Project Title: Car Tinder

Summary: Car Tinder is an interactive mobile/web application designed to simplify the used car buying experience. Buying a car can be overwhelming due to fluctuating prices, inconsistent vehicle histories, varying fuel economy, and reliability concerns. Our app transforms the process into a fun and engaging experience by allowing users to swipe through personalized car recommendations based on their preference. It's the same high stakes and high financial implications as Tinder, but for cars!!

Description:
Purchasing a car is understandably a very hard decision with the many factors that must be taken into account. With our app, we are looking to create a uniform platform for information on all these factors that a prospective buyer may seek out when in the market for a car. Users will begin by creating their profiles, personalized with features they prioritize in a prospective car. The main functionality of the app comes in the form of a stack of cars, each featuring notable metrics, the user will swipe on to indicate their interest level. Not only are we aiming to make information more accessible, we are looking to provide suggestions tailored to the user. Our algorithm will take the user preferences along with the swipe tendencies into account to choose which cars to prompt the user with, creating a personalized experience.

Technically challenging feature:

We plan to make the app resemble tinder, where users can swipe either right or left on a car if they like it or not. To accomplish this, we will use various js visualization libraries to assist with this, giving the user statistics and other relevant information about the car for them to make their decision. This dynamic visualization will include information such as price, mileage, model year, fuel efficiency, and possibly ratings from other users. Once the user makes a decision, we will store information about their preferences and develop an algorithm to show them better suggestions in the future. This means that each user will have a personalized account, showing different suggestions based on how someone interacts with the application.

Usefulness
Buying a new car is a stressful process for people, as it involves a very large financial decision, in addition to a large commitment. This app provides them a fun, less stressful way of coming to a decision for what car to purchase. Users are prompted initially with various filters that they fill out, and the cars that fit their requirements are shown to them in a fun, interactive way. This is significantly more useful than the alternative of searching multiple websites, and having to comb through results manually. There are similar websites out there; CarGurus, Cars.com, AutoTrader, and even craigslist all show cars to prospective buyers. What sets our app apart is the fact that none of those help you choose a car. They all have filters, but that's it. For the most part you need to know what you are looking for when buying a car on those sites. On our

website, we help people figure out what they are looking for in a fun and engaging way. Not everyone is a car expert, and we hope to make the car buying experience much simpler.

