The Auto Shop

R1

The Auto Shop is going to be a web-based service which allows users to both post their cars for sale and browse through cars for purchase. The sample dataset for this project will be created by hand using common knowledge and a few sample descriptions and images. The real dataset will be created through a program (will be accessible through the Git) using either a generative text API or some form of random but controlled generation. The administrators will be the Technology Operations team at The Auto Shop (just Araad Shams in this case), and the users will be any individual who has a car to sell or who needs to buy a car.

In specifics, this application will provide a user-to-user car selling service. Before making the purchase, the buyer will be able to see a description about the car, an image of the car, the company which made the car, the model name of the car, the make year of the car, the price of the car, contact information for the seller, and the odometer reading of the car. Additionally, the user will be able to see other cars made by the same company, other cars made in the same year, other cars near the same price range, and ratings/reviews of the seller. Additionally, the buyer will be able to see cars within a certain radius of his location. The seller will also be able to edit and delete their posts. The buyer will be able to review the seller provided that they have purchased from the seller.

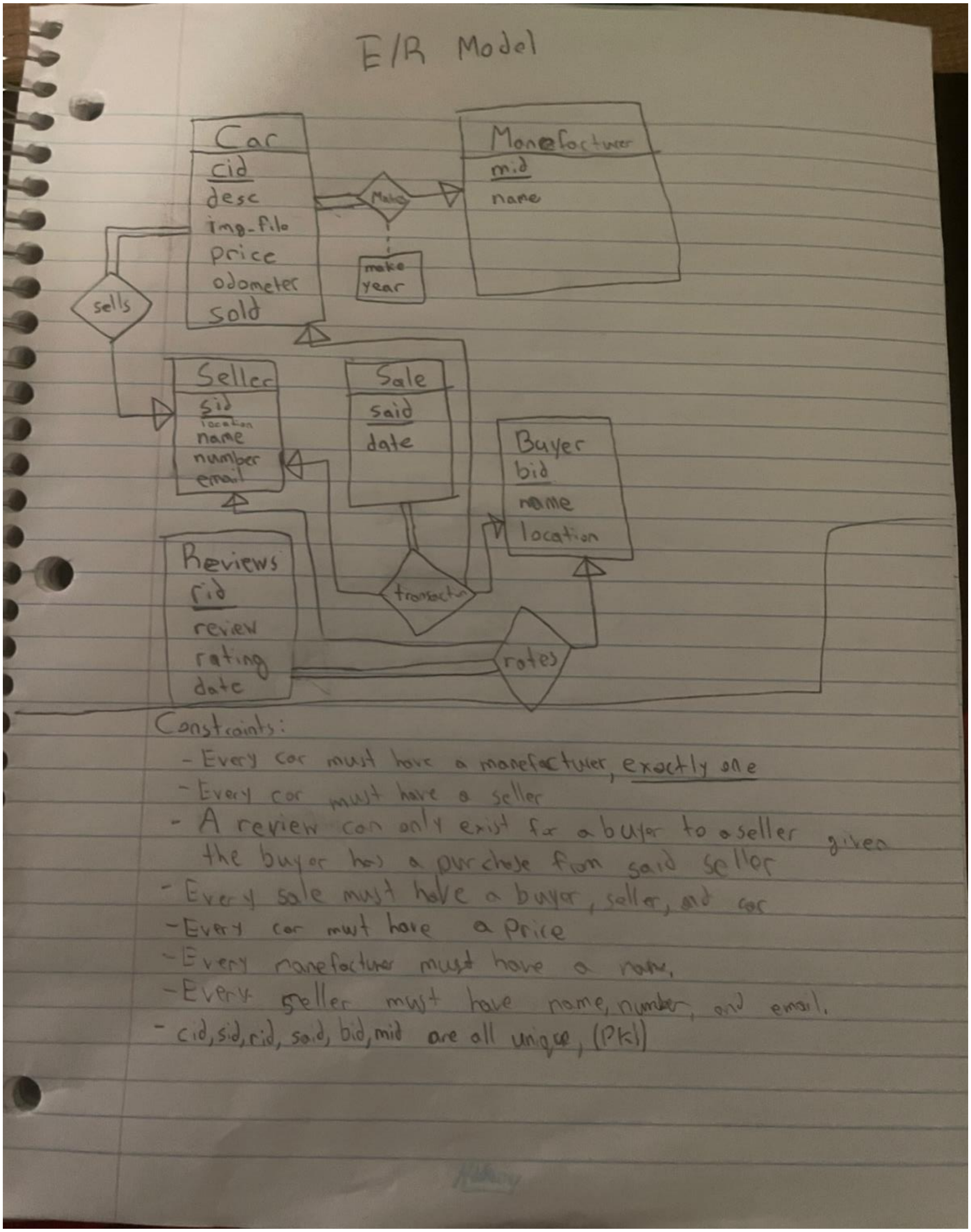
R2

The Auto Shop will be built using MySQL as the DBMS support for the backend, and an express server connecting it to a React application. React will essentially be the entirety of the front-end and MySQL + Node.js+ Express will compose the backend. Postman will be used to test the MySQL queries, and once working, they will be integrated into the React App. That is the entire Stack that will be used for developing The Auto Shop.

R3

For the sample data, it will be manually hand entered and the plan of the attack for the real data that will populate the database is to create a program that, either through some Generative Text processing or through some randomization algorithm created, gives us appropriate entities and tuples that can work together and populate the database. They will be created according to the schema shown in the next section.

