Used car data collection and analyzing website

Xiannan Chen
100995076

Submitted to: Dr.
Anil Somayaji
School of Computer Science
Carleton University

13-12-2020

Contents

Figures

Tables

1. Introduction

1-1. Project Information

Table 1

Project Name:	Used car data climber and analyzing website
Author:	Xiannan Chen
Supervisor:	Anil Somayaji
Department	Faculty of Science - School of Computer Science
Date	December 11, 2020

1-2. Development Team

Xiannan Chen

1-3. Abstract

This project creates a website to help users choose used car based on the users' filter condition. Users will be asked to enter four filter conditions, which are Body type, Postal Code, Min Year and Budget. Then, the backend will run a puppeteer Node.JS file to collect the first 99 newest car information and store the information in database. When the enough data has been collected. A Python file will be run to calculate which 6 used car are most worth to buy and display them to users.

1-4. Overview of Document

The second part of this document is designed to state this website development processes, design patterns and analyze requirement of making this puppeteer-used car analyzation website.

The third part of this document will explain what problem was met and solved. What is the goal of this project and how the goals be achieved. Besides, this part will introduce the methodology I used such as puppeteer, SQL and Express. Then discuss the overall structure of the project and UML diagrams.

2. Proposed System

All students must take COMP 3004. In this course, we learnt software development processes, requirement specification, etc. This part of document shows the whole project development processes. Started from list all functional and most non-functional requirements. Few non-functional requirements may vary such as performance requirement during developing. Following step is design the system architecture and model. All these steps could make the development much more efficient.

2-1. Functional Requirements

The functional requirements in this website for a User are input four filter condition and view 6 used car which are worth to buy. Besides, the system must create and drop table, collect information, and calculate result correctly.

Table 2

F-01	User must be able to input filter conditions
	F-01-01 Edit body type
	F-01-02 Edit Postal Code
	F-01-03 Edit Min year of cars
	F-01-04 Edit Budget
F-02	User must be able to view the recommendation result
	F-02-01 System must be able to recommend 6 used cars(including 3 luxury brand cars) correctly based on algorithm
	F-02-01 The table must be deleted in the end
F-03	The table in database must be created correctly
F-04	The enough car information must be collected correctly
	F-04-01 Users' filter conditions must be inputted to target website correctly.

F-04-02	Car information(year, make, model, drivetrain, mileage, and price) must be collected correctly.
F-04-03	The above data must be able to store in database correctly.

2-2. Non-Functional Requirements

The non-functional requirements on the website are those related to quality systematic.

Table 3

Usability requirement	
NF-01	The user interface must be clear and use commonly understood UI element such as windows and dropdown list.
NF-02	The user interface must support user input by mouse and keyboard.
NF-03	All errors must be reported to the user.
Reliability and performance requirement	
NF-04	The system should be able to run the computer completely offline on a single computer.
NF-05	The database should support up to 100 used car information.
NF-06	The whole collecting used car information process should take less than 5 minutes.
NF-07	The result should be saw in 3 seconds after the enough used car information been collected.
NF-08	The collecting information process should start in 5 second after users click submit button.
Manageability requirement	
NF-09	The system should be extensible to support any type of used car.
NF-10	The system should be extensible to support other similar used car website with some simple change.
NF-11	The system should be easily extensible to support multiple languages.
NF-12	The system should be scalable to network solutions to allow multiple clients

	are used at the same time.
Environmental requirement	
NF-13	The system should be able to run on both windows and mac operation system.
	Scalability requirement
NF-14	The database of this system should be able to store any used car data.
	Packaging requirement
NF-15	The system should be able to download as an archive.
NF-16	The system should be run directly without any addition installation except npm install.
Serviceability requirement	
NF-17	The system should be only use Node.JS and Python to do any update.
Security requirement	
NF-18	The system is not require user registration or logon. User should be able to directly to use this system without any personal information collection.
Data Integrity requirement	
NF-19	The system should allow for software updates without affecting stored data.
NF-20	The collected data should be store in database correctly.

2-3. System Models

Use case Model

The use case diagram state the relation between the user and the different use case in this website.

