Find Your Perfect Car

David Del Moral
College of Engineering and Computer Science
Texas A&M University-Corpus Christi
Corpus Christi, TX, US
ddelmoral@islander.tamucc.edu

Ean Shi
College of Engineering and Computer Science
Texas A&M University-Corpus Christi
Corpus Christi, TX, USA
eshi@islander.tamucc.edu

Matthew Tomme
College of Engineering and Computer Science
Texas A&M University-Corpus Christi
Corpus Christi, TX, US
mtomme@islander.tamucc.edu

Ricky Godsey
College of Engineering and Computer Science
Texas A&M University-Corpus Christi
Corpus Christi, TX, US
rgodsey@islander.tamucc.edu

Zachary Kao
College of Engineering and Computer Science
Texas A&M University-Corpus Christi
Corpus Christi, TX, USA
zkao@islander.tamucc.edu

***Abstract*—In this proposal we suggest and explore the development of an AI chatbot designed to assist consumers in searching for the vehicle that best suits their wants and needs. We propose the development of an AI chatbot that leverages natural language processing (NLP), and machine learning algorithms coupled with a comprehensive database of vehicles to engage users in a natural, conversational interface that can understand their preferences and provide personalized car recommendations. The goal is to simplify the car selection process and reduce the stress and burden placed on buyers who lack prior knowledge of car details. We explore related solutions and propose potential approaches to create a user-centric and efficient automotive search process. The project culminates in the deployment of a demonstration of the proposed chatbot on our project webpage.**

I. Introduction

In the modern automotive market, consumers are overwhelmed by choices. There are myriads of car models with many features and specifications. In addition to different manufacturers, models, price, and fuel efficiency, modern car buyers need to decide fuel types, autonomous assistance features, connectivity, and infotainment. This complexity can lead to stress and dissatisfied purchases. The advancement of modern technology has helped to enhance the car-buying experience by using artificial intelligence that can answer individual needs.

This paper proposes the development of an AI-assisted chatbot designed to guide users in finding a car that best suits their specific wants and needs. The central problem addressed by this research is the difficulty consumers encounter in navigating the vast array of vehicle options and making informed choices. Traditional methods of car shopping often involve time-consuming research and reliance of generic advice that may not fully align with an individual's unique lifestyle, budget, and preferences. The chatbot helps the user by simplifying the decision-making process using user inputs to deliver sensible recommendations from a comprehensive database of vehicles.

The support of artificial intelligence in car-buying goes beyond convenience, it saves money and reduces stress to search for a car that can improve satisfaction. By employing AI technologies, this proposal seeks to contribute to transforming the automotive sales experience into a confident encounter.

II. Related Works

As AI chatbots and database search engines are not novel ideas, we explored related works that we can draw inspiration from.

A. Akinator

Akinator is a web game that employs machine-learning and a vast database of knowledge to deduce fictional and non-fictional characters and objects [1]. It is similar to the game *Twenty Questions*, in which players ask each other simple yes or no questions to deduce what the other player is thinking about. This process of using user input to narrow down possible results based on a database of knowledge serves as the basis for our proposed research. The problem with an *Akinator* style approach is that *Akinator's* database is built by users and adjusted by user input. This data provided by users can potentially be incorrect, so we propose the use of a carefully curated and comprehensive database to ensure accurate data for our chatbot to draw information from.

B. ChatGPT

ChatGPT is a popular AI chatbot that uses NLP to generate human-like conversations and written content [2]. It leverages a generative pre-trained transformer (GPT), a type of large language model (LLM), that is fine-tuned for conversational applications. It produces human-like responses through a combination of supervised learning and reinforcement learning from human feedback. *ChatGPT* provides a user-friendly and conversational interface for users to interact with that we would like to adopt. However, although *ChatGPT* can be used to search for cars, it is not possible to verify the validity of the information it provides. *ChatGPT* is notorious for potentially providing inaccurate or false information, known as hallucinations [6]. For our purposes, we would like to ensure that all information provided by our chatbot is accurate and citable.

C. CarGPT

CarGPT is a website that allows a user to enter features, styles, or other preferences for their ideal car along with their budget to find cars that match their criteria using AI to provide results [18]. This is a good representation of our ideal implementation; however, it does not provide an unexperienced user with an adequate number and type of categories to choose from right away which could lead to user dissatisfaction. The website also only uses the euro to represent currency, which can cause confusion for people who do not live in an area that utilizes the euro and forces users to convert from their native currency to euros to use the application. We would like to ensure that these are not going to be problems that our users encounter in our program.

D. Traditional Search Methods

Traditional search methods for car buyers involve manual searches, the use of websites such as cars.com [4] to find available cars nearby, and a reliance on car dealership sales representatives to find information and narrow down choices. These options either place the burden of the search on the consumer or a sales representative who may not have the consumer's best interest as their priority.

E. Automobile Consumer Search Behavior

The proposed chatbot aims to provide value to consumers by decreasing customer dissatisfaction through improving the search process. To achieve this, research into vehicle consumer behavior and satisfaction must be done. A relevant study done by Punj in 1983 [12] explores the relationship between searching for vehicle information and customer satisfaction. Their research made three major conclusions. First, they found that people with less prior general car knowledge found greater benefit in searching for car information. Second, they found that search activity indirectly increases customer satisfaction by providing a means to achieve a better purchase. Third, they found that less prior car knowledge a person had, the more search activities they performed.

We drew several conclusions from this research that provided a basis for our proposed chatbot to improve customer satisfaction. As consumers with a large prior base of car knowledge are less likely to need to search for information and will still have the information necessary to make a satisfying purchase, our target audience is consumers with a small prior base of car knowledge. These consumers will benefit from searching for information, but as Punj's research showed, consumers do not directly draw satisfaction from the search activity itself. Customer satisfaction is gained from the application of their knowledge to make an informed and good purchase. Therefore, we aim to increase the efficiency of the vehicle information search. By lowering the amount of searching a customer must do and increasing the amount of useful car information they draw from the search activity, we can increase the efficiency of customer satisfaction.

F. Personalized Product Searches

Online product search is not confined to just retrieving products relevant to the consumer's search. It also focuses on finding items that satisfy the consumer's individual needs and preferences. This makes the personalization of product searches valuable to both consumers and producers. We aim to provide personalized searches through our chatbot, so relevant research into this field is applicable to us. A relevant study by Ai in 2017 [13] proposed the use of a hierarchical embedding model using a deep neural network to improve personalized product searches. Their research found that their hierarchical embedding model for personalized product search significantly outperformed the state-of-the-art base-lines on Amazon benchmarks. This research indicated that personalization of searches is fruitful. By using an AI chatbot to assist consumers in searching, we can leverage the personalization inherent in the technology to increase the effectiveness of vehicle searches.