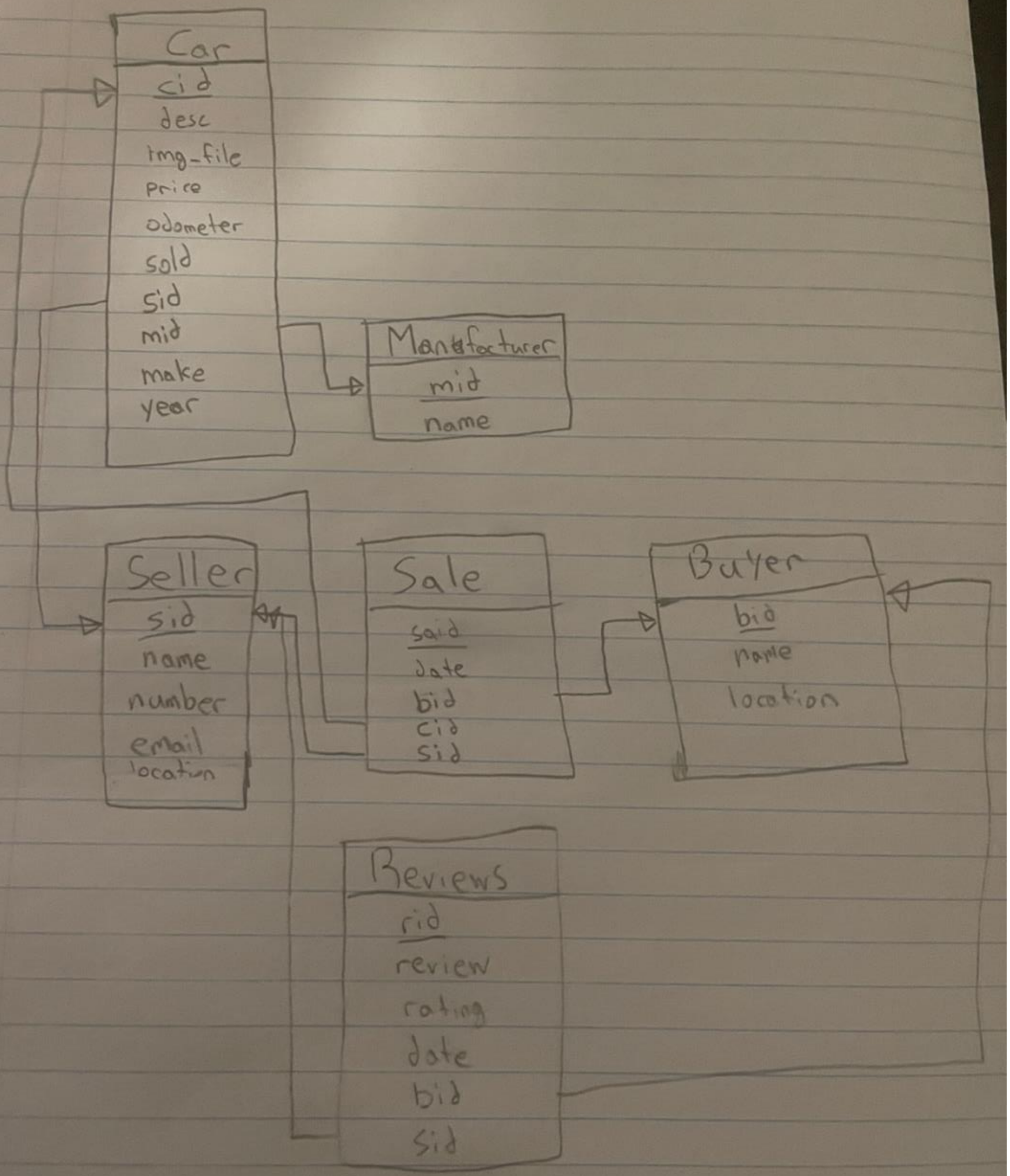
R5



Constraints:

- Every car must have a manefactuer, exactly one
- Every car must have a seller
- A review can only exist for a buyer to a seller given the buyer has a purchase from said seller
- Every sale must have a buyer, seller, and car
- Every car must have a price
- Every manefacturer must have a name.
- Every seller must have name, number, and email.
- cid, sid, rid, said, bid, mid are all unique, (PK)

Relational Data Model



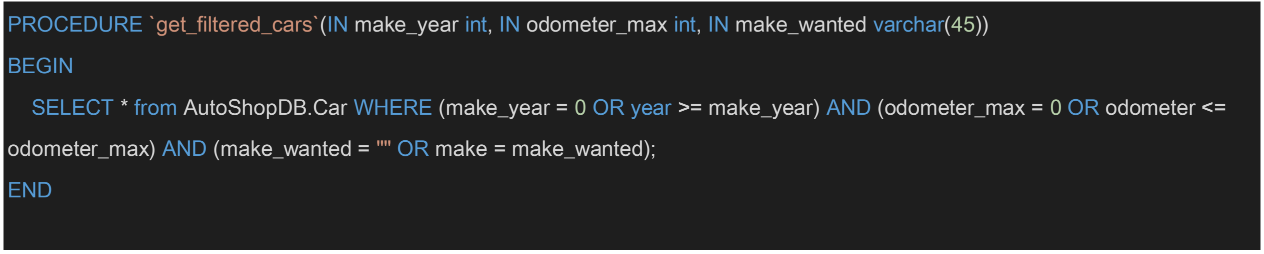
R6a – Interface

This feature will be a basic search filter. For instance, if a person who is looking for a car has certain requirements on the type of car he is looking for where it may be made in a certain year, have less than a certain number