# FINAL CODE

| | |
|---|---|
| Date | 16 November 2022 |
| Team ID | PNT2022TMID21789 |
| Project Name | Project - Intelligent Vehicle Damage Assessment and Cost Estimator for Insurance Companies. |
| Maximum Marks | 4 Marks |

## IMAGE PRE PROCESSING

### Body:

### 1. IMPORT THE IMAGEDATAGENERATOR LIBRARY:

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

### 2. CONFIGURE IMAGEDATAGENERATOR CLASS IMAGE DATA AUGMENTATION:

```
train_datagen = ImageDataGenerator(rescale = 1./255,
shear_range = 0.1, zoom_range = 0.1, horizontal_flip =
True) test_datagen = ImageDataGenerator(rescale = 1./255)
```

### 3. APPLY IMAGE DATA GENERATOR FUNCTIONALITY TO TRAINSET AND TESTSET:

```
training_set =
train_datagen.flow_from_directory('/content/drive/MyDrive/body/trainin
g',target_size = (224, 224),batch_size = 10,class_mode = 'categorical')
test_set =
test_datagen.flow_from_directory('/content/drive/MyDrive/body/validati
on',target_size = (224, 224),batch_size = 10,class_mode = 'categorical')
```

```
Found 979 images belonging to 3 classes.
Found 171 images belonging to 3 classes.
```

## Level:

### 1. Import The ImageDataGenerator Library:

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

### 2. Configure ImageDataGenerator Class:

```
train_datagen = ImageDataGenerator(rescale = 1./255,  shear_range = 0.1,
zoom_range = 0.1, horizontal_flip =  True) test_datagen =
ImageDataGenerator(rescale =  1./255)
```

### 3. Apply ImageDataGenerator Functionality to Trainset And Test set:

```
training_set =
train_datagen.flow_from_directory('/content/drive/MyDrive/level/training',targe
t_size = (224, 224),batch_size = 10,class_mode =   'categorical')
test_set =
test_datagen.flow_from_directory('/content/drive/MyDrive/level/validat
ion',target_size = (224, 224),batch_size = 10,class_mode = 'categorical')
```

```
Found 979 images belonging to 3 classes.
Found 171 images belonging to 3 classes.
```

## MODEL BUILDING

## Body:

### 1. Importing The Model Building Libraries

```
import tensorflow as tf from tensorflow.keras.layers
import Input, Lambda, Dense, Flatten from
tensorflow.keras.models
import Model from tensorflow.keras.applications.vgg16
import VGG16 from tensorflow.keras.applications.vgg19
import VGG19 from tensorflow.keras.preprocessing
```

```
import image from tensorflow.keras.preprocessing.image
import ImageDataGenerator,load_img
from tensorflow.keras.models import Sequential
import numpy as np
from glob import glob
```

## 2. Loading The Model

```
IMAGE_SIZE = [224, 224]
train_path =
'/content/drive/MyDrive/body/training'
valid_path =
'/content/drive/MyDrive/body/validation'
vgg16 =
VGG16(input_shape=IMAGE_SIZE +
[3], weights='imagenet',
include_top=False)
```

```
Downloading data from
https://storage.googleapis.com/tensorflow/kerasapplications/vgg16/vgg1
6_weights_tf_dim_ordering_ tf_kernels_notop.h5 58889256/58889256
[==============================] - 0s 0us/step
```

## 3. Adding Flatten Layer

```
for layer in vgg16.layers:layer.trainable = False
folders = glob('/content/drive/MyDrive/body/training/*')
folders ['/content/drive/MyDrive/body/training/02-side',
 '/content/drive/MyDrive/body/training/00-front',
'/content/drive/MyDrive/body/training/01-rear'] x = Flatten()(vgg16.output)
len(folders)3
```

