

Second-Hand Car Valuation User Manual

* Birgül Yüksel - 211805066
* Betül Yılmaz - 211805064
* Derya Nazlıcan Akannaç – 211805019
* <https://github.com/birgulyuksel/DjangoProjem>
* <https://github.com/betulylmaz/DjangoProjem>
* <https://github.com/deryanazlican/DjangoProjem>

**Table of Contents**

[1 Introduction 4](#_Toc187421972)

[1.1 Brief overview of the project 4](#_Toc187421973)

[1.2 Objectives and goals 4](#_Toc187421974)

[1.3 Scope of the project 4](#_Toc187421975)

[2 System Requirements 4](#_Toc187421976)

[2.1 Hardware Requirements (e.g., CPU, RAM, Storage, etc.) 4](#_Toc187421977)

[2.2 Software Requirements (e.g., Operating System, Development Tools, Libraries) 5](#_Toc187421978)

[3 Installation Guide 5](#_Toc187421979)

[3.1 Step-by-step instructions for installing the project 5](#_Toc187421980)

[4 User Interface Overview 18](#_Toc187421981)

[4.1 Description of the main screens or pages 18](#_Toc187421982)

[4.1.1 Home Page (home.html): (Figure 25) 18](#_Toc187421983)

[4.1.2 Trend Analysis Page (trend-analysis.html) (Figure 26) 18](#_Toc187421984)

[4.1.3 Prediction Page (predict.html) (Figure 27) 19](#_Toc187421985)

[4.1.4 Collected Data Page (collected-data.html) (Figure 28) 21](#_Toc187421986)

[4.2 Explanation of the navigation structure 21](#_Toc187421987)

[4.3 Key buttons and their functions 22](#_Toc187421988)

[4.3.1 Veri Çek (Fetch Data) Button (Figure 30) 22](#_Toc187421989)

[4.3.2 Tahmin Et (Predict) Button 23](#_Toc187421990)

[4.3.3 Filtrele (Filter) Button 23](#_Toc187421991)

[5 Features and Functionalities 23](#_Toc187421992)

[5.1 List of Primary Features 23](#_Toc187421993)

[5.1.1 Listing Vehicle Data 24](#_Toc187421994)

[5.1.2 Filtering Process 24](#_Toc187421995)

[5.1.3 Data Extraction Process 25](#_Toc187421996)

[5.1.4 Trend Analysis 25](#_Toc187421997)

[5.1.5 Estimated Price Determination: 26](#_Toc187421998)

[5.2 How each feature works? 26](#_Toc187421999)

[5.2.1 How Listing Vehicle Data Works? 26](#_Toc187422000)

[5.2.2 How Filtering Process Works? 28](#_Toc187422001)

[5.2.3 How Data Extraction Process Works? 30](#_Toc187422002)

[5.2.4 How Trend Analysis Works? 31](#_Toc187422003)

[5.2.5 How Estimated Price Determination Works? 32](#_Toc187422004)

[6 Usage Instructions 33](#_Toc187422005)

[6.1 Step-by-step guide on how to use the system 33](#_Toc187422006)

[6.1.1 Login to the System 33](#_Toc187422007)

[6.1.2 Home Page Usage 34](#_Toc187422008)

[6.1.3 New Data Scraping 35](#_Toc187422009)

[6.1.4 Viewing Trend Analysis 36](#_Toc187422010)

[6.1.5 Estimated Price Determination 36](#_Toc187422011)

[6.2 Examples of typical use cases 37](#_Toc187422012)

[6.2.1 Finding and Listing Vehicles 37](#_Toc187422013)

[6.2.2 Scraping New Data 38](#_Toc187422014)

[39](#_Toc187422015)

[6.2.3 Viewing Trend Analyzes 39](#_Toc187422016)

[6.2.4 Estimated Price Determination 40](#_Toc187422017)

[7 Troubleshooting and Maintenance 41](#_Toc187422018)

[7.1 Common issues and their solutions 41](#_Toc187422019)

[7.2 Error messages and how to resolve them 43](#_Toc187422020)

[8 References 44](#_Toc187422021)

# Introduction

## Brief overview of the project

The "Second Hand Car Valuation" project is designed to analyze and estimate the value of second-hand vehicles. It utilizes a pre-existing database containing detailed vehicle data, which users can filter based on attributes like brand, model, color, and more. This project offers a comprehensive solution for analyzing and estimating second-hand vehicle values while providing meaningful insights through data analysis.

## Objectives and goals

* Enable users to view and filter vehicle data efficiently.
* Provide automated trend analysis for popular brands, colors, and other preferences. Implement a prediction model to estimate vehicle prices based on inputs such as brand, model, year, and mileage.
* Facilitate dynamic data scraping for updating the database with new vehicle listings.

## Scope of the project

* **Filtering and Visualization:** Users can filter and view existing vehicle data with customizable parameters.
* ***Trend Analysis*:** Automatically generated reports and tables present insights into market trends, such as the most preferred brands or colors.
* **Price Prediction:** A machine learning model predicts vehicle prices by analyzing input parameters against the database.
* **Data Scraping:** A feature to scrape vehicle data from online sources, displaying new listings in a separate section with the same filtering options.

# System Requirements

## Hardware Requirements (e.g., CPU, RAM, Storage, etc.)

To run the "Second Hand Car Valuation" project efficiently, the following hardware specifications are recommended:

* **Processor (CPU)**: Dual-core processor with a minimum speed of 2.5 GHz (e.g., Intel Core i3 or equivalent).
* **Memory (RAM)**: At least 4 GB of RAM (8 GB recommended for better performance).
* **Storage**:
  + 500 MB free disk space for project files and dependencies.
  + Additional space may be required for large datasets.
* **Display**: A monitor with a minimum resolution of 1366x768 pixels.
* **Internet Connection**: Required for data scraping functionality and downloading dependencies.

## Software Requirements (e.g., Operating System, Development Tools, Libraries)

The following software and tools are required to set up and run the project:

* **Operating System**:
  + Windows 10 or later / Linux (Ubuntu 18.04 or later) / macOS 10.15 or later.
* **Development Environment**:
  + Visual Studio Code (with Python and HTML extensions installed).
* **Programming Languages**:
  + Python 3.8+ (for backend and data analysis).
  + HTML/CSS (for frontend, if applicable).
* **Libraries and Dependencies**:
  + - **Django:** Django.shortcuts, Django.http, Django.core.management, Django.urls.
    - **Python Standard:** json, time, pathlib.
    - **Data Analysis:** pandas, job library.
    - **Machine Learning:** sklearn.
    - **Web Scraping:** undetected\_chromedriver, selenium.
* **Browser and WebDriver**:
  + Google Chrome (latest version).
  + ChromeDriver compatible with the installed Chrome version.

This configuration ensures smooth execution of all project features, including data filtering, trend analysis, machine learning predictions, and web scraping functionalities.

# Installation Guide

## Step-by-step instructions for installing the project

In this section, the steps required for the preparation and integration of the "Second Hand Vehicle Valuation" project are explained in detail. You can install the project without any problems by following the instructions below.

**Step 1: Install Visual Studio Code (VS Code)**

1. **Download VS Code:**

Go to [Visual Studio Code](https://code.visualstudio.com/) official website.

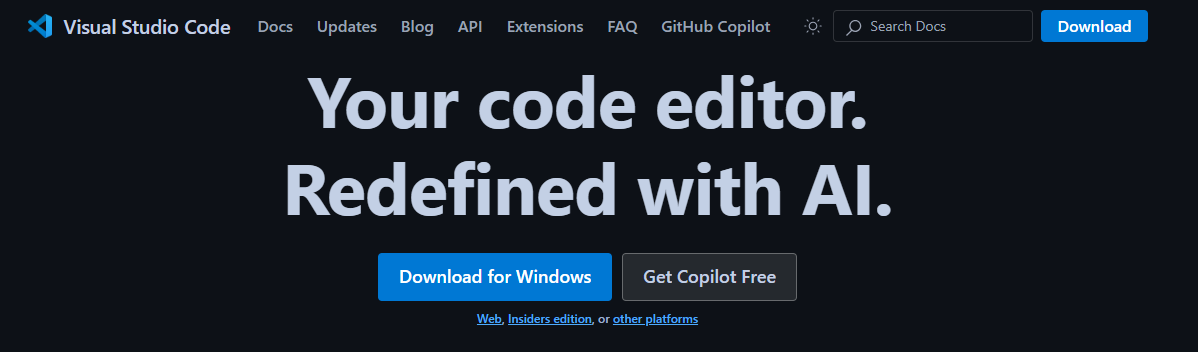
Download the version appropriate for your operating system (Windows, macOS or Linux).(Figure 1)

Figure : VS Code Website

1. **Install VS Code**:

Run the downloaded file and follow the installation wizard. (Figure 2,4)

Check the "Add to PATH" option during installation (recommended). (Figure 3)

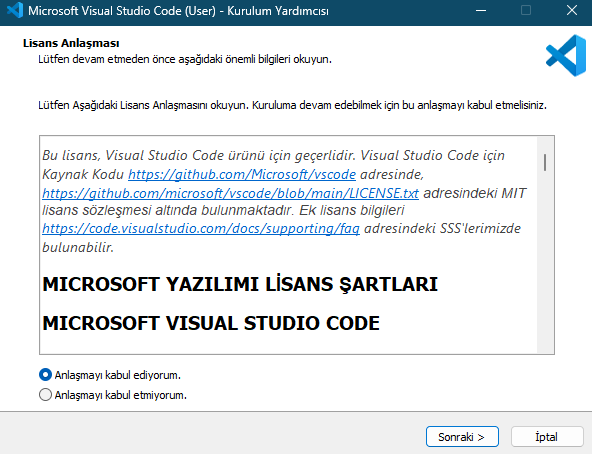


Figure : Install VS Code Page

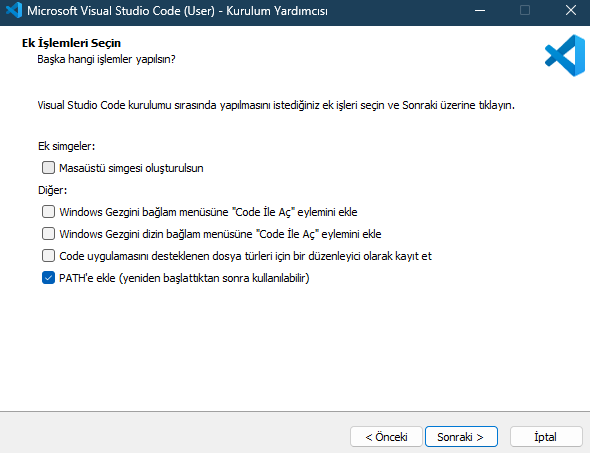


Figure : Add to Path

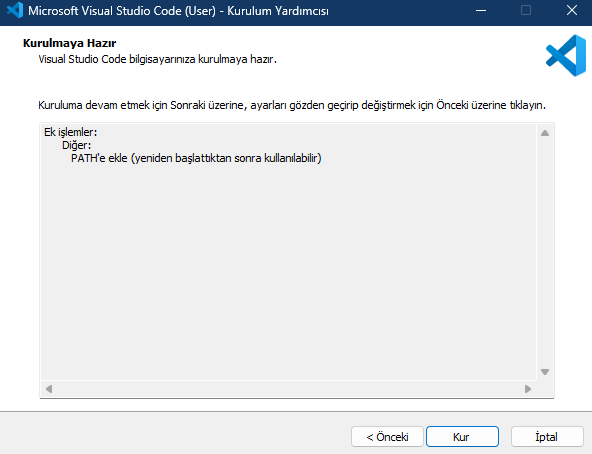
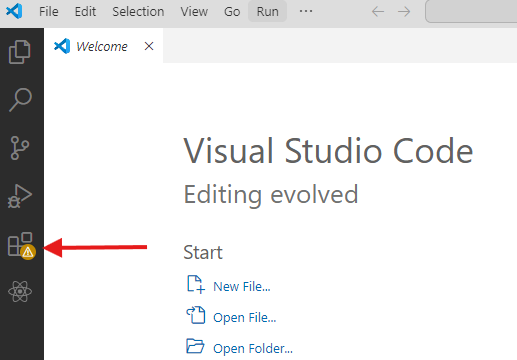


Figure : Install Vs Code Page (Continue)

1. **Install Extensions**:

Open VS Code and click the "Extensions" tab on the left. (Figure 5)

Figure : Vs Code Extensions Tab



For Python, install the "Python" extension. (Figure 6)

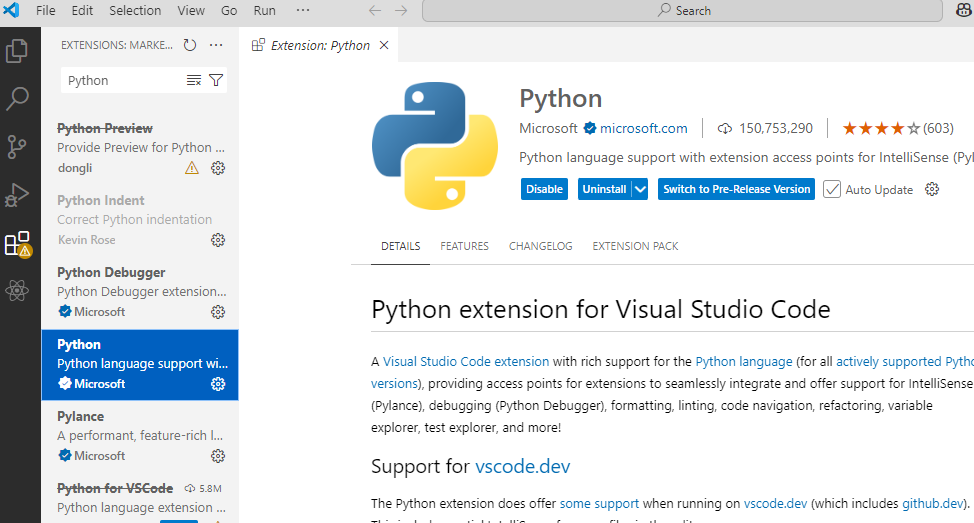


Figure : Install Python Extension

To work with HTML files, install the "HTML CSS Support" plugin. (Figure 7)

Figure :Install HTML CSS Support

**Step 2: Install Python**

1. **Download Python**:

Go to [Python official website](https://www.python.org/). (Figure 8,9)

Download Python 3.8 or newer.

