

AI601: DATA ENGINEERING FOR AI SYSTEMS

Used Car Price Prediction, Search and Analysis

Kashaf Gohar | Eeman Adnan | Muhammad Annus Shabbir

PROJECT OVERVIEW

Goal: The aim of the project is to build an end-to-end data engineering pipeline to predict the price of used cars by preprocessing the data, then train machine learning models for price prediction, storage of data using structured datasets and giving output with analytics and an interactive UI.

Domain/Theme: Data Engineering, Machine Learning and MLOps

Data:

Source: PakWheels used cars listings data..

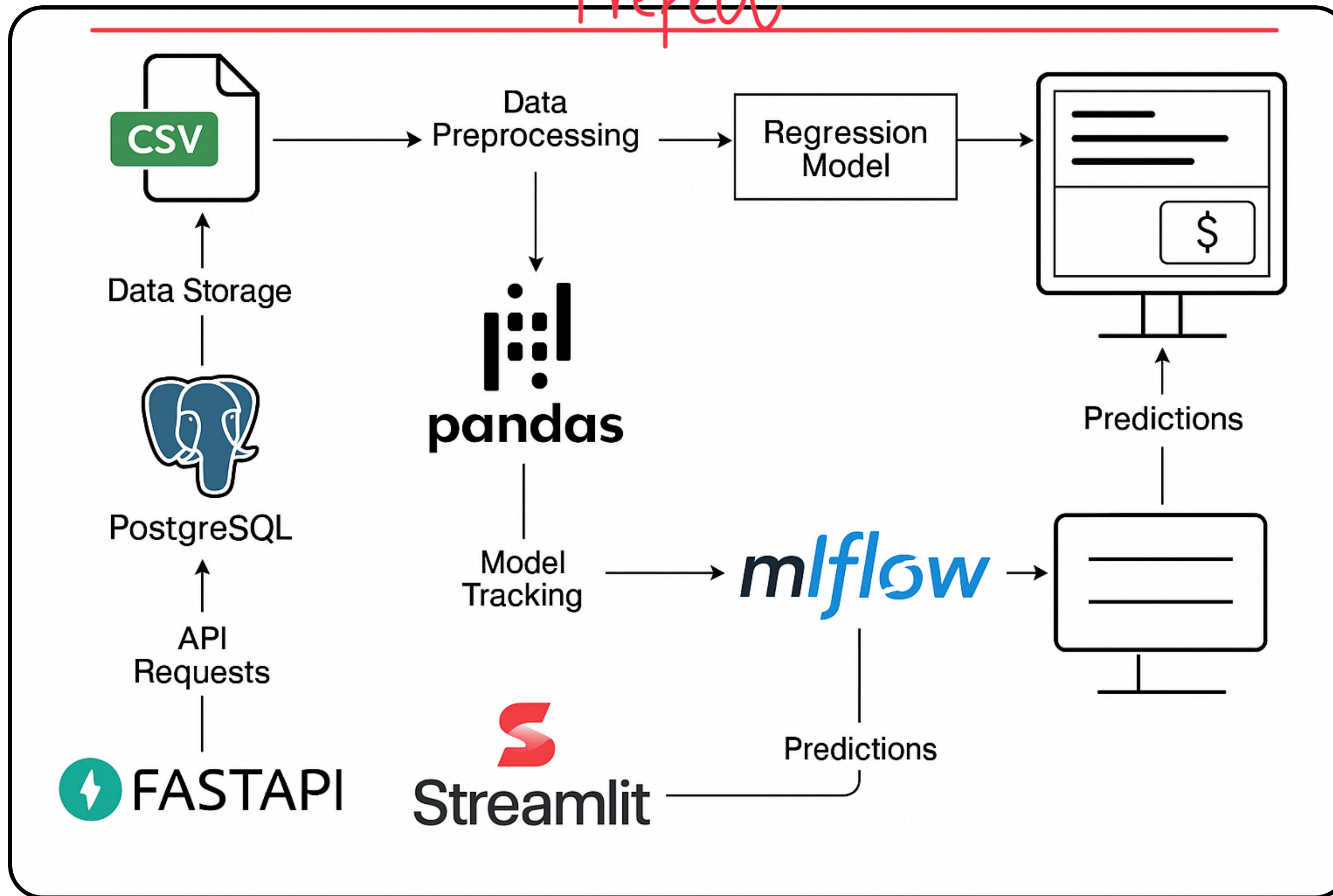
Type: Structured data.