## 4. Adding Output Layer

```
 prediction = Dense(len(folders), activation='softmax')(x)
```

## 5. Creating A Model Object

```
model = Model(inputs=vgg16.input, outputs=prediction)
model.summary()
Model: "model"
```

```
_________________________________________________________________
_________ Layer (type)
Output Shape              Param #
=================================================================
========== input_1
(InputLayer)      [(None, 224, 224, 3)]    0                    block1_conv1
(Conv2D)     (None, 224, 224, 64)     1792                    block1_conv2
(Conv2D)     (None, 224, 224, 64)     36928                     block1_pool
(MaxPooling2D)  (None, 112, 112, 64)     0                    block2_conv1
(Conv2D)                  (None,   112,   112,   128)              73856
block2_conv2  (Conv2D)           (None,  112,  112,  128)        147584
block2_pool   (MaxPooling2D)     (None,  56,  56,  128)              0
block3_conv1  (Conv2D)           (None,  56,  56,  256)          295168
block3_conv2  (Conv2D)           (None,  56,  56,  256)          590080
block3_conv3  (Conv2D)           (None,  56,  56,  256)          590080
block3_pool   (MaxPooling2D)     (None,  28,  28,  256)              0
block4_conv1  (Conv2D)           (None,  28,  28,  512)         1180160
block4_conv2  (Conv2D)           (None,  28,  28,  512)         2359808
block4_conv3  (Conv2D)           (None,  28,  28,  512)         2359808
block4_pool   (MaxPooling2D)     (None,  14,  14,  512)              0
block5_conv1  (Conv2D)           (None,  14,  14,  512)         2359808
block5_conv2  (Conv2D)           (None,  14,  14,  512)         2359808
block5_conv3  (Conv2D)           (None,  14,  14,  512)         2359808
block5_pool   (MaxPooling2D)     (None,  7,  7,  512)                0
flatten (Flatten)         (None, 25088)             0                    dense
(Dense)             (None, 3)                 75267
=================================================================
==========
Total params: 14,789,955
Trainable params: 75,267
Non-trainable params: 14,714,688
_________________________________________________________________
_________
```

### 6. Configure The Learning Process

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

### 7. Train The Model

### 8. Save The Model

```
from tensorflow.keras.models import load_model
model.save('/content/drive/MyDrive/ibm project/Intelligent Vehicle Damage Assessment & Cost Estimator/MODEL/BODY.h5')
```

### 9. Test The Model

```
from tensorflow.keras.models import load_model
import cv2 from skimage.transform import resize

model = load_model('/content/drive/MyDrive/ibm project/Intelligent Vehicle Damage Assessment & Cost Estimator/MODEL/BODY.h5')

def detect(frame): img = cv2.resize(frame,(224,224)) img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
if(np.max(img)>1): img = img/255.0 img = np.array([img]) prediction = model.predict(img)
label = ["front","rear","side"] preds = label[np.argmax(prediction)] return preds

data = "/content/drive/MyDrive/body/training/00-front/0007.JPEG" image = cv2.imread(data) print(detect(image))

1/1 [==============================] - 1s 812ms/step front
```

## Level:

### 1. Importing The Model Building Libraries

```
import tensorflow as tf from tensorflow.keras.layers
import Input, Lambda, Dense, Flatten from
tensorflow.keras.models import Model from
tensorflow.keras.applications.vgg16 import VGG16 from
tensorflow.keras.applications.vgg19 import VGG19 from
tensorflow.keras.preprocessing import image from
tensorflow.keras.preprocessing.image import ImageDataGenerator,load_img
from tensorflow.keras.models import Sequential import numpy as
np from glob import glob
```

## 2. Loading The Model

```
IMAGE_SIZE = [224, 224] train_path =
'/content/drive/MyDrive/level/training'
valid_path =
'/content/drive/MyDrive/level/validation'
```

