Course: IST 615 Cloud Management
Title: Used Car Retail Website

Project Description:

Built a used car retail website using Microsoft Azure Cloud services where buyers can buy and enlist cars based on manufacturers, mileage, distance traveled, etc.

Dataset Description:

The dataset was taken from Kaggle in the form of a csv file. It is originally from Craigslist. The dataset consisted of 426881 rows and 26 columns. Columns consisted of details of used car information like price, year, manufacturer, model, make, transmission, cylinders, mileage, distance travelled, condition, etc.

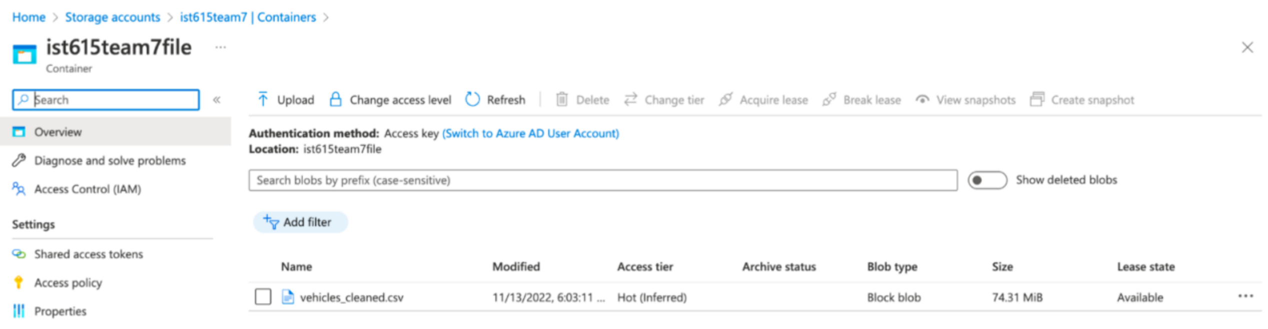
Data Exploration and Cleansing:

Dataset was cleaned by removing rows with null values in columns like condition, manufacturer, model, fuel, odometer, and transmission. Column with the County name was also dropped as it had more than 50% null values. Similarly, the size and region URL were also removed due to a lot of missing values. The description column was also dropped as the column contained identical descriptions for more than 50% of the records.

Cloud Services Implemented:

Azure Blob Storage:

Our cleaned dataset file 'vehicles_cleaned.csv' was uploaded to Azure blob service and stored in a container.


Home > Storage accounts > ist615team7 | Containers >

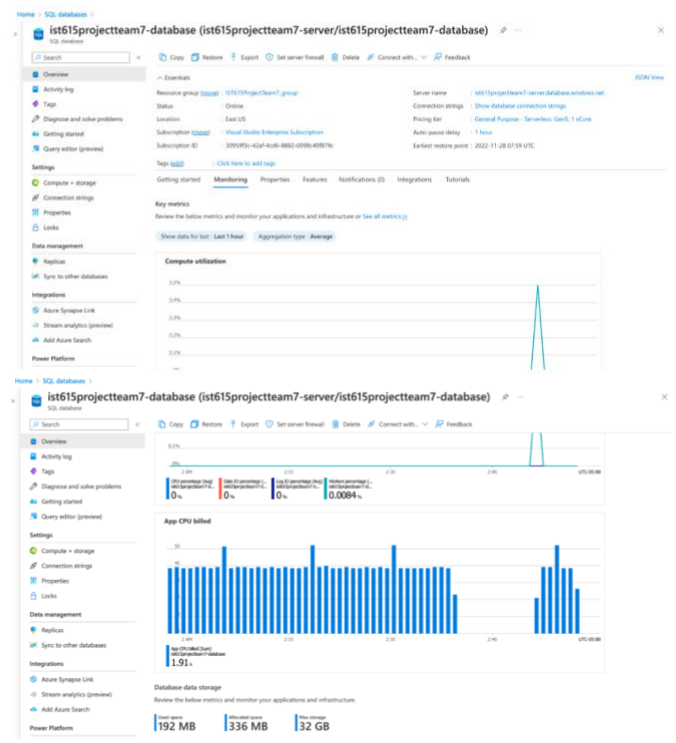
ist615team7file
Container

Authentication method: Access key (Switch to Azure AD User Account)
Location: ist615team7file

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
vehicles_cleaned.csv	11/13/2022, 6:03:11 ...	Hot (Inferred)		Block blob	74.31 MiB	Available

Azure SQL Database:

The SQL Database was set up so that the data from the blob storage can be integrated with the Database which was communicating with the website.