Figure 1

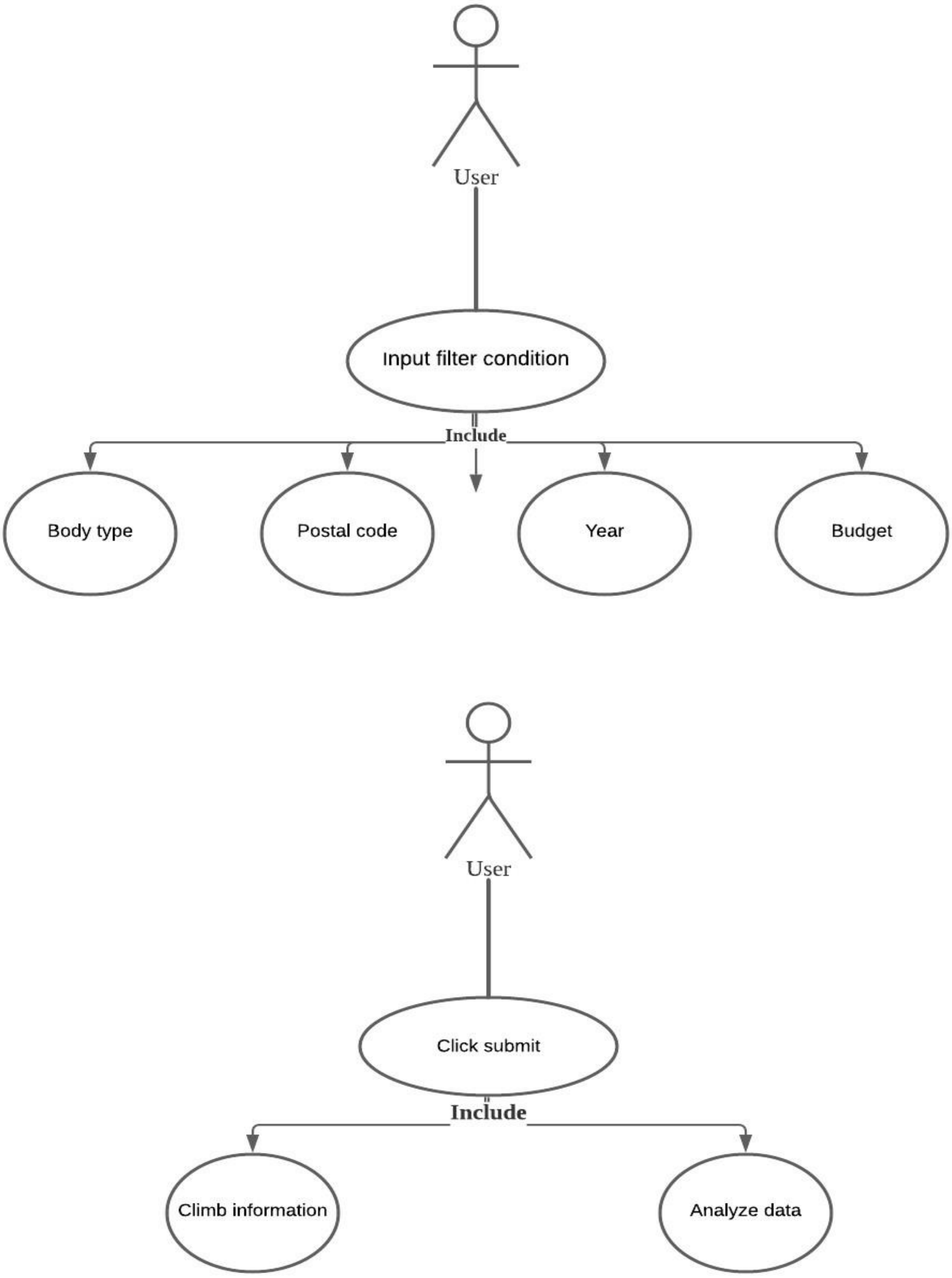


Table 4

Use case identifier	UC-01
Use case name	Input filter condition
Participating	Initiated by User
Flow of events	1.The user select body type from dropdown list 2.The user input postal code 3.The user edit Min year and Budget
Entry condition	The user must enter this website
Exit condition	The user must input all condition valid and click submit
Traceability	F-01, NF-01, NF-02

Use case identifier	UC-02
Use case name	Climb and store information
Participating	Initiated by system
Flow of events	4.The user click submit button 5.The system open a new browse 6.The system control the new browse then collect and store data
Entry condition	The user must click submit button.
Exit condition	The enough information must be collected.
Traceability	F-03, F-04, NF-05, NF-06, NF-08, NF-14, NF-20

Use case identifier	UC-03
Use case name	Get result
Participating	Initiated by system
Flow of events	7. The system run the algorithm and display the result.
Entry condition	The collect information process have completed.
Exit condition	The result been displayed.
Traceability	F-02, NF-7

Object Model

The object model states the classes in this system and their relationship

Figure 2

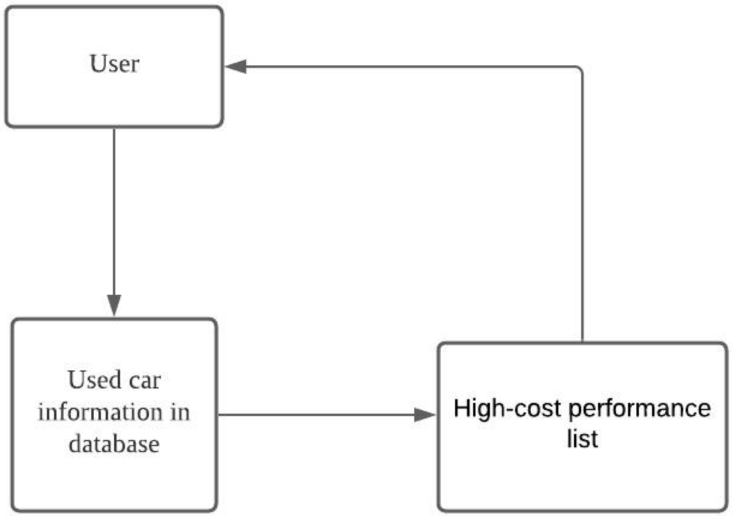


Table 5

Entity Object	User(UC-01)
Attributes	N/A
Def	All users can enter this website and use this website. Users do not need to register or logon. There is only one type of user(No administrator).