## 3. Adding Flatten Layer

```
vgg16 = VGG16(input_shape=IMAGE_SIZE + [3],
weights='imagenet', include_top=False) Downloading data
from
https://storage.googleapis.com/tensorflow/kerasapplications/
vgg16/vg g16_weights_tf_dim_ordering_
tf_kernels_notop.h5
   58889256/58889256 [==============================] - 2s
   0us/step  for layer in vgg16.layers:layer.trainable = False
folders =
glob('/content/drive/MyDrive/level/training/*') folders
```

```
['/content/drive/MyDrive/level/training/02-moderate',
 '/content/drive/MyDrive/level/training/03-severe',
'/content/drive/MyDrive/level/training/01-minor']   x   =
Flatten()(vgg16.output) len(folders)  3
```

## 4. Adding Output Layer

```
 prediction = Dense(len(folders), activation='softmax')(x)
```

## 5. Creating A Model Object

```
model = Model(inputs=vgg16.input, outputs=prediction)
model.summary()
```

```
Model: "model"
_________________________________________________________________
_________  Layer (type)
Output Shape              Param #
```

```
=================================================================
========== input_1
(InputLayer)                 [(None, 224, 224, 3)]          0
block1_conv1 (Conv2D)        (None, 224, 224, 64)        1792
block1_conv2 (Conv2D)        (None, 224, 224, 64)        36928
block1_pool (MaxPooling2D)   (None, 112, 112, 64)        0
block2_conv1 (Conv2D)        (None, 112, 112, 128)       73856
block2_conv2 (Conv2D)        (None, 112, 112, 128)       147584
block2_pool (MaxPooling2D)   (None, 56, 56, 128)         0
block3_conv1 (Conv2D)        (None, 56, 56, 256)         295168
block3_conv2 (Conv2D)        (None, 56, 56, 256)         590080
block3_conv3 (Conv2D)        (None, 56, 56, 256)         590080
block3_pool (MaxPooling2D)   (None, 28, 28, 256)         0
block4_conv1 (Conv2D)        (None, 28, 28, 512)         1180160
block4_conv2 (Conv2D)        (None, 28, 28, 512)         2359808
block4_conv3 (Conv2D)        (None, 28, 28, 512)         2359808
block4_pool (MaxPooling2D)   (None, 14, 14, 512)         0
block5_conv1 (Conv2D)        (None, 14, 14, 512)         2359808
block5_conv2 (Conv2D)        (None, 14, 14, 512)         2359808
block5_conv3 (Conv2D)        (None, 14, 14, 512)         2359808
block5_pool (MaxPooling2D)   (None, 7, 7, 512)           0
flatten (Flatten)            (None, 25088)               0
dense
(Dense)             (None, 3)                 75267
=================================================================
================== Total params: 14,789,955
Trainable params: 75,267
Non-trainable params: 14,714,688
_________________________________________________________________
_________
```

## 6. Configure The Learning Process

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

## 7. Train The Model

```
r = model.fit_generator( training_set,
validation_data=test_set, epochs=5,
steps_per_epoch=len(training_set),
validation_steps=len(test_set) )  /usr/local/lib/python3.7/dist-
packages/ipykernel_launcher.py:6: UserWarning:
`Model.fit_generator` is deprecated and will be removed in a future version.
Please use `Model.fit`, which supports generators.

Epoch 1/5
98/98 [==============================] - 407s 4s/step -
loss: 1.2409 - accuracy: 0.5628 - val_loss:
1.2019 - val_accuracy: 0.5614 Epoch 2/5
98/98 [==============================] - 18s 179ms/step - loss:
0.7316
- accuracy: 0.7191 - val_loss: 0.9586 - val_accuracy: 0.6082  Epoch 3/5
98/98 [==============================] - 16s 164ms/step - loss:
0.5469
- accuracy: 0.7957 - val_loss: 1.0207 - val_accuracy: 0.6140  Epoch 4/5
98/98 [==============================] - 16s 167ms/step - loss:
0.4278
- accuracy: 0.8223 - val_loss: 1.6515 - val_accuracy: 0.5965
Epoch 5/5
98/98 [==============================] - 17s
177ms/step - loss: 0.4449 - accuracy: 0.8284 - val_loss:
1.2299 - val_accuracy: 0.6199
```