Home > SQL databases >

ist615projectteam7-database (ist615projectteam7-server/ist615projectteam7-database)

SQL database

Essentials

Resource group (move)	IST615ProjectTeam7_group	Server name	ist615projectteam7-server.database.windows.net
Status	Online	Connection strings	Show database connection strings
Location	East US	Pricing tier	General Purpose - Serverless: Gen5, 1 vCore
Subscription (move)	Visual Studio Enterprise Subscription	Auto-pause delay	1 hour
Subscription ID	30959f3c-42af-4cd6-8882-0098c40f879c	Earliest restore point	2022-11-28 07:59 UTC
Tags (edit)	Click here to add tags		

Key metrics

Review the below metrics and monitor your applications and infrastructure or See all metrics

Show data for last : Last 1 hour   Aggregation type : Average

Compute utilization

CPU percentage (Avg) 0 %; Data IO percentage 0 %; Log IO percentage (Avg) 0 %; Workers percentage 0.0084 %

App CPU billed

App CPU billed (Sum) ist615projectteam7-database 1.91 k

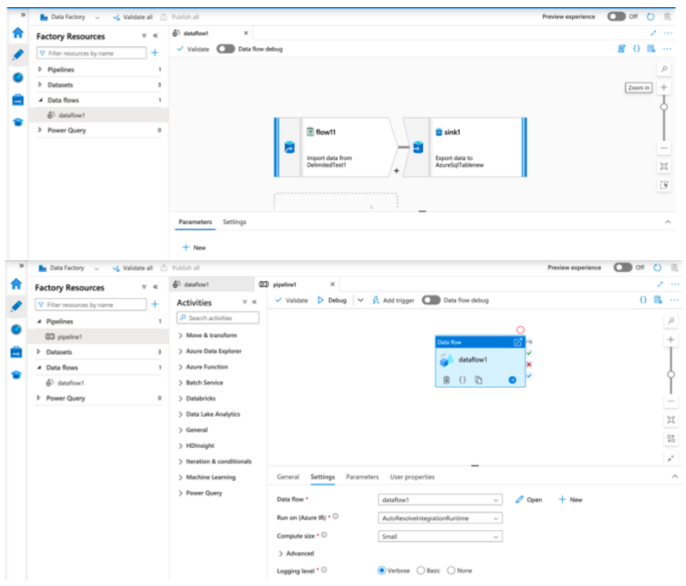
Database data storage

Review the below metrics and monitor your applications and infrastructure.

Used space 192 MB; Allocated space 336 MB; Max storage 32 GB

Azure Data Factory:

Azure Data Factory was used to facilitate data engineering and data integration. A data flow was created to get data from the Azure Blob storage container to Azure SQL DB. This was executed using a pipeline for which we had to set up Azure Blob storage in a right way so that the column headers and the values are accurately analyzed. Data flow was used so that each field in the csv file from the Azure Blob storage was mapped to the column headers.


flow1: Import data from DelimitedText1

sink1: Export data to AzureSqlTablenew

Data flow *: dataflow1

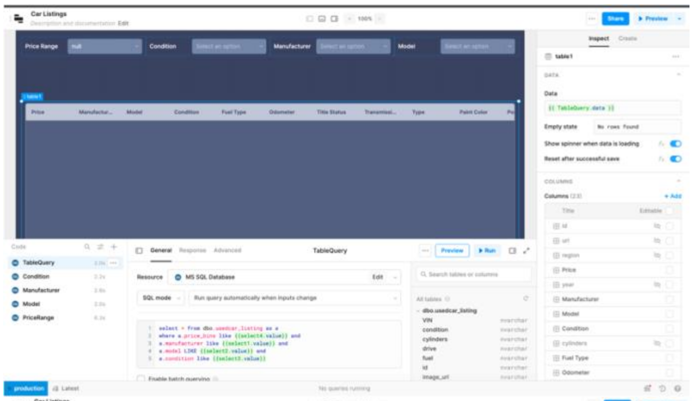
Run on (Azure IR) *: AutoResolveIntegrationRuntime

Compute size *: Small

Logging level *: [✓ Verbose] Basic None

Retool Service:

Retool is not a Microsoft Azure service but is a platform where users can connect to databases and create an application which acts like a CRUD application. To create our website we created three different pages, one in which we can look for the cars listed on the website. To do that basic SQL + JavaScript was used. For Retool to communicate with the database some IPs were added to the firewall rules of the database. Each of the cells were attached to a query that got values from the database based on the selections made by the user.



