# Feature: Car News Center

**Feature Developer:** Vi Nguyen

**Date Review Submitted:** 4/9/2024

**Peer Reviewer:** Rainier Getuaban

**Date Review Completed:** 4/21/2024

## Major Positives

1. There were some appropriate fail scenarios for database and API failures.
2. Fail scenarios included specific tables in the database.

## Major Negatives

1. SQL Statements
    a. For CNC-1, your sql statement can use a distinct constraint so that there is only one result per combination of make, model, and year. For example, if someone owns two 1994 Honda Civics, the current sql statement would return two rows containing the same information. One of the two rows would become redundant, making the payload of the database call larger than needed. This problem can also lead into another problem in the API call that you use to get the news articles, where you would request for the same information twice.
2. Fail Scenarios
    a. Incomplete or missing fail scenarios.
3. Clarification on where values are retrieved from
    a. For CNC-2, you state “Requests information from API using the make, model, and year of the vehicle” but never mention how you get the make, model, and year of the vehicle. If you need to get the information from the database, then denote that. If you get the information from the session storage, denote that.
4. Input Data Types

a. The goal for CNC-7 is to retrieve every alert on a single page. You pass in a vehicle's VIN into the manager to get all the alerts for a user. A vehicle's VIN would only be able to retrieve all alerts from a user if they join onto the UserAccount table, then onto the Notifications table. This is really inefficient, especially when you can easily get the UserAccount using their tokens from the front end. Not only that, but there is no parameter for the VIN from the browser to the Entry Point.

## Unmet Requirements

1. CNC-1 Fail Scenario

   There is no fail scenario for if no news articles are retrieved from the API call. Since the API is another point of failure that is out of our control, it is important to catch if we cannot retrieve information from the API.
2. CNC-3, CNC-4, CNC-5, and CNC-6 do not meet the functionality written in the BRD.

   The purpose of these user stories are to retrieve and send notifications to a user when new alerts are made. The current design for these user stories only write to the database and do not send the email and/or sms message to the user. It should be the job of the VehicleMarketplace, ServiceLog, and UserAdministration to create the alerts and Car News Center should be the one to alert the user.

## Design Recommendations

1. Clarify where you get values from. Denote where you get Make, Model, and Year inputs for CNC-2

2. Name methods so that a developer can at least guess the functionality of the method. Method naming should be descriptive enough to at least sort of tell what they will do. Method names that only have a noun are not descriptive enough.
3. Use the correct HTTP methods in your method signatures. Since a user must be authenticated and authorized to view your feature, the HTTP methods for your endpoints should be POST.
4. For CNC-7, the purpose of the user story is to have a single page to view all alerts for a user. Pass in the UserAccount as parameters into the manager and service layer and take what you need from that to get a user's alerts/notifications.
5. Try to add pagination into CNC-7 so that upon loading the view, they only see new and relevant alerts rather than everything all at once. It helps with both user experience seeing only the ones relevant and database execution. Though it will take multiple database executions retrieving the earliest alert, each package will be smaller in size.
6. Reevaluate the design of CNC-3, CNC-4, CNC-5, and CNC-6.

## Test Recommendations

1. Test scenarios where you feed in bad data that will make our database will throw an error.
2. Test scenarios where you feed in bad data that will be allowed in our database but will make the APIs used throw invalid errors or no data back.
3. Test scenarios where the data will be thrown out before even being turned into an sql statement.