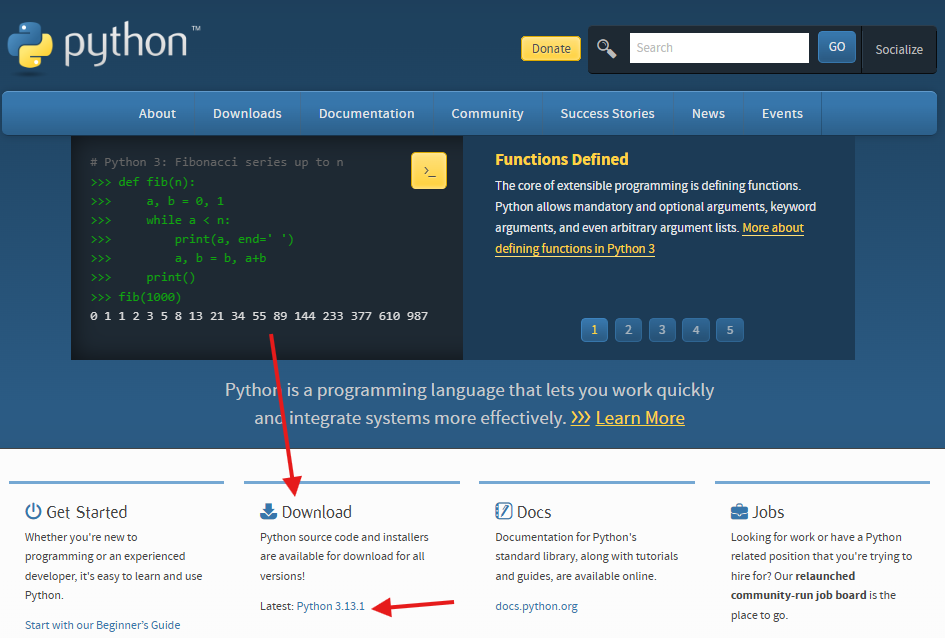


Figure 8 : Download Python

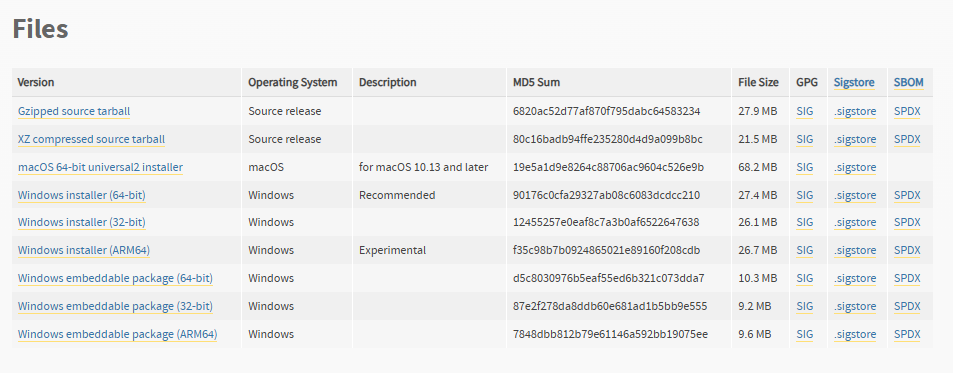
****

Figure : Download Python (Continue)

1. **Install Python:**

Run the downloaded file.

Make sure you check the "Add Python to PATH" option.

Complete the installation by following the installation wizard.

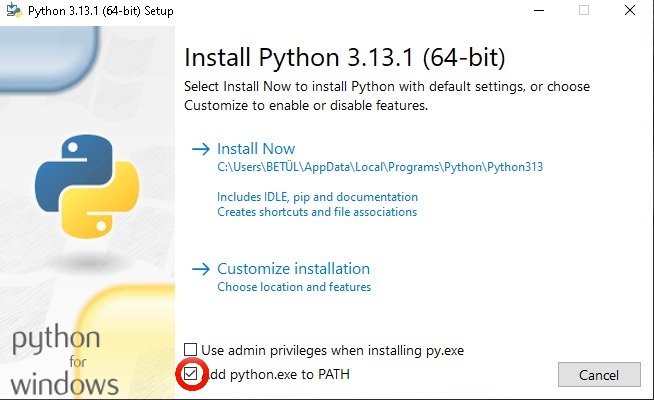


Figure : Add to PATH

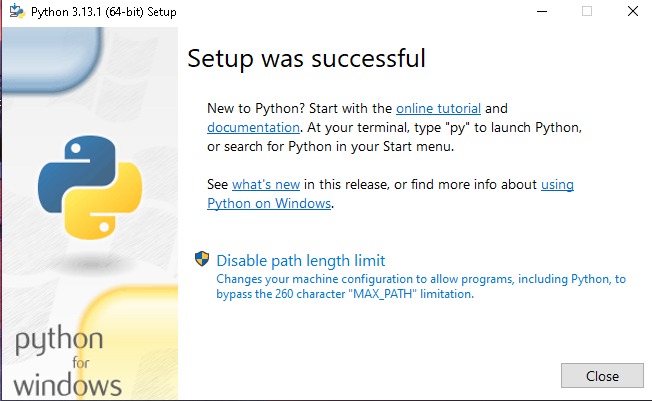


Figure : Install Python

1. **Verify Python Installation:**

Open Terminal (CTRL+SHIFT+” ”) or Command Prompt (WINDOWS + X) and verify that Python has been installed successfully by running the **“** ***python –version* “ or “ *py –version* “** command. (Figure 11)



Figure : Verifying Python Installation

**Step 3: Install Google Chrome and ChromeDriver**

1. **Install Google Chrome**:

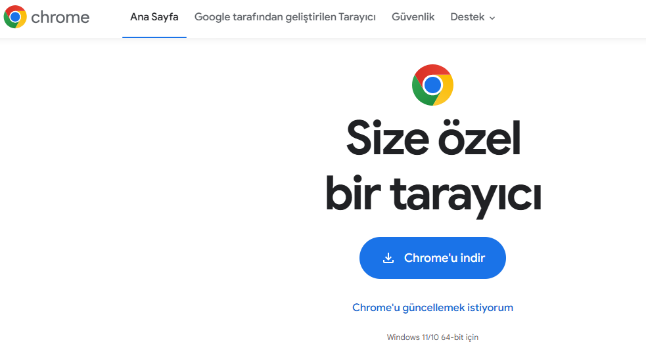
Go to the [Google Chrome website](https://www.google.com.tr/intl/tr/chrome/?brand=CHBD&ds_kid=43700049407063865&gad_source=1&gclid=Cj0KCQiAvvO7BhC-ARIsAGFyToVKTEqKSkUTJzYeLlFOc060Rx5sThevLeFUhPhtm43villpSeVlCh8aAmXMEALw_wcB&gclsrc=aw.ds) and download the browser.(Figure 12 )

Figure : Install Google Chrome

1. **Install ChromeDriver:**

To download the ChromeDriver suitable for your version of Chrome, go to the ChromeDriver official page. (Figure 13)

Add the downloaded file to your system's PATH variable.

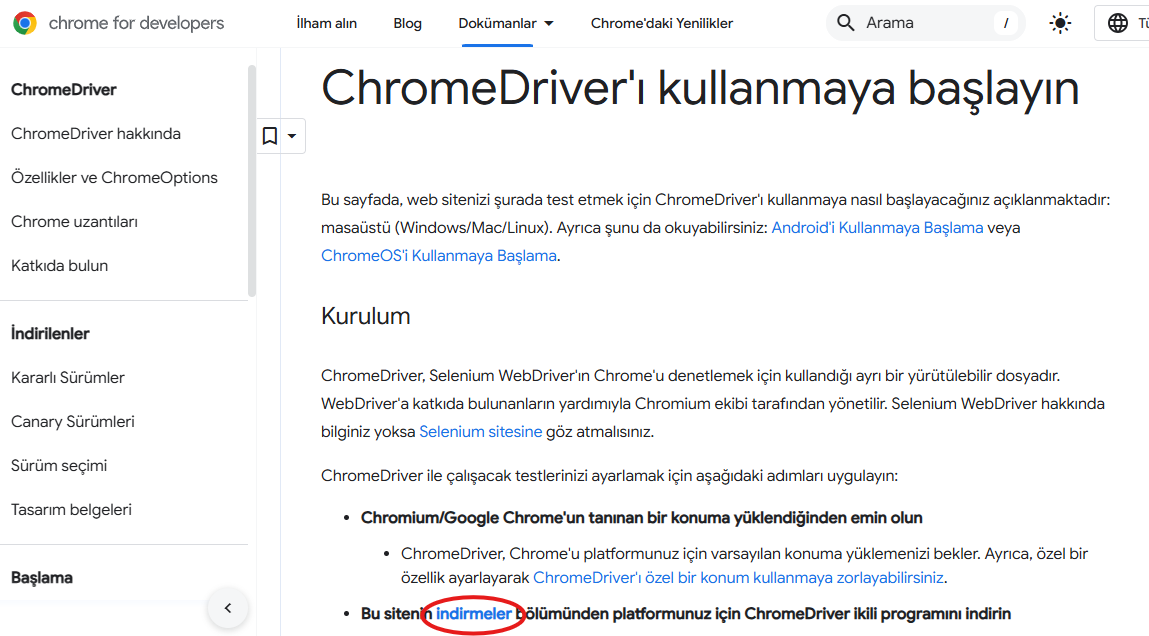


Figure : Install ChromeDriver

**Step 4: Clone or Download the Project**

1. **Obtain the Project:**

If Git is installed, you can clone the project using the following command in the terminal:

“ ***git clone <*** [***https://github.com/birgulyuksel/DjangoProjem***](https://github.com/birgulyuksel/DjangoProjem)***> “***

Alternatively, you can download the project in .zip format from [***https://github.com/birgulyuksel/DjangoProjem***](https://github.com/birgulyuksel/DjangoProjem) ***.***

1. **Obtain the Project (Recommended):**

You can download the zip version of our project from the folder sent to you. After downloading, extract it to the Desktop. (Figure 14)

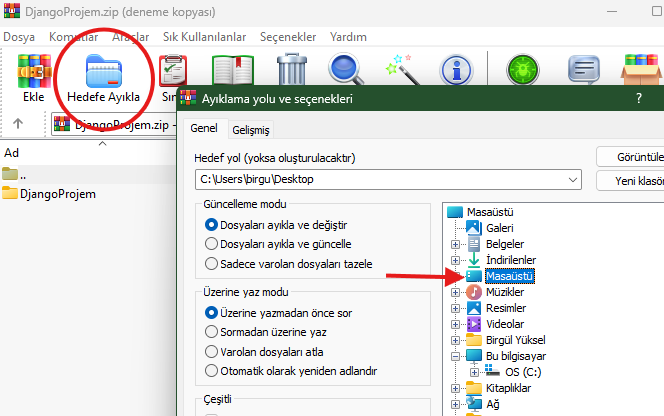


Figure : Extract to Desktop

1. **Navigate to the Project Directory:**

First, open the folder (Figure 15,16) in VsCode, then in Terminal or Command Prompt, navigate to the project folder: (Figure 17)

