

Data Analysis on Cars

Ganguru, Rohith Sai, Gorantla, Manoj, Yarlagadda, Akhil Babu, Makkena,
Ajay, Chundu, Avinash, Smith, Carter

Northwest Missouri State University, Maryville MO 64468, USA

1 Introduction

There are many theories about how the market decides the prices of used cars offered for sale by private parties, particularly in comparison to new cars. According to one theory, new car dealers can generate more demand by making a greater effort to sell a car. This dataset describes about the Cars and the make model of the car manufactured. Full detailed information about the car is included in the data set for example the Mileage of the car and body type and the estimated cost of the used car according to the market.

2 A detailed explanation of your proposed methodology with respect to 5 Vs

Volume: Big data can be used to describe a set of data that is sufficiently massive. However, the definition of big data is subjective and subject to change depending on the market's supply of computer power. Since our dataset is related about cars. How many cars are being buy/sold in particular time and based on the specific brands.

Velocity: For businesses who require their data to flow fast so that it is available when needed to make the best business decisions, this is a crucial factor. Since there will be many models of cars in the dataset it contains the data about mileage of the car that have been driven where we need to retrieve the results soon based on the type users search.

Variety: Data might originate both inside and outside of an organization. The standardization and distribution of all the data being gathered pose a problem in terms of variety. Since this dataset might contain unstructured data like the NA, NULL, NO DATA FIELDS of the particular film.

Veracity: Veracity, overall, refers to the level of trust there is in the collected data. since the data format overall, all cars are going to be the same we may not notice veracity.

Value: It is necessary to be able to extract value from big data because the value of big data greatly depends on the insights that can be obtained from them. The value of the data set is typical going cost of the cars, that actually depends on the model type, the condition of the car and the mileage on the car, by considering all these aspects the value of the car can be decided in the market.

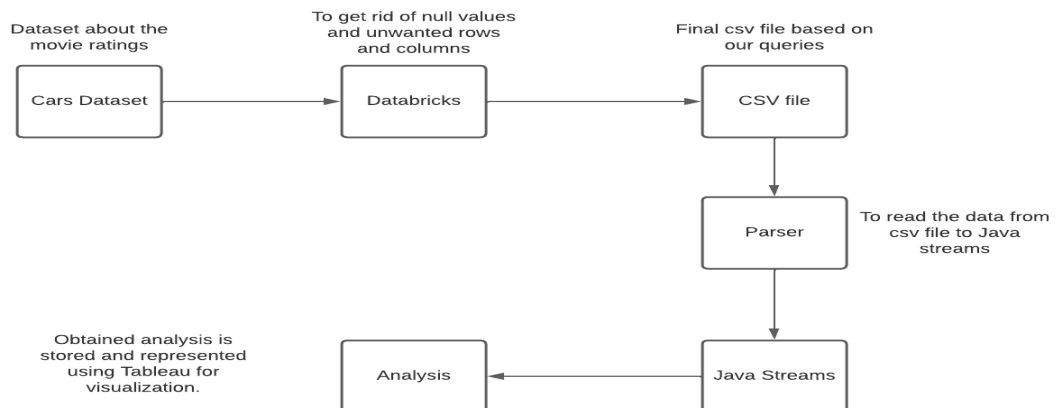
3 Goals of your project

- 1. List all Maruthi Brand With Model Alto?
- 2. What are the car Brands Manufactured between 2011 to 2015?
- 3. What are the brands which run on diesel fuel?
- 4. Specify all the Car Brands, Models, years, and prices between 3L to 6L?
- 5. List Brand with Model and year which are in the location Delhi?
- 6. List out all the brands, models, and counts with BMW?
- 7. List SUV body type Brand and Models?
- 8. List all first-owner car brands with Models?
- 9. List the car model, Brand, Year, and Km that are driven less than 10000 kilometers.
- 10. Name the honda cars with the model, Body type, and year manufactured after 2015.

4 Tools and Technologies you plan to use:

- Data Bricks
- IO Streams
- Java
- Spring Boot
- Overleaf
- To segregate the data we use data bricks i.e, to remove the null values and duplicate data. Also, we will remove unwanted rows and columns from that static dataset by developing queries. We will take the final output CSV file and using IO streams we will do some analysis and display the data in text format.

5 A detailed methodology in the form of a block diagram



6 A thorough and detailed explanation of all the steps of your implementation

- Initially we need to find the data set to display the set of goals depend on the data.
- By using the data bricks tool to clean the unstructured data and extracting the csv file from the data bricks.
- After that created a project with name BigDataFinalProject using spring boot Application.
- After creating project we need to create a package, then we need to import the csv data set file into resource folder.
- We need to create a main class and fetch the csv file using "path" library, then we created an data object class named as Car and we need to initiate the required attributes with corresponding data types and using those attributes created a constructor.
- In main class,from the data set the each row will be read and inserted in the Car object and store the object into the list by using collections and Map.
- Using the Car object List we are created the Goal Analysis by using the IOStreams we are analysed the result of the goal.
- The result output is storing into the text file by using PrintWritable Library.
- we reading the time stamp while executing the each goal and calculating the time elapsed to executing the goal.

7 Discussion of your results of each goal in detail by providing screenshots

7.1 List all Maruthi Brand With Model Alto?

```

43
44 private static void Goal1(List<Car> carsList) throws IOException {
45     long startTime = System.nanoTime();
46     PrintWriter pw = new PrintWriter(Files.newBufferedWriter(Paths.get("src/main/resources/goal1.txt")));
47     carsList.stream()
48         .filter(car -> car.getBrand().equalsIgnoreCase("Maruti") && car.getModel().equalsIgnoreCase("Alto"))
49         .map(Car::toString)
50         .forEach(pw::println);
51     pw.close();
52     long elapsedTime = System.nanoTime() - startTime;
53     System.out.println("\n\nTotal execution time for Goal1 in milli_seconds: " + elapsedTime / 1000000);
54
55 }
56

```

Console: <terminated> CSVFileStream [Java Application] C:\Program Files\Eclipse Adoptium\jdk-17.0.2.8-hotspot\bin\javaw.exe (Dec 3, 2022, 10:40:59 PM - 10:41:00 PM)