G. Inquisitive Chatbots

Chatbots can respond to natural language input, but in many scenarios, the user's query may not contain adequate information to provide an answer. In these cases, the chatbot needs to be inquisitive so that it can be interactive and collect the data required to answer the query. A relevant study by Sadhana in 2016 [14] researches this topic. This study achieved an inquisitive chatbot by using a chat engine that can identify what types of information is missing, which can then be obtained by issuing inquisitive queries. We propose a similar approach in the development of our chatbot by utilizing a database, from which missing information can be identified.

H. Chatbots for Searching

In our research, we found several instances of chatbots being employed to improve upon traditional search methods. One relevant study was done by Cantador in 2021 [15] on the impact of using a chatbot to search for open government data. This chatbot was used by non-expert users and found that the proposed system outperformed traditional search methods.

Another relevant study by Pitel in 2024 [16] utilized popular AI-assisted search engines such as ChatGPT to investigate the accuracy of queries in relation to cancer variant interpretation. They found that these popular and public search engines tended to falsely overestimate cancer variant interpretation. They concluded that AI-assisted search engines are a useful supplement in a highly specialized field, but that it required expert human oversight. To account for this, we propose the use of a expertly curated database for our chatbot to use as its knowledge base so that it does not draw false conclusions from past information.

I. Chat Bot Classifications & Techniques

There are many types of chat bots. A very common type is **Task-orientated.** The goal of these bots is to assist the user with completing a task. This could be anything from purchasing a plane ticket or navigating a website. Task oriented chat bots operate in restricted domains as well. **Non-Task oriented** chat bots operate in a more complex matter. Even though responses are still pre-defined, they also generate responses based on pattern recognition. It does this by Generative-Based (generates proper responses during interaction) and Retrieval-Based approaches, (Uses a repository to learn and make more informed responses). **Domain Specific** chat bots can either operate in open or closed domains. Open domains are used for non-specific conversations, and domain specific are used for specific conversations with a goal. By specific conversations it's based off the domain associated (healthcare, or car finder).

There are also different interaction types, **Text-Based** and **Voice-Based.** Competitive models tend to include speech-to-text and require more complex programming models. As they need to be able to filter noise and accurately translate speech to text.

As briefly mentioned, chat bots use different models to generate responses, **Rule-Based** and **Self-learning** are the main distinctions between the two. Self-learning models can either be Generative-Based or Retrieval-Based and then these models implement a variety of techniques. **Parsing** and **Pattern Matching** are essential to a good chat bot. They also utilize **Artificial Intelligence Markup Language**, **Markov chain models**, and **Artificial Neural Networks**. Our chat bot needs to utilize complex techniques to Generate responses or retrieve from repositories.

III. Proposed Approaches

We propose the development of an AI chatbot that leverages NLP and a comprehensive vehicle database to provide users with a user-friendly solution to searching for a car that suits their needs and preferences. To achieve this, we propose an incremental approach to development. The first priority is the creation of a chatbot that uses simple inputs like Yes and No from customers to search results from a database. Once this basic model of the chatbot is created, we can proceed to implementing more complex and user-friendly AI features similar to ChatGPT. The AI can interact with customers in a more real and dynamic setting.

A. Basic Model

To achieve a basic model of our chatbot, we propose the creation of a chatbot similar to *Akinator* [1]. To achieve this, we have found a tutorial on how to create such an application [3]. Using this tutorial as our basis, we can expand upon it and customize it to work with cars instead of characters. For our chatbot to have an accurate and comprehensive database of vehicles to search through, we propose the use of the CarAPI database [5], an automotive API created for developers that provides a free vehicle database.

This basic model of the chatbot prompts the user with a series of simple questions, such as “What is your budget?” and “Would you prefer to drive gas, hybrid, or electric?”. The chatbot processes the answers to these questions, searches the database for cars that fit the given criteria, and provides the user with suggestions that meet the requirements they provided.

This model implements the core features that we believe are integral to this project. It provides a user-friendly text-based interface that searches a reliable and comprehensive database and provides users with customized suggestions.

B. GPT Model

Using the basic model as our base, we improved its user-friendliness by leveraging modern NLP to provide a human-like and conversational interface that can process more complex input.

To achieve a human-like and conversational model of our chatbot that can process complex input, we propose the use of GPTs to create an interface similar to *ChatGPT*. This model will build off the basic model and improve upon how the user interacts with the chatbot. Through the use of GPTs, we hope to create a version of our chatbot that can process more complex user input. This will allow us to create a more conversational approach to searching for a car. For example, the user may be able to start a conversation with the chatbot by listing multiple criteria they want fulfilled, and the chatbot will process all the information and provide follow-up questions.

To accomplish this, we have investigated the use of several generative AI APIs. Google’s *Gemini* [7] and *Google AI Studio* [8]. Google offers free models of their generative AI APIs [9] which makes it an appealing and powerful option for us to leverage in the development of this project. However, we decided that *ChatGPT* is a better known and better documented and easy to use product. While OpenAI’s custom GPTs look to be the most promising and well-known solution available, the custom GPT service is a premium service [10] and as such is not a feasible option for us. Instead, we propose the use of the OpenAI API, which offers access to several pretrained GPT models that are used by *ChatGPT*. We have decided to use the gpt-4o model, the latest model trained for public release by OpenAI.

C. Natural Language Processing

The Chatbot uses NLP (Natural Language Processing) to communicate with customers. Conversations are converted from texts into meaningful information that can power automated search of Data Sets. Chatbot is objective and efficient because it uses NLP to filter out subjective dialogues.

Customer’s queries are tokenized and dissembled into units of texts. They are lemmatized to the root form and changed to lowercase. Punctuations, special characters, and numbers are removed. The texts are transformed into structured data. Word embeddings capture the semantic relationship of texts. The meaning of texts is analyzed by dependent and semantic parsing, they identify the grammar and tone of the conversation. The Chatbot uses Natural Language Toolkit in Python programming language as a conversation medium.

D. Evaluation Metrics

To properly evaluate the performance of the proposed chatbot, we propose the measurement of several metrics to observe both the performance and the usefulness of the chatbot. These metrics can be compared between several search methods, such as manual search, the basic chatbot model, and the GPT chatbot model. The evaluation metrics are as follows.

1) Accuracy of Suggestions

The chatbot’s suggestions must match the user’s queries. We track the number of suggestions that are irrelevant to the user’s input. This number can then be compared to the ideal number of errors (0).

2) Response Time

The chatbot should be engaging to use, and as such needs to have a quick response time. We track the response time of both chatbot models for comparison.

3) Intent Recognition

Since we want to employ NLP in our chatbot, we want to ensure that the chatbot understands the user’s intent. For example, if the user states they are searching for a “family car”, it will return results of cars with larger numbers of seats, safety features, and non-sport cars. We track the number of confused results returned by the chatbot in order to analyze this metric.

4) Fallback Rate

We want to minimize the frequency at which the chatbot is unable to understand a user's query and has to fallback to a default, robotic response. We track the number of queries the chatbot fails to understand.

5) Search Simplicity

The proposed chatbot is designed with the intent to simplify the vehicle search process, so it is imperative that we measure the usefulness of the system in that regard. We propose the measurement of simplicity by tracking the amount of queries the user has to make to get the same result across different search methods.

