Car Depreciation Prediction App – Technical Report

1. Project Overview

This project is a machine learning-based web application that predicts:

- The **current market value** of a used car
- The **future price** after 1–3 years

The system is built with a complete machine learning pipeline, cloud-based training and storage, and CI/CD automation using AWS. It provides a responsive web interface for public access, deployed via Elastic Beanstalk.

2. Tools, Languages, Libraries, and Services Used

Category	Tool/Library/Service	Purpose
Programming Language	Python	Core language for data processing, ML, and backend development
Data Handling	pandas, numpy	Data cleaning, transformation, and numerical operations
Machine Learning	scikit-learn	Model training (Linear Regression), feature engineering, evaluation
Serialization	dill	Save/load ML models and transformers (model.pkl, preprocessor.pkl)
System Utilities	os, sys, datetime	File paths, environment handling, date-based calculations
Logging & Errors	logging, CustomException	Custom logging and error handling for traceability
Web Framework	Flask	REST API and user interface to serve predictions
Web Server	Flask built-in server	Used for running the app during development and basic deployment
Frontend Templates	HTML	Web forms and result presentation

Category	Tool/Library/Service	Purpose
,Cloud Storage	Amazon S3	Store datasets, trained models, and preprocessor artifacts
Cloud Training	Amazon SageMaker	Train models on large datasets using managed cloud compute
App Deployment	AWS Elastic Beanstalk	Host the Dockerized app with autoscaling and load balancing
CI/CD Automation	AWS CodePipeline	Automatically build and deploy from GitHub to Elastic Beanstalk
Cloud Hosting (VM)	AWS EC2 (via Beanstalk)	Underlying compute resource for the application

3. Core Components of the App

File	Description
data_ingestion.py	Loads dataset, splits into train/test
data_transformation.py	Applies encoding, scaling, and transformation pipelines
model_trainer.py	Trains model, computes RMSE, R^2, and saves final model
predict_pipeline.py	Loads model & preprocessor, makes predictions for user input
application.py	Flask web server that handles form submission and results
utils.py	Custom functions to save/load objects, handle logs
exception.py	Defines custom exception class for debugging