Entity Object	Used car list(UC-02)
Attributes	name, make, model, year, price, drivetrain
Def	All collected car data stored in this object

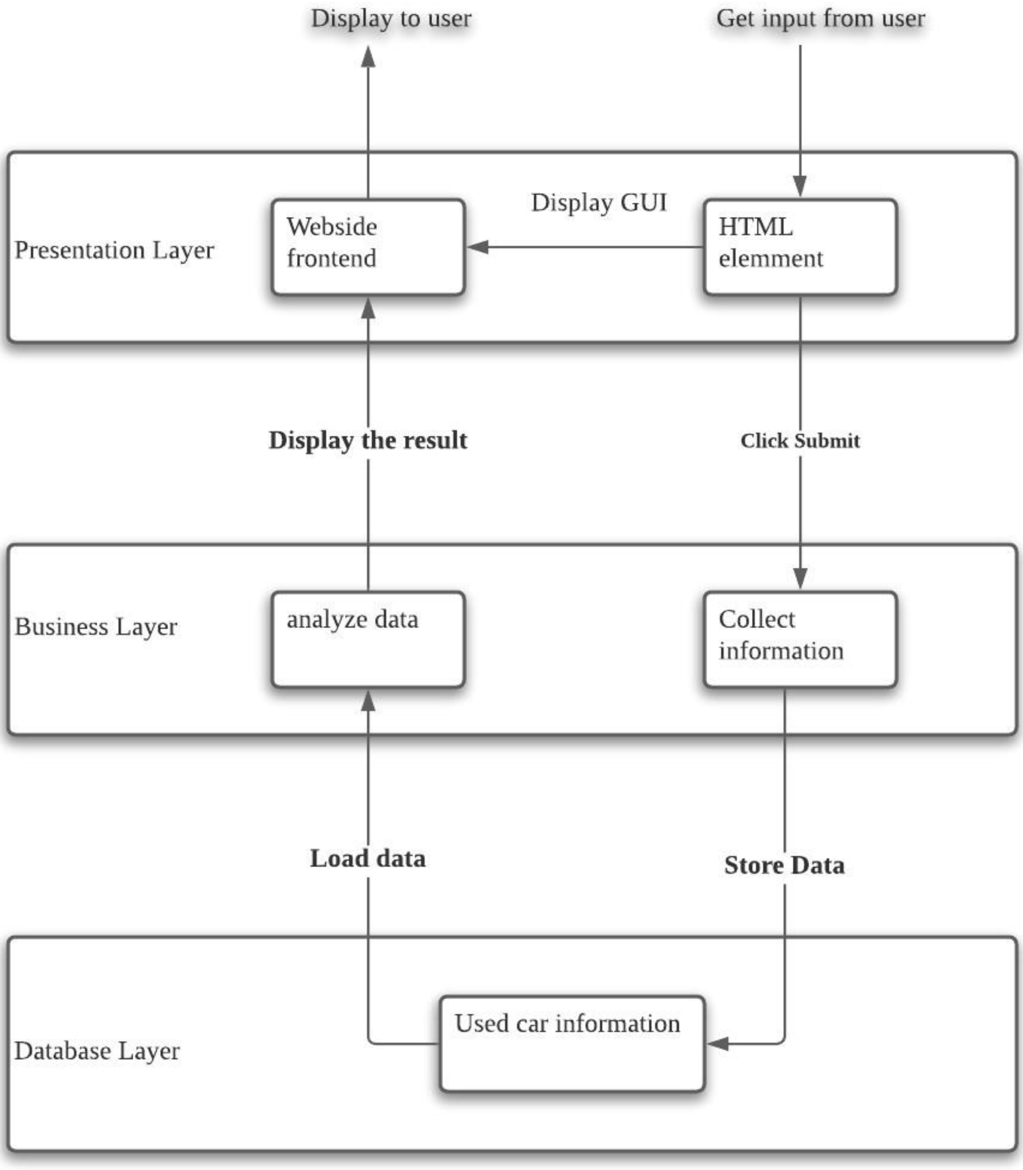
Entity Object	High-cost performance list (UC-03)
Attributes	name, make, model, year, price, drivetrain
Def	After calculation, the top 6 cars will be recommended and displayed to user.

2-4. Architecture

Layered architecture

This project is improved based on the above Layer architecture. The main purpose of design the framework is making development more efficient and supporting future Editor as reference. As the most common architecture style, each layer of hierarchical architecture has a clear division Labor. Just figure out which layer to write to (Business layer, data layer, etc.), and then Connect to the interface during the development process, and you can easily determine the logic between them Layer.

Figure 3



3. Report

3-1. Report Introduction

All COMP student must take COMP 2406, Fundamental of web applications. Which introduce internet application development and Node.JS. The most backend of this project is written by two library of Node called puppeteer and express. The other small part backend use python to write.

3-1-1. Problem

What problem this website solved?

When a potential used car buyer tries to find a used car. There are thousand of options. And used cars do not have retail price, each used car is different. Which would bring lots of confusion to the potential used car buyer.

What challenge of this project for me and what problem was met during developing?

The first part of this project is developed by a library name puppeteer of Node.JS. However, I learnt Node.JS not very well. I just got a grade of C- in COMP 2406. Thus, the process went slowly at start. Fortunately, this part is not too difficult. Reading the puppeteer documentation and some sample code are very helpful to review the knowledge of COMP2406 and complete the first part of this project. After that, the more challenge was met in the second part of this project, which is responsible to analyzing the data collected by the first part. The more details will be pointed in section 3-3-5(To be completed). Thus, two main backend command line program have finished. The next challenge is combining them up to be executed in order web server-side. It is more difficult than most online web create tutorial. The two .js file named js_call_py and js_scrape use child process to handle this issue.