## 8. Save The Model

```
from tensorflow.keras.models import
load_model
model.save('/content/drive/MyDrive/ibm project/Intelligent Vehicle Damage Assessment & Cost Estimator/MODEL/LEVEL.h5')
```

## 9. Test The Model

```
from tensorflow.keras.models import
load_model import cv2 from
skimage.transform import resize

model = load_model('/content/drive/MyDrive/ibm project/Intelligent Vehicle Damage Assessment & Cost Estimator/MODEL/LEVEL.h5')
```

```
def detect(frame):   img = cv2.resize(frame,(224,224))   img =
cv2.cvtColor(img,cv2.COLOR_BGR2RG
B)   if(np.max(img)>1):     img = img/255.0     img =
np.array([img])     prediction = model.predict(img)
                                    label   =
["minor","moderate","severe"]     preds =
label[np.argmax(prediction)]     return preds
```

```
data = "/content/drive/MyDrive/level/training/01-minor/0007.JPEG" image
= cv2.imread(data)  print(detect(image))
```

```
1/1 [==============================] - 0s 157ms/step minor
```

## HTML File

Index:

```
<!DOCTYPE html>
<!-- saved from url=(0051)https://haripit193.wixsite.com/vehicle-
damageinsur -->
<html lang="en"><head><meta http-equiv="Content-Type"
content="text/html; charset=UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1"
id="wixDesktopViewport">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="generator" content="Wix.com Website Builder
<link rel="icon" sizes="192x192" href="https://www.wix.com/favicon.ico">
<link rel="shortcut icon" href="https://www.wix.com/favicon.ico"
type="image/x-icon">
  <link rel="apple-touch-icon" href="https://www.wix.com/favicon.ico"
type="image/x-icon">
  <!-- Safari Pinned Tab Icon -->
  <!-- <link rel="mask-icon" href="https://www.wix.com/favicon.ico"> -->
  <!-- Legacy Polyfills -->
  <script nomodule="" src="./index_files/minified.js.download"></script>
  <script nomodule="" src="./index_files/focus-
withinpolyfill.js.download"></script>   <script
nomodule=""
src="./index_files/polyfill.min.js.download"></script>
  <!-- Performance API Polyfills -->
  <script>   (function
() {
```

```
   var noop = function noop() {};     if
("performance" in window === false) {
window.performance = {};
   }
   window.performance.mark = performance.mark || noop;
window.performance.measure = performance.measure || noop;
if ("now" in window.performance === false) {       var
nowOffset = Date.now();
     if (performance.timing && performance.timing.navigationStart) {
nowOffset = performance.timing.navigationStart;
     }
     window.performance.now = function now() {
return Date.now() - nowOffset;
     };
 })();
 </script>

 <!-- Globals Definitions -->
 <script>     (function () {       var now = Date.now()
window.initialTimestamps = {         initialTimestamp: now,
initialRequestTimestamp: Math.round(performance.timeOrigin ?
performance.timeOrigin : now - performance.now())
     }
     window.thunderboltTag = "libs-releases-GA-local"
window.thunderboltVersion = "1.11233.0"
   })();
 </script><!-- Old Browsers Deprecation -->
   <script data-
url="https://static.parastorage.com/services/wixthunderbolt.....
```