on the odometer, be made by a certain company, etc. In the main page, the user will have certain check boxes and text inputs below the search bar that let them choose the filters, and then upon hitting an apply button, the query will run and the page will update to show cars that meet their requirements.

R6b – SQL Template

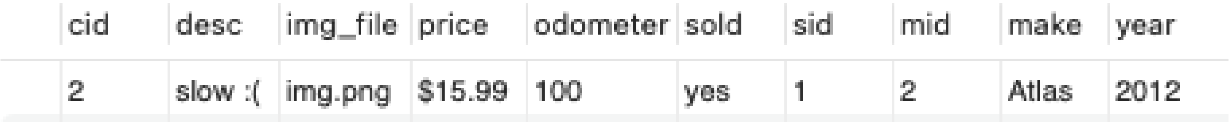
CALL AutoShopDB.get_filtered_cars(2012, 10000000, "Atlas");

This uses the stored procedure defined as below:



```
PROCEDURE `get_filtered_cars`(IN make_year int, IN odometer_max int, IN make_wanted varchar(45))
BEGIN
   SELECT * from AutoShopDB.Car WHERE (make_year = 0 OR year >= make_year) AND (odometer_max = 0 OR odometer <=
odometer_max) AND (make_wanted = "" OR make = make_wanted);
END
```