3-1-2. Motivation

As above mention. I did not do a good job on COMP 2406. Thus, this project was considered to some extent as retake COMP 2406 by myself to me. This is my personal motivation. Besides, the other motivation for this project came from choosing a used car. It was so confused. Potential used car buyers are impossible to compare each options in their area.

3-1-3. Goals

This website is designed to help users found the highest cost performance used car from thousands of options based on their filter conditions.

3-1-4. Objectives

The goal will be attained by separate the project into three smaller parts and use multiple libraries. The first part to get users input and use puppeteer to climb data and store the data in database. The next step would be analyzing the collected data and display result to users. The last step is creating a front-end GUI and connect it to back-end.

3-2. Methodology

3-2-1. Node.JS

Node.js is an open-source server environment, uses JavaScript on the server and runs on various platforms such as Windows, Linux and Mac.Node.js also can collect from data and modify data in the database.

3-2-1-1. Puppeteer

Puppeteer is a Node library which provides a high-level API to control Chrome or Chromium over the DevTools Protocol. Puppeteer is used to control Chrome enter target webpage and collect and store specific information in this project.

3-2-1-2. Express

Express.js is a Node.js web application server framework for server-side programming, which allows users to build network application quickly. Express is used to connect two backend file and one HTLM frontend file.

3-2-2. Python

Python is a general-purpose coding language. This project use python to analyzing the data in database.

3-2-3. SQL

SQL (Structured Query Language) is a special purpose language designed for managing data held in a relational database management system. This project use SQL to create and drop table, insert and update data.

3-2-4. NPM

Npm allows user to download all npm public software packages without any registration or logon.

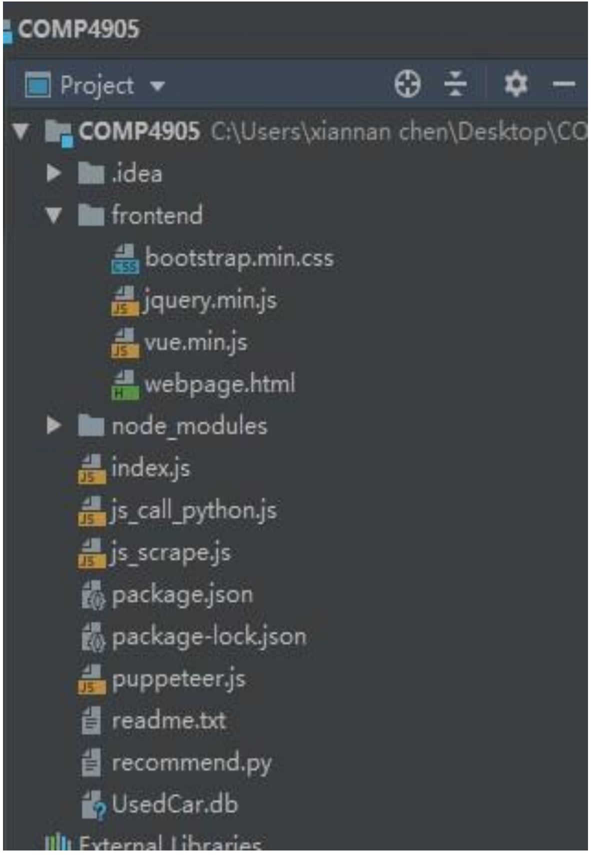
3-3. Result

This project has been completed. It is a long since its proposal. This project also has been a great learning experience. Lots of knowledge and skills were reviewed and learnt from this project.

The GUI of this project is not complex but very clear to users and easily to understand. In addition, it connects backend well. However, there are still two upgrades could be done to this project in the future. The first is increase the puppeteer stable. The other more important is upgrade recommend algorithm.

3-3-1. File and package structure

Figure 4



```
COMP4905
Project
COMP4905 C:\Users\xiannan chen\Desktop\CO
  .idea
  frontend
      bootstrap.min.css
      jquery.min.js
      vue.min.js
      webpage.html
  node_modules
  index.js
  js_call_python.js
  js_scrape.js
  package.json
  package-lock.json
  puppeteer.js
  readme.txt
  recommend.py
  UsedCar.db
External Libraries
```

Figure 4 shows the structure of the project.

Frontend: contains all static files

Webpage.htlm: Handle the webpage looks like

Node_modules: Store the installed libraries

Index.js: Start the web server and combine everything together.

Js_call_python: Set up analyzing data process.

Js_scrape.js: Control the two main backend files.

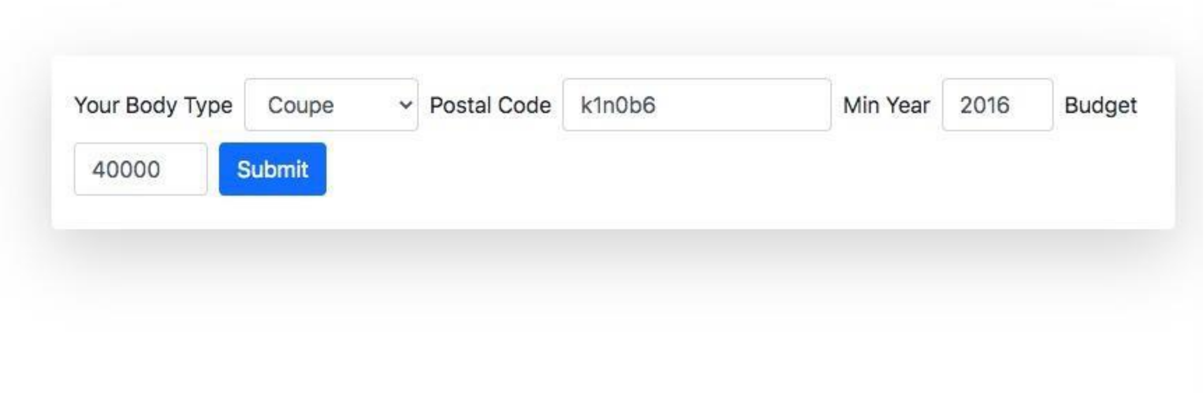
Package.json: List all libraries required to be installed.

