant for our project. We did not use Python or implement a feed reader

Code review score: 3

Category	Description	Reviewers Comment	Action
Build	Could you clone from Git and build using the README file?	Able to clone and build, but with slightly tweaked instructions. The README suggests using npm run for both the server and client, which did not work for me. However, entering npm run brought up a 'help' menu of sorts, which let me know to use npm start instead. This worked for both. I'm not sure if this is intended or not. If not, the README should be tweaked slightly so that even the most uninformed user will be able to run it.	<p>The README says to use 'npm run dev'.</p> <p>'npm run dev' will start the application correctly.</p>
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	Good flow, everything is easy to follow. It seems to follow best practices and is thoroughly documented in most areas. In planValidation.js, there is a large chunk of awaiting constraints, which could use a little documentation. I know that the actual constraints are documented in planConstraints.js, but for the average reader it would be helpful to just have a line there reminding the reader which one it is. It is worth noting that the names themselves are intuitive and make sense, however.	

Implementation	is it shorter/easier/faster/cleaner/safer to write functionally equivalent code? Do you see useful abstractions?	I don't see any glaringly obvious ways to improve the quality of the code. Nothing seems shorter, easier, faster, or cleaner to rewrite. Session handling (student vs advisor) is written in the simplest way that I can think of, as a simple Boolean value. This is what I would've done as well. Decent validation is in place in the form of states that are changed if something is not validated.	
Maintainability	Are there unit tests? Should there be? Are the test covering interesting cases? Are they readable?	There are tests that check all the validation and constraints for the users and plan data. Zachary mentioned that they aren't the most thorough, but from looking at them for a short while I think they're definitely adequate to test everything they wanted to test.	
Other	Are there other things that stand out that can be improved?	Not that I recall. Everything is pretty much good as-is, and my very minor observations are included in the other sections.	