Attributes: City, Make, Model, Year, Engine, Fuel, Mileage, Transmission, Price, etc.

Additions: Synthetic data added for robustness and variation.

ARCHITECTURE DIAGRAM

Project



SCHEMA DIAGRAM

car_data	
id	SERIAL
city	varchar(50)
assembly	varchar(50)
body	varchar(50)
make	varchar(50)
model	varchar(50)
year	int
engine	varchar(50)
transmission	varchar(50)
fuel	varchar(50)
color	varchar(50)
registered	varchar(50)
mileage	int
price	float

Database Explorer

postgres@localhost

public

car_data

columns 14

id integer = nextval('ca')

city varchar(50)

assembly varchar(50)

body varchar(50)

make varchar(50)

model varchar(50)

year integer

engine varchar(50)

transmission varchar(50)

fuel varchar(50)

color varchar(50)

registered varchar(50)

mileage integer

price double precision

keys 1

indexes 4

console

car_data

WHERE

ORDER BY

	id	city	assembly	body	make	model	year	engine
1	7642697	Islamabad	<null>	Compact SUV	KIA	Sorento	2021	3500.0
2	7909608	Sadiqabad	Imported	Hatchback	Daihatsu	Mira	2020	660.0
3	7929224	Peshawar	Imported	Hatchback	Toyota	Vitz	2018	1000.0
4	7832366	Lahore	<null>	Sedan	Toyota	Corolla	2019	1600.0
5	7866725	Islamabad	<null>	Sedan	Toyota	Corolla	2022	1600.0
6	7902938	Lahore	<null>	Sedan	Honda	Civic	2019	1800.0
7	7881747	Gujranwala	Imported	Crossover	Honda	Vezel	2016	1500.0
8	7909055	Bahawalpur	<null>	Sedan	Toyota	Corolla	2018	1300.0
9	7893330	Islamabad	<null>	Hatchback	Suzuki	Wagon	2018	1000.0
10	7812080	Karachi	<null>	SUV	Hyundai	Tucson	2022	2000.0
11	7897667	Karachi	<null>	Sedan	Toyota	Corolla	2018	1300.0
12	7796905	Lahore	<null>	Compact sedan	Honda	City	2021	1200.0
13	7934803	Islamabad	<null>	Sedan	Honda	City	2001	1300.0
14	7805864	Lahore	<null>	Sedan	Honda	City	2017	1300.0
15	7902178	Vehari	<null>	Sedan	Toyota	Corolla	2015	1300.0
16	7926129	Rawalpindi	<null>	Hatchback	Suzuki	Mehran	2019	800.0
17	7891178	Faisalabad	<null>			Saga	<null>	1300.0

Database > postgres@localhost > postgres > public > tables > car_data

SUM: 0 1:2

AI Usage

- Synthetic data generation to add to data source
- FastAPI integration with Database

CHALLENGES

- ***Database Connection and Integration***

Establishing a reliable and persistent connection between FastAPI and PostgreSQL was challenging, especially during local development and Docker deployment.

- ***FastAPI Model Serving***

Integrating MLflow models into FastAPI and ensuring consistent model performance was a non-trivial task. The MLflow models loaded dynamically using `mlflow.sklearn.load_model()` which caused delays on initial API call (cold start problem) and there was dependency mismatch (e.g., NumPy or scikit-learn versions) during loading trained models.

- ***API-Frontend Integration (FastAPI + Streamlit)***

Connecting Streamlit with FastAPI (via `requests.post()` and `requests.get()`) caused CORS issues and broken API endpoints during testing. Localhost ports (8000 for FastAPI and 8501 for Streamlit) had to be configured manually. Streamlit displayed generic errors when API call failed without helpful debug info.

LEARNINGS

- End-to-End ML and Data Engineering Pipeline Architecture
- Model Tracking with MLflow
- Database Management
- Frontend–Backend Communication
- API Development with FastAPI

DEMO