“ ***cd projefinal*** ” then “ ***cd car\_project*** ”

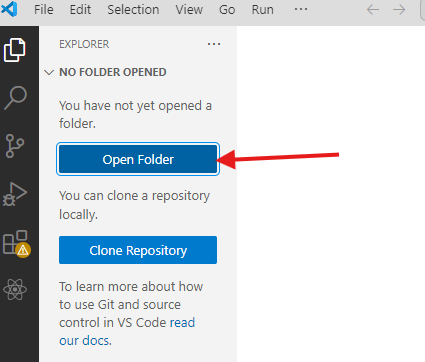


Figure : Open Folder

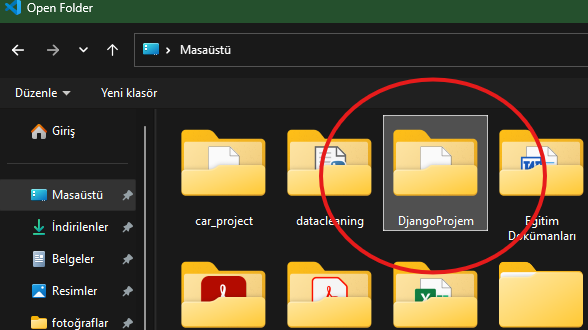


Figure : Open Folder (Continue)

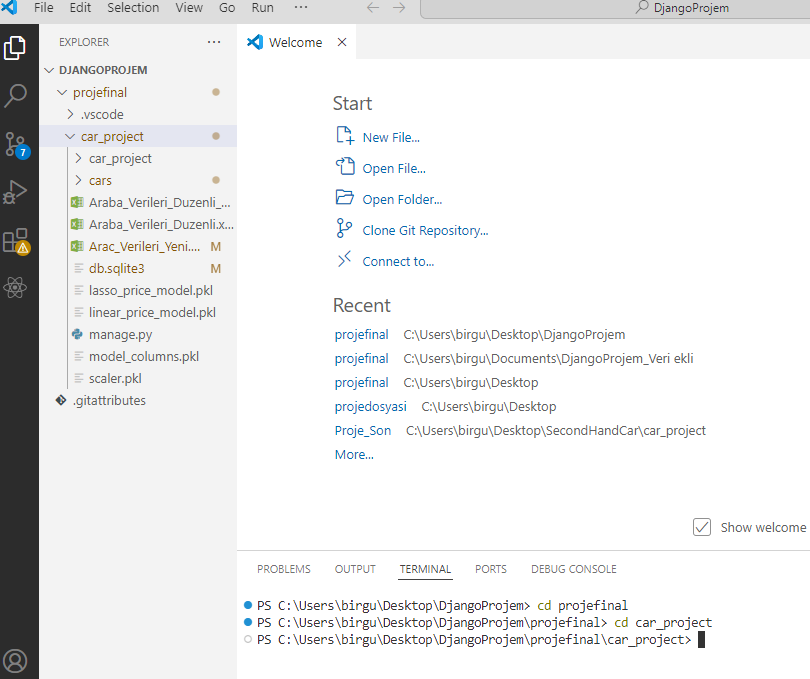


Figure : Navigating the Project folder

**Step 5: Set Up a Virtual Environment**

1. **Create a Virtual Environment:**

Run the following command in terminal: (Figure 18)

“ ***python -m venv venv*** “

1. **Activate the Virtual Environment:** (Figure 18)

***Windows: macOS/Linux:***

***“ venv\Scripts\activate “ “ source venv/bin/activate “***

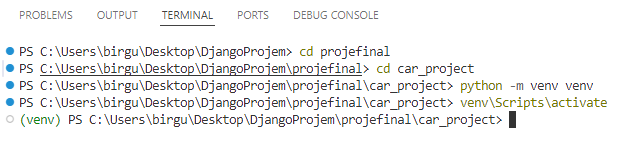


Figure : Create And Activate the Virtual Environment

Make sure you see (venv) on the command line.

**Step 6: Install Required Dependencies**

1. **Install Libraries**:

Install required libraries in one line (may take some time): (Figure 19)

“ ***pip install openpyxl pandas numpy matplotlib seaborn scikit-learn joblib selenium undetected-chromedriver Django*** “

If you get an error, try adding the following:

“ ***python.exe -m pip install --upgrade pip*** “

“ ***pip install --upgrade setuptools***”

“ ***pip install --upgrade undetected-chromedriver*** “

“ ***pip freeze*** “

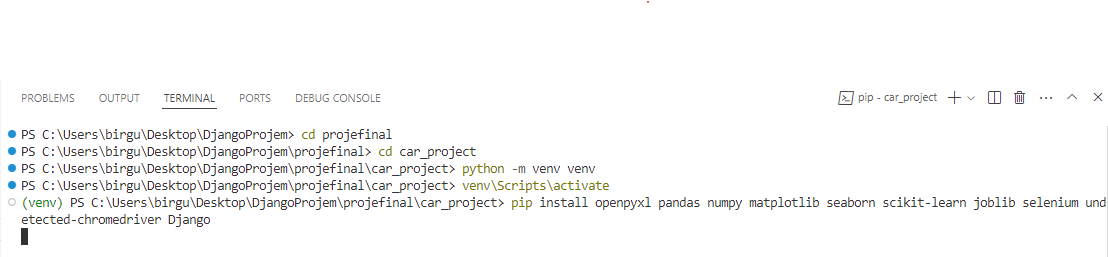


Figure : Install Libraries

1. **Verify Installation**:

To make sure the dependencies are installed correctly, run this command: (Figure 20)

“ ***pip list*** “

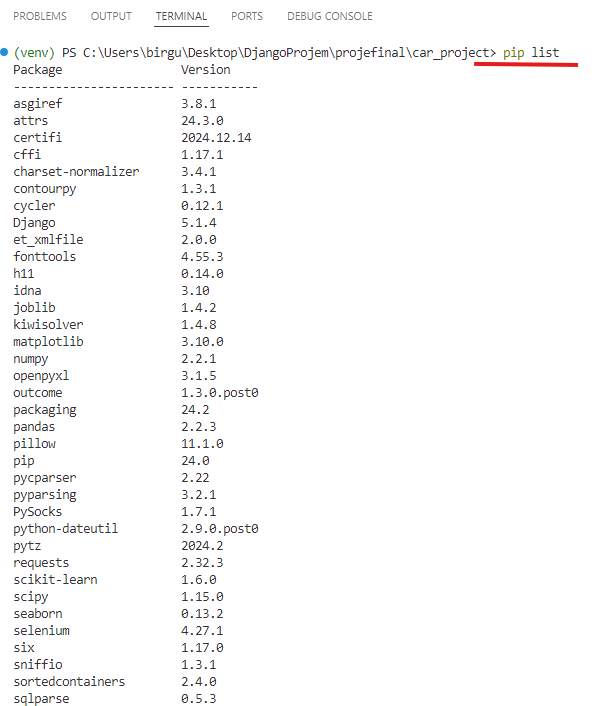


Figure : Verifying Installation

**Step 7: Set Up the Database**

Make sure that the path directory is like this, if you have moved it to the desktop, your current directory should be: “ [\\Desktop\\DjangoProjem\\projectfinal\\car\_project\\](file:///\\Desktop\\DjangoProjem\\projectfinal\\car_project\\) “

1. **Run Migrations**:

Run the following commands to create the database tables: (Figure 21)

“ ***python manage.py makemigrations***

***python manage.py migrate*** “

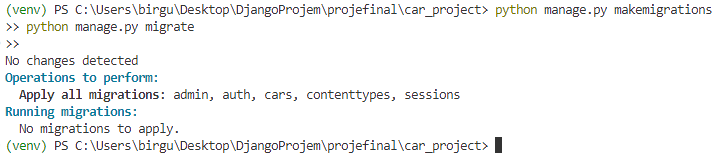


Figure : Running Migrations

**Step 8: Run the Project**

1. **Start the Server:**

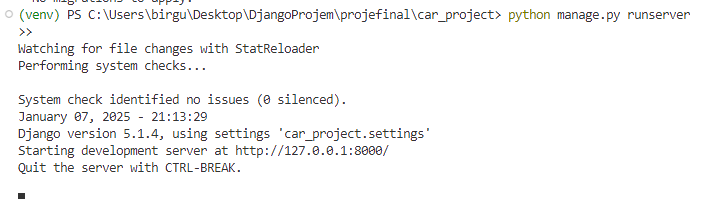
To run the project, run this command: “ *python manage.py runserver* “ (Figure 22)

Figure : Running the server

1. **Access the Project**:

Hover over the link http://127.0.0.1:8000 and CTRL + CLICK (Figure 21) or open your browser and visit this URL: (Figure 24)

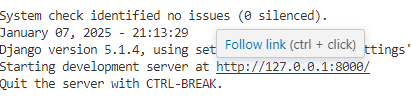


Figure : Access the Project

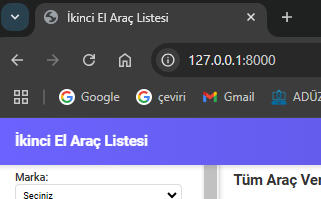


Figure : Visiting the URL

**If you've finished the steps so far, Congratulations! You have successfully reached our server.**

**If you encounter any error, you can find the solution** [**on this page**](#_Troubleshooting_and_Maintenance)**.**

# User Interface Overview

## Description of the main screens or pages

### Home Page (home.html): (Figure 25)

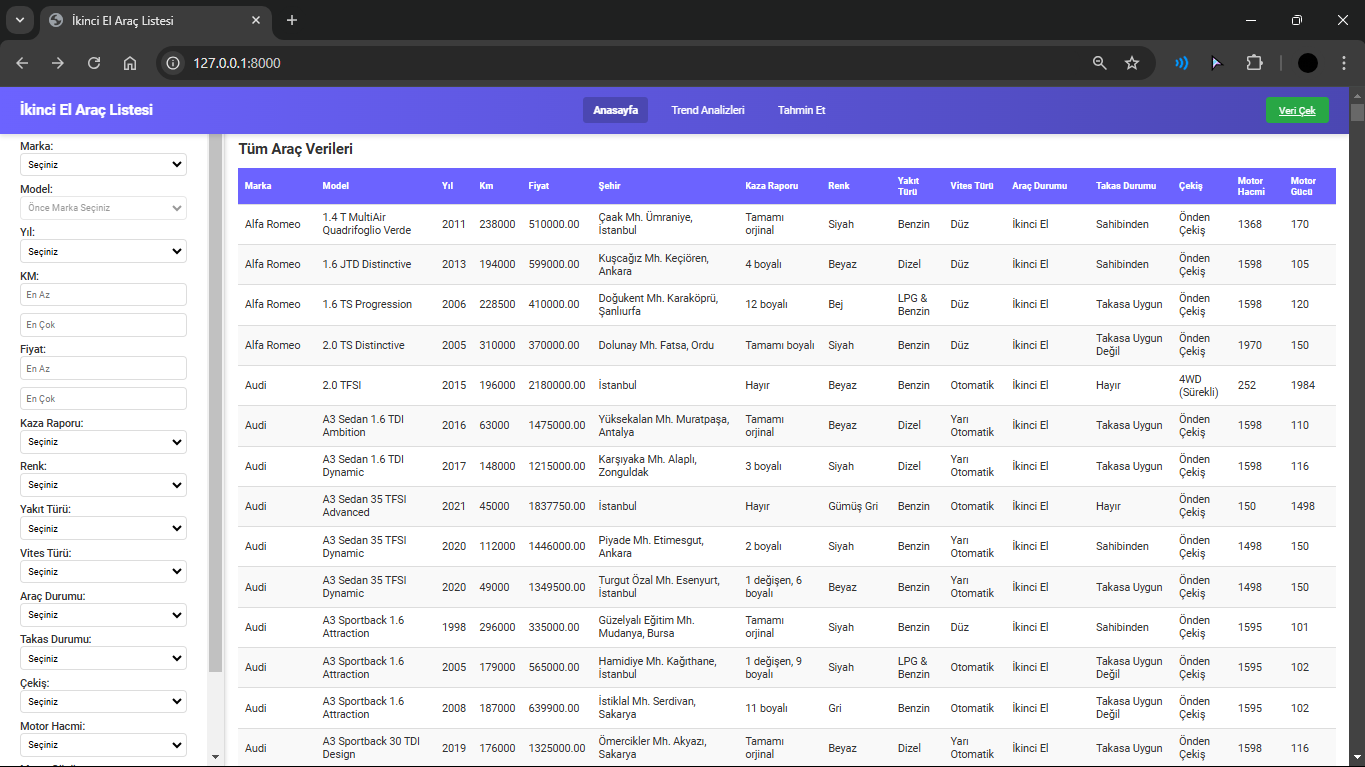


Figure : Home Page Interface

**Explanation:**

This page is the main screen of the application. All vehicles registered in the database are listed here and users can easily access the data using various filtering options.