Login:

```
 <!DOCTYP
 E html>
              <!-- saved from url=(0059)https://haripit193.wixsite.com/vehicle-
              damage-
            insur/blank-1 -->
              <html lang="en"><head><meta http-equiv="Content-Type"
              content="text/html;
           charset=UTF-8">
```

```
    <meta name="viewport" content="width=device-width, initial-
   scale=1"
  id="wixDesktopViewport">
 <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="generator" content="Wix.com
   Website Builder">

<link rel="icon" sizes="192x192" href="https://www.wix.com/favicon.ico">
    <link rel="shortcut icon"
   href="https://www.wix.com/favicon.ico"
  type="image/x-icon">
    <link rel="apple-touch-icon"
   href="https://www.wix.com/favicon.ico"
  type="image/x-icon">
    <!-- Safari Pinned Tab Icon -->
    <!-- <link rel="mask-icon"
   href="https://www.wix.com/favicon.ico"> -->

    <!-- Legacy Polyfills -->
 <script nomodule="" src="./Login_files/minified.js.download"></script>
    <script nomodule="" src="./Login_files/focus-within
 polyfill.js.download"></script>
    <script nomodule=""
   src="./Login_files/polyfill.min.js.download"></script>

  <!-- Performance API Polyfills -->
    <script>
    (function () {
 var noop = function noop() {};      if
("performance" in window === false) {
    window.performance = {};
      }
  window.performance.mark = performance.mark || noop;
  window.performance.measure = performance.measure ||
noop;      if ("now" in window.performance === false) {
    var nowOffset = Date.now();
```

```
                if (performance.timing &&
            performance.timing.navigationStart) {        nowOffset =
            performance.timing.navigationStart;
                }        window.performance.now =
function now() {        return Date.now() -
nowOffset;
            };
                }
              })();
             </script>


            <!-- Globals Definitions -->
             <script>
          (function () {          var now =
Date.now()
window.initialTimestamps = {
initialTimestamp: now,
             initialRequestTimestamp: Math.round(performance.timeOrigin ?
        performance.timeOrigin : now - performance.now())
                }


        window.thunderboltTag = "libs-releases-GA-local"
window.thunderboltVersion = "1.11233.0"
                })();
             </script>



          <!-- Old Browsers Deprecation -->
             <script data-
url="https://static.parasto.....
```

Register:

```
<!DOCTYP
E html>
            <!-- saved from url=(0059)https://haripit193.wixsite.com/vehicle-
            damage-
```

```
insur/blank-2 -->
<html lang="en"><head><meta http-equiv="Content-Type"
content="text/html;
charset=UTF-8">


<meta name="viewport" content="width=device-width, initial-
scale=1"
id="wixDesktopViewport">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="generator" content="Wix.com
Website Builder">

<link rel="icon" sizes="192x192" href="https://www.wix.com/favicon.ico">
<link rel="shortcut icon"
href="https://www.wix.com/favicon.ico"
type="image/x-icon">
<link rel="apple-touch-icon"
href="https://www.wix.com/favicon.ico"
type="image/x-icon">
<!-- Safari Pinned Tab Icon -->
<!-- <link rel="mask-icon"
href="https://www.wix.com/favicon.ico"> -->

<!-- Legacy Polyfills -->
<script nomodule="" src="./Register_files/minified.js.download"></script>
<script nomodule="" src="./Register_files/focus-within
polyfill.js.download"></script>
<script nomodule=""

src="./Register_files/polyfill.min.js.download"></
script>

<!-- Performance API Polyfills -->
<script>
(function () {
var noop = function noop() {};     if
("performance" in window === false) {
```

```
                window.performance = {};
                  }
          window.performance.mark = performance.mark || noop;
          window.performance.measure = performance.measure ||
noop;             if ("now" in window.performance === false) {
                  var nowOffset = Date.now();
            if (performance.timing && performance.timing.navigationStart) {
              nowOffset = performance.timing.navigationStart;
                    }
            window.performance.now = function now() {
return Date.now() - nowOffset;
                    };
                  }
                 })();
                </script>


              <!-- Globals Definitions -->
               <script>
          (function () {           var now =
Date.now()
window.initialTimestamps = {
initialTimestamp: now,
                     initialRequestTimestamp:
              Math.round(performance.timeOrigin ?
        performance.timeOrigin : now - performance.now())
                    }


     window.thunderboltTag = "libs-releases-GA-local"
window.thunderboltVersion = "1.11233.0"
                  })();
                </script>



              <!-- Old Browsers Deprecation -->
                 <script data-url="https://static.parastorage.com/.....
```