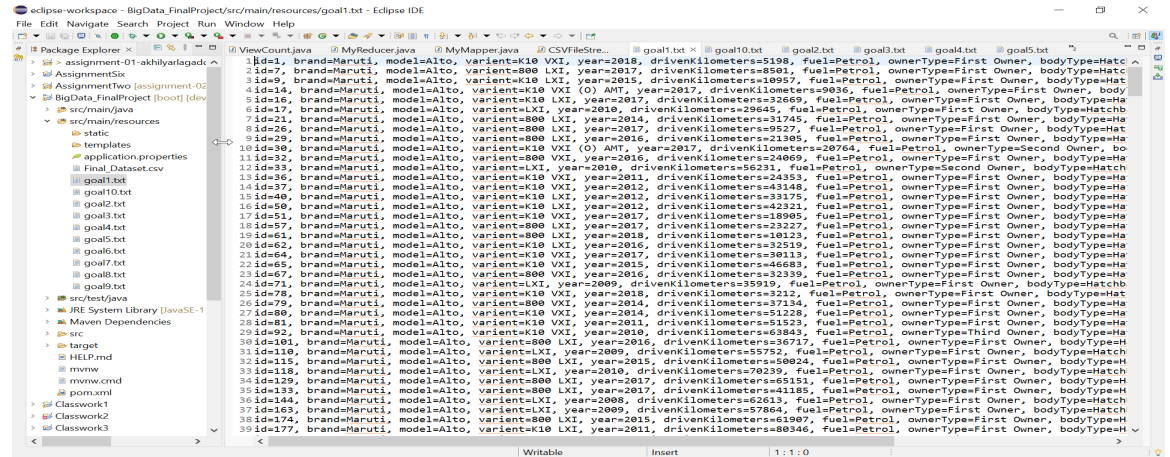
Total execution time for Goal1 in milli_seconds: 22

- To display the cars based on the specific brand like "Maruthi" and specific model "Alto".

4 Authors Suppressed Due to Excessive Length

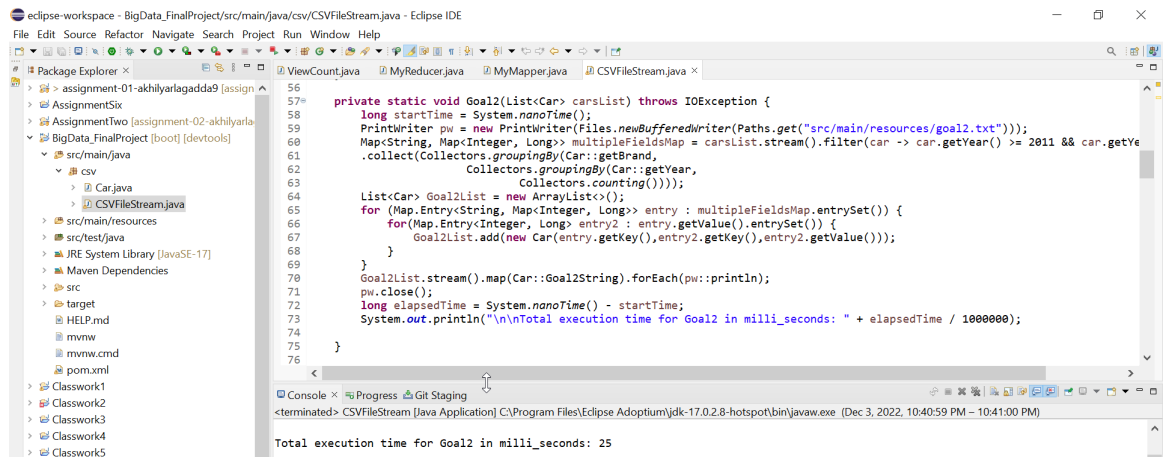
- No difficulty find while we executing this goal.

OUTPUT 1



```
1 id=1, brand=Maruti, model=Alto, variant=K10 VXi, year=2018, drivenKilometers=5198, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
2 id=7, brand=Maruti, model=Alto, variant=800 LXI, year=2017, drivenKilometers=8581, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
3 id=9, brand=Maruti, model=Alto, variant=K10 LXI, year=2015, drivenKilometers=18957, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
4 id=14, brand=Maruti, model=Alto, variant=K10 VXi (O) AMT, year=2017, drivenKilometers=9836, fuel=Petrol, ownerType=First Owner, body
5 id=16, brand=Maruti, model=Alto, variant=K10 LXI, year=2017, drivenKilometers=32669, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
6 id=17, brand=Maruti, model=Alto, variant=LXI, year=2010, drivenKilometers=29645, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
7 id=21, brand=Maruti, model=Alto, variant=800 LXI, year=2014, drivenKilometers=31745, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
8 id=26, brand=Maruti, model=Alto, variant=800 LXI, year=2017, drivenKilometers=9527, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
9 id=28, brand=Maruti, model=Alto, variant=800 LXI, year=2016, drivenKilometers=21385, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
10 id=30, brand=Maruti, model=Alto, variant=K10 VXi (O) AMT, year=2017, drivenKilometers=20764, fuel=Petrol, ownerType=Second Owner, bo
11 id=32, brand=Maruti, model=Alto, variant=800 VXi, year=2016, drivenKilometers=24069, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
12 id=33, brand=Maruti, model=Alto, variant=LXI, year=2010, drivenKilometers=56231, fuel=Petrol, ownerType=Second Owner, bodyType=Hatch
13 id=36, brand=Maruti, model=Alto, variant=K10 VXi, year=2011, drivenKilometers=24353, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
14 id=37, brand=Maruti, model=Alto, variant=K10 VXi, year=2012, drivenKilometers=43148, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
15 id=40, brand=Maruti, model=Alto, variant=K10 LXI, year=2012, drivenKilometers=33175, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
16 id=46, brand=Maruti, model=Alto, variant=K10 LXI, year=2012, drivenKilometers=42321, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
17 id=51, brand=Maruti, model=Alto, variant=K10 VXi, year=2017, drivenKilometers=18905, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
18 id=57, brand=Maruti, model=Alto, variant=800 LXI, year=2017, drivenKilometers=23227, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
19 id=61, brand=Maruti, model=Alto, variant=800 LXI, year=2018, drivenKilometers=18123, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
20 id=62, brand=Maruti, model=Alto, variant=K10 LXI, year=2016, drivenKilometers=32519, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
21 id=64, brand=Maruti, model=Alto, variant=K10 VXi, year=2017, drivenKilometers=38113, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
22 id=65, brand=Maruti, model=Alto, variant=K10 VXi, year=2015, drivenKilometers=46683, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
23 id=67, brand=Maruti, model=Alto, variant=800 VXi, year=2016, drivenKilometers=32339, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
24 id=71, brand=Maruti, model=Alto, variant=LXI, year=2009, drivenKilometers=35919, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
25 id=78, brand=Maruti, model=Alto, variant=K10 VXi, year=2018, drivenKilometers=3212, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
26 id=79, brand=Maruti, model=Alto, variant=800 VXi, year=2009, drivenKilometers=37134, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
27 id=80, brand=Maruti, model=Alto, variant=K10 LXI, year=2014, drivenKilometers=51226, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
28 id=81, brand=Maruti, model=Alto, variant=K10 VXi, year=2011, drivenKilometers=51523, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
29 id=92, brand=Maruti, model=Alto, variant=K10 VXi, year=2015, drivenKilometers=61843, fuel=Petrol, ownerType=Third Owner, bodyType=Hatch
30 id=101, brand=Maruti, model=Alto, variant=800 LXI, year=2016, drivenKilometers=36717, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
31 id=110, brand=Maruti, model=Alto, variant=LXI, year=2009, drivenKilometers=55752, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
32 id=115, brand=Maruti, model=Alto, variant=800 LXI, year=2015, drivenKilometers=80024, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
33 id=118, brand=Maruti, model=Alto, variant=LXI, year=2010, drivenKilometers=70239, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
34 id=129, brand=Maruti, model=Alto, variant=K10 LXI, year=2017, drivenKilometers=65151, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
35 id=133, brand=Maruti, model=Alto, variant=800 LXI, year=2017, drivenKilometers=44185, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
36 id=144, brand=Maruti, model=Alto, variant=LXI, year=2008, drivenKilometers=62613, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
37 id=163, brand=Maruti, model=Alto, variant=LXI, year=2009, drivenKilometers=57864, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
38 id=174, brand=Maruti, model=Alto, variant=800 LXI, year=2015, drivenKilometers=69197, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
39 id=177, brand=Maruti, model=Alto, variant=K10 LXI, year=2011, drivenKilometers=80346, fuel=Petrol, ownerType=First Owner, bodyType=Hatch
```