**Main Features:**

* **Vehicle List**: All vehicles in the database are shown in a table.
* **Filtering**: Filtering can be done according to criteria such as brand, model, year, color, km.
* **"Veri Çek" Button**: Located in the upper right corner, this button directs the user to the collected-data.html page and starts the data extraction process.
* **Data Refresh:** When new data is pulled, this data is added to the list when the page is refreshed and is subject to the same filtering options.

**Additional Settings:**

Data extraction is limited to 5 tools by default. However, this number can be increased or decreased by editing the relevant code blocks.

### Trend Analysis Page (trend-analysis.html) (Figure 26)

**Explanation:**

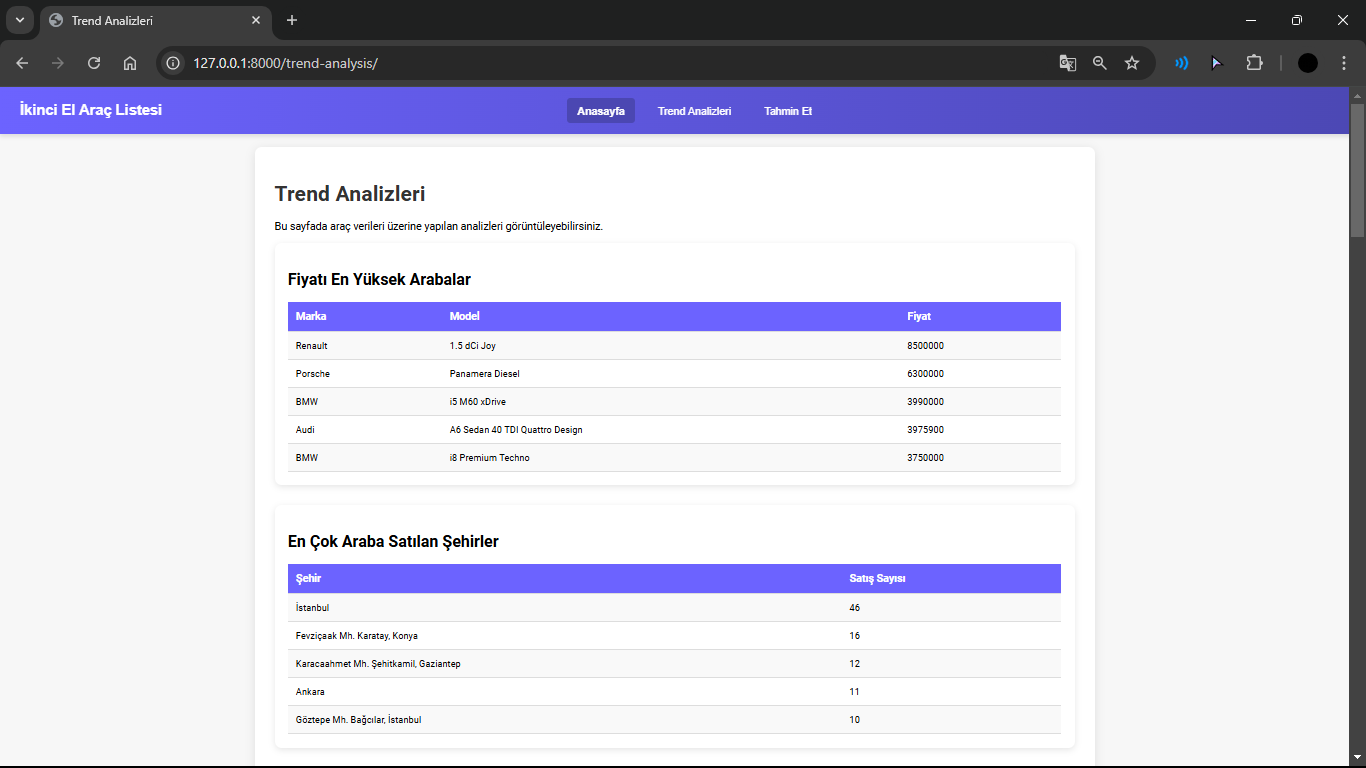
This page presents the user with detailed results of the analyzes performed on the vehicles registered in the database.

Figure : Trend Analaysis Page Interface

**Main Features:**

* **Analysis Data:**
* The most preferred brands.
* The most popular colors.
* Various analyzes such as price trends according to vehicle age are shown in the table.
* **Navigation:** The navigation bar at the top of the page provides easy access to other pages.

### Prediction Page (predict.html) (Figure 27)

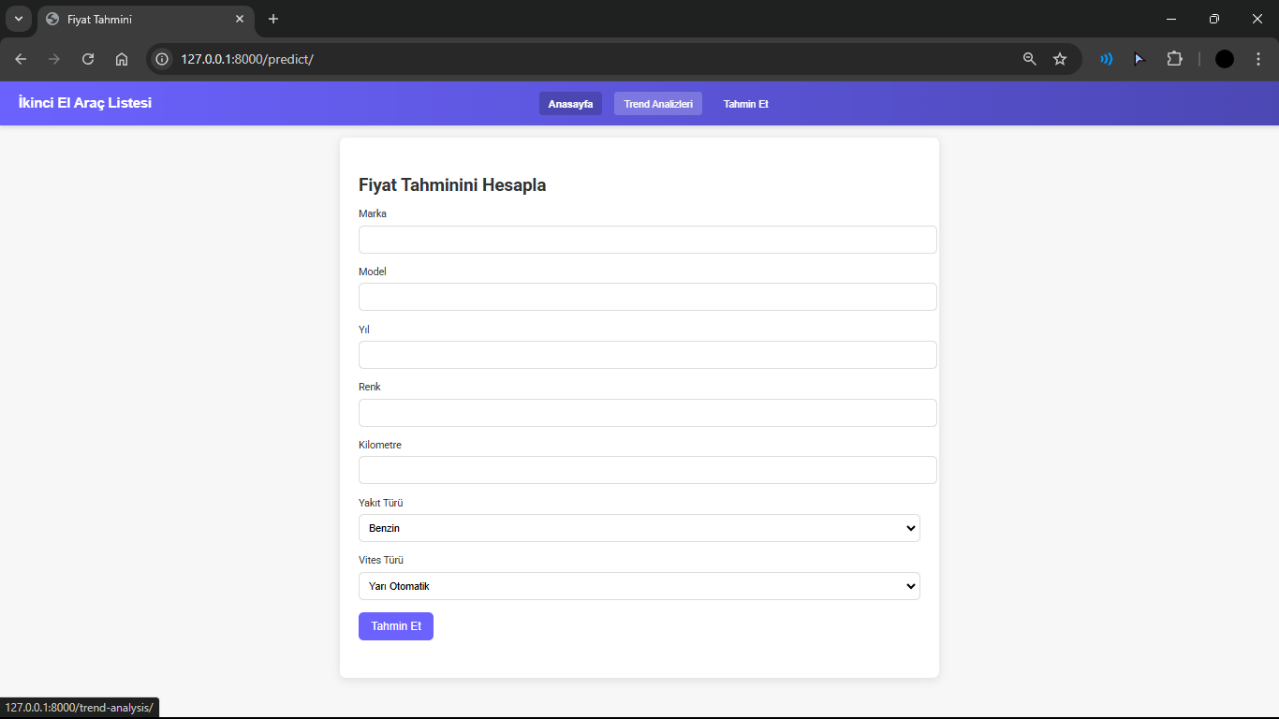
**Explanation****:** This is the page where users can learn estimated vehicle prices.

Figure : Price Prediction Interface

**Main Features:**

* Parameter Entry: The user is asked for vehicle characteristics (e.g. Brand, Model, Color, Year, Km).
* Prediction Button: After entering the required information, click on the "Tahmin Et" button.

The machine learning model determines an estimated price by analyzing similar vehicles in its database. The estimated price is shown on the screen.

### Collected Data Page (collected-data.html) (Figure 28)

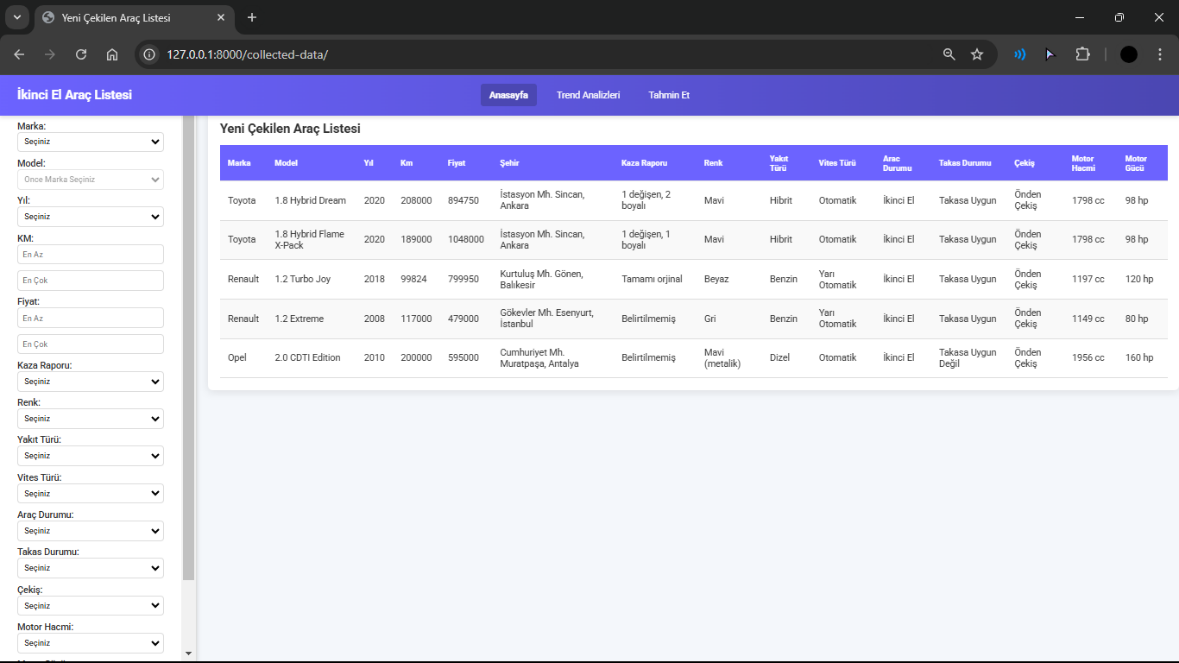
**Explanation:**This page is the screen where the "Veri Çek" process is managed and new vehicle data is captured.

Figure : Collected Data Page

**Main Features:**

* **Data Extraction Process:** When the user presses the "Veri Çek" button, the data extraction process starts.
* **Default Settings:** The number of data captured is initially set to 5.
* **Data List:** New vehicles captured are listed here when the page is refreshed.
* **Filtering Options:** Filtering can be applied to the new data captured according to criteria such as brand, model and color.

## Explanation of the navigation structure



Figure : Navbar

**Navigation Bar (Navbar): (Figure 29)**

It's a constantly visible bar at the top of the app. It helps users get quick access to different pages.

The navigation bar contains the following links:

* **Home Page**: It directs the user to the main screen where vehicles are listed and filtering operations can be performed.
* **Collected Data (Veri Çek) Button**: It directs the user to the screen where the data extraction process takes place. On this page, new vehicle data is added by starting web scraping.
* **Trend Analysis**: It directs the user to the page where the analyzes made with vehicle data are shown as a table.
* **Prediction**: It directs the user to the page where he can calculate the estimated price by entering vehicle information.

**Navigation Flow:**

The user can quickly switch from any page to another.

The navigation bar is displayed the same way on all pages, making navigation consistent.

Page links are clear and distinct and have clear names to guide the user correctly.

## Key buttons and their functions

### Veri Çek (Fetch Data) Button (Figure 30)

* **Location:**

It is located in the upper right corner of the home page (home.html).

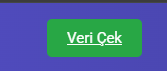


Figure : Veri Çek Button

* **Function:**

When the user clicks this button:

You are directed to the Collected Data page.

Web scraping is started and 5 new vehicle data are pulled by default.

Once the process is complete, the data will be listed when the page is refreshed.

* **Usage Scenario**:

The new tool is used when you want to pull data.