Realness

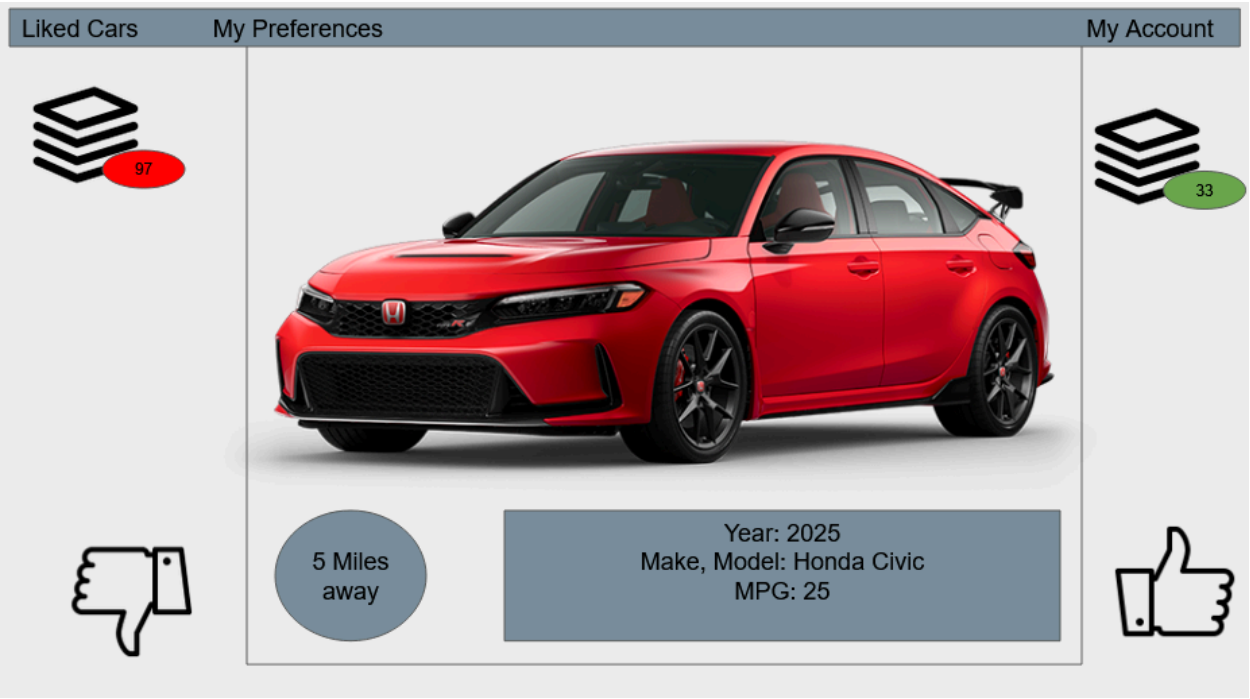
We will use several datasets to provide detailed information about different makes and models of cars. The Vehicle Fuel Economy dataset provides extremely detailed information about a wide range of cars with a focus on nuanced fuel economy statistics (annual fuel cost, tailpipe CO2 emissions, hours to charge an EV on 120V or 240V) with additional technical specifications (cylinder count, presence of turbochargers, etc.). The dataset is a CSV file with 38,000 rows and 81 columns, allowing our application to provide all the details a user could want. Our second dataset will be the Cars Datasets (2025) which has considerable overlap with the Vehicle Fuel Economy dataset but with the crucial addition of prices for new vehicles. This dataset is also a CSV File and has 1200 rows. To complement this we will also use the Used Car Price Prediction dataset, which contains records of individual sales of used cars. This rich dataset includes columns for accident history and whether it had a clean title, which will allow us to provide a distribution of prices to users interested in purchasing a used car. This dataset is also a CSV file with 4000 rows, although some of the entries refer to distinct purchases of cars with a given make and model.

Lastly, to provide engaging visuals we primarily use the cars196 dataset which contains 16,000 images split across 196 classes, which are at the level of make, model, year. The data is accessed with a python script, but a method of mapping images to class is provided. If this dataset proves insufficient we can use the 60,000+ Images of Cars dataset, which contains clearly labelled images. However, the images in this dataset are of lower quality than the cars196 dataset.

Functionality

1. List of functionality
    a. Users create profiles indicating features they are looking for in a car upon first accessing the application
    b. Once user profiles are complete, users are presented with a stack of vehicles one by one, and are prompted to either swipe left if they are not interested in the car, or right if they are interested
    c. Each prompted car will include key metrics for the user to make an educated decision when deciding to swipe on a car
    d. The app will contain a tab to see past cars users have expressed interest in, where they will be able to learn more about each vehicle and revisit the information as needed
    e. User are able to access their profiles to modify their preferences as what they desire changes

2. UI-Mockup



3. Project work distribution
    a. Prepare datasets -Brian, -Ryed
        i. Some datasets already have primary keys, some have none
        ii. Identify makes and models that appear in all or at least multiple datasets
        iii. Remove personally identifying information
    b. Think of what queries we need -Rohan
        i. What data do we need where?
        ii. What tables make sense
        iii. Schema design
    c. Building the recommendation system -Ryed
    d. Create endpoints for user actions -Josh
        i. Logging in/out
        ii. Liking a car
        iii. Removing a car from likes
    e. Complete service classes for endpoint functionality -Rohan
    f. Complete Repository classes (Interaction with db) -Brian
    g. Setting up hosting/basic backend infrastructure -Josh
    h. Create Frontend pages -Josh, Rohan
        i. Preference selector
        ii. Left or right page
        iii. Liked cars