7.2 What are the car Brands Manufactured between 2011 to 2015?



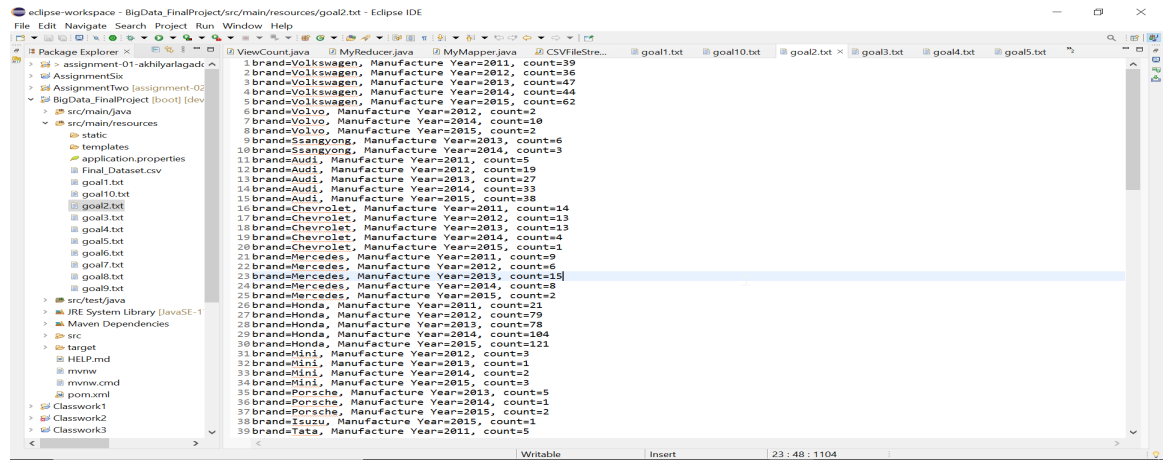
```
56 private static void Goal2(List<Car> carsList) throws IOException {
57     long startTime = System.nanoTime();
58     PrintWriter pw = new PrintWriter(Files.newBufferedWriter(Paths.get("src/main/resources/goal2.txt")));
59     Map<String, Map<Integer, Long>> multipleFieldsMap = carsList.stream().filter(car -> car.getYear() == 2011 && car.getYear() == 2015)
60     .collect(Collectors.groupingBy(Car::getBrand,
61     Collectors.groupingBy(Car::getYear,
62     Collectors.counting())));
63     List<Car> Goal2List = new ArrayList<>();
64     for (Map.Entry<String, Map<Integer, Long>> entry : multipleFieldsMap.entrySet()) {
65         for (Map.Entry<Integer, Long> entry2 : entry.getValue().entrySet()) {
66             Goal2List.add(new Car(entry.getKey(), entry2.getKey(), entry2.getValue()));
67         }
68     }
69     Goal2List.stream().map(Car::Goal2String).forEach(pw::println);
70     pw.close();
71     long elapsedTime = System.nanoTime() - startTime;
72     System.out.println("\n\nTotal execution time for Goal2 in milli_seconds: " + elapsedTime / 1000000);
73 }
74
75
76
```

Console Output:

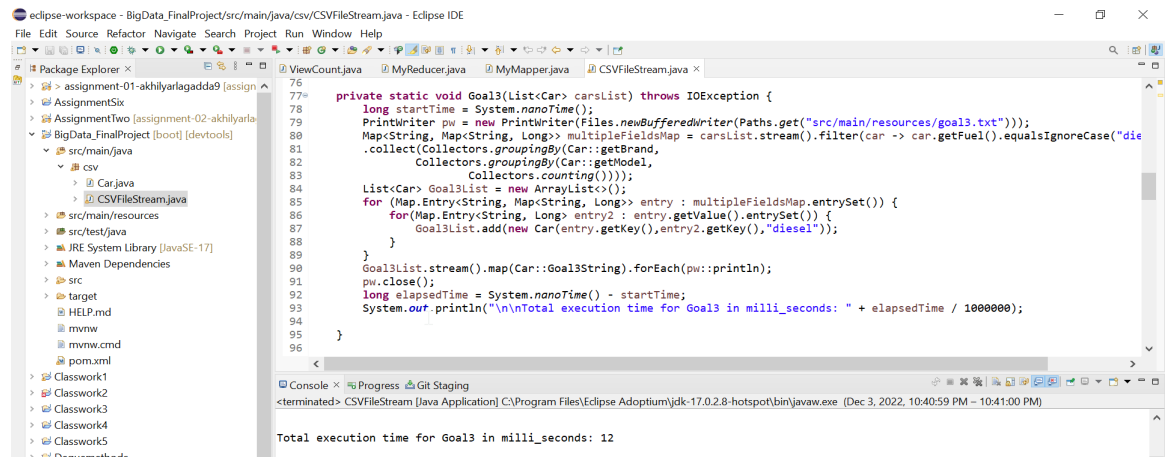
```
<terminated> CSVFileStream [Java Application] C:\Program Files\Eclipse Adoptium\jdk-17.0.2-hotspot\bin\javaw.exe (Dec 3, 2022, 10:40:59 PM - 10:41:00 PM)
Total execution time for Goal2 in milli_seconds: 25
```