### Tahmin Et (Predict) Button

* **Location:**

Located on the Predict page (prediction.html).



Figure : Tahmin Et Button

* **Function:**

When the user clicks this button after entering the required vehicle information (brand, model, color, year, km):

* + - The machine learning model is run.
    - The estimated vehicle price in accordance with the specified parameters is shown to the user on the screen.
* **Usage Scenario:**

It is used to estimate prices based on vehicle information.

### Filtrele (Filter) Button

* **Location:**

It is available on the home page and on the vehicle list on the collected data site (home.html, collected\_data.html).



Figure : Filtrele Button

* **Function:**

When the user clicks this button after selecting filtering options (brand, model, color, km, etc.):

* + - The list is updated according to the selected criteria.
    - Relevant tools are shown to the user.
* **Usage Scenario:**

It is used when you want to see vehicles that meet certain criteria.

# Features and Functionalities

## List of Primary Features

### Listing Vehicle Data

All tools in the database are presented to the user as a table on the home page. (Figure 33)



Figure : All Vehicle Data

### Filtering Process

Vehicles can be filtered according to features such as brand, model, color, year and km. For example, data for Brand = “MINI” : (Figure 34)



Figure : MINI Model Vehicles

### Data Extr**action Process**

With web scraping, new vehicle data is extracted and added to the database. (Figure 35)

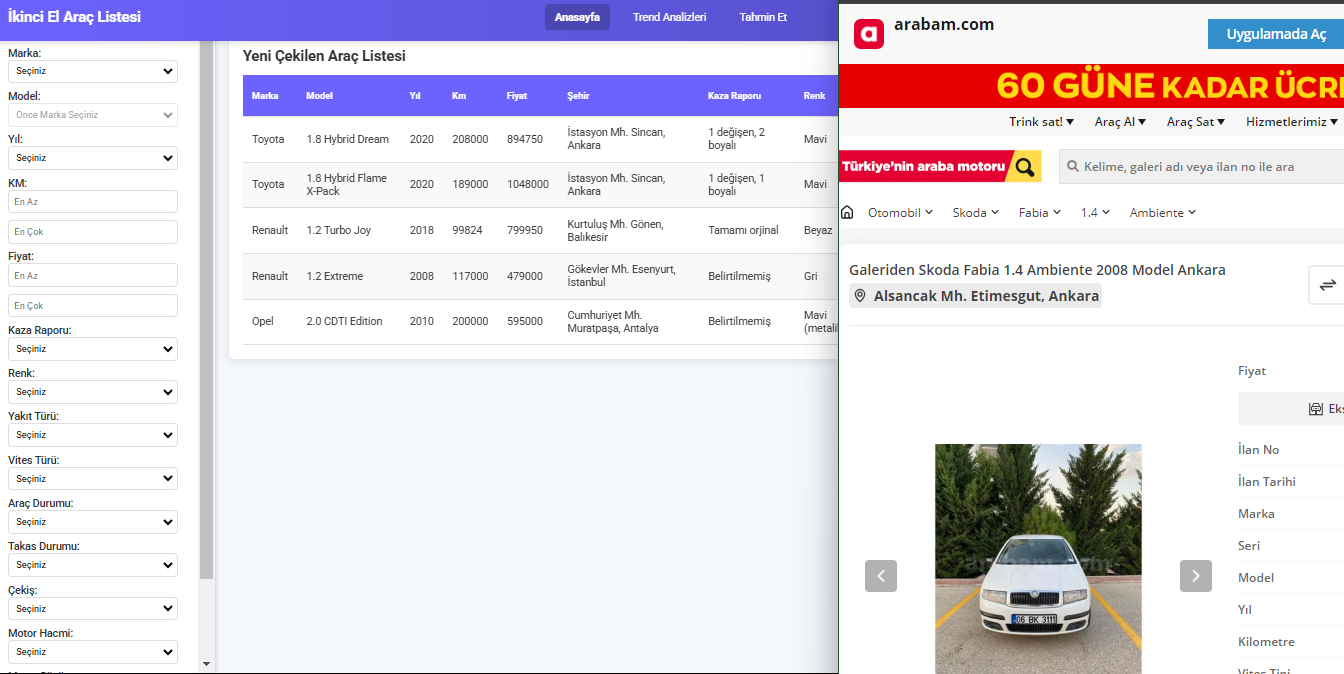


Figure : Web Scraping

### Trend Analysis

Analysis results such as popular brands, colors, and price trends based on vehicle data are displayed. (Figure 36)

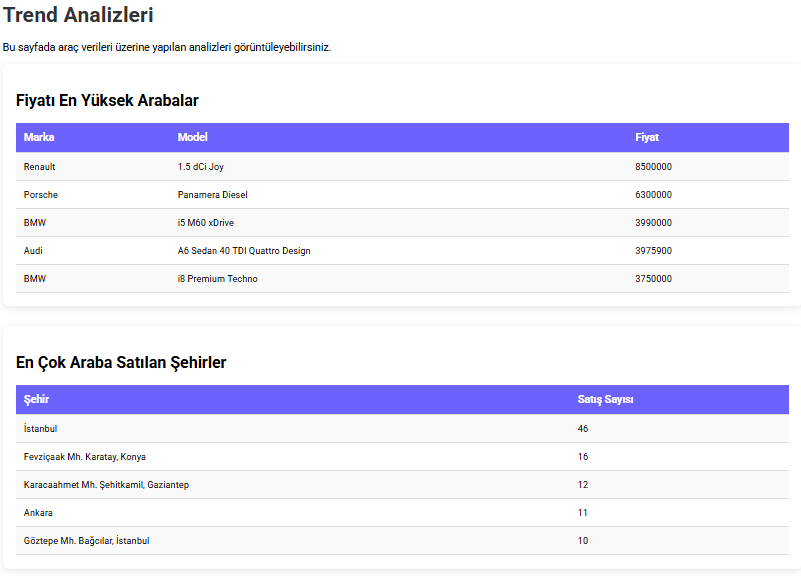


Figure : Trend Analysis Results

### Estimated Price Determination:

Based on the parameters entered by the user, the machine learning model calculates an estimated price. (Figure 37)



Figure : Estimated Price

## How each feature works?

### How Listing Vehicle Data Works?

1. **General Working Logic**

All vehicles in the database are pulled from the Car model.

This data is processed in Django's views module and transferred to an HTML template (home.html).

When the user opens the page, it displays all the tools in a table.

2. **Relevant Codes and Explanations**

1. **Models (Database Model):** Vehicle data is stored in the database via the Car model. (Figure 38)

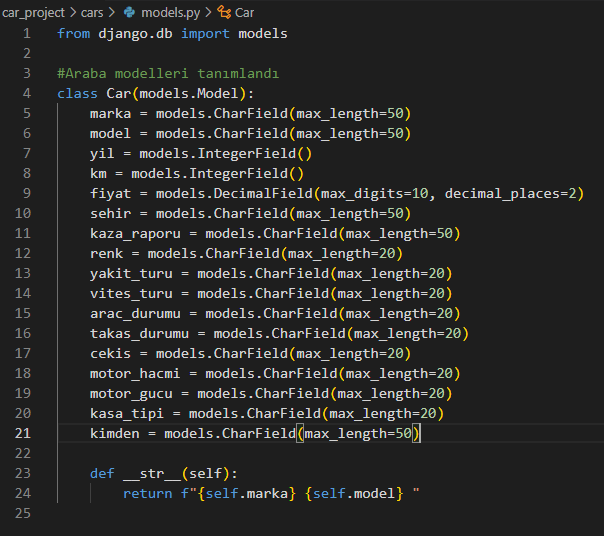


Figure : Car Models

1. **Views (Processing and Sending of Data):** A view is created to retrieve vehicle data. (Figure 39)

* Car.objects.all(): Returns all vehicle records.
* render: Transfers data to the home.html template.

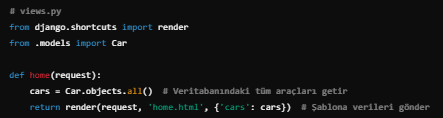


Figure : Example Of View

1. **Template (HTML and CSS) :** The home.html template displays data in tabular form. (Figure 40)
   * Expressions like {{ car.brand }} export fields from the Car model to HTML.
   * The for loop sorts all vehicles into the table.



Figure : Example of HTML Code

1. **Filtering Process (Optional)**

The tool can work with data filtering feature. An example filtering code:

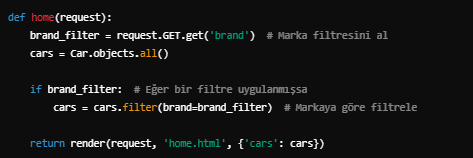


Figure : Example Code Of Filtering

### How Filtering Process Works?

1. **General Working Logic**

The user fills in filtering options (for example, brand or color selection) on the home.html page.

Filtering parameters received from the form are processed in the views.py file.

Vehicles that meet certain criteria are queried from the database and the results are sent to the page.

1. **Relevant Codes and Explanations**
2. **HTML Form (Filtering Options)**

A piece of form code where the user will enter data to filter: (Figure 42)



Figure : A Piece Of Form Code

1. **Views.py (Filtering Process)**

The data from the form is taken and filtered: (Figure 43)

* request.GET.get(): Retrieves the data sent by the user from the form.
* filter(): Queries the results that meet the specified criteria in the database.
* icontains: Searches regardless of case.

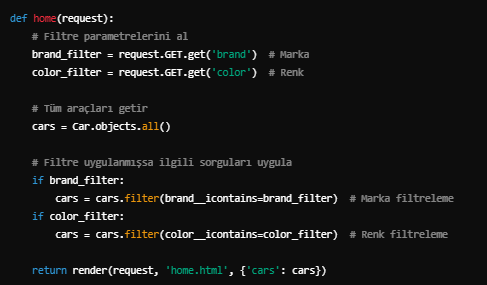


Figure : Example of Filtering Data Code

1. **Case Study (Figure 44)**
   1. The user filters by selecting "Make: BMW" and "Color: White".
   2. The system finds vehicles with brand='BMW' and color='White' in the database.
   3. The page displays only vehicles that meet these criteria.

****

Figure : BMW-White Car Data

### How Data Extraction Process Works?

1. **General Working Logic**

When the user clicks on the "Veri Çek" button, the data extraction process starts.

Vehicle information is collected from web pages using Selenium.

The captured data is saved in the database and shown to the user.

1. **Relevant Codes and Explanations**
   1. **Data Extraction Process (Selenium-Chromedriver)**

Data is retrieved with the collect\_data command. Sample simplified selenium code: (Figure 45)

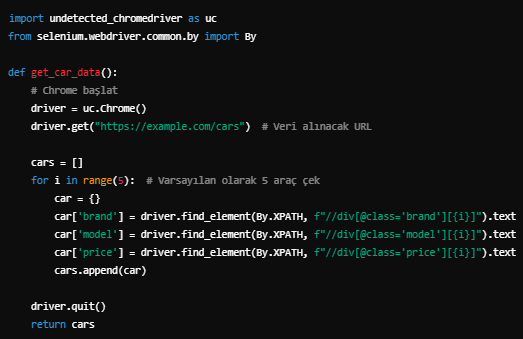
****

Figure : Sample Code

* 1. **Saving to Database**

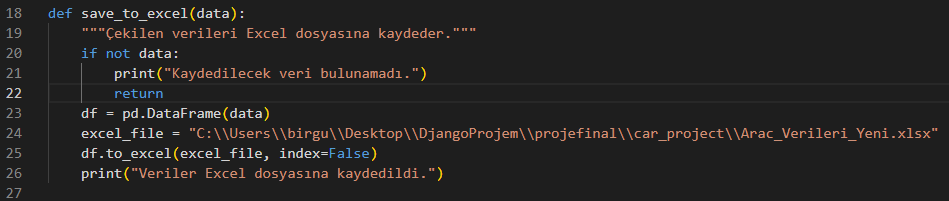
The captured data is saved in the Car model: (Figure 46)

Figure : Save To Excel Function

### How Trend Analysis Works?

1. **General Working Logic**

Vehicle data in the database is analyzed.

Using Pandas, information such as the most preferred brand, color, or price range is obtained.

Analysis results are presented to the user in tables.