Prediction:

```
<!DOCTYP
E html>
            <!-- saved from url=(0059)https://haripit193.wixsite.com/vehicle-
            damage-
          insur/blank-3 -->
            <html lang="en"><head><meta http-equiv="Content-Type"
            content="text/html;
         charset=UTF-8">


             <meta name="viewport" content="width=device-width, initial-
            scale=1"
       id="wixDesktopViewport">
     <meta http-equiv="X-UA-Compatible" content="IE=edge">
             <meta name="generator" content="Wix.com
           Website Builder">

  <link rel="icon" sizes="192x192" href="https://www.wix.com/favicon.ico">
             <link rel="shortcut icon"
           href="https://www.wix.com/favicon.ico"
        type="image/x-icon">
             <link rel="apple-touch-icon"
           href="https://www.wix.com/favicon.ico" type="image/x-icon">
             <!-- Safari Pinned Tab Icon -->
             <!-- <link rel="mask-icon"
           href="https://www.wix.com/favicon.ico"> -->

           <!-- Legacy Polyfills -->
             <script nomodule=""
      src="./Prediction_files/minified.js.download"></script>
             <script nomodule="" src="./Prediction_files/focus-within
    polyfill.js.download"></script>
             <script nomodule=""

            src="./Prediction_files/polyfill.min.js.download"><
           /script>

          <!-- Performance API Polyfills -->
            <script>
```

```
(function () {
    var noop = function noop() {};      if
("performance" in window === false) {
                window.performance = {};
                  }
          window.performance.mark = performance.mark || noop;
          window.performance.measure = performance.measure ||
noop;             if ("now" in window.performance === false) {
                 var nowOffset = Date.now();
            if (performance.timing && performance.timing.navigationStart) {
              nowOffset = performance.timing.navigationStart;
                    }
            window.performance.now = function now() {
return Date.now() - nowOffset;
                    };
                  }
                 })();
                </script>



              <!-- Globals Definitions -->
                <script>
           (function () {           var now =
Date.now()
window.initialTimestamps = {
initialTimestamp: now,
                     initialRequestTimestamp:
               Math.round(performance.timeOrigin ?
         performance.timeOrigin : now - performance.now())
                    }


                                window.thunderboltTag = "libs-releases-GA-local"
                                        window.thunderboltVersion = "1.11233.0"
                  })();
                </script>



              <!-- Old Browsers Deprecation -->
                 <script data-url="https://static.parastorage.com/servi.....
```

Logout:

```
<!DO
CTY
PE
html>

        <!-- saved from url=(0059)https://haripit193.wixsite.com/vehicle-
        damage-
      insur/blank-1 -->
        <html lang="en"><head><meta http-equiv="Content-Type"
        content="text/html; charset=UTF-8">


          <meta name="viewport" content="width=device-width, initial-
        scale=1"
    id="wixDesktopViewport">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
          <meta name="generator" content="Wix.com
        Website Builder">

<link rel="icon" sizes="192x192" href="https://www.wix.com/favicon.ico">
          <link rel="shortcut icon"
        href="https://www.wix.com/favicon.ico"
      type="image/x-icon">
          <link rel="apple-touch-icon"
        href="https://www.wix.com/favicon.ico"
      type="image/x-icon">
        <!-- Safari Pinned Tab Icon -->
          <!-- <link rel="mask-icon"
        href="https://www.wix.com/favicon.ico"> -->

          <!-- Legacy Polyfills -->
  <script nomodule="" src="./Login_files/minified.js.download"></script>
          <script nomodule="" src="./Login_files/focus-within-
      polyfill.js.download"></script>
          <script nomodule=""
        src="./Login_files/polyfill.min.js.download"></script>
```

```
<!-- Performance API Polyfills -->
<script>
(function () {
var noop = function noop() {};  if ("performance" in window === false) {
window.performance = {};
}
window.performance.mark = performance.mark || noop;
window.performance.measure = performance.measure || noop;  if ("now" in window.performance === false) {
var nowOffset = Date.now();
if (performance.timing && performance.timing.navigationStart) {
nowOffset = performance.timing.navigationStart;
}
window.performance.now = function now() {
return Date.now() - nowOffset;
};
}
})();
</script>


<!-- Globals Definitions -->
<script>
(function () {  var now = Date.now()
window.initialTimestamps = {
initialTimestamp: now,
initialRequestTimestamp: Math.round(performance.timeOrigin ? performance.timeOrigin : now - performance.now())
}

window.thunderboltTag = "libs-releases-GA-local"
window.thunderboltVersion = "1.11233.0"
})();
</script>
```