E. Technique

The chatbot utilizes reinforcement learning from human feedback. In the real-world environment, a car salesman needs to ask many questions to a potential customer for a proper purchase. The customer provides feedback to shape types of cars that will be offered by the salesman. In reward shaping, pseudo-rewards can be physical cues from nodding heads and smiling for making positive progress in the search. In hierarchical reinforcement learning, a long sequence of action is broken down into smaller actions until the learning becomes easy. The chatbot starts as a partial program that outlines the behavior, primitives for unspecified choices are added in. "While true do choose(A(s))." The chatbot uses the learning process of how choices are made to work out the details of the complete program.

IV. System Design

We propose a system that integrates an AI chatbot with a vehicle database. To accomplish this, we have investigated several tools and libraries that assisted in the development of this project. We outline our proposed system design choices below.

A. Architecture

Our proposed system architecture consists of a chatbot portion and a database searching portion. The chatbot is programmed as described in sections III.A and III.B. It is able to process either simple input in response to questions, or complex input in the form of conversational phrases. After processing the user's input, the chatbot uses this information to search the database.

The database consists of a number of cars with their features and specifications. The chatbot is able to search this database and narrow down the results using the answers provided by the user. These results are then returned to the chatbot to display to the user.

B. Languages

The go-to language for AI development in recent years has been Python. Its ease of use combined with extensive libraries tailored for AI and machine learning make it an ideal language for AI development. The tutorials we have found also make use of Python, and *Gemini*'s API includes a Python library. Python also includes tools for interoperability with C/C++, which makes it useful to us in the case that we need to implement a high-performance section of code in a language we are familiar with. As such, Python is our language of choice for the development of this project.

We also use HTML, CSS, and JavaScript to create and design the webpage. HTML provides us with the main structure and content of the webpage, CSS gives us the style and design, and JavaScript gives us the ability to integrate the functionality of the chatbot into our website.

C. Software Tools

We have outlined the tools and libraries in the proposal of this project. Here are our choices and our reasoning for utilizing them.

1) ChatGPT & OpenAI API

ChatGPT is a generative AI tool that offers their API service with several of their GPT models. It boasts the ability to build agents for contexts such as data exploration, content searching, and assistant creation [11], which gives us the confidence that *ChatGPT* provides us with the means to create the chatbot we envision. In addition to potentially providing a conversational interface for our chatbot, it has the potential to help us search through our database.

ChatGPT is also a generative AI tool offering free use of their API services and its ability to answer complex answers and write code as needed. This tool is primarily used for code editing, but fixing, and conversion of straight code into something that works within the website itself or an executable app.

D. Data Sets

For our reliable and comprehensive vehicle database, we are employing the use of CarAPI's free vehicle database. The free database is limited to the years 2015-2020, but this is more than enough for us to demonstrate the use and utility that our chatbot can provide. CarAPI provides a free comma-separated value (CSV) file as well as the ability to query and retrieve vehicles using the API in a JavaScript object notation (JSON) format. This acts as the library the AI pulls from when deciding what car is the right one for you.

The dataset we used is a free CSV from CarAPI containing information on all vehicles sold in the years 2015-2020. The dataset contains 17,662 vehicle records and includes 57 features. These features are categorized into the "Year, Make, Model, Trim", "Body Specs", "Engine Specs", and "Mileage Specs" categories. Table 1 details the features in these categories in full.

TABLE 1. FEATURES OF THE DATASET

Feature Category	**Features**
Year, Make, Model, Trim	Make ID, Make Name, Model ID, Model Name, Trim ID, Trim Year, Trim Name, Trim Description, Trim MSRP, Trim Invoice, Trim Created, Trim Modified
Body Specs	Body ID, Body Type, Body Doors, Body Seats, Body Length, Body Width, Body Height, Body Wheel Base, Body Front Track, Body

	Rear Track, Body Ground Clearance, Body Cargo Capacity, Body Max Cargo Capacity, Body Curb Weight, Body Gross Weight, Body Max Payload, Body Max Towing Capacity
Engine Specs	Engine ID, Engine Type, Engine Fuel Type, Engine Cylinders, Engine Size, Engine Horsepower HP, Engine Horsepower RPM, Engine Torque Ft Lbs, Engine Torque RPM, Engine Valves, Engine Valve Timing, Engine Cam Type, Engine Drive Type, Engine Transmission
Mileage Specs	Mileage ID, Mileage Fuel Tank Capacity, Mileage Combined MPG, Mileage EPA City MPG, Mileage EPA Highway MPG, Mileage Range City, Mileage Range Highway, Mileage EPA Combined MPG Electric, Mileage EPA City MPG Electric, Mileage EPA Highway MPG Electric, Mileage Range Electric, Mileage EPA kWh 100 Mi Electric, Mileage EPA Time to Charge Hr 240V Electric, Mileage Battery Capacity Electric

E. Data Preprocessing

Due to the large number of features present in this dataset, we decided to only use search for results from this dataset using the most relevant features. We believe these features to be simple and straightforward enough for the average customer to understand, while also being detailed enough to find a good match for the user's search queries. This means removing features such as "Body Ground Clearance", "Body Gross Weight", and "Trim Created". These are features that the average consumer does not pay attention to. Instead, we focus our search queries on features such as "Make Name", "Trim MSRP", and "Mileage". Features such as these are common points of comparison between vehicles and are important features to search and compare. The following table details the narrowed list of features in the dataset.

TABLE 2. NARROWED FEATURES OF THE DATASET

Feature Category	**Features**
Year, Make, Model, Trim	Make Name, Model Name, Trim Year, Trim Name, Trim MSRP
Body Specs	Body Type, Body Doors, Body Seats
Engine Specs	Engine Fuel Type, Engine Cylinders
Mileage Specs	Mileage Combined Mpg (Gas + Electric)

As shown in Table 2, this narrowed dataset is much simpler while still retaining key information that users are interested in when searching for a vehicle. Using this narrowed set of features allows us to simplify the search parameters for the user while still retaining enough feature information to find a specific vehicle to match the user's queries.

Even with the narrowed list of features, there are still 11 features and 17,662 entries in this dataset. This dataset is too large to feasibly search using a service like ChatGPT. In order to efficiently search through this dataset, we propose some initial sorting of the dataset by splitting them into several, smaller datasets. These smaller datasets split the dataset into datasets that consist of vehicles that have the same make and body type. This allows us to still allow the user to search for vehicles of certain makes and body types, while also allowing us to filter our dataset so that the AI can efficiently search it.

After preprocessing the dataset, we ended up with 205 datasets, organized into 45 different makes. Each dataset has 9 features and at most 200 entries. This is a much more manageable size of dataset to use for our chatbot.