1. **Relevant Codes and Explanations**

The system analyzes and returns results such as "Most Popular Brand: BMW", "Most Popular Colour: White". (Figure 47)

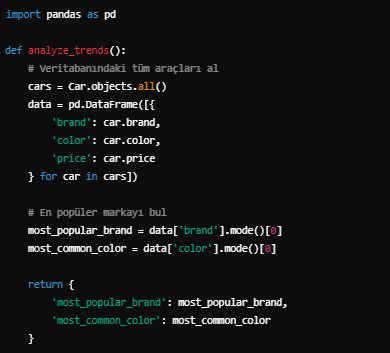


Figure : Example Of Pandas Code

1. **Case Study:**

Most Preferred Fuel Types: (Figure 48)

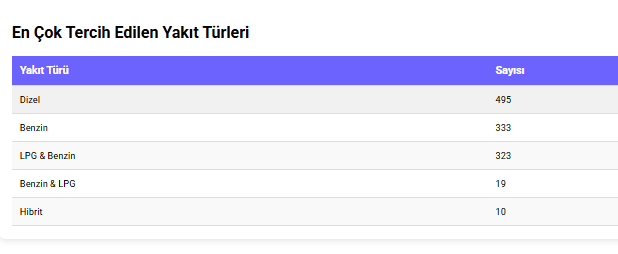


Figure : Most Preferred Fuel Types

### How Estimated Price Determination Works?

1. **General Working Logic**

The user enters parameters such as brand, model, color, year.

The Machine Learning Model evaluates these parameters and calculates an estimated price.

1. **Relevant Codes and Explanations**
   1. **Model Training:** A regression model is trained using vehicle data from the database (Figure 49)

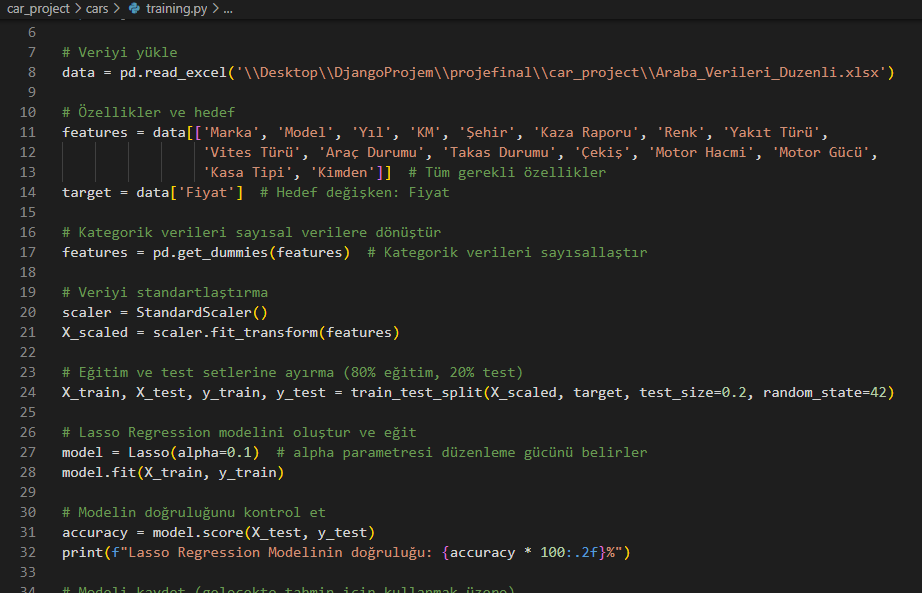
****

Figure : Model Training Code

* 1. **Estimation Process:** The parameters entered by the user are transferred to the model: (Figure 50)

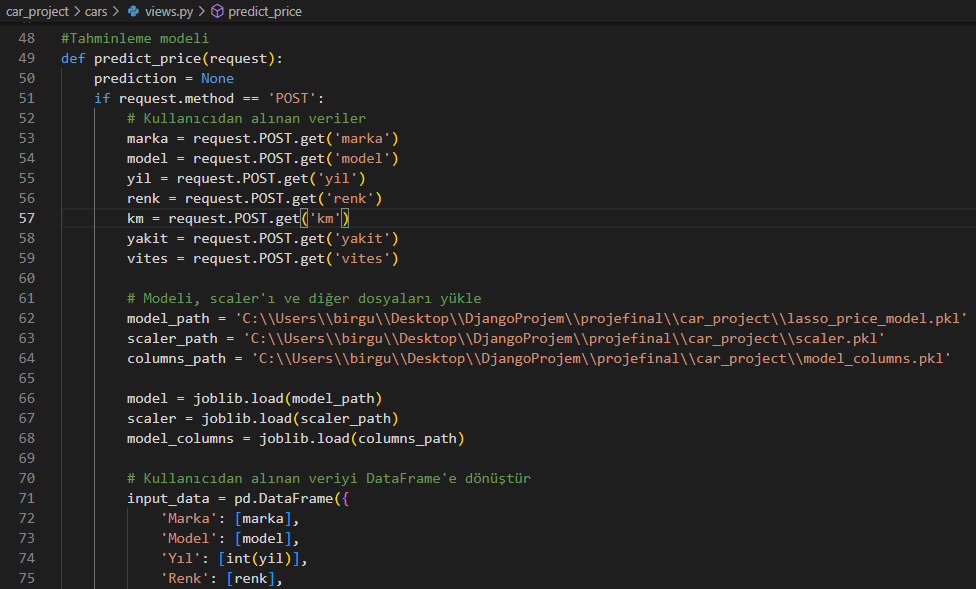


Figure : A Piece Of Function Code

1. **Case Study**

If we enter the data like this: Brand = Audi , Model = A6 Sedan 2.0 TDI , Year = 2020 , Colour = Red , Kilometer = 20000 , Fuel Type = Benzin, Gear Type = Otomatik . (Figure 51)



Figure : Estimated Data

# Usage Instructions

## Step-by-step guide on how to use the system

### Login to the System

Open Visual Studio Code installed on your computer.

Open the folder containing the project and launch the terminal window. (Figure 52)

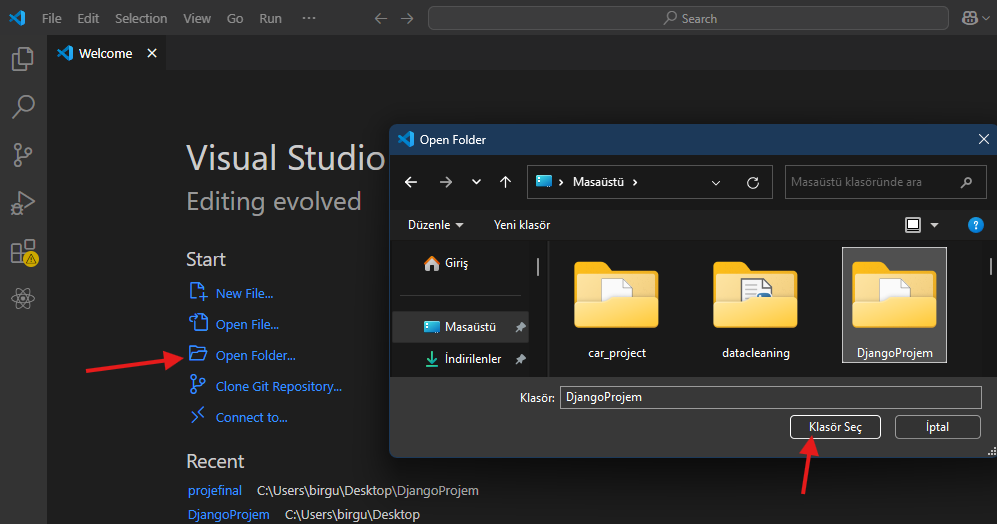


Figure : Open Folder in VSCode

Run the Django project by typing the following command in the terminal: (Figure 53)

***“ python -m venv venv*** ”

Windows: “ ***.\venv\Scripts\activate*** “

Linux/MacOS: “ ***source venv/bin/activate*** “

“ ***python manage.py runserver*** “

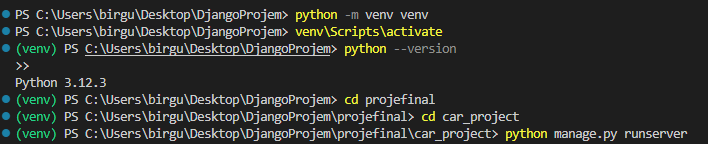


Figure : Run the Code

Start the system by pasting the link in the terminal into your browser or shift + click on the link.

### Home Page Usage

1. **Listing Vehicles in the Database**

When you start the system, the home.html page opens automatically.

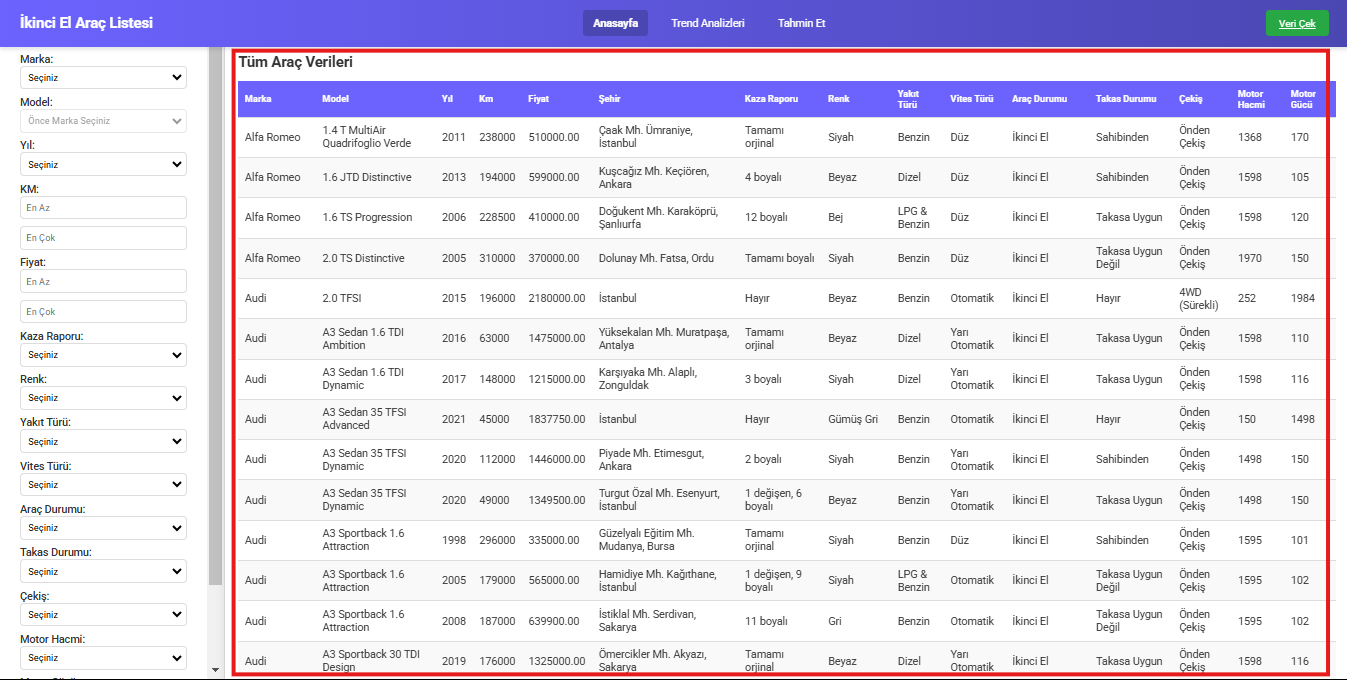
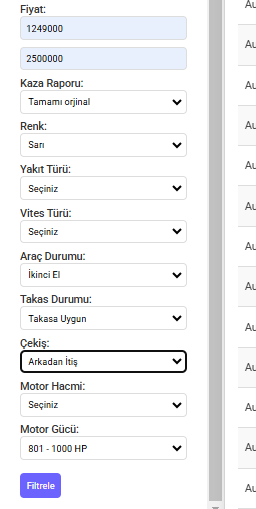
This page lists all vehicles in the current database. (Figure 54)

Figure : All Vehicles Are Here



1. **Filtering Tools**

You can filter vehicles by brand, color or other criteria by using the filtering options on the left.

(Figure 55)

1. **Initiating Data Scraping**

You can start the data extraction process by clicking the "Veri Çek" button at the top right.

The system will redirect you to the collected-data.html page.

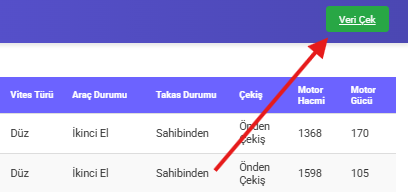


Figure : Filtering Tools

Figure : Veri Çek Buttton

### New Data Scraping

Click the "Veri Çek" button and wait for the system to automatically pull data.

The number of data to be captured is 5 by default.