Puppeteer.js: Control climbing information.

Recommend.py: Control analyzing data.

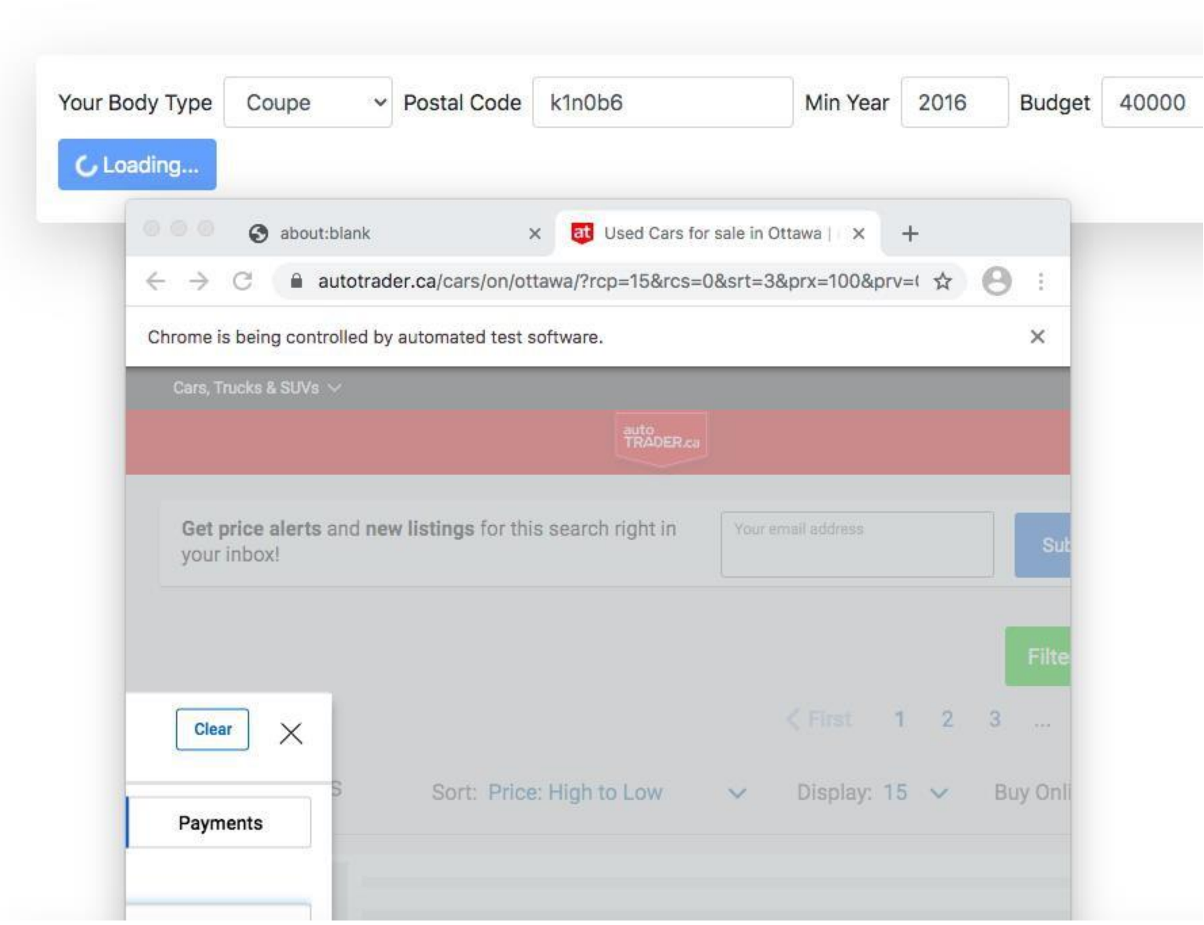
3-3-2. Frontend

Figure 5



Your Body Type: Coupe
Postal Code: k1n0b6
Min Year: 2016
Budget: 40000

Figure 6



Your Body Type: Coupe
Postal Code: k1n0b6
Min Year: 2016
Budget: 40000

about:blank

Used Cars for sale in Ottawa |

autotrader.ca/cars/on/ottawa/?rcp=15&rcs=0&srt=3&prx=100&prv=(

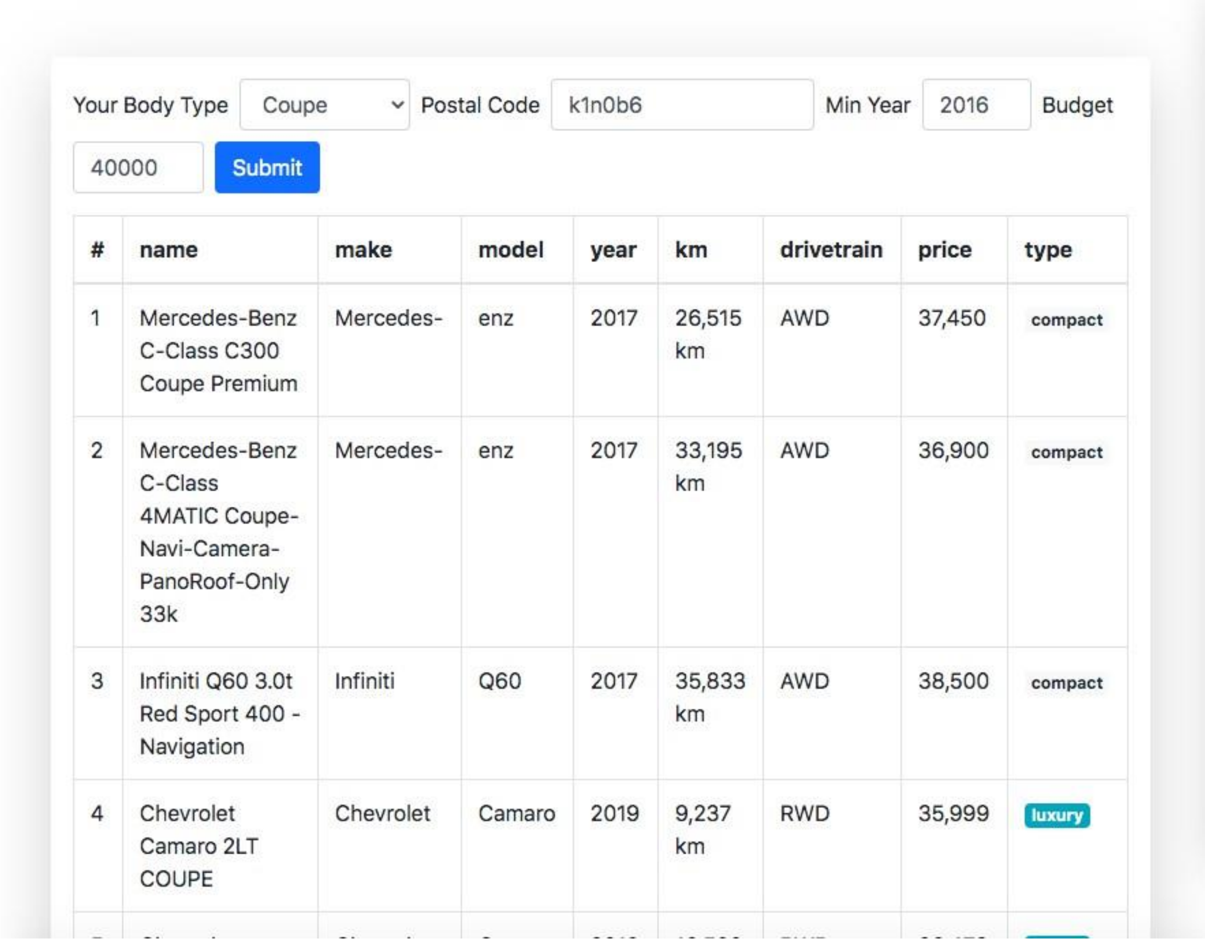
Chrome is being controlled by automated test software.

Get price alerts and new listings for this search right in your inbox!

Sort: Price: High to Low

Display: 15

Figure 7



Your Body Type: Coupe
Postal Code: k1n0b6
Min Year: 2016
Budget: 40000

#	name	make	model	year	km	drivetrain	price	type
1	Mercedes-Benz C-Class C300 Coupe Premium	Mercedes-	enz	2017	26,515 km	AWD	37,450	compact
2	Mercedes-Benz C-Class 4MATIC Coupe-Navi-Camera-PanoRoof-Only 33k	Mercedes-	enz	2017	33,195 km	AWD	36,900	compact
3	Infiniti Q60 3.0t Red Sport 400 - Navigation	Infiniti	Q60	2017	35,833 km	AWD	38,500	compact
4	Chevrolet Camaro 2LT COUPE	Chevrolet	Camaro	2019	9,237 km	RWD	35,999	luxury