- In this goal we are displaying the cars that are manufactured from the year 2011 to 2015.
- No difficulty find while we executing this goal.

OUTPUT 2



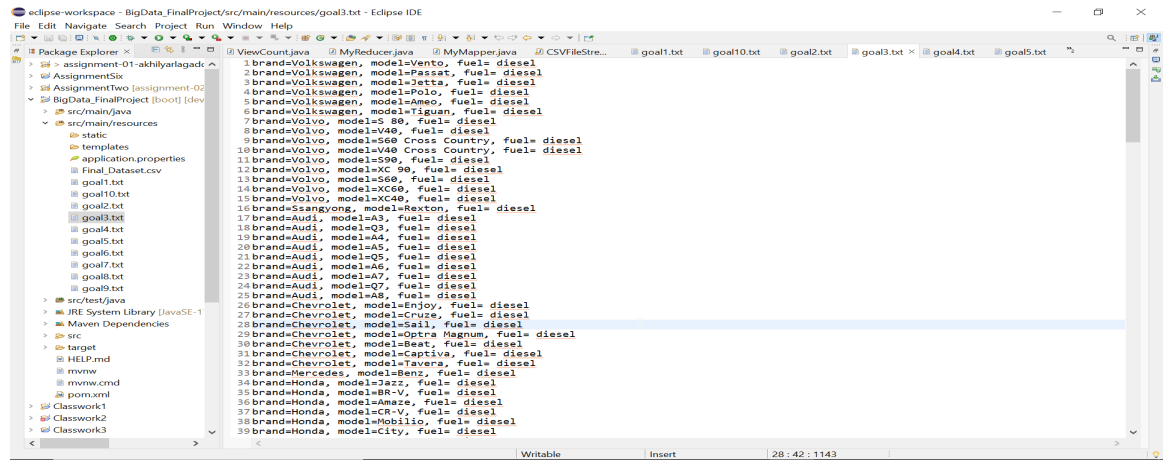
7.3 What are the brands which run on diesel fuel?



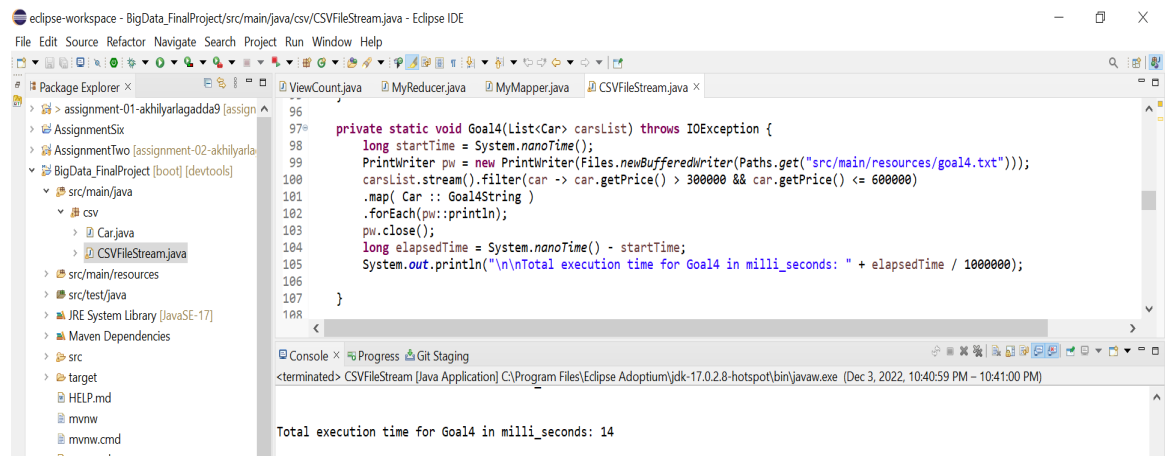
- Here we are presenting the cars that runs on using the fuel type either it is using diesel or fuel.
- No difficulty find while we executing this goal.