TableQuery

Resource: MS SQL Database

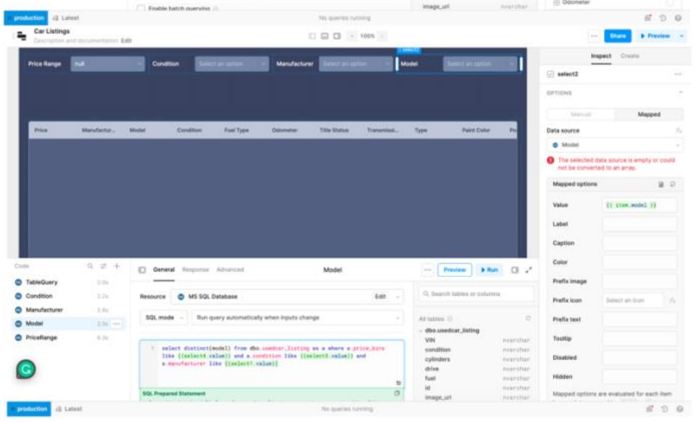
SQL mode: Run query automatically when inputs change

```
select * from dbo.usedcar_listing as a
where a.price_bins like {{select4.value}} and
a.manufacturer like {{select1.value}} and
a.model LIKE {{select2.value}} and
a.condition like {{select3.value}}
```

Data: {{ TableQuery.data }}

Empty state: No rows found

No queries running



Model

Resource: MS SQL Database

SQL mode: Run query automatically when inputs change

```
select distinct(model) from dbo.usedcar_listing as a where a.price_bins
like {{select4.value}} and a.condition like {{select3.value}} and
a.manufacturer like {{select1.value}}
```

SQL Prepared Statement

Data source: Model

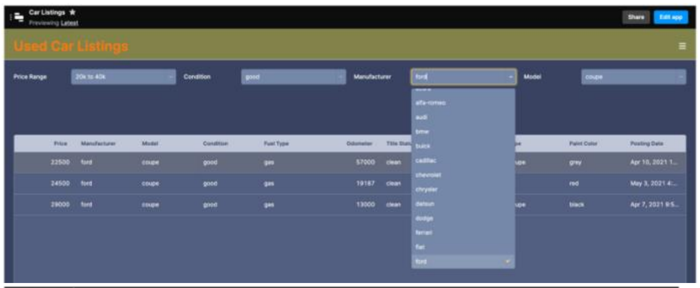
The selected data source is empty or could not be converted to an array.

Mapped options — Value: {{ item.model }}

No queries running

Similarly, another webpage was created for the user to check the value of their car - the highest and the lowest estimated resale value according to the data that we have in our database. The query to find the highest/lowest estimate is the highest/lowest value for that car that has been listed in our database i.e by the manufacturer, model, condition and the title status.

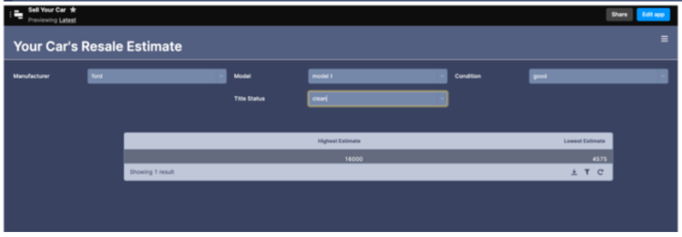
Once the user selects all the fields in a webpage, they will be able to the listings that are available in our database for that specific webpage. Different webpages could be accessed by clicking the navigation button on the top.


Car Listings ★
Previewing Latest

Used Car Listings

Price Range: 20k to 40k | Condition: good | Manufacturer: ford | Model: coupe

Price	Manufacturer	Model	Condition	Fuel Type	Odometer	Title Status		Paint Color	Posting Date
22500	ford	coupe	good	gas	57000	clean	upe	grey	Apr 10, 2021 1...
24500	ford	coupe	good	gas	19187	clean		red	May 3, 2021 4:...
29000	ford	coupe	good	gas	13000	clean	upe	black	Apr 7, 2021 9:5...