F. Data Analysis and Manipulation

The Artificial Intelligence chatbot uses Pandas to organize the data using Python. Pandas is a library used for data indexing, axis labeling, and alignment. It is used to read and write CSV and text files, Microsoft Excel, SQL databases, and fast HDF5 format. There is data alignment and integrated handling of missing data. A very large data set can be intelligently indexed. The code is optimized for efficiency.

There are Series and DataFrame in data structure. Series is a one-dimensional labeled array of type integer, string, float, and objects. It can be a Python dict, an ndarray, and a scalar value. The operations offered by Pandas include Vectorized operations and label alignment. DataFrame can be a 2-dimensional labeled data of Series, 2-D numpy.ndarray, structured or record ndarray, and another DataFrame. The Pandas operations are used for data indexing and column selection.

G. Graphical User Interface

The user interacts with the chat bot in a Graphical User Interface (GUI). This is achieved through a web browser accessing our repository on GitHub Pages. For this to work the web browser implements CSS for stylistic choices. Through CSS, designs for the web page would include things like colors, fonts, positioning, display options, interactive menus, and an indicator of where the chatbot is and how to interact with it. It also uses MVC's to be able to handle users logging in. The GUI acts as front-end development and makes it easier for the user to interact with the chat bot. As of right now it is through command line interface (CLI) which isn't necessarily user friendly. The challenge that may occur because of this is backend and front-end communication.

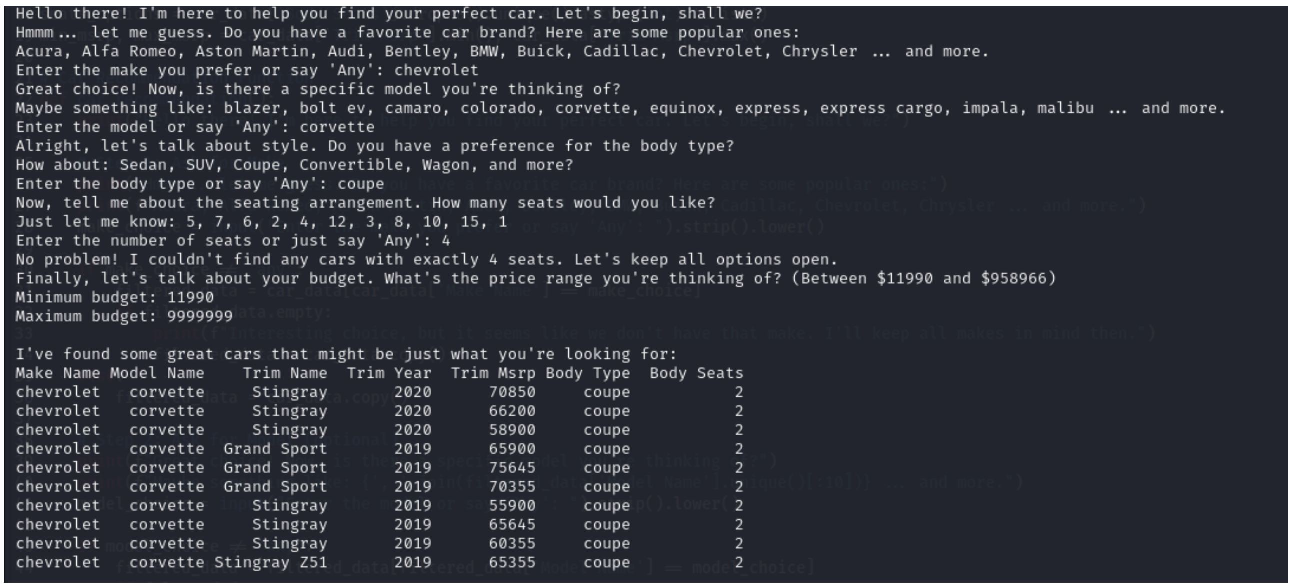
Both programs need to be able to read and identify data being output and stored during action phases. The tools needed for this to work are IDEs like VS code. As in past projects VS code was able to handle HTML pages while also using python scripts to perform actions. Although more of a stylistic choice to pair with an AI project, it is important to make the process seamless and simple, as potential users may be more attracted to GUIs over CLIs.

V. IMPLEMENTATION

A. Basic Chatbot

The first version of our chatbot as described in section III.A is implemented as a basic Akinator-style chatbot and is based off the previously mentioned Akinator tutorial [3] and uses the CarAPI database [5]. It prompts the user with a series of inquiries, which it uses to search the database and narrow down the vehicle options that it presents to the user as a result. As shown in Figure 1, this version of the chatbot can successfully deliver search results to the user while being fault-tolerant enough to prevent a lack of prior knowledge by the user from negatively affecting the search results.

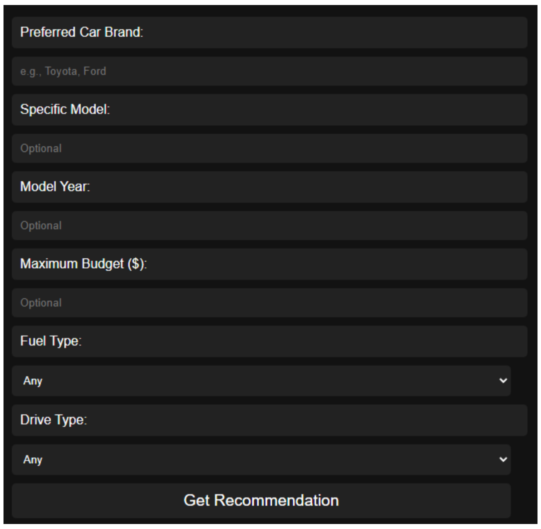
The webpage interface of this basic version of the chatbot is shown in Figure 2. It takes the form of a basic questionnaire, which simulates the question-and-answer nature of the program in an easily testable format. As our focus is the GPT Chatbot model, we wanted to quickly deploy an easily testable interface instead of spending time on an interface that would not be used. Using this as a test and basis for our chatbot, we moved forward with the integration of the chatbot features and AI.



```
Hello there! I'm here to help you find your perfect car. Let's begin, shall we?
Hmmm... let me guess. Do you have a favorite car brand? Here are some popular ones:
Acura, Alfa Romeo, Aston Martin, Audi, Bentley, BMW, Buick, Cadillac, Chevrolet, Chrysler ... and more.
Enter the make you prefer or say 'Any': chevrolet
Great choice! Now, is there a specific model you're thinking of?
Maybe something like: blazer, bolt ev, camaro, colorado, corvette, equinox, express, express cargo, impala, malibu ... and more.
Enter the model or say 'Any': corvette
Alright, let's talk about style. Do you have a preference for the body type?
How about: Sedan, SUV, Coupe, Convertible, Wagon, and more?
Enter the body type or say 'Any': coupe
Now, tell me about the seating arrangement. How many seats would you like?
Just let me know: 5, 7, 6, 2, 4, 12, 3, 8, 10, 15, 1
Enter the number of seats or just say 'Any': 4
No problem! I couldn't find any cars with exactly 4 seats. Let's keep all options open.
Finally, let's talk about your budget. What's the price range you're thinking of? (Between $11990 and $958966)
Minimum budget: 11990
Maximum budget: 9999999

I've found some great cars that might be just what you're looking for:
Make Name Model Name    Trim Name  Trim Year  Trim Msrp Body Type  Body Seats
chevrolet   corvette     Stingray     2020      70850     coupe          2
chevrolet   corvette     Stingray     2020      66200     coupe          2
chevrolet   corvette     Stingray     2020      58900     coupe          2
chevrolet   corvette  Grand Sport     2019      65900     coupe          2
chevrolet   corvette  Grand Sport     2019      75645     coupe          2
chevrolet   corvette  Grand Sport     2019      70355     coupe          2
chevrolet   corvette     Stingray     2019      55900     coupe          2
chevrolet   corvette     Stingray     2019      65645     coupe          2
chevrolet   corvette     Stingray     2019      60355     coupe          2
chevrolet   corvette Stingray Z51     2019      65355     coupe          2
```