OUTPUT 3

6 Authors Suppressed Due to Excessive Length

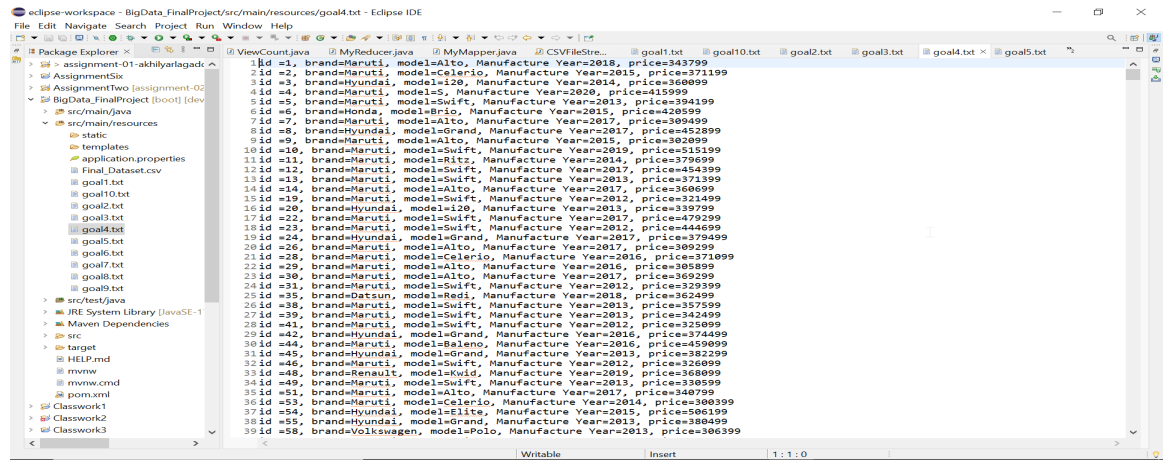


7.4 Specify all the Car Brands, Models, years, and price between 3L to 6L?

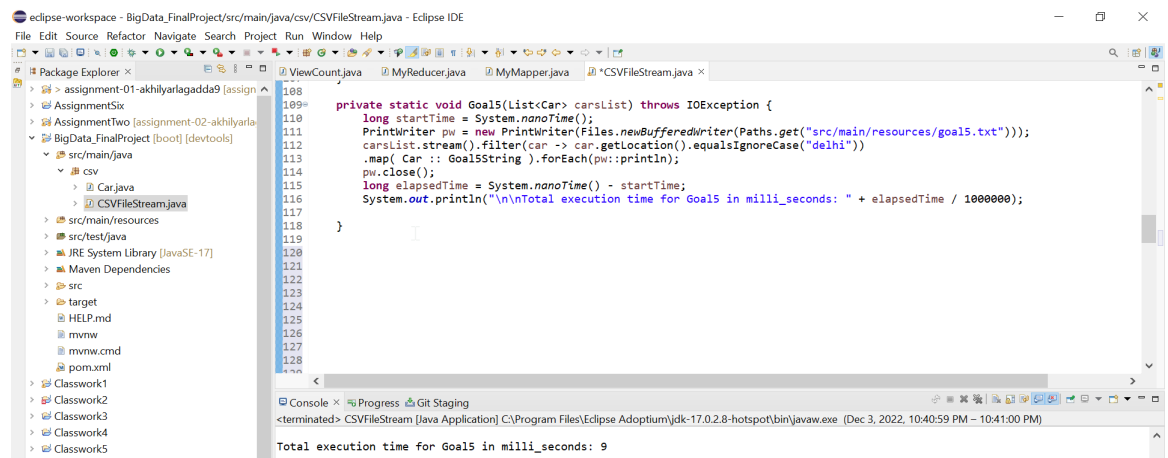


- To specify the cars that are ranging from the price 3 lakhs to 6 lakhs depends on all brands of cars.
- No difficulty find while we executing this goal.

OUTPUT 4



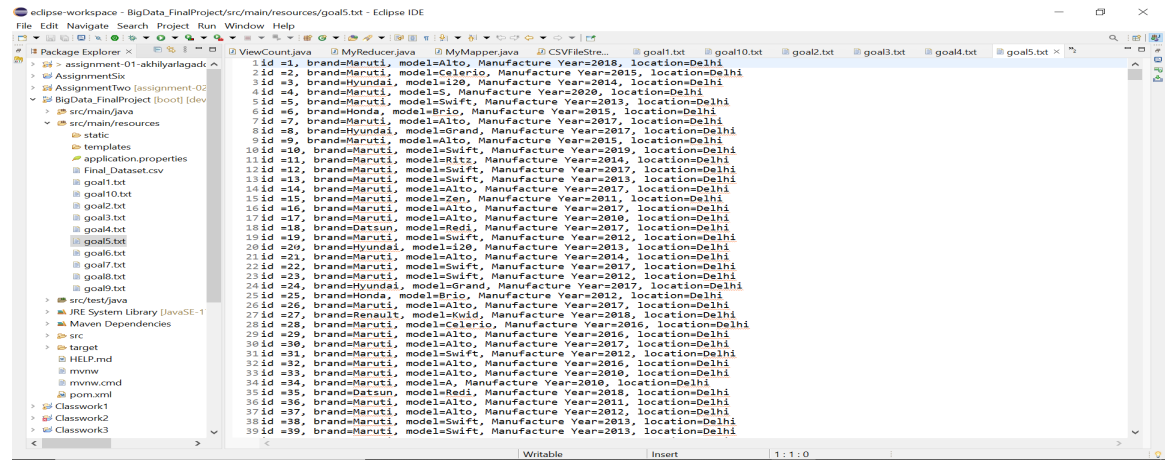
7.5 List Brand with Model and year which are in the location Delhi?



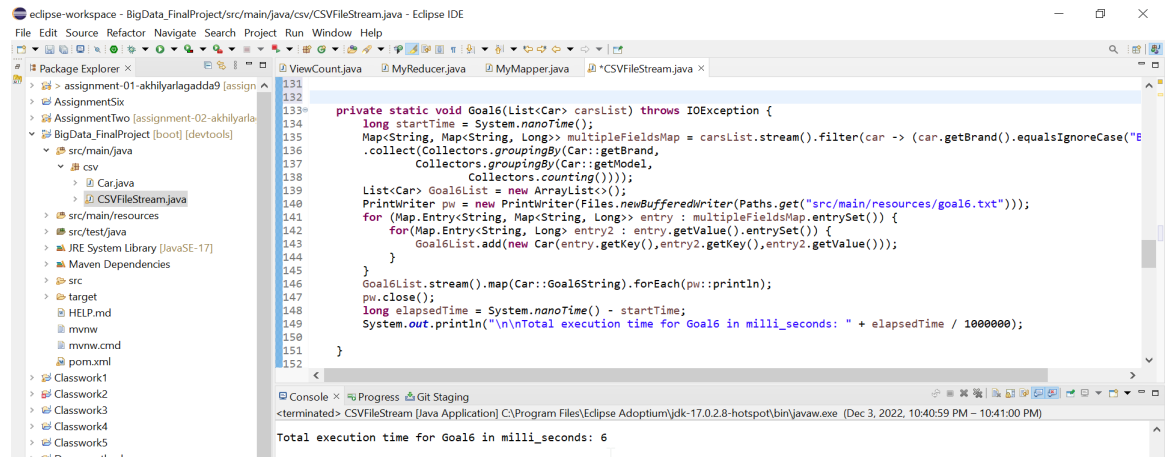
- Here we are displaying all the cars that are in the Delhi location with all the details like Brand of the car with the model and the year.
- No difficulty find while we executing this goal.

