CS 21: Data Mining for Mobile System Performance

Code Freeze / Code Review Responses

GitHub Repo: https://github.com/guyera/Direct-Key-Dashboard

We received reviews from exactly five students. Following are responses to each of those reviews. Each response is contained within its own table. Category descriptions are left out to improve readability.

Response 1

Category	Reviewer's Comment	Reviewee's Response
Build	I was unable to try to build the code myself due to the NDA and proprietary data.	N/A
Legibility	Follows camelCase for variables and PascalCase for classes and function which I believe is standard for C#. Overall style looks good and the code is well commented, variable names appear to be appropriately chosen. This is a very minor issue, but there was a small inconsistency in the line breaks for me; sometimes line breaks were introduced to keep the code legible in a smaller frame, especially with comments, while other times such as line 98 of ApiBarChartViewComponent.cs there were 200+ characters in a single line.	We went through every source file and broke lines apart where the characters could not all fit on a standard desktop monitor at standard zoom levels.
Implementation	I'm not super familiar with C# and it's libraries, so there could be something I'm missing, but nothing stood out to me as redundant or inefficient.	N/A
Maintainability	I didn't find any unit tests. I think it might be possible to include unit tests to ensure that future changes don't break the typecasting by feeding the charts dummy data	Unit testing is hard for UI applications. Nevertheless, we implemented some unit tests to test data projections and summaries. See the new "Testing" directory in the

		repository root. At the very least, it's a registry to encourage unit testing of more complex features should they be implemented in the future.
Requirements	From my understanding of the project based on the discussion in class, it met requirements awhile ago and the code I reviewed was actually stretch goal material.	This is correct. N/A
Other	There's no other major points that really stand out for me. Without being able to test the code	N/A

Response 2

Category	Reviewer's Comment	Reviewee's Response
Build	I was able to clone repository. However, I did not see the README file. If you can add a README file to explain the project and the features of each part of code, it will be much better.	Added a README explaining how to build and run the project, although it only applicable to the client, and only somewhat so, given the requirement for API certificates, it being an in-house application, and it being written in the same Framework and with similar dependencies as all of the client's other projects.
Legibility	The flow is legible. There are many detailed comments which are very helpful for audience to understand the code.	Thank you.
Implementation	I think the implementation is perfect. The code is very clean.	Thank you.
Maintainability	I did not find any unit tests.	Unit testing is hard for UI applications. Nevertheless, we

		implemented some unit tests to test data projections and summaries. See the new "Testing" directory in the repository root. At the very least, it's a registry to encourage unit testing of more complex features should they be implemented in the future.
Requirements	I think the code fulfill all the requirements.	N/A
Other	Group 21 did a really good job. The only one suggestion is to add the README file and unit tests.	Did both of these things. See previous responses

Response 3

Category	Reviewer's Comment	Reviewee's Response
Build	This project is with a client which required the members to sign an NDA. Thus building and setting this project up on a personal computer cannot be done. As I recall you did bring up the absence of a README during the code review, and while your application is private, I think a README explaining the structure of your project and why the project was created would be helpful for your client regardless. Once you complete your work you have no idea who will maintain the code or when it will be touched next and having some digital paper trail explaining why	Added a README explaining how to build and run the project, although it only applicable to the client, and only somewhat so, given the requirement for API certificates, it being an in-house application, and it being written in the same Framework and with similar dependencies as all of the client's other projects.

	the project was done in the first place would be a good idea to prevent it from getting shelved and collecting dust.	
Legibility	The code is extremely legible and clean. The directory structure of your project is also well organized and easy to follow. I can intuitively navigate the project and know where to look for code relating to database queries, cache management, and application views. Functions were kept short, succinct, and properly-scoped. Code formatting is also consistently structured throughout each file.	Thank you.
Implementation	Implementation looks great. I would like to commend your team for using the right tools in the right places such as Entity Framework as an ORM, Redis for web request caching, and Chart.js for responsive beautiful graphs on the front end. I think a nice addition to the project could be a dockerfile, there are many .NET core base images available that would suit your needs.	Thank you. Given that the client plans on running this project in-house and has no future plans on containerizing or virtualizing this project to be run on multiple platforms or instances of a single platform, we do not believe a dockerfile or other containerization technology is necessary at this stage of the project.
Maintainability	As previously mentioned, the code is legible and well-scoped. This lends to the codes maintainability. When a developer at Carrier is handed this they will have no difficulty picking it up quickly. My one critique regarding maintainability is that I was unable to locate unit tests, but I believe this is something that could be readily included.	Thank you. Unit testing is hard for UI applications. Nevertheless, we implemented some unit tests to test data projections and summaries. See the new "Testing" directory in the repository root. At the very least, it's a registry to encourage unit testing of more

		complex features should they be implemented in the future.
Requirements	It is difficult to say as an external spectator if all the requirements were met for this project though I am sure, seeing as you signed an NDA, your client will be able to verify this. I did watch the beta functionality video and listen to your code walkthrough but given that this application requires specific certificates to run I cannot personally verify with 100% accuracy that the requirements were met. By inspecting the code however it appears that the requirements were met and some. It looks like you can fetch key and lock data by user and your front end charts are gorgeous. I am sure your client will be very pleased with the final result.	Thank you. We can verify with 100% certainty that the requirements are met, and that it is capable of the features presented in the beta demonstration video and much more.
Other	Reviewer left this field empty	N/A