Fig. 1. Basic Version of Chatbot



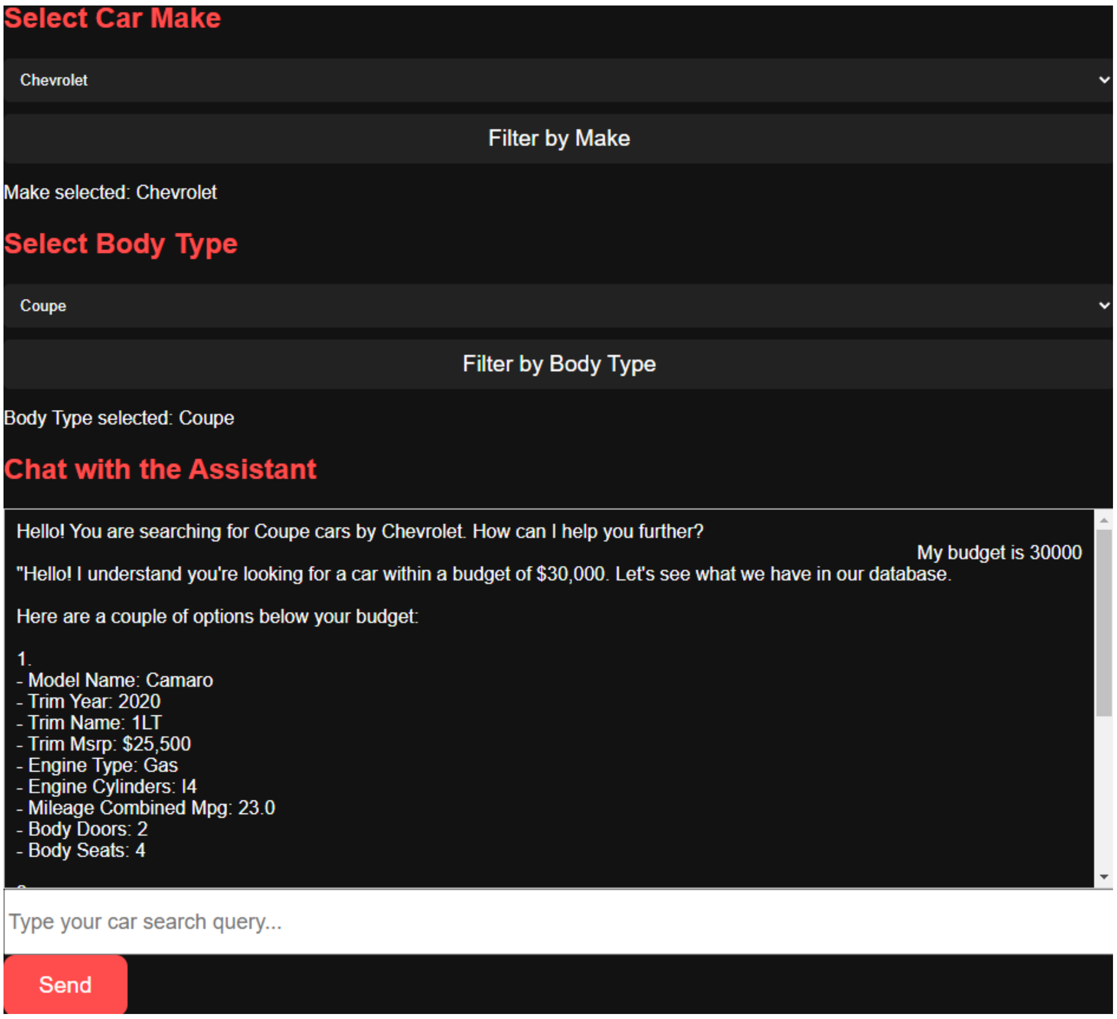
Preferred Car Brand:	e.g., Toyota, Ford
Specific Model:	Optional
Model Year:	Optional
Maximum Budget ($):	Optional
Fuel Type:	Any
Drive Type:	Any

Fig. 2. Basic Model Interface

B. GPT Chatbot

The second version of our chatbot as described in section III.B is implemented as a ChatGPT style chatbot. It integrates the OpenAI API and uses the ChatGPT gpt-4o model to search the database and provide answers to user queries. It prompts the user with a series of questions in order to find a specific dataset that the user wants to search from. This dataset corresponds to a list of vehicles with a specific make and body type. After the dataset is selected through questioning, the user is able to ask the chatbot any query and it is answered using ChatGPT's NLP. As shown in Figure 3, since our chatbot leverages ChatGPT's NLP, it is able to handle queries in the form of natural language.

This chatbot is hosted on our GitHub Pages website, and the frontend is coded using HTML, CSS, and JavaScript. When the chatbot is accessed, the user is first prompted to select their desired car make and body type. This is done by the user to assist the chatbot in filtering which dataset to use, which was split into multiple subsets as discussed in section IV.E. The chosen vehicle brand and body type are then sent to the backend, where it processes the information and selects the corresponding CSV file. Once the CSV file is chosen, the chat box is displayed and from there the user can chat with the chatbot. The chatbot is implemented using OpenAI's gpt-4o model and their OpenAI API. The chatbot is instructed to engage with the user in a specific manner and to display search results in a specific format. Since it leverages OpenAI's ChatGPT model, it makes use of their pretrained model to perform NLP to process human language queries. It implements session history and chat history by using cookies, so that the user can continuously chat with the chatbot and refine their search results.



Make selected: Chevrolet

Body Type selected: Coupe

Hello! You are searching for Coupe cars by Chevrolet. How can I help you further?

My budget is 30000

"Hello! I understand you're looking for a car within a budget of $30,000. Let's see what we have in our database.