OUTPUT 5

8 Authors Suppressed Due to Excessive Length

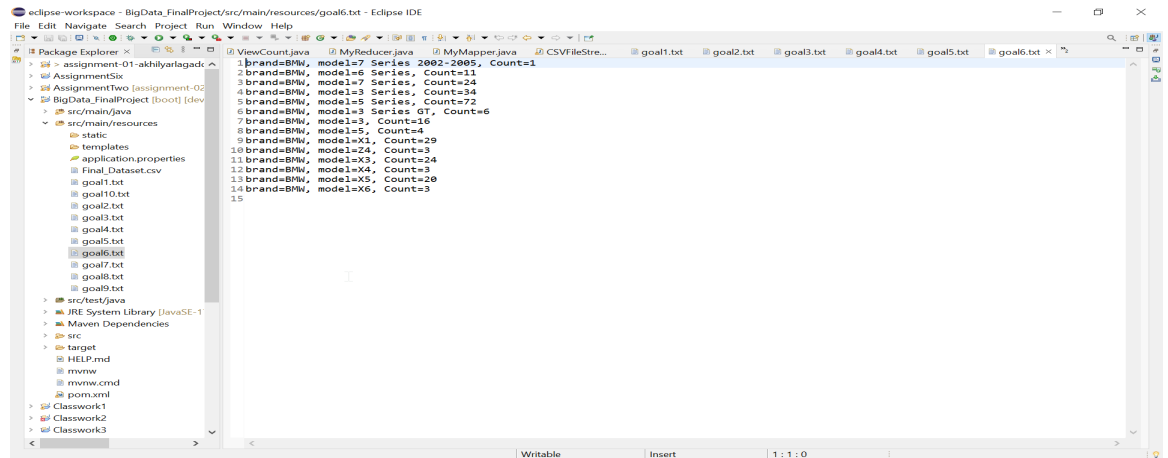


7.6 List out all the brand, model and count with BMW?

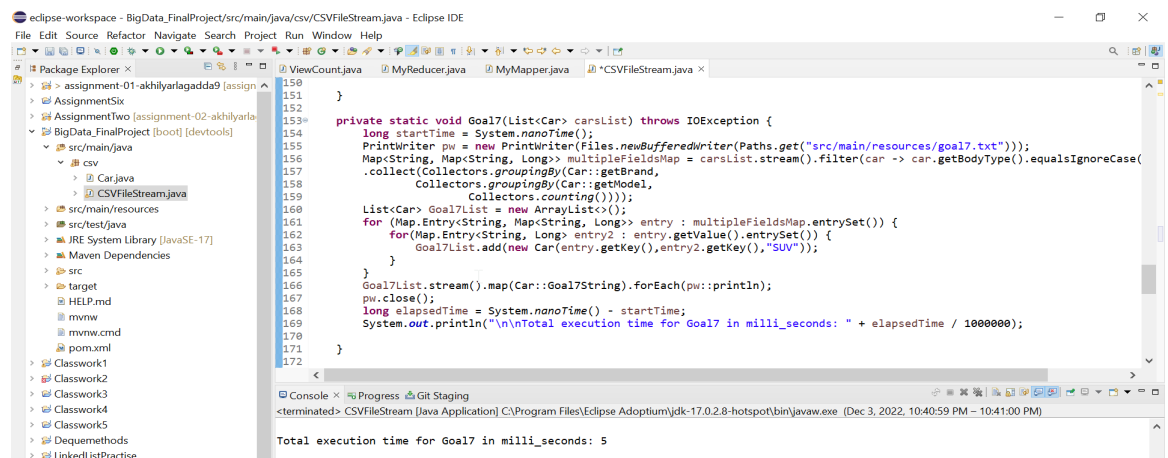


- Calculating the number of manufactured cars for each model of the BMW cars.
- No difficulty find while we executing this goal.

OUTPUT 6

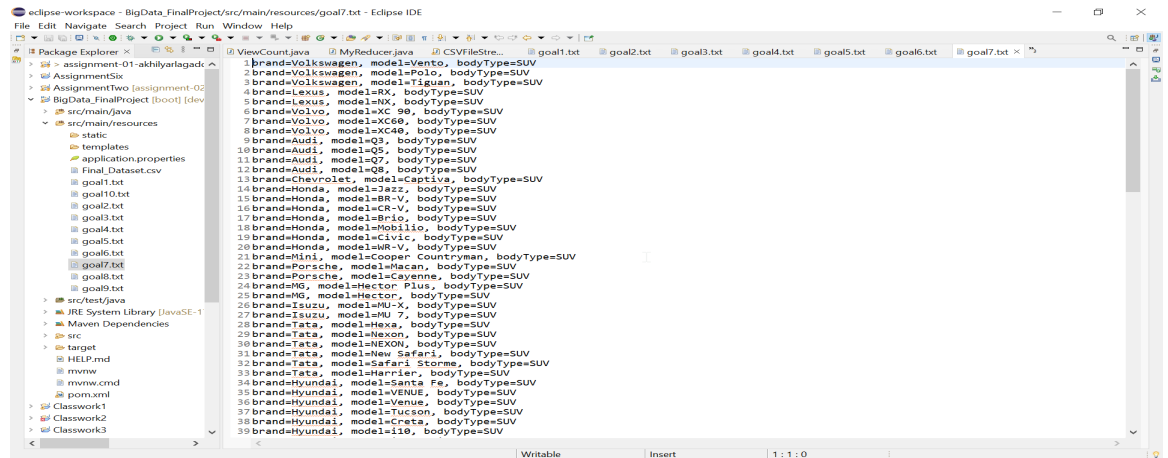


7.7 List SUV body type Brand and Models?

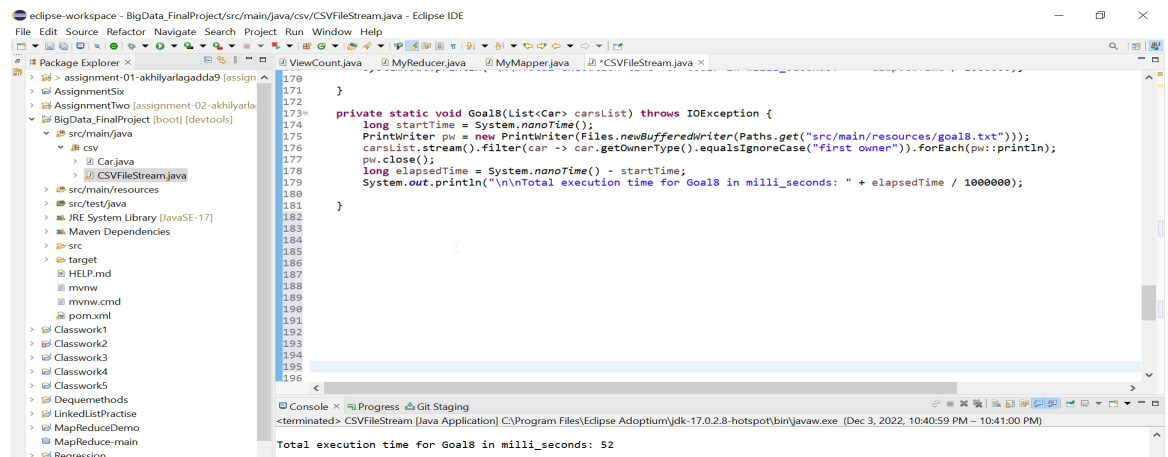


- Providing the data of the Brand and Models if the SUV body typed cars.
- No difficulty find while we executing this goal.