<!-- Old Browsers Deprecation -->
<script data-url="https://static.parastorage.com/ser.....

Python code

**Body:**

```
from keras.models import Sequential from
keras.layers import Convolution2D from
keras.layers import MaxPooling2D from
keras.layers import Flatten from
keras.layers import Dense from
keras.models import model_from_json
import matplotlib.pyplot as plt import
warnings
warnings.filterwarnings('ignore') batch_size =
32

from tensorflow.keras.preprocessing.image import
ImageDataGenerator

# All images will be rescaled by 1./255 train_datagen =
ImageDataGenerator(rescale=1/255)

# Flow training images in batches of 128 using train_datagen
generator
train_generator = train_datagen.flow_from_directory(
'body',  # This is the source directory for training images
        target_size=(200, 200),  # All images will be resized to 200 x
200
        batch_size=batch_size,         # Specify the classes explicitly
classes = ['00-front','01-rear','02-side'],         # Since we use
categorical_crossentropy loss, we need categorical labels
        class_mode='categorical')

import tensorflow as tf
#cnn Model
model = tf.keras.models.Sequential([
    # Note the input shape is the desired size of the image
```

```
200x 200 with 3 bytes color
# The first convolution
   tf.keras.layers.Conv2D(16, (3,3), activation='relu',
input_shape=(200, 200, 3)),
tf.keras.layers.MaxPooling2D(2, 2),     # The second
convolution
   tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
tf.keras.layers.MaxPooling2D(2,2),     # The third
convolution
   tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
tf.keras.layers.MaxPooling2D(2,2),     # The fourth
convolution
   tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
tf.keras.layers.MaxPooling2D(2,2),     # The fifth convolution
   tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
tf.keras.layers.MaxPooling2D(2,2),
   # Flatten the results to feed into a dense layer
tf.keras.layers.Flatten(),
   # 128 neuron in the fully-connected layer
tf.keras.layers.Dense(128, activation='relu'),     # 5
output neurons for 5 classes with the softmax activation
   tf.keras.layers.Dense(3, activation='softmax')
])
model.summary()

from tensorflow.keras.optimizers import RMSprop early
=
tf.keras.callbacks.EarlyStopping(monitor='val_loss',patience=5)
model.compile(loss='categorical_crossentropy',
optimizer=RMSprop(lr=0.001),              metrics=['accuracy'])

total_sample=train_generator.n n_epochs
= 20
 history = model.fit_generator(
train_generator,
       steps_per_epoch=int(total_sample/batch_size),
epochs=n_epochs,        verbose=1)
model.save('body.h5')

acc = history.history['accuracy']

loss = history.history['loss']
```

```
epochs = range(1, len(acc) + 1)

# Train and validation accuracy
plt.plot(epochs, acc, 'b', label=' accurarcy')
 plt.title('  accurarcy')
plt.legend()

plt.figure()

# Train and validation loss plt.plot(epochs,
loss, 'b', label=' loss') plt.title('     loss')
plt.legend() plt.show()
```

