# [redacted] AI/ML Engineer Take Home Assessment
# Image Classification model training

[redacted]

## Instruction

- Please follow closely to the Assignment Instruction
- You can make assumptions to any information not specified within the Assignment Instruction
- You can make references to any other technical literature as required
- Please complete the assignment and email the required deliverables within 7 calendar days upon receiving this email
- Reach out to us if you have any queries via email:
  [redacted]
- Submit Deliverables to:
  [redacted]

## Background

A [redacted] Department has reached out to [redacted] to request for a computer vision based solution to identify car type from images. As an AI/ML engineer working in [redacted], you are required to develop this solution using AI and deploy the solution as a service for use by the requesting users. You have been tasked to train a deep learning image classification model that is able to recognise the classifications of the car images.

## Assessment Instruction

You are tasked to train an image classification model and serve the model via API. Here are the requirements:

1. You are required to complete the assessment using Python 3
2. You may use either Tensorflow2.0-Keras or Pytorch framework to train the model
3. You are encouraged to use any pre-trained open-source classifier models
4. You are required to train the model using Jupyter notebook **(Artefact #1)**.
   In your notebook, you must:
   - Explain the data pre-processing steps
   - Clearly state the model used
   - List and explain any assumptions you have made about the requirements
   - Explain the model training process
   - Present the final training and validation results (confusion matrix, graph output etc.)
   - Save the trained model in either .pth or .pt(Pytorch) , h5 or .hdf5 (Tensorflow)

- Provide the necessary python environment and dependencies (i.e., conda or pip)

5. Once the model is trained, you are also required to wrap the trained model as an API function. You can use either Flask or FastAPI to implement the API functions.
6. You shall implement 2 API functions exposed through *Port 4000* **(Artefact #2)**:

| | | |
|---|---|---|
| **/ping** | To poll if the service is alive and running | The response should be in JSON serialised format with key = 'message' and value = 'pong' |
| **/infer** | To post the image for inference and receive the inference result | An image shall be passed in using the 'files' parameter in a POST request as a base64 binary with key = 'image' and value= <binary value of the image file><br><br>The base64 binary of the image file can be obtained by function *open(test_image_path.jpg, "rb")* or any other equivalent functions<br><br>The response should be in JSON serialised format with key = 'class' and value = " Make, Model, Year" where this represents the classification for each API request. |

7. Finally, containerize the API service using Docker container. The Docker file, along with the execution instruction, is to be submitted for evaluation **(Artefact #3)**. Provide a Readme.md file for your project on the execution steps to build your docker image.

## Dataset

In this assessment, you are only to use the Stanford Cars dataset provided. The data can be downloaded here.

The Stanford Cars dataset contains 16,185 images of 196 classes of cars. The data is split into 8,144 training images and 8,041 testing images, where each class has been split in a 50-50 split. Classes are typically at the level of Make, Model, Year, e.g., 2012 Tesla Model S or 2012 BMW M3 coupe.

## Deliverables

The following artefacts must be submitted within 7 calendar days

1. Artefact #1: Jupyter notebook for model training and evaluation
2. Artefact #2: Source code file(s) for API service
3. Artefact #3: Docker file to containerize API service