OUTPUT 7

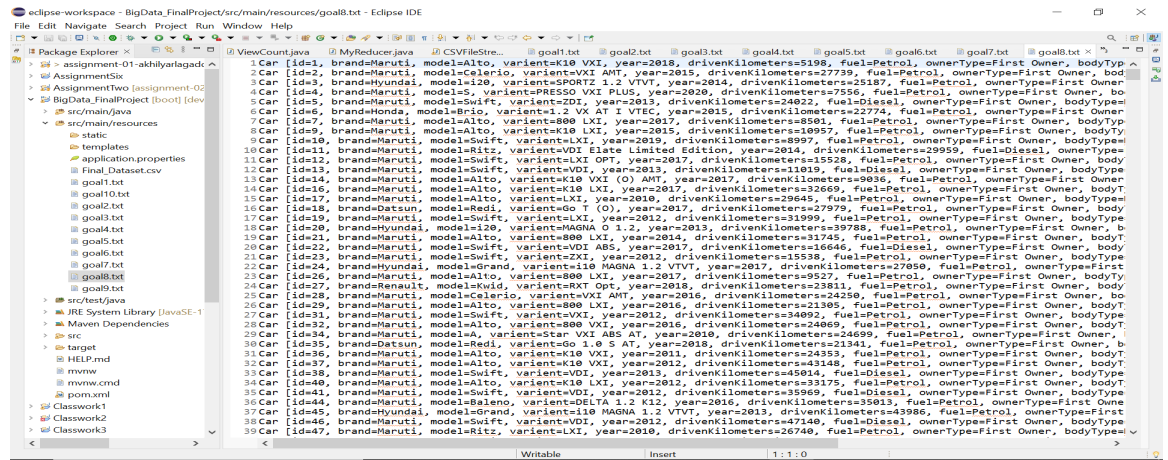


7.8 List the all first owner car brand with Models?

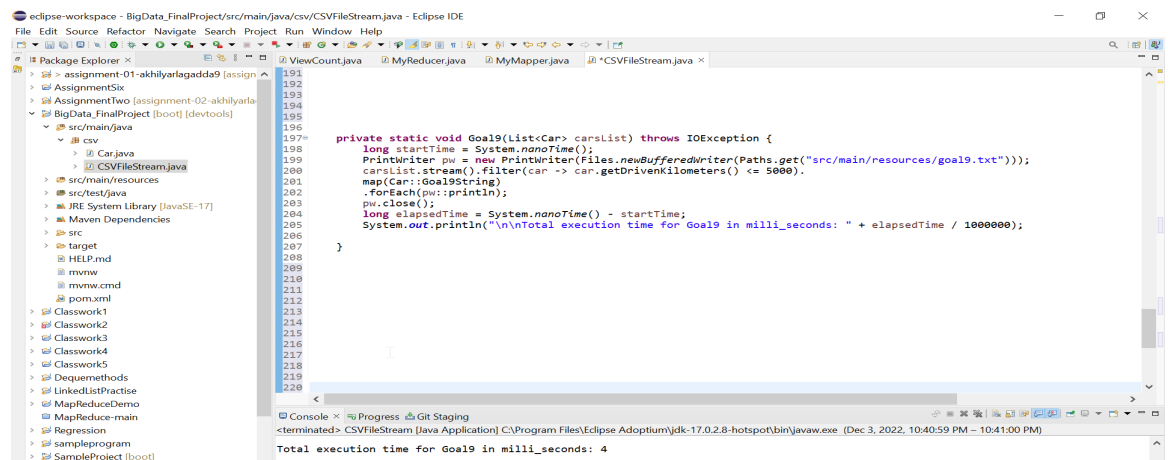


- Providing the data of the First owner who used particular brand and model of the cars.
- No difficulty find while we executing this goal.

OUTPUT 8



7.9 List the car model, Brand, Year, and Km that are driven less than 10000 kilometers?



- To display the cars that are driven 10000 kilometers with specific Car model, Brand and the Year.
- No difficulty find while we executing this goal.