Here are a couple of options below your budget:

1.
- Model Name: Camaro
- Trim Year: 2020
- Trim Name: 1LT
- Trim Msrp: $25,500
- Engine Type: Gas
- Engine Cylinders: I4
- Mileage Combined Mpg: 23.0
- Body Doors: 2
- Body Seats: 4

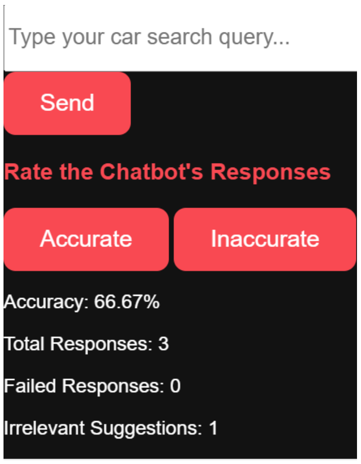
Fig. 3. GPT Version of Chatbot

VI. Results

To gather evaluation metrics there needed to be some adjustments to the original page, as well as the backend program. For the test version there is a time script, automatic relevancy function, Failed response tracker, total response tracker, and irrelevant response tracker. The purpose of these implementations was to gather data useful to testing the program.

A. Accuracy of Suggestions

The chatbots suggestions must match the user's queries. To evaluate this, we needed a way to measure and analyze the chatbot's responses. This is where the automatic relevancy function comes into use. It analyzes both user input and chatbot output to determine if there are matching words and items in both texts. If deemed irrelevant then irrelevancy tracker is incremented. There is also a manual element to this as well where the user can determine if a response is accurate or not through the "accurate" and "inaccurate" buttons. If not deemed relevant, then the irrelevant tracker is incremented.



Accuracy: 66.67%

Total Responses: 3

Failed Responses: 0

Irrelevant Suggestions: 1

Fig. 4. Test GPT Version of Chatbot

Figure 4 is displaying the result of 3 queries with one being deemed as "Inaccurate". Which in turn increases the irrelevant suggestions and affects the accuracy of the responses. (This is an example of this function working).

B. *Response Time*

The chatbot should be engaging to use and as such needs to have quick response times. This is a very important factor in gauging the useability of the chatbot. To record this a time tracker is implemented to show how long it takes for the chatbot to search the database and come up with relevant responses to the user. For the test there were ten queries that were executed.

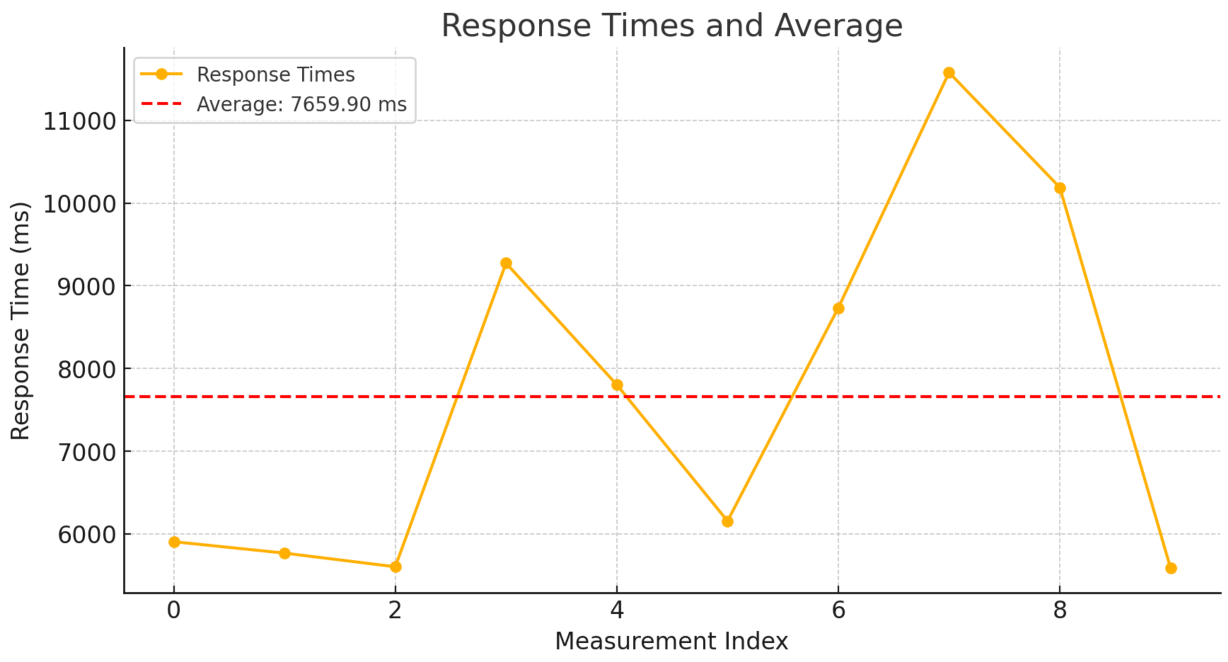
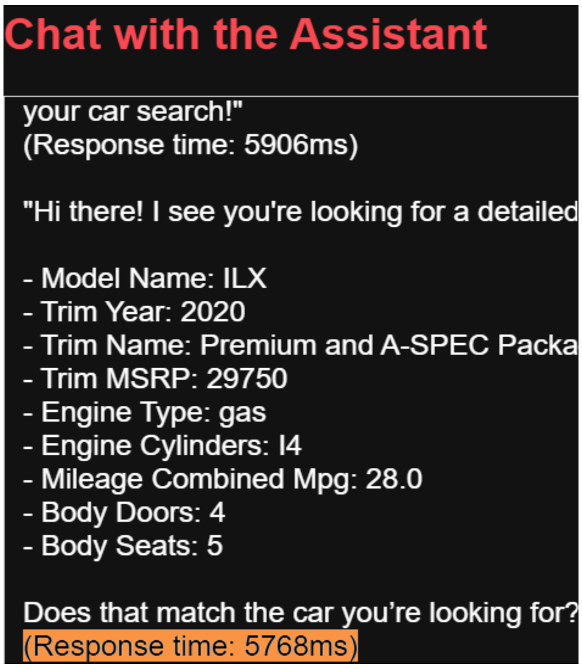


Fig. 6. Test Data of Response Times



Chat with the Assistant

your car search!"
(Response time: 5906ms)

"Hi there! I see you're looking for a detailed

- Model Name: ILX
- Trim Year: 2020
- Trim Name: Premium and A-SPEC Packa
- Trim MSRP: 29750
- Engine Type: gas
- Engine Cylinders: I4
- Mileage Combined Mpg: 28.0
- Body Doors: 4
- Body Seats: 5

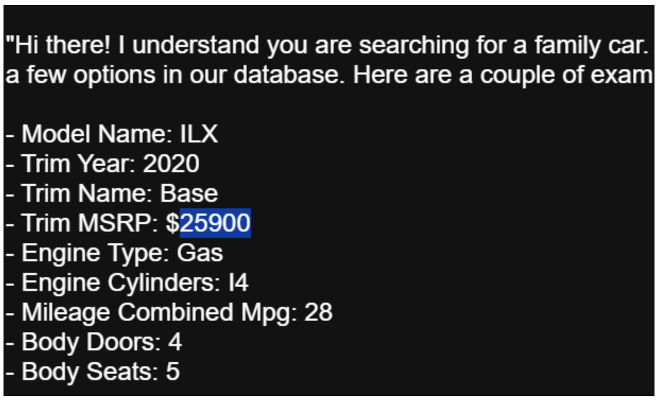
Does that match the car you're looking for?
(Response time: 5768ms)

Fig. 5. Response Time Test GPT Version

Figure 5 shows how the response time is integrated into the front-end code. This is so the test user can record how long it took for the chatbot to query the database; for relevant information and create an accurate response. Figure 6 is a graph of 10 queries that were recorded during the test. The first 2 queries were made for rows close to the beginning of the table. Most of the queries were made in random sections not near the beginning of the table. It seems the further down an item is in the table the longer the response time may be. The average response time for the 10 test queries was 7659.9 ms.