Sell Your Car ★
Previewing Latest

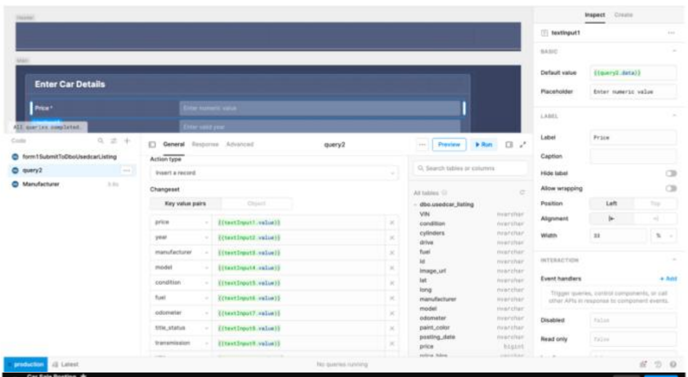
Your Car's Resale Estimate

Manufacturer: ford | Model: model t | Condition: good
Title Status: clean

Highest Estimate	Lowest Estimate
16000	4575

Showing 1 result

A third page was created for users to post their own car for sale. Each of the fields had to be filled to post their car for sale. To post the car, a query with Insert action was used which works exactly like the SQL insert into database command. Each field was given some constraints to enter the information.


Enter Car Details
Price * — Enter numeric value
Enter valid year
All queries completed.

Code: form1SubmitToDboUsedcarListing, query2, Manufacturer

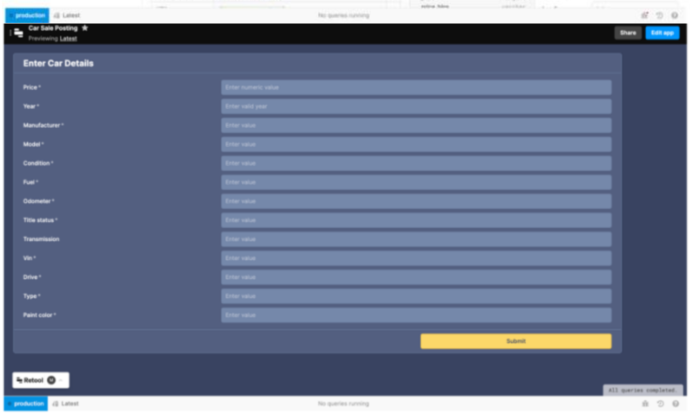
query2 — General | Response | Advanced
Action type: Insert a record
Changeset: Key value pairs

price	{{textInput1.value}}
year	{{textInput2.value}}
manufacturer	{{textInput3.value}}
model	{{textInput4.value}}
condition	{{textInput5.value}}
fuel	{{textInput6.value}}
odometer	{{textInput7.value}}
title_status	{{textInput8.value}}
transmission	{{textInput9.value}}

All tables — dbo.usedcar_listing

VIN	nvarchar
condition	nvarchar
cylinders	nvarchar
drive	nvarchar
fuel	nvarchar
id	nvarchar
image_url	nvarchar
lat	nvarchar
long	nvarchar
manufacturer	nvarchar
model	nvarchar
odometer	nvarchar
paint_color	nvarchar
posting_date	nvarchar
price	bigint

Inspect — textInput1
Default value: {{query2.data}}
Placeholder: Enter numeric value
Label: Price
Position: Left
Width: 33 %
Event handlers: Trigger queries, control components, or call other APIs in response to component events.
Disabled: false
Read only: false


Car Sale Posting ★
Previewing Latest

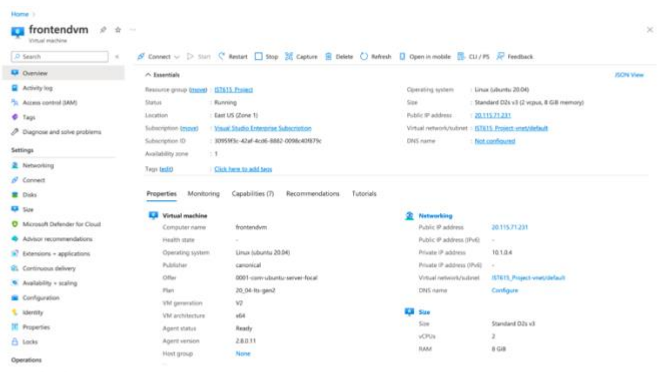
Enter Car Details

Price *	Enter numeric value
Year *	Enter valid year
Manufacturer *	Enter value
Model *	Enter value
Condition *	Enter value
Fuel *	Enter value
Odometer *	Enter value
Title status *	Enter value
Transmission	Enter value
Vin *	Enter value
Drive *	Enter value
Type *	Enter value
Paint color *	Enter value

All queries completed.