OUTPUT 9

12 Authors Suppressed Due to Excessive Length

```

1 id=78, brand=Maruti, model=Alto, Manufacture Year=2018, drivenKilometers=3212
2 id=337, brand=Renault, model=Kwid, Manufacture Year=2019, drivenKilometers=4037
3 id=375, brand=Maruti, model=S, Manufacture Year=2019, drivenKilometers=3353
4 id=415, brand=Maruti, model=Alto, Manufacture Year=2017, drivenKilometers=2678
5 id=416, brand=Tata, model=Tiago, Manufacture Year=2018, drivenKilometers=4013
6 id=427, brand=Hyundai, model=Eon, Manufacture Year=2015, drivenKilometers=4711
7 id=559, brand=Renault, model=Kwid, Manufacture Year=2019, drivenKilometers=2663
8 id=584, brand=Maruti, model=Baleno, Manufacture Year=2016, drivenKilometers=474
9 id=610, brand=Tata, model=Tiago, Manufacture Year=2019, drivenKilometers=4321
10 id=634, brand=Maruti, model=Alto, Manufacture Year=2020, drivenKilometers=3942
11 id=822, brand=Maruti, model=Dzire, Manufacture Year=2020, drivenKilometers=800
12 id=889, brand=Ford, model=Figgo, Manufacture Year=2018, drivenKilometers=4963
13 id=1078, brand=Hyundai, model=VENUE, Manufacture Year=2021, drivenKilometers=2589
14 id=1152, brand=Hyundai, model=creta, Manufacture Year=2019, drivenKilometers=3249
15 id=1374, brand=Maruti, model=Alto, Manufacture Year=2018, drivenKilometers=3212
16 id=1634, brand=Renault, model=Kwid, Manufacture Year=2019, drivenKilometers=4037
17 id=1668, brand=Maruti, model=S, Manufacture Year=2019, drivenKilometers=3353
18 id=1706, brand=Maruti, model=Alto, Manufacture Year=2017, drivenKilometers=2678
19 id=1787, brand=Tata, model=Tiago, Manufacture Year=2018, drivenKilometers=4013
20 id=1717, brand=Hyundai, model=Eon, Manufacture Year=2015, drivenKilometers=4711
21 id=1829, brand=Maruti, model=Alto, Manufacture Year=2020, drivenKilometers=3942
22 id=2015, brand=Maruti, model=Dzire, Manufacture Year=2020, drivenKilometers=800
23 id=2017, brand=Honda, model=Amaze, Manufacture Year=2019, drivenKilometers=4693
24 id=2083, brand=Ford, model=Figgo, Manufacture Year=2018, drivenKilometers=4963
25 id=2219, brand=Hyundai, model=VENUE, Manufacture Year=2021, drivenKilometers=2589
26 id=2459, brand=Maruti, model=Alto, Manufacture Year=2018, drivenKilometers=3212
27 id=2717, brand=Renault, model=Kwid, Manufacture Year=2019, drivenKilometers=4037
28 id=2751, brand=Maruti, model=S, Manufacture Year=2019, drivenKilometers=3353
29 id=2787, brand=Maruti, model=Alto, Manufacture Year=2017, drivenKilometers=2678
30 id=2788, brand=Tata, model=Tiago, Manufacture Year=2018, drivenKilometers=4013
31 id=2798, brand=Hyundai, model=Eon, Manufacture Year=2015, drivenKilometers=4711
32 id=2911, brand=Maruti, model=Alto, Manufacture Year=2020, drivenKilometers=3942
33 id=3096, brand=Maruti, model=Dzire, Manufacture Year=2020, drivenKilometers=800
34 id=3098, brand=Honda, model=Amaze, Manufacture Year=2019, drivenKilometers=4693
35 id=3164, brand=Ford, model=Figgo, Manufacture Year=2018, drivenKilometers=4963
36 id=3299, brand=Hyundai, model=VENUE, Manufacture Year=2021, drivenKilometers=2589
37 id=3472, brand=Maruti, model=Celerio, Manufacture Year=2017, drivenKilometers=4760
38 id=3489, brand=Maruti, model=Alto, Manufacture Year=2015, drivenKilometers=3378
39 id=3593, brand=Maruti, model=Alto, Manufacture Year=2018, drivenKilometers=4343
  
```

7.10 Name the honda cars with the model, Body type, and year manufactured after 2015.

```

214 private static void Goal10(List<Car> carsList) throws IOException {
215     long startTime = System.nanoTime();
216     PrintWriter pw = new PrintWriter(Files.newBufferedReader(Paths.get("src/main/resources/goal10.txt")));
217     Map<String, Map<Integer, Long>> multipleFieldsMap = carsList.stream().filter(car -> (car.getBrand().equalsIgnoreCase("
218     .collect(Collectors.groupingBy(Car::getYear,
219         Collectors.counting()));
220     List<Car> goal10List = new ArrayList<>();
221     for (Map.Entry<String, Map<Integer, Long>> entry : multipleFieldsMap.entrySet()) {
222         for (Map.Entry<Integer, Long> entry2 : entry.getValue().entrySet()) {
223             goal10List.add(new Car("Honda", entry.getKey(), entry2.getValue()));
224         }
225     }
226     goal10List.stream().map(Car::toString).forEach(pw::println);
227     pw.close();
228     long elapsedTime = System.nanoTime() - startTime;
229     System.out.println("\nTotal execution time for Goal10 in milli_seconds: " + elapsedTime / 1000000);
230 }
231
232
233
234
235
236
237
238
239
240
241
242
243 }
244
245
  
```

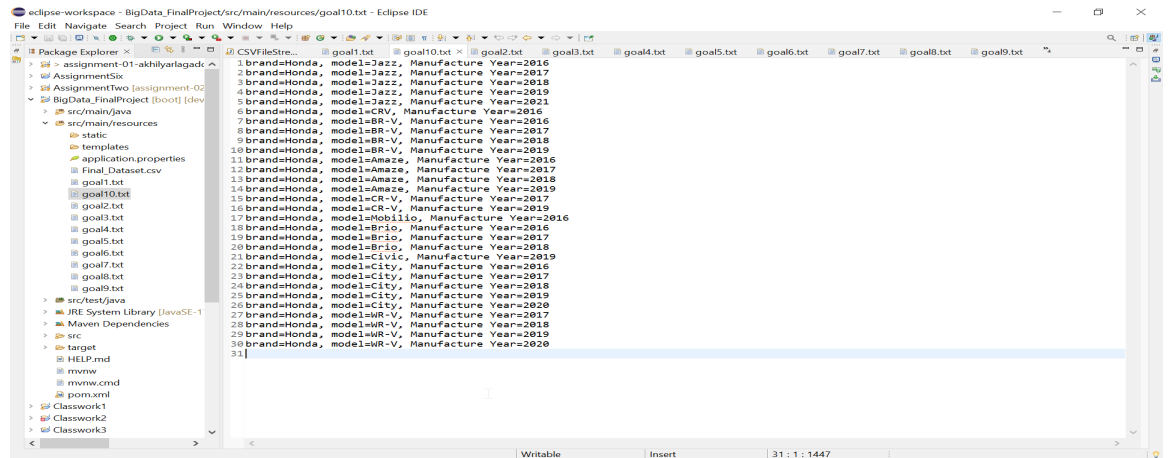
Console Output:

```

<terminated> CSVFileStream [Java Application] C:\Program Files\Eclipse Adoptium\jdk-17.0.2-hotspot\bin\javaw.exe (Dec 3, 2022, 10:40:59 PM - 10:41:00 PM)
Total execution time for Goal10 in milli_seconds: 6
  
```

- Providing the cars that are manufactured after 2015 with the model, Body type.
- No difficulty find while we executing this goal.

OUTPUT 10



8 Conclusion of your analysis

To Finalize, the data we have taken is to show the numbers of the cars in the market are being sold based on the Make Model of the cars by its body type and the location of the buyer and seller with the price of the car based on the market price.

9 Provide the GitHub link for your project

GITHUB LINK