C. *Intent Recognition*

Since we want to employ NLP in our chatbot, we want to ensure that the chatbot understands the user's intent. To test this feature, we chose to use the query "Family car." The chat bot was able to recognize this request and recommended the "best" option on this criterion: Affordability, seat number, and milage statistics. It reasoned that a good family car would have 4-5 seats. It responded with the most affordable option within the specified database table. It also searched for "good" mileage statistics.



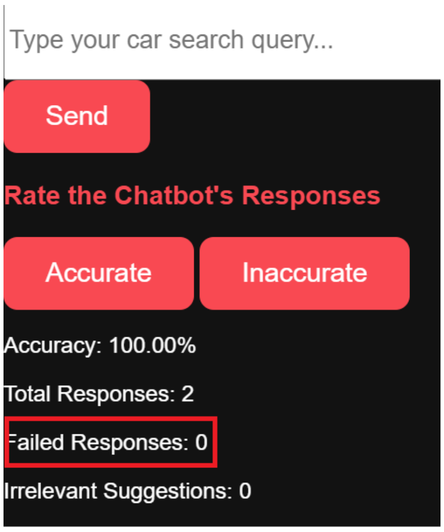
"Hi there! I understand you are searching for a family car.
a few options in our database. Here are a couple of exam

- Model Name: ILX
- Trim Year: 2020
- Trim Name: Base
- Trim MSRP: $25900
- Engine Type: Gas
- Engine Cylinders: I4
- Mileage Combined Mpg: 28
- Body Doors: 4
- Body Seats: 5

Fig.7. NLP Test GPT Version

D. Fall Back Rate

We want to minimize the frequency at which the chatbot is unable to understand a user's query and has to fallback to a default, robotic response. To record if there were any failed requests the failed responses tracker was implemented. The way the test works is that if the chatbot is unable to produce a response or there is a network error the backend program will send an error message. Once this error message is read by the front end the failed response tracker is increased. Fortunately, there were no failed responses during testing.



Accuracy: 100.00%

Total Responses: 2

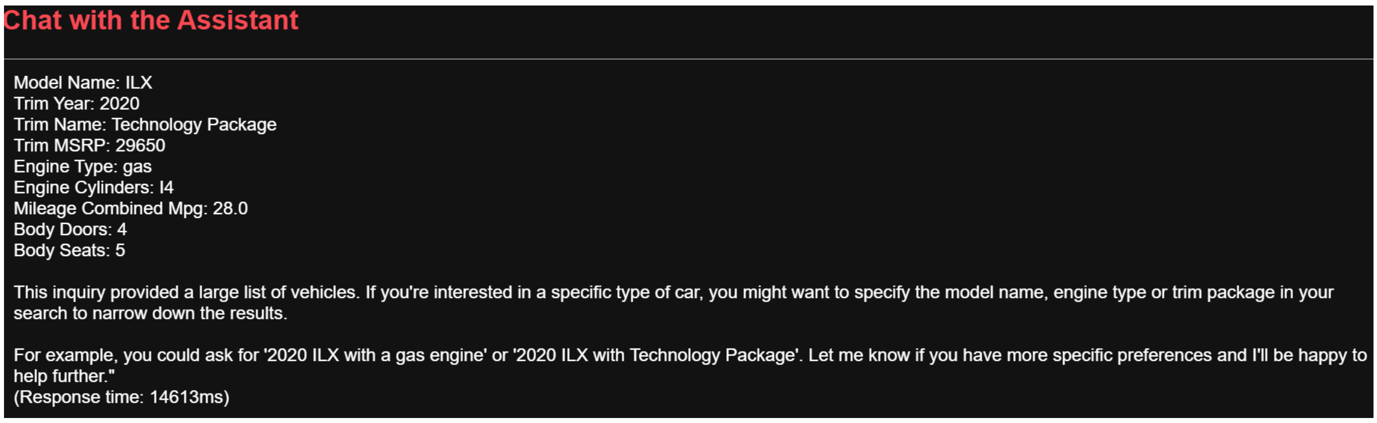
Failed Responses: 0

Irrelevant Suggestions: 0

Fig. 8. Fall Back Rate Test GPT Version

E. Search Simplicity

The proposed chatbot is designed with the intent to simplify the vehicle search process, so it is imperative that we measure the usefulness of the system in that regard. To test this feature, we used broad search queries, for example: Only putting in the year of a model we wanted. That isn't enough for the chatbot to recommend a specific option, instead it would just list cars in that year. So instead, it guides the user into narrowing down the search so that they would be able to select a specific model.


Chat with the Assistant

Model Name: ILX
Trim Year: 2020
Trim Name: Technology Package
Trim MSRP: 29650
Engine Type: gas
Engine Cylinders: I4
Mileage Combined Mpg: 28.0
Body Doors: 4
Body Seats: 5

This inquiry provided a large list of vehicles. If you're interested in a specific type of car, you might want to specify the model name, engine type or trim package in your search to narrow down the results.

For example, you could ask for '2020 ILX with a gas engine' or '2020 ILX with Technology Package'. Let me know if you have more specific preferences and I'll be happy to help further."
(Response time: 14613ms)

Fig.9. Search Simplicity Test GPT

VII. Conclusion

In this project we proposed a potential solution to ease the vehicle purchase process for consumers. There are multiple distinguishing features of each vehicle, which makes finding a vehicle that suits a consumer's unique needs can be a difficult process. To assist consumers in this process, we proposed the development of a chatbot to assist them in searching for a vehicle that suits their needs. In our research, we found that consumers who had little prior vehicle knowledge had to do the most additional searching and benefitted the most from performing search activities. This is why we believe that by creating a chatbot that is accessible and user friendly, we can help the consumers that can benefit the most from assistance in the vehicle search process.

Our chatbot architecture consists of two major parts: the AI chatbot that engages with the user and assists in searching the database, and the curated database that has reliable information. For the chatbot we utilized OpenAI's API to create an interface similar to ChatGPT, and for the database we utilized CarAPI's free sample database. Using this architecture, we were able to successfully deploy a chatbot to our website that can assist users in searching for vehicles. While the database sample is limited to the years 2015-2020, the concept has been proven, and the dataset could be easily swapped out for a more recent one.

