Import Libraries

```
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
```

Load the dataset

```
[ ] from google.colab import drive
    drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
[ ] !unzip "/content/drive/MyDrive/Animal_dataset (2).zip"
```

```
Streaming output truncated to the last 5000 lines.
  inflating: animals/animals/bison/73b10839bc.jpg
  inflating: animals/animals/bison/74dc1dd32a.jpg
  inflating: animals/animals/bison/75ab166a7d.jpg
  inflating: animals/animals/bison/79e1610d01.jpg
  inflating: animals/animals/bison/7ba5eee91f.jpg
  inflating: animals/animals/bison/7d214aab4b.jpg
  inflating: animals/animals/bison/7f2ef9ab03.jpg
  inflating: animals/animals/bison/82ec723a4f.jpg
  inflating: animals/animals/bison/85e145af6c.jpg
  inflating: animals/animals/bison/862e2c6379.jpg
  inflating: animals/animals/bison/87a5c700ad.jpg
  inflating: animals/animals/bison/88b723ce90.jpg
  inflating: animals/animals/bison/8c3af742c3.jpg
  inflating: animals/animals/bison/8e34044b98.jpg
  inflating: animals/animals/bison/8e6adbdc40.jpg
  inflating: animals/animals/bison/8fce80d5ba.jpg
  inflating: animals/animals/bison/961a895686.jpg
  inflating: animals/animals/bison/984a5b583d.jpg
  inflating: animals/animals/bison/9e150d02f9.jpg
  inflating: animals/animals/boar/0903d0db72.jpg
  inflating: animals/animals/boar/0b2e935388.jpg
  inflating: animals/animals/boar/0d168ffd8a.jpg
  inflating: animals/animals/boar/12c58c5e8c.jpg
  inflating: animals/animals/boar/13e49ce33a.jpg
  inflating: animals/animals/boar/149cf25549.jpg
  inflating: animals/animals/boar/1643dbb824.jpg
  inflating: animals/animals/boar/17c67e42a7.jpg
  inflating: animals/animals/boar/183f0cd1c2.jpg
  inflating: animals/animals/boar/1e28d918cf.jpg
  inflating: animals/animals/boar/2005eb9945.jpg
  inflating: animals/animals/boar/25ec8c79ee.jpg
  inflating: animals/animals/boar/287ebb3999.jpg
  inflating: animals/animals/boar/28aea02d8b.jpg
  inflating: animals/animals/boar/2bec6d90a6.jpg
  inflating: animals/animals/boar/2c23b537e2.jpg
  inflating: animals/animals/boar/2e85f933b3.jpg
  inflating: animals/animals/boar/2fa8d16887.jpg
  inflating: animals/animals/boar/343c23952b.jpg
  inflating: animals/animals/boar/351fc18170.jpg
  inflating: animals/animals/boar/3c34dc709e.jpg
  inflating: animals/animals/boar/3d28ab32a5.jpg
  inflating: animals/animals/boar/3fc82d21eb.jpg
  inflating: animals/animals/boar/481db3c9bd.jpg
  inflating: animals/animals/boar/4a799151a2.jpg
  inflating: animals/animals/boar/4b3234a9cf.jpg
  inflating: animals/animals/boar/4c9de46c24.jpg
  inflating: animals/animals/boar/4d216b409a.jpg
  inflating: animals/animals/boar/4f5b8e247e.jpg
  inflating: animals/animals/boar/4f712dd94c.jpg
  inflating: animals/animals/boar/517c9f6d24.jpg
  inflating: animals/animals/boar/51c7d65b34.jpg
  inflating: animals/animals/boar/5297f21f92.jpg
  inflating: animals/animals/boar/52fca892e9.jpg
  inflating: animals/animals/boar/549c6255cb.jpg
  inflating: animals/animals/boar/580d7cc298.jpg
  inflating: animals/animals/boar/583c3a1d3e.jpg
  inflating: animals/animals/boar/5aa8637cf7.jpg
```

Data Augmentation

```
[ ] from tensorflow.keras.preprocessing.image import ImageDataGenerator,array_to_img, img_to_array, load_img
```

```
[ ] datagen = ImageDataGenerator(
            rotation_range = 40,
            shear_range = 0.2,
            zoom_range = 0.2,
            horizontal_flip = True,
            brightness_range = (0.5, 1.5))
```

```
[ ] img = load_img('/content/animals/animals/bear/093836b753.jpg')
```

```
[ ] x = img_to_array(img)
```

```
[ ] x
```

```
array([[[152., 166., 177.],
```

```
        [151., 165., 176.],
        [151., 163., 175.],
        ...,
        [149., 156., 166.],
        [150., 157., 167.],
        [147., 154., 164.]],

       [[156., 170., 181.],
        [155., 169., 180.],
        [155., 167., 179.],
        ...,
        [143., 150., 160.],
        [146., 153., 163.],
        [149., 156., 166.]],

       [[153., 167., 178.],
        [153., 167., 178.],
        [153., 165., 177.],
        ...,
        [141., 150., 159.],
        [143., 150., 160.],
        [149., 156., 166.]],

       ...,

       [[156., 136., 111.],
        [157., 137., 112.],
        [156., 136., 111.],
        ...,
        [169., 139., 115.],
        [172., 144., 120.],
        [168., 142., 117.]],

       [[160., 140., 115.],
        [161., 141., 116.],
        [160., 140., 115.],
        ...,
        [164., 134., 110.],
        [166., 138., 114.],
        [162., 136., 111.]],

       [[163., 143., 118.],
        [164., 144., 119.],
        [163., 143., 118.],
        ...,
        [158., 128., 104.],
        [160., 132., 108.],
        [155., 129., 104.]]], dtype=float32)
```

[ ]
```
train_datagen=ImageDataGenerator(rescale=1./255,zoom_range=0.2,horizontal_flip=True,vertical_flip=False)
```

[ ]
```
train_datagen
```

```
<keras.preprocessing.image.ImageDataGenerator at 0x7fdee6732650>
```

Convolution & Pooling layer

[ ]
```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,Convolution2D,MaxPooling2D,Flatten
```

[ ]
```
model=Sequential()
```

[ ]
```
model.add(Convolution2D(32,(3,3),input_shape=(64,64,3),activation='relu'))
```

[ ]
```
model.add(MaxPooling2D(pool_size=(2,2)))
```

[ ]
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 32)        896

 max_pooling2d (MaxPooling2D  (None, 31, 31, 32)       0
 )

=================================================================
Total params: 896
Trainable params: 896
Non-trainable params: 0
_________________________________________________________________
```

Flatten layer

[ ]
```
model.add(Flatten())
```

Minimum of 2 Hidden layers

[ ]
```
model.add(Dense(300,activation='relu'))
model.add(Dense(150,activation='relu'))
```

Output layer

[ ]
```
model.add(Dense(500,activation="relu"))

model.add(Dense(2,activation="softmax"))
```

```
[ ] model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
```

Test the Model

```
[ ] from tensorflow.keras.utils import load_img, img_to_array

    test_image = load_img('/content/animals/animals/bat/0315a4b6b7.jpg', target_size = (64, 64))
```

```
[ ] test_image
```

```
[ ] x = img_to_array(test_image)
    x = np.expand_dims(x, axis =0)

    x.shape
```

```
(1, 64, 64, 3)
```

```
[ ] pred = np.argmax(model.predict(x), axis = 1)

    pred
```

```
1/1 [==============================] - 0s 41ms/step
array([0])
```

```
[ ] print(pred)
```

```
[0]
```