Azure Virtual Machine:

Azure Virtual Machine was used to host out website in which we cloned a git docker environment which is responsible for hosting our website. This environment is created by Retool itself. Necessary security and network rules were added to it so that Retool can communicate with Azure. Retool service was hosted on port 3000. To access it we had to put in our public IP address {{pulic_ip}}/3000.



Home >

frontendvm

Virtual machine

Essentials

Resource group (move)	IST615_Project
Status	Running
Location	East US (Zone 1)
Subscription (move)	Visual Studio Enterprise Subscription
Subscription ID	3095f93c-42af-4cd6-8882-0098c40f879c
Availability zone	1
Tags (edit)	Click here to add tags
Operating system	Linux (ubuntu 20.04)
Size	Standard D2s v3 (2 vcpus, 8 GiB memory)
Public IP address	20.115.71.231
Virtual network/subnet	IST615_Project-vnet/default
DNS name	Not configured

Virtual machine

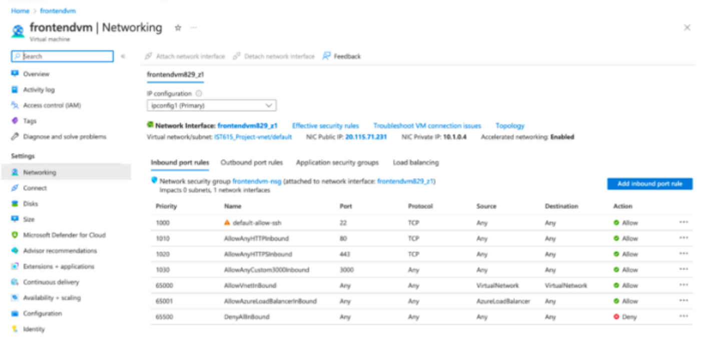
Computer name	frontendvm
Health state	-
Operating system	Linux (ubuntu 20.04)
Publisher	canonical
Offer	0001-com-ubuntu-server-focal
Plan	20_04-lts-gen2
VM generation	V2
VM architecture	x64
Agent status	Ready
Agent version	2.8.0.11
Host group	None

Networking

Public IP address	20.115.71.231
Public IP address (IPv6)	-
Private IP address	10.1.0.4
Private IP address (IPv6)	-
Virtual network/subnet	IST615_Project-vnet/default
DNS name	Configure

Size

Size	Standard D2s v3
vCPUs	2
RAM	8 GiB



Home > frontendvm

frontendvm | Networking

Virtual machine

frontendvm829_z1

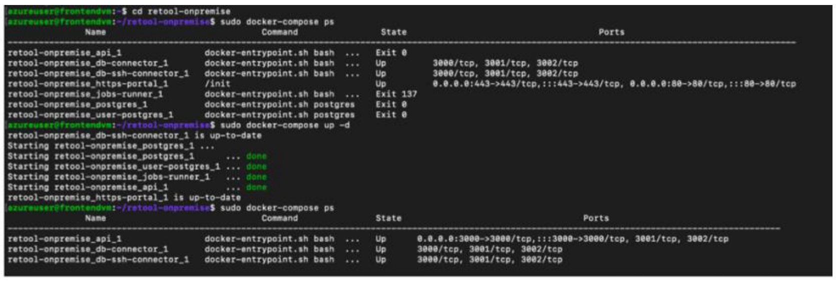
IP configuration: ipconfig1 (Primary)

Network Interface: frontendvm829_z1 Effective security rules Troubleshoot VM connection issues Topology

Virtual network/subnet: IST615_Project-vnet/default NIC Public IP: 20.115.71.231 NIC Private IP: 10.1.0.4 Accelerated networking: Enabled

Network security group frontendvm-nsg (attached to network interface: frontendvm829_z1)
Impacts 0 subnets, 1 network interfaces

Priority	Name	Port	Protocol	Source	Destination	Action
1000	default-allow-ssh	22	TCP	Any	Any	Allow
1010	AllowAnyHTTPInbound	80	TCP	Any	Any	Allow
1020	AllowAnyHTTPSInbound	443	TCP	Any	Any	Allow
1030	AllowAnyCustom3000Inbound	3000	Any	Any	Any	Allow
65000	AllowVnetInBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowAzureLoadBalancerInBound	Any	Any	AzureLoadBalancer	Any	Allow
65500	DenyAllInBound	Any	Any	Any	Any	Deny



