CarCool

By: Mikaila Hirschler, Temisan Iwere, Jeremy Gralla, and Salman Fazal

Our app, CarCool, utilizes a 3-tier architecture, with the database being an SQL database, and the back-end (being the express-server.js file) accessing it, the middleware (the script.js file) processing it, and the front-end (the index.html file) displaying it.

Our web app has a main page on which users can sign in, sign up, or even look at listings without being signed in (they aren't permitted to select a car unless they sign in, however). After a user signs in, they will automatically be redirected to their own profile page, which displays their rating/recommendation (made by other users), the comments made by other users, and their car information (if they are a CarRuler – someone looking to rent out their car). From this page, they have the option (a button) to change their password, and also a button to search for listings or create a listing. When a user searches for a listing, they will be asked for their current location, and a specified radial distance they want the car they rent to be within. After they submit the request, they will be taken to a page of listings of cars within the specified distance, sorted with the highest rating/recommendation at the top. They can select a user and will then be redirected to that user's profile page, where they can look at their information more in depth and request to use their car. A notification will be sent to the user with the car telling them that a user wants to rent their car and a link to the renter's profile will be provided. If they are willing to give them the car, the can press the button to accept the offer, and they will be able to message each other about when/where to meet, etc. Once the car is returned, the owner of the car will press a button saying that it was returned, and both users will be sent a form to rate/recommend each other and give an optional comment.

Some features we did not have the chance to implement are:

- The user can't upload a profile picture or pictures of their car(s). To do this, they would provide the source of their picture and we would insert it into a table in the database, inserting it whenever the profile is viewed
- The messaging system would be based on having a table of messages composed of tuples of 3 components, the sending user's username, the date/time the message was sent, and the message content itself.
- An email confirmation of the user's account which would send a message to the email that the user specified in their sign up form for CarCooler
- Sending notifications to users when someone wants to rent their car/accepts a rental offer, or when we need them to rate other users.

We have not tested our app as we did not have time to do so. We’re hoping this will not affect our report mark since there is a separate marking section for the testing.

We have hashed our users’ passwords when putting them in the database as a form of security. We did not implement a second form or test the security as we did not have time to do so. We’re hoping this will not affect our report mark since there is a separate marking section for the security.

We did not feel the need to improve the performance, and did not have time to test this. We’re hoping this will not affect our report mark since there is a separate marking section for the performance testing.

One enhancement we made was implementing the radial distance feature when users search for cars. We had to use a Google API to do this and it makes searching easier for the user instead of searching through all possible cars in any location.