R6b – Generated Output



	cid	desc	img_file	price	odometer	sold	sid	mid	make	year
	2	slow :(	img.png	$15.99	100	yes	1	2	Atlas	2012

R6b – Generated Output Text

```
# cid  desc  img_file      price  odometer   sold   sid    mid    make year
2      slow :(      img.png      $15.99100    yes    1      2      Atlas  2012
```

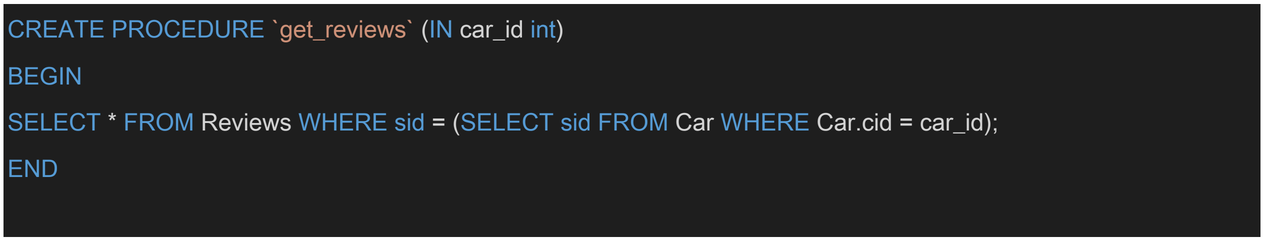
R7a – Interface

This feature will be a basic reviews selection on the seller. Based on the car that the person is looking at, on the same page, reviews about the seller will be queried and shown with date and timestamps at the bottom of the page. The user's rating will also be shown next to their profile name. This will allow the buyer to make a more informed purchase and they will know how trustworthy the person they are dealing with is

R7b – SQL Template

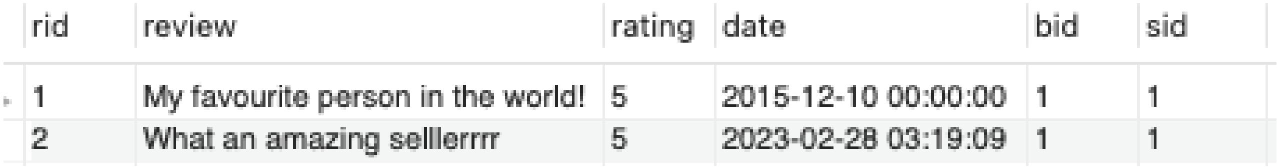
CALL AutoShopDB.get_reviews(1);

This uses the stored procedure defined as below:



```
CREATE PROCEDURE `get_reviews` (IN car_id int)
BEGIN
SELECT * FROM Reviews WHERE sid = (SELECT sid FROM Car WHERE Car.cid = car_id);
END
```

R7b – Generated Output



rid	review	rating	date	bid	sid
1	My favourite person in the world!	5	2015-12-10 00:00:00	1	1
2	What an amazing selllerrrr	5	2023-02-28 03:19:09	1	1

R7b – Generated Output Text

```
# rid   review          rating date    bid     sid
1       My favourite person in the world!       5       2015-12-10 00:00:00 1       1
2       What an amazing selllerrrr      5       2023-02-28 03:19:091        1
```

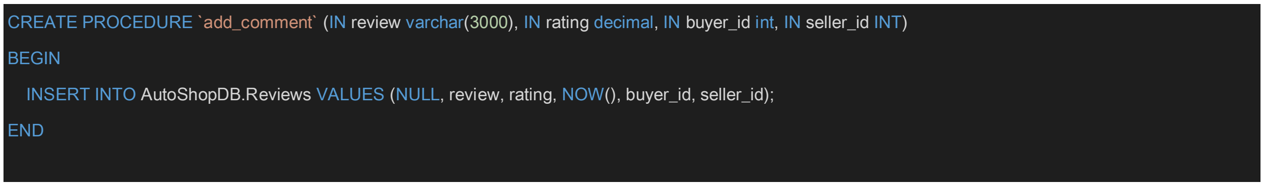
R8a – Interface

This feature will allow a buyer to buy a car from a seller. Once the car has been bought, the buyer will then be allowed to comment and write a review about the seller by navigating to the seller's profile page and, once recognized that a valid transaction has been made between this buyer and seller, the buyer will be able to leave a review about the seller.