VIII. References

[1] Elokence, “Everything you’ve always wanted to know about Akinator,” Akinator.com, 2024. https://en.akinator.com/content/6/Everything-you-039-ve-always-wanted-to-know-about-Akinator (accessed Sep. 29, 2024).

[2] “Introducing ChatGPT,” OpenAI, Nov. 30, 2022. https://openai.com/index/chatgpt/ (accessed Sep. 28, 2024).

[3] addykoder, “Youtube_Tutorial/Akinator.py at master · addykoder/Youtube_Tutorial,” GitHub, 2021. https://github.com/addykoder/Youtube_Tutorial/blob/master/Akinator.py (accessed Sep. 29, 2024).

[4] Cars.com, “New Cars, Used Cars, Car Dealers, Prices & Reviews | Cars.com,” Cars.com, 2019. https://www.cars.com/

[5] CarAPI. https://carapi.app/ (accessed Sep. 28, 2024).

[6] H. Alkaissi and S. McFarlane, “Artificial hallucinations in ChatGPT: Implications in scientific writing,” Cureus, vol. 15, no. 2, Feb. 2023, doi: https://doi.org/10.7759/cureus.35179.

[7] “Build with the Gemini API,” Google AI for Developers. https://ai.google.dev/

[8] S. Pichai and D. Hassabis, “Introducing Gemini: our largest and most capable AI model,” Google, Dec. 06, 2023. https://blog.google/technology/ai/google-gemini-ai/

[9] “Gemini API Pricing,” Google AI for Developers. https://ai.google.dev/pricing

[10] OpenAI, “ChatGPT ,” ChatGPT, 2024. https://chatgpt.com/

[11] OpenAI, “OpenAI Platform,” Openai.com, 2024. https://platform.openai.com/docs/overview

[12] G. N. Punj and R. Staelin, “A Model of Consumer Information Search Behavior for New Automobiles,” Journal of Consumer Research, vol. 9, no. 4, p. 366, Mar. 1983, doi: https://doi.org/10.1086/208931.

[13] Q. Ai, Y. Zhang, K. Bi, X. Chen, and W. Bruce Croft, “Learning a Hierarchical Embedding Model for Personalized Product Search,” International ACM SIGIR Conference on Research and Development in Information Retrieval, Aug. 2017, doi: https://doi.org/10.1145/3077136.3080813.

[14] S. Reshmi and K. Balakrishnan, “Implementation of an inquisitive chatbot for database supported knowledge bases,” Sādhanā, vol. 41, no. 10, pp. 1173–1178, Oct. 2016, doi: https://doi.org/10.1007/s12046-016-0544-1.

[15] Iván Cantador, Jesús Viejo-Tardío, M. E. Cortés-Cediel, and M. Pedro, “A Chatbot for Searching and Exploring Open Data: Implementation and Evaluation in E-Government,” Biblos-e Archivo (Universidad Autónoma de Madrid), Jun. 2021, doi: https://doi.org/10.1145/3463677.3463681.

[16] B. Pitel et al., “63. An introduction to publicly available AI-assisted chatbot-style search engines for cancer variant curation,” Cancer Genetics, vol. 286–287, pp. S20–S20, Aug. 2024, doi: https://doi.org/10.1016/j.cancergen.2024.08.065.

[17] IBM. (2024, October 25). *What is NLP (Natural Language Processing)?*. IBM. https://www.ibm.com/topics/natural-language-processing

[18] CarGPT, “CarGPT,” CarGPT. [Online]. Available: <https://www.cargpt.ai/>. [Accessed: Oct. 27, 2024].

IX. Appendices

A. *Appendix A - Task Assignments*

1) *David Del Moral*

- Paper writing
  - Related works
- Similar implementation search
- Create one-page summary

2) *Ricky Godsey*

- Paper writing
  - Proposed approaches
  - Results

3) *Ean Shi*

- Paper writing
  - Propose approaches
  - System design

4) *Zachary Kao*

- Paper writing
  - Related works
  - Proposed approaches
  - System design
  - Dataset Analysis
  - Proposed metrics
  - Implementation
  - Conclusion
- Scholarly research search
- App development
  - Create backend logic for chatbot AI
  - Host backend logic on server
  - Create frontend for chatbot

5) *Matthew Tomme*

- Webpage setup
- Implement prototype of basic version of chatbot
- Record video demonstration

B. *Appendix B - Schedule*

- Deadline 1: EOD September 29, 2024
  - Project Proposal
  - 3 pages
  - Project Webpage
- Deadline 2: EOD October 27, 2024
  - Project Checkpoint 1
  - 5 pages
  - Progress reports from each member
- Deadline 3: EOD November 24, 2024
  - Project Checkpoint 2
  - 7 pages
  - Progress reports from each member
- Deadline 4: EOD December 8, 2024
  - Final Project Report
  - 10 pages
  - YouTube video
  - Project presentation slides
  - One-page project summary
  - Project source code/data/readme
  - Peer evaluations
- Deadline 5: Dec 12, 2024
  - Project Presentation
  - 15-20 minute presentation including Q&A
  - 5 minute YouTube video demonstrating the application

C. Appendix C – Progress Report 1

1) David Del Moral

For checkpoint 1, I conducted online searches for related works, more specifically, products or concepts that currently exist that share most of the criteria for our project. I described what they do and how it compares to our ideal implementation.

2) Ricky Godsey

For checkpoint 1, I researched chat bots. This includes the different types of models and techniques they use to generate responses. Familiar terms like Neural Networks and self-learning are integral parts of successful chat bots. I expanded upon the proposed approaches section.

3) Ean Shi

For checkpoint 1, I have read about Natural Language Processing and described it in the paper.

4) Zachary Kao

For checkpoint 1, I performed searches for related works on Google Scholar and found relevant studies that pertained to consumer vehicle searches, personalized product searches, and the use of chatbots in searching for information. Using these sources, I expanded upon the related works section.

5) Matthew Tomme

For checkpoint 1, I worked on and created a proof of concept for the Akinator styled car finder. I also updated the website to include the updated documentation on said car finder and got the image included in the document.

D. Appendix D – Progress Report 2

1) David Del Moral

2) Ricky Godsey

For checkpoint 2, researched about the graphical user interface of the project. The tools we will need to make this work are Vs code and we will need to use CSS styles to implement visually appealing components. If we plan to use logins and saves for past chats MVCs (Multi-View-Controllers) will need to be implemented as well.

3) Ean Shi

For checkpoint 2, I have researched Data Structure and the open-source library Pandas used to read and use the CSV file.

4) Zachary Kao

For checkpoint 2, I analyzed the dataset and its features and wrote about it in the paper. I also proposed certain metrics by which we can evaluate the proposed system.

5) Matthew Tomme

For checkpoint 2, I worked on advancing the code and getting the first bits of AI implemented but not functional yet. More advanced reading of the .csv file was added, and this allows for more accurate answers to be obtained and also for the AI to properly read the file once it is working.