After the data extraction process is completed, you can see the new vehicles by refreshing the page. New vehicle data will also be added to the home page.(Figure 57)



Figure : New Vehicle Data

If you want to change the number of data to be pulled, you can edit the relevant part of the code:

**1.Collect\_data.py : (Figure 58)**

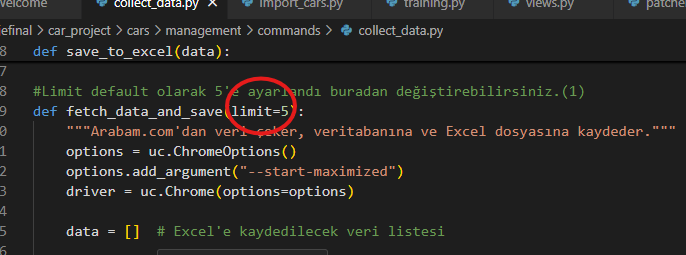


Figure : Collect Data Code

2.**Views.py: (Figure 59)**

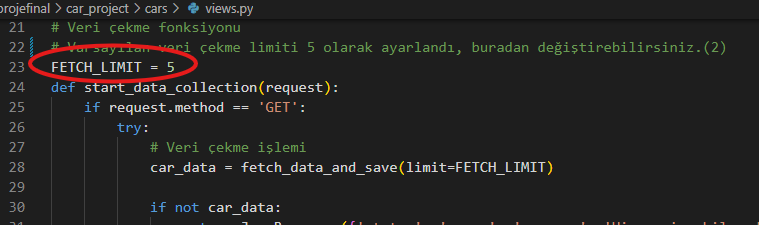


Figure : Views.py Code

### Viewing Trend Analysis

Click on the "Trend Analysis" tab in the top menu.

You can view the analyzes made on the trend-analysis.html page that opens. (Figure 60)

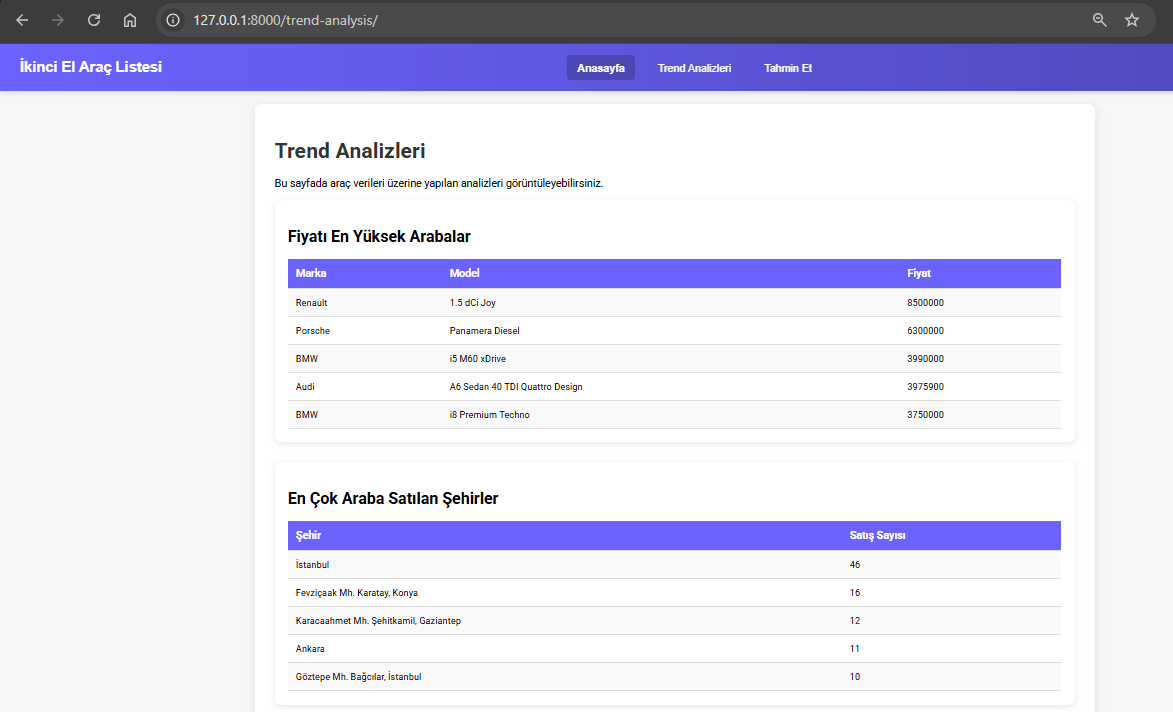


Figure : Trend Analysis Page

### Estimated Price Determination

Click on the "Tahmin Et” tab in the top menu.

Enter the parameters on the page that opens.

Click the "Tahmin Et" button. Wait for 5-10 seconds.

The system calculates the estimated vehicle price and displays it on the screen. (Figure 61)

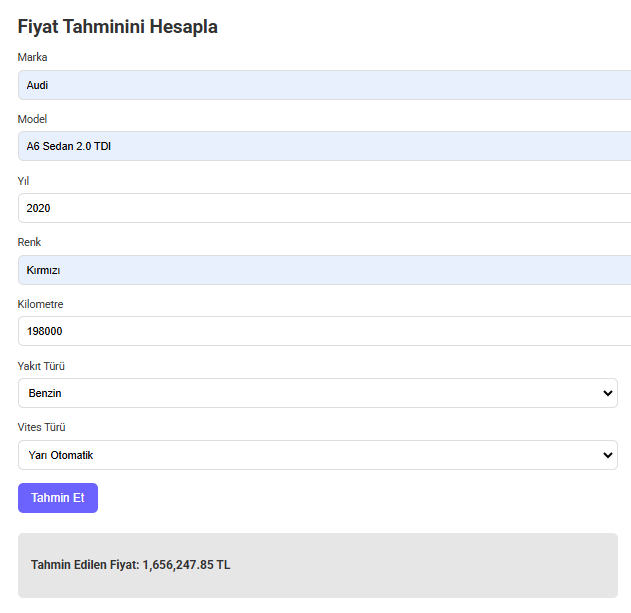


Figure : Predict Page

## Examples of typical use cases

### Finding and Listing Vehicles

**Scenario:** The user wants to view the available vehicles and see the white colored and Automatic Transmission vehicles of the BMW brand.

**Steps**:

1. Select "BMW" in the "Marka" section.
2. Select "Beyaz" in the "Renk" section.
3. Select "Otomatik" from the "Vites Türü" section.
4. Press the "Filtrele" button.
5. Only white vehicles with automatic transmission belonging to the BMW brand are displayed in the list. (Figure 62)

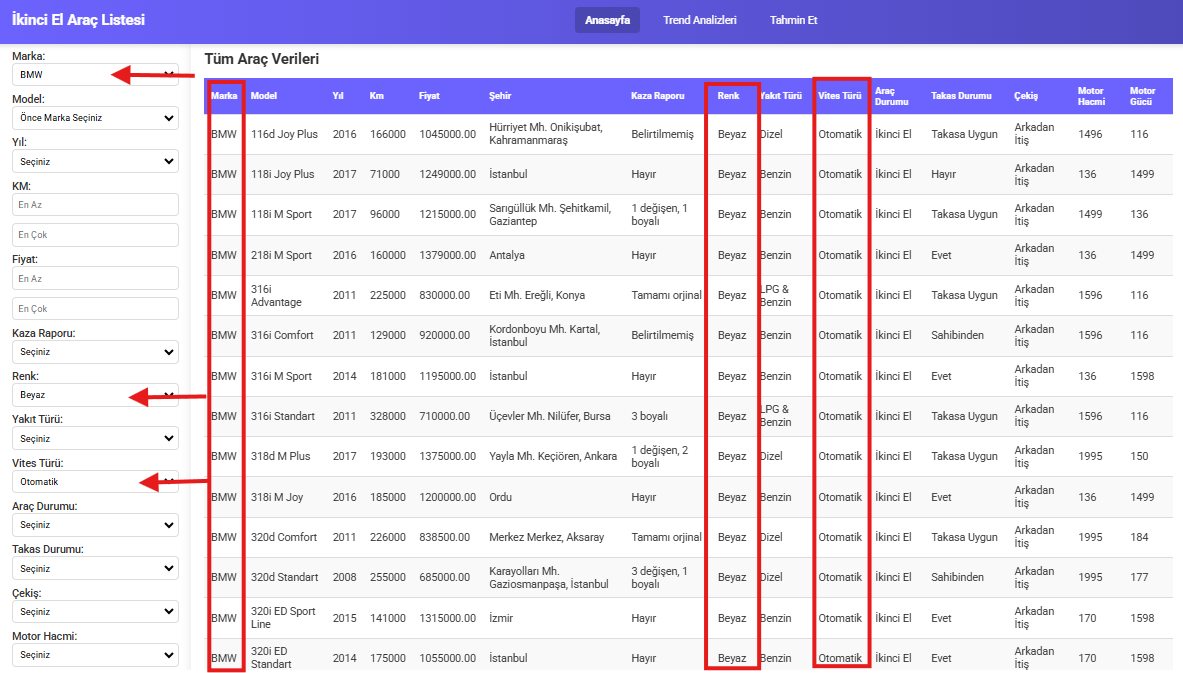


Figure : Use Case For Filtering Data

### Scraping New Data

**Scenario:** The user wants to add 5 new vehicle data to the system's database.

**Steps:**

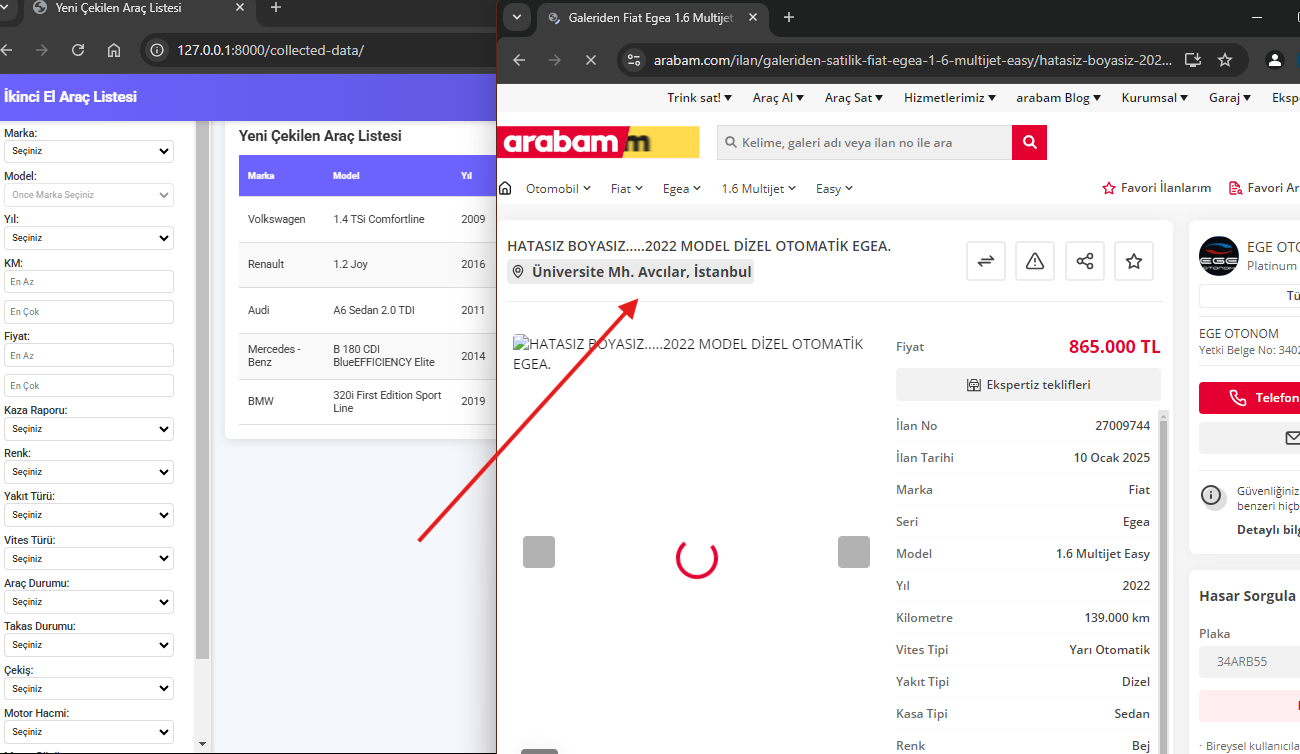
1. Click the "Veri Çek" button.
2. Do not close the opened site, wait and do not touch anything else. (Figure 63)
3. By default, the system extracts and adds 5 vehicle information (approximately 2 minutes).
4. Refresh the page to view new data. (Figure 64)