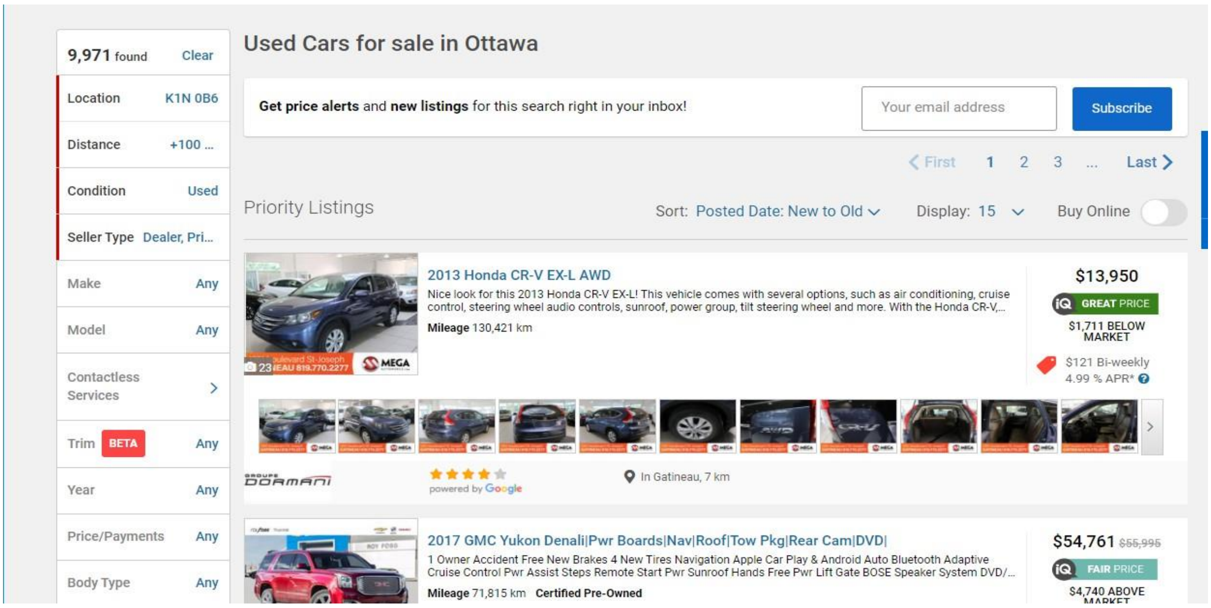
Users will use the above GUI to interact with the system. Users are required to choose body type and input the postal code, min year, budget and click submit. Then, users may wait for about 5 minutes to view to result. The figure 6 shows the GUI when users wait for the result, meanwhile, a new Chrome will be controlled to climb information. When the climbing process completed, webpage will jump to figure 7.

3-3-3.
Backend

The backend is responsible for climbing the used car information from autotrader website and store the data in database, then analyzing the collected data. A .js file called puppeteer handles collecting information. A new Chrome browser will be launched and open a new page first, then this page will jump to autotrader.ca. The next step is passing postal code from user to the autotrader website and starting search. Now, the page similar with figure

8 will appear. The other 3 filter condition which are budget, price, and body type will be set on this page. Meanwhile, the sort will be set to Posted Date: New to Old and page size will be set to 50. Thus, the page will be updated. Now, the target information appears. There is a selector list containing every car in this page. The backend file will control the browser enter each single car page one by one. After the single car page is loaded complete. The required data will be collected and written into database. Then the page will go back to enter the next car page. Usually, this process will end until 99 cars information been store in database unless there are no 99 available cars based on the users' filter condition. In the end, the page and database will close till the enough data has been collected. It is analyzing data turn for now. A .py file called recommend.py will handle analyzing data. The analyzing is not complex and will be upgrade in the future, which will be explained in the next part(3-3-5). For current analyzing algorithm. A luxrybrand_list will be created and contains all luxury car brand and recommend_list1 and recommend_list2 will be created. Because the result will give recommend user 3 luxury brand cars such as BMW and 3 general brand cars such as Honda. The algorithm contains two functions, one is comparing two cars the other is adding one car to recommend list. In comparing function, we set every one year after worth $6000 or $8000 depends on car brand type, every one km worth $0.15 or $0.25 depends on car brand type and AWD drivetrain worth $5000 or $6000. And the price different between these two cars will be calculated to see which one more worth wo purchase. The other function is addcar. In this function. if the length of recommend_list is less than 3, the current car will be directly added to recommend_list. Otherwise, comparing function will be called. The current car will be compared with the cars at end of recommend_list to decide if the current car is more worth to buy. If yes, The current car will replace the car at end of recommend_list. Then, the current car will continue to compare the cars before itself. Until the current car cannot win the comparison or the current car already be the first element of the recommend_list. If no, the system will go through to the next car in the database. After iterating all cars in the database, the table will be dropped, and the result will be displayed to users.

Figure 8



9,971 found	Clear
Location	K1N 0B6
Distance	+100 ...
Condition	Used
Seller Type	Dealer, Pri...
Make	Any
Model	Any
Contactless Services	>
Trim BETA	Any
Year	Any
Price/Payments	Any
Body Type	Any

Used Cars for sale in Ottawa

Get price alerts and **new listings** for this search right in your inbox!

Your email address

Subscribe

< First 1 2 3 ... Last >

Priority Listings

Sort: Posted Date: New to Old

Display: 15

Buy Online

2013 Honda CR-V EX-L AWD

Nice look for this 2013 Honda CR-V EX-L! This vehicle comes with several options, such as air conditioning, cruise control, steering wheel audio controls, sunroof, power group, tilt steering wheel and more. With the Honda CR-V,...

Mileage 130,421 km

$13,950

iQ GREAT PRICE

$1,711 BELOW MARKET

$121 Bi-weekly 4.99 % APR*

In Gatineau, 7 km

powered by Google

2017 GMC Yukon Denali|Pwr Boards|Nav|Roof|Tow Pkg|Rear Cam|DVD|

1 Owner Accident Free New Brakes 4 New Tires Navigation Apple Car Play & Android Auto Bluetooth Adaptive Cruise Control Pwr Assist Steps Remote Start Pwr Sunroof Hands Free Pwr Lift Gate BOSE Speaker System DVD/...