Response 4

Category	Reviewer's Comment	Reviewee's Response
Build	The group provided no readme to follow. The group mentioned that the build was not possible since we did not have the certificates to actually run and build the code, but this is still no exception to not have a readme document. In the presentation it was asked why there was not a readme and the group mentioned that their client knew how to build the product, but this is not a valid	Added a README explaining how to build and run the project, although it only applicable to the client, and only somewhat so, given the requirement for API certificates, it being an in-house application, and it being written in the same Framework and with similar dependencies as all of the client's other projects.

	reason to not have one. A readme serves as a reference and guide for getting dependencies and building. The employer may know how to build the code for now, but in a year's time that may change or they may want to onboard a new member. The readme is meant to streamline the process of onboarding and running code. If the client cannot build the code at a later date then they may need to reach back out to members of this team slowing down the process of their work flow. Please add a well- documented readme for professionalism.	
Legibility	This group has implemented very legible code. All the sections are appropriately commented, and variable names are descriptive and consistent. The group used camel casing for all their variables which makes the code look like it was written by one person. They use a good amount of proper object-oriented principles with polymorphism. The group talked about the model control structure which is currently the most accepted formatting for web development applications like theirs. I would say that this fact contributes to a good flow.	Thank you.
Implementation	I do not have any C# programming experience, which is a bulk of their code but, I did wonder why they did not use JavaScript instead. The group mentioned that they are using a library called chart.js for creating their graphics which I have a little of experience with, it seems like	This project is a "big data" project. It involves processing, caching, and summarizing very large amounts of data. Implementing the backend in an interpreted language run within a userland runtime environment, like Node, would

	moving between C# and the JavaScript library could be a little cumbersome. There are also a lot of especially useful JavaScript frameworks that could apply all of their techniques. For example, ReactJS allows for easy component building and also could then use SQLite for their offline database storage. I think that this is merely just an alternate route they could have taken. C# is a very acceptable implementation and I think the decision comes down to comfortability for the developers, so there is no real reason to penalize them.	result in significant performance hits. Secondly, moving between JavaScript and C# is not all that difficult; C#'s razor pages rendering engine supports C# embedded into HTML. Passing simple fields between a C# POCO model and a JavaScript environment is as simple as embedding it in the variable assignment. Complex objects can be passed by serializing and deserializing between JSON. This is how the chart models are provided to the JavaScript chart rendering engine. ViewComponents can load external scripts and pass information to them through global handlers and a queue of handler IDs with no chance of race conditions. Lastly, our client required the project to be built in the ASP.NET framework, so even if a JavaScript backend was to be considered a relevant option, it would still be a moot point.
Maintainability	No there are not unit tests however unit testing a web application is quite a challenging process and really can be a big undertaking. The group would likely have to sacrifice meeting some of their requirements if they developed in a test driven way. I think that meeting the requirements for them was more important and appropriately so.	We agree that unit testing is hard for UI applications. Nevertheless, we implemented some unit tests to test data projections and summaries. See the new "Testing" directory in the repository root. At the very least, it's a registry to encourage unit testing of more complex

		features should they be implemented in the future.
Requirements	Yes! The group even mentioned that they are starting on their stretch goals now. They have completed all that was asked and are still working on making it better. I will say that the goals the client set for them seemed quite low so I was glad to hear that they are taking on the client's stretch goals as this seems like it would be a more fitting amount of work for a group of 3. That being said though doing things quickly versus doing them right will yield two different results. Just meeting the goals doesn't always mean that the delivered product is good. I think what this group has showed off is good and even if the stretch goals are not met they have produced a professional product.	Thank you.
Other	The group mentioned that a side bar is a static item in their presentation. I would say this is not the case. Not every page needs a sidebar and pages can change the content of the sidebar depending on the main data presented in the page. Don't create unnecessary components on pages that don't need them. This is in other because I did not see this issue in their code or built website, just merely in their slide presentation.	The side bar can be left out using a separate layout. It is included in the default shared layout because it should be present in almost every page of the application. If the need to remove it in a particular page arises, a separate layout will be created.

Response 5

Category	Reviewer's Comment	Reviewee's Response

Build	I can get access to your GitHub repository. However, there is no Readme.md file in your GitHub repository. So, I don't really know how to run your program.	Added a README explaining how to build and run the project, although it only applicable to the client, and only somewhat so, given the requirement for API certificates, it being an in-house application, and it being written in the same Framework and with similar dependencies as all of the client's other projects.
Legibility	I read some code and functions. I think the code is good. The code is clean and has no redundancy. The code is modular. The structure of the code relative to each other is very clear. The relationship between each file is also very clear. There are folders like view, controllers, and models. So, the project has good legibility.	Thank you.
Implementation	From the videos that the group showed. I think implementation is very goods. The implementation is very smooth.	Thank you.
Maintainability	I think your group need more comments on each function. That helps to do maintainability. I think you guys need comments to describe the function.	We did notice that a few classes were left fairly undocumented. We went back through every source file and made sure to at least include a description of each class and its purpose. In cases where the code was not self-explanatory, additional in-line comments were provided as well. Special attention was paid to the Charting/Domain directory and its included source code files, given that models and business logic tend to be less self-explanatory than views or controllers.

Requirements	I check your Requirements document. I think you guys have met the requirements. Maybe your guys need to improve something, but I think your guys have good implementation and met the requirements.	Thank you.
Other	The reviewer left this field empty	N/A