R8b – SQL Template

CALL AutoShopDB.add_comment("What an amazing selllerrrr", 4.9, 1, 1);

This uses the stored procedure defined as below:



```
CREATE PROCEDURE `add_comment` (IN review varchar(3000), IN rating decimal, IN buyer_id int, IN seller_id INT)
BEGIN
   INSERT INTO AutoShopDB.Reviews VALUES (NULL, review, rating, NOW(), buyer_id, seller_id);
END
```

R8b – Generated Output

N/A, Insert Statement (will show some confirmation on the website UI though)

R8b – Generated Output Text

N/A, Insert Statement (will show some confirmation on the website UI though)

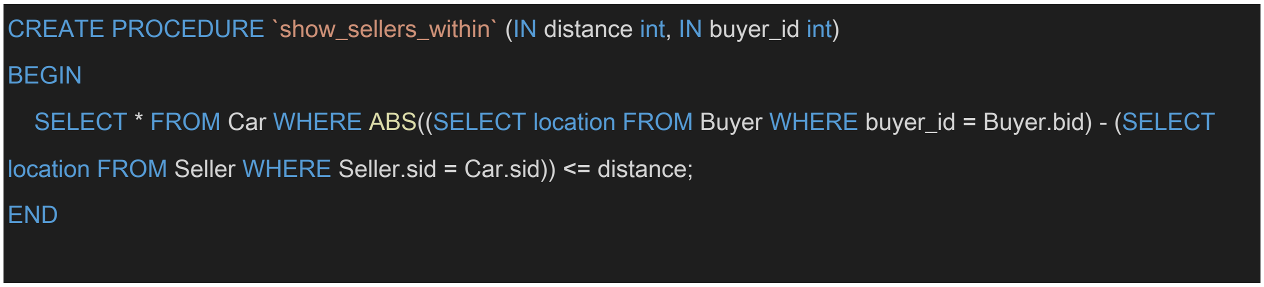
R9a – Interface

This feature will allow users to only show sellers who are in a certain radius of their location. Using Long, Lat coordinates (determined at time of sign up), the buyer will be able to, on the main page, define a radius in which they want to search in, and then the SQL Query will remove all results that are outside of said radius.

R9b – SQL Template

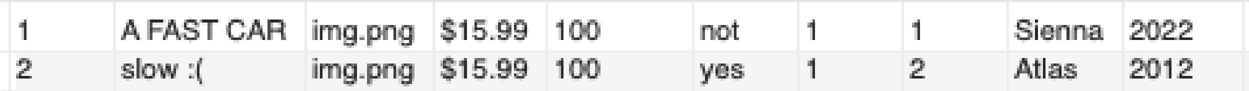
CALL AutoShopDB.show_sellers_within(100, 1);

This uses the stored procedure defined as below:



```
CREATE PROCEDURE `show_sellers_within` (IN distance int, IN buyer_id int)
BEGIN
   SELECT * FROM Car WHERE ABS((SELECT location FROM Buyer WHERE buyer_id = Buyer.bid) - (SELECT
location FROM Seller WHERE Seller.sid = Car.sid)) <= distance;
END
```

R9b – Generated Output



1	A FAST CAR	img.png	$15.99	100	not	1	1	Sienna	2022
2	slow :(	img.png	$15.99	100	yes	1	2	Atlas	2012

R9b – Generated Output Text

```
# cid  desc  img_file       price  odometer    sold   sid      mid     make year
1         A FAST CARimg.png        $15.99100    not    1         1        Sienna         2022
2         slow :(         img.png        $15.99100    yes    1         2        Atlas  2012
```

R17

This section does not pertain as I am working alone :D