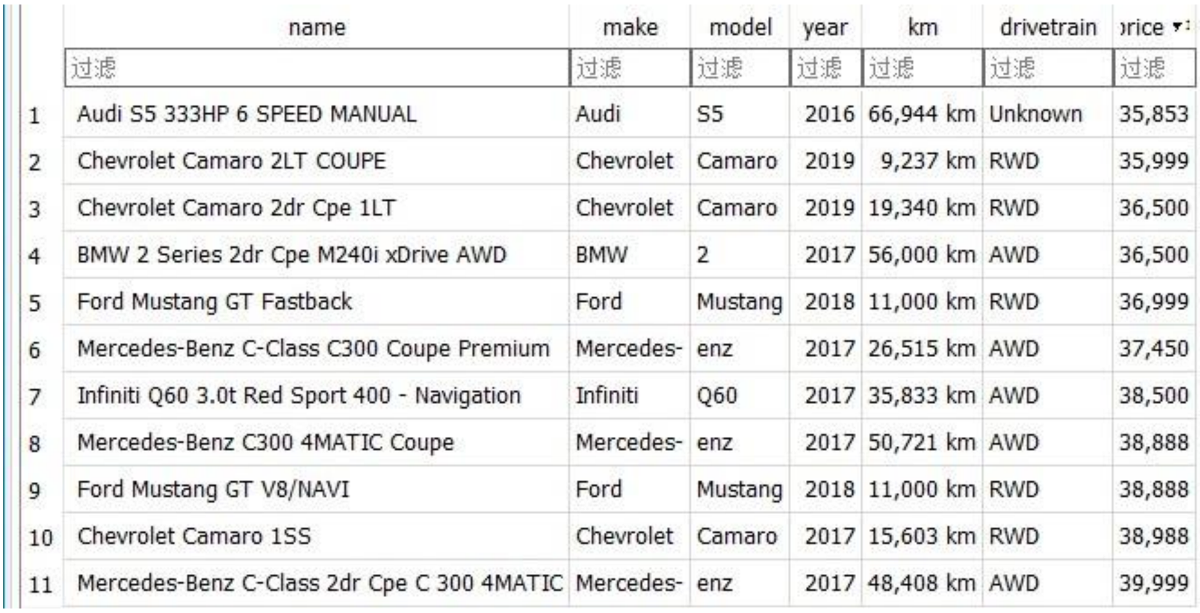
Mileage 71,815 km **Certified Pre-Owned**

$54,761 ~~$55,995~~

iQ FAIR PRICE

$4,740 ABOVE MARKET

Figure 9



	name	make	model	year	km	drivetrain	rice
	过滤	过滤	过滤	过滤	过滤	过滤	过滤
1	Audi S5 333HP 6 SPEED MANUAL	Audi	S5	2016	66,944 km	Unknown	35,853
2	Chevrolet Camaro 2LT COUPE	Chevrolet	Camaro	2019	9,237 km	RWD	35,999
3	Chevrolet Camaro 2dr Cpe 1LT	Chevrolet	Camaro	2019	19,340 km	RWD	36,500
4	BMW 2 Series 2dr Cpe M240i xDrive AWD	BMW	2	2017	56,000 km	AWD	36,500
5	Ford Mustang GT Fastback	Ford	Mustang	2018	11,000 km	RWD	36,999
6	Mercedes-Benz C-Class C300 Coupe Premium	Mercedes-	enz	2017	26,515 km	AWD	37,450
7	Infiniti Q60 3.0t Red Sport 400 - Navigation	Infiniti	Q60	2017	35,833 km	AWD	38,500
8	Mercedes-Benz C300 4MATIC Coupe	Mercedes-	enz	2017	50,721 km	AWD	38,888
9	Ford Mustang GT V8/NAVI	Ford	Mustang	2018	11,000 km	RWD	38,888
10	Chevrolet Camaro 1SS	Chevrolet	Camaro	2017	15,603 km	RWD	38,988
11	Mercedes-Benz C-Class 2dr Cpe C 300 4MATIC	Mercedes-	enz	2017	48,408 km	AWD	39,999

3-3-4. Database

The figure 9 shoes how database of this project looks like. The table name is CarInfo and contains 7 columns, which are name, make, model, year, km, drivetrain and price. The collected data is stored in this table and these data will be read for analyzing.

3-3-5. To be completed

This project completes all three parts mentioned in proposal. However, this project is not perfect for now in some points. The first one is about stable, the other one is about analyzing data algorithm.

For the stable problem, in most case, the system can display the result the result successful. However, if the network speed is too low. The system also may crash. And if the computer goes sleep before the climbing information process done, am disconnection error will be reported to user.

The other one is more important. The analyzing data method is not advanced technology such as machine learning. The current system recommends cars based on simple compare each car in the database, which is not accurate enough. The recommend result will be upgraded to base on used car price estimator. The main reason why the current system does not use this technology is I did not take the machine learning course(COMP 4900). I must learn it by myself. There is not enough time to finish it by the due date.

4. Reference

Lind, V. (2017). The Beginner's Guide: Understanding Node.js and Express.js fundamentals. Retrieved from https://medium.com/@LindaVivah/the-beginners-guide-understanding-node-js-express-js-fundamentals-e15493462be1.

Mjzffr. (2020). Headless Chrome Node.js APL. Retrieved from https://github.com/puppeteer/puppeteer.

W3C. (n.d.). Node.js Introduction Retrieved from https://www.w3schools.com/nodejs/nodejs_intro.asp

W3C. (n.d.). What is npm. Retrieved from https://www.w3schools.com/whatis/whatis_npm.asp.

W3cubDocs. (n.d.). Puppeteer Documentation. Retrieved from https://docs.w3cub.com/puppeteer/

JAVSCRIPT.FNFO (n.d.). Promised, async/await. Retrieved from https://javascript.info/async-await#error-handling.