Figure : Data Scraping Is Started

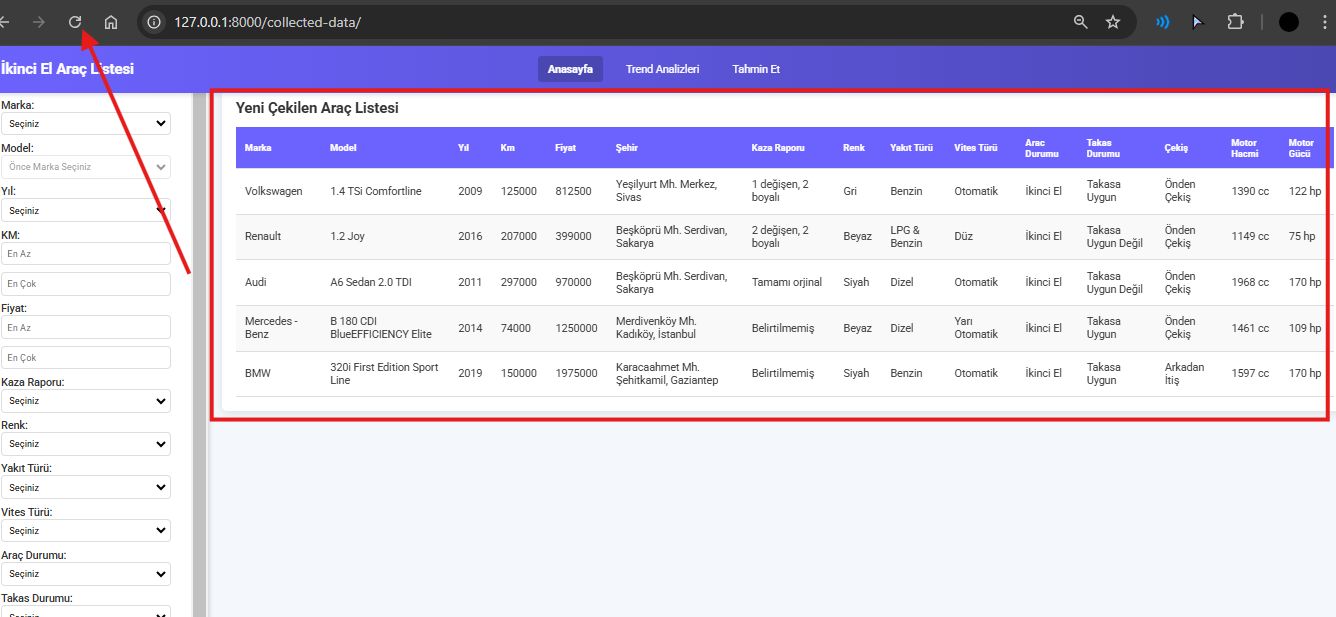


Figure : New Vehicle Datas Are Here

### 

Figure : New Vehicle Datas Are Here

### Viewing Trend Analyzes

**Scenario:** User wants to know which brand is most popular (Figure 66), Cars with Highest Mileage (Figure 67) and Most Preferred Fuel Types. (Figure 68)

**Steps:**

Click on the "Trend Analysis" tab from the top menu.

Examine the analysis tables displayed.

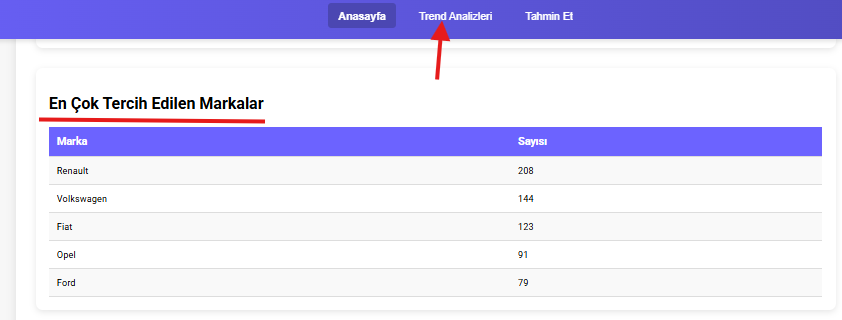


Figure 66 : Most Popular Brands



Figure : Cars with Highest Mileage



Figure : Most Preferred Fuel Types

### Estimated Price Determination

**Scenario:** The user wants to know the estimated price of a red Audi A6 Sedan 2.0 TDI model, 2020 model and 150000 km, Fuel Type: Gasoline, Gear Type: Automatic.

**Steps:**

1. Write "Audi" in the "Marka" section.
2. In the "Model" section, write "A6 Sedan 2.0 TDI".
3. Enter "2020" in the "Yıl" field.
4. Enter "150000" in the "KM" section.
5. Select "Yakıt Türü" as “Benzin” and "Vites Türü" as “Otomatik”.
6. Click the "Tahmin Et " button.

The system calculates the estimated price and displays it on the screen (for example: **Tahmin Edilen Fiyat: 1,584,953.44 TL**). (Figure 69)

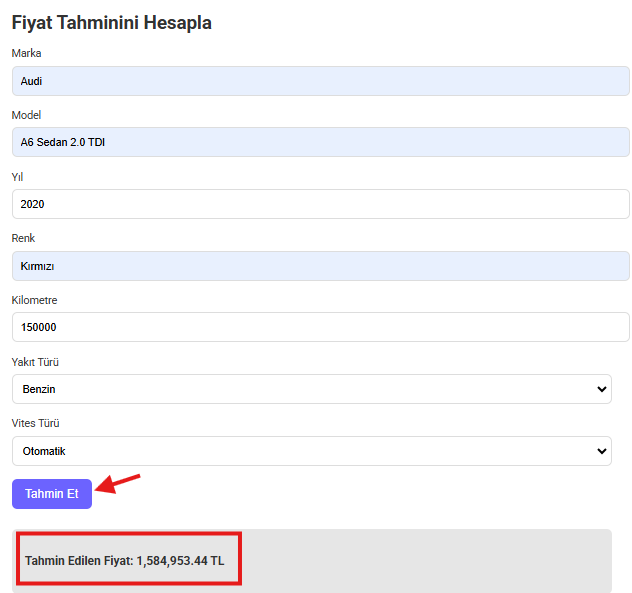


Figure : Use Case For Predict Page

# Troubleshooting and Maintenance

## Common issues and their solutions

This section explains common system-related problems, solutions, and error messages.

| **Problem** | **Possible Cause** | **Solution** |
| --- | --- | --- |
| **VS Code cannot be downloaded or used.** | Errors occur during download, or installation fails. | Visit the VS Code [support page](https://code.visualstudio.com/docs/supporting/faq) and ensure you are downloading the correct version for your operating system. |
| **Issues during Python installation.** | Python installation is incomplete, or it is not added to the PATH. | Visit the Python [help page](https://www.python.org/about/help/). During installation, make sure to check the "Add Python to PATH" option. |
| **Virtual environment (venv) not created.** | Execution policies in PowerShell block script execution. | Temporarily bypass the execution policy in PowerShell using: Set-ExecutionPolicy -Scope Process -ExecutionPolicy Bypass. Then run python -m venv myenv to create the virtual environment. |
| **venv cannot be selected or long error messages appear.** | The virtual environment is not properly set or requires manual configuration. | Press Ctrl+Shift+P in VS Code, select **Python: Select Interpreter**, and manually set the full path of your virtual environment. (C:\Users\*YourUserName*\project\venv\Scripts\python.exe) |
| **ModuleNotFoundError appears.** | Required libraries are not installed. | Install the missing library using the terminal, e.g., pip install pandas. Additionally, ensure pip is up to date with pip install --upgrade pip. For the pip list to download, go to [Step 6: Install Required Dependencies](#_Step-by-step_instructions_for). |
| **File path not found.** | File paths in the code are not configured for your computer. | Update the file and Excel paths in import\_cars.py, views.py, price\_prediction.py, collect\_data.py, and training.py according to your system's specific file locations. To find the correct file path, right-click on the file, select "Copy file path," and update the respective section in the code. |
| **Data extraction fails.** | Unstable internet connection or interference during the process. | Ensure your internet connection is stable. Do not close the page or interact with the system until the data extraction process (default 5 items) completes successfully. |
| **Filtering does not work.** | Incorrect or non-existent filter values are entered. | Ensure the filter values are valid. For example, avoid entering a brand or model name that does not exist in the database. |
| **Prediction functionality fails.** | Incorrect or incomplete input parameters are provided. | Verify that the brand and model names entered exist in the database. For more accurate results, ensure realistic and appropriate values are provided for the inputs. |

## Error messages and how to resolve them

| **Error Message** | **Cause** | **Solution** |
| --- | --- | --- |
| **ModuleNotFoundError: No module named '...'** | A required library is not installed. | Use the terminal to install the missing library. For example: pip install pandas. Ensure that the virtual environment (if used) is activated before running the installation. |
| **NoReverseMatch** | Error in URL configuration. | Check the urls.py file for incorrect or missing path configurations. Ensure the view functions and route names match correctly. |
| **AttributeError: 'NoneType' object has no attribute '...'** | Missing or incorrect data. | Verify that the required data is being fetched correctly. Check for null or missing values in the database or during the data extraction process. |
| **ConnectionError: Failed to establish a new connection.** | Browser or internet connection issue. | Ensure that the correct browser driver is installed and updated. Check your internet connection and retry. |
| **PermissionError: [WinError 5] Access is denied.** | Lack of permissions for certain operations. | Run the terminal or IDE as an administrator and ensure that the necessary directories and files are accessible. |
| **FileNotFoundError: [Errno 2] No such file or directory: '...'** | The specified file path does not exist or is incorrectly defined. | Update the file paths in the code to match your local system. To find the file path, right-click on the file, select "Copy file path," and replace the path in the respective code section. |
| **ImportError: cannot import name '...'** | The module or function being imported is unavailable or incorrectly spelled. | Verify that the module is installed and the import statement matches the exact function or class name. Ensure there are no circular imports in your project. |
| **Selenium WebDriverException: Message: ...** | The browser driver is missing, outdated, or incompatible with your browser. | Ensure the Selenium WebDriver is downloaded and matches your browser version. For example, download the ChromeDriver that matches your Google Chrome version. |

If the error persists after trying these solutions, you may consider reviewing the official documentation of the library or tool you are using or reaching out to online communities for further support**.**

**For further support, you can reach us at:** [**secondhandcardestek@gmail.com**](mailto:secondhandcardestek@gmail.com?subject=Uygulama%20Hk%20Hata%20Yardımı) **24/7.**

**We provide support within 30-60 minutes maximum!**

# References

**Documentation and Official Guides**

1. **Python Official Documentation**
   * Resource: <https://docs.python.org/3/>
   * Purpose: Used to understand Python's syntax, libraries, and best practices.
2. **Django Framework Documentation**
   * Resource: <https://docs.djangoproject.com/en/>
   * Purpose: Reference for implementing Django models, views, and templates.
3. **Pandas Library Documentation**
   * Resource: <https://pandas.pydata.org/docs/>
   * Purpose: Assisted in data manipulation and analysis for filtering and trend analysis features.
4. **Scikit-learn Documentation**
   * Resource: <https://scikit-learn.org/stable/documentation.html>
   * Purpose: Used to implement the machine learning model for price estimation.
5. **Selenium WebDriver Documentation**
   * Resource: <https://www.selenium.dev/documentation/>
   * Purpose: Guidance for automating web scraping tasks.

**Tutorials and Learning Platforms**

1. **YouTube Tutorials on Django**
   * Resource: [Sadık Turan Django Eğitimi Listesi](https://www.youtube.com/playlist?list=PLXuv2PShkuHzrqh-_ZYuDcHZcoZfeAnad%20)
   * Purpose: Helped in learning Django-specific techniques and troubleshooting.
2. **W3Schools HTML & CSS Reference**
   * Resource: <https://www.w3schools.com/>
   * Purpose: Used for building and styling the user interface.
3. **Real Python Articles**
   * Resource: [RealPython/Python For Data Analysis Article](https://realpython.com/python-for-data-analysis/)
   * Resource: [RealPython/Linear Regression In Python Article](https://realpython.com/linear-regression-in-python)
   * Resource: [RealPython/Python Web Scraping Article](https://realpython.com/python-web-scraping-practical-introduction/)
   * Purpose: Provided in-depth articles on Python, Django, and common development patterns.

**Additional Online Resources**

1. **Stack Overflow Discussions**

* Resource: [Stackoverflow/NoModuleDistutilsProblem](https://stackoverflow.com/questions/69919970/no-module-named-distutils-util-but-distutils-installed)
* Resource: [Stackoverflow/CannotRunTheServer](https://stackoverflow.com/questions/78322459/cannot-run-the-server-with-python-manage-py-runserver?noredirect=1&lq=1)
* Purpose: Used to resolve common coding and debugging issues encountered during development.