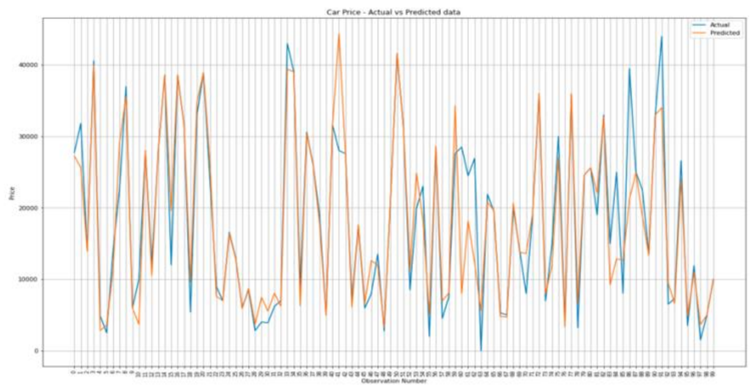
```
azureuser@frontendvm:~$ cd retool-onpremise
azureuser@frontendvm:~/retool-onpremise$ sudo docker-compose ps
              Name                             Command               State                                   Ports
--------------------------------------------------------------------------------------------------------------------------------------------
retool-onpremise_api_1             docker-entrypoint.sh bash  ...   Exit 0
retool-onpremise_db-connector_1    docker-entrypoint.sh bash  ...   Up        3000/tcp, 3001/tcp, 3002/tcp
retool-onpremise_db-ssh-connector_1   docker-entrypoint.sh bash  ...   Up        3000/tcp, 3001/tcp, 3002/tcp
retool-onpremise_https-portal_1    /init                            Up        0.0.0.0:443->443/tcp,:::443->443/tcp, 0.0.0.0:80->80/tcp,:::80->80/tcp
retool-onpremise_jobs-runner_1     docker-entrypoint.sh bash  ...   Exit 137
retool-onpremise_postgres_1        docker-entrypoint.sh postgres    Exit 0
retool-onpremise_user-postgres_1   docker-entrypoint.sh postgres    Exit 0
azureuser@frontendvm:~/retool-onpremise$ sudo docker-compose up -d
retool-onpremise_db-ssh-connector_1 is up-to-date
Starting retool-onpremise_postgres_1 ...
Starting retool-onpremise_postgres_1      ... done
Starting retool-onpremise_user-postgres_1 ... done
Starting retool-onpremise_jobs-runner_1   ... done
Starting retool-onpremise_api_1           ... done
retool-onpremise_https-portal_1 is up-to-date
azureuser@frontendvm:~/retool-onpremise$ sudo docker-compose ps
              Name                             Command               State                                Ports
-----------------------------------------------------------------------------------------------------------------------------------
retool-onpremise_api_1             docker-entrypoint.sh bash  ...   Up      0.0.0.0:3000->3000/tcp,:::3000->3000/tcp, 3001/tcp, 3002/tcp
retool-onpremise_db-connector_1    docker-entrypoint.sh bash  ...   Up      3000/tcp, 3001/tcp, 3002/tcp
retool-onpremise_db-ssh-connector_1   docker-entrypoint.sh bash  ...   Up      3000/tcp, 3001/tcp, 3002/tcp
```

Data Modelling:

Azure Databricks:

Azure Databricks was used for the purpose of python scripting, data analysis and model creation in our project. We wrote a script to integrate Azure Blob storage and Azure Databricks. Using this script, we can read data from our container in blob storage. Some additional data cleansing was done and new columns were added to add on to the features to build price estimation models for users' cars that they had to post online for sale. Three models - Linear Regression, Random Forest and XGBoost were built to predict estimated price. Our best prediction model was Random Forest and it was used to predict prices for these used cars.

Screenshot 17: Graph for comparison of Random Forest Model price predictions and actual prices of cars.



Future Scope:

- Include our model predicted prices in the price estimator webpage of our website.
- Establish connection to PowerBI and build dashboards to display our data analysis, predicted prices and our model performance over time.

Specific contribution to the project:

- Assisted with data exploration and data cleansing.
- Assisted with Retool to build our website.
- Databricks analysis and multiple model building.

Learning outcomes from the project:

- Even though I was aware of the theoretical aspects of cloud services before and had implemented parts of the services for IST 615 assignments, but I never had the chance to practically implement a whole project using cloud services which was very insightful.
- Integration of various cloud services to implement a project.