**Level:**

```
from
keras.mod
el s import
Sequential

            from keras.layers import Convolution2D
           from keras.layers import MaxPooling2D
                from keras.layers import Flatten
                from keras.layers import Dense   from
keras.models import model_from_json          from
tensorflow.keras.applications.vgg16 import VGG16
 import matplotlib.pyplot as plt   import
warnings
 warnings.filterwarnings('ignore')        batch_size
= 32


                from tensorflow.keras.preprocessing.image
                import
             ImageDataGenerator


 # All images will be rescaled by 1./255        train_datagen
= ImageDataGenerator(rescale=1/255)
```

```
# Flow training images in batches of 128 using
train_datagen generator
train_generator = train_datagen.flow_from_directory(
'level',  # This is the source directory for
training images
target_size=(200, 200),  # All images will be
resized to 200 x 200 batch_size=batch_size,
# Specify the classes explicitly classes = ['01-minor','02-
moderate','03-severe'],
# Since we use categorical_crossentropy
loss, we
need categorical labels
class_mode='categorical')


import tensorflow as tf
#cnn Model
model = tf.keras.models.Sequential([
# Note the input shape is the desired size of the
image
200x 200 with 3 bytes color
# The first convolution
tf.keras.layers.Conv2D(16, (3,3),
activation='relu',
input_shape=(200, 200, 3)),
tf.keras.layers.MaxPooling2D(2, 2),  # The second
convolution  tf.keras.layers.Conv2D(32, (3,3),
activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),  # The third
convolution  tf.keras.layers.Conv2D(64, (3,3),
activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
# The fourth convolution
tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
# The fifth convolution
    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
tf.keras.layers.MaxPooling2D(2,2),  # Flatten the results
to feed into a dense layer  tf.keras.layers.Flatten(),  #
128 neuron in the fully-connected layer
tf.keras.layers.Dense(128, activation='relu'),
```

```
    # 5 output neurons for 5 classes with the
   softmax
  activation
   tf.keras.layers.Dense(3, activation='softmax')
  ])



  model.summary()



from tensorflow.keras.optimizers import RMSprop
    early =
    tf.keras.callbacks.EarlyStopping(monitor='val_loss',pat
    ienc
   e=5)
model.compile(loss='categorical_crossentropy',
optimizer=RMSprop(lr=0.001),
                metrics=['accuracy'])



  total_sample=train_generator.n



  n_epochs = 20



history = model.fit_generator(      train_generator,
          steps_per_epoch=int(total_sample/batch_size),
            epochs=n_epochs,
             verbose=1)
```

```
model.save('level.h5')

acc = history.history['accuracy']

loss = history.history['loss']

epochs = range(1, len(acc) + 1)

 # Train and validation accuracy          plt.plot(epochs,
acc, 'b', label=' accurarcy')

 plt.title('  accurarcy')        plt.legend()

plt.figure()

 # Train and validation loss  plt.plot(epochs, loss,
'b', label=' loss')
plt.title('  loss')
plt.legend()
plt.show()
```

**Main:**

```
from flask
import Flask,
render_templ
a te, flash,
request,sessi
on
```

```
from cloudant.client import  Cloudant


 import cv2


  client = Cloudant.iam("eb55a2b7-ae45-4df8-8d1c-
  69c5229ffdbe-
  bluemix","YzG5FZg9Vs_HScOBZaWyVXm7PpN
  jbPrmPaPMfHx7w3X9",co
 nnect=True)
my_database = client.create_database("database-dharan")



app = Flask(__name__)   app.config.from_object(__name__)
app.config['SECRET_KEY'] =
          '7d441f27d441f27567d441f2b6176a'



@app.route("/")     def
homepage():


          return render_template('index.html')



@app.route("/userhome")          def
userhome():
```

```
                return render_template('userhome.html')
            @app.route("/addamount")


@app.route("/NewUser")          def
NewUser():


    return render_template('NewUser.html')    @app.route("/user")
def user():


                return render_template('user.html')




@app.route("/newuse",methods=['GET','POST'])    def
newuse():
                if request.method == 'POST':#


                    x = [x for x in request.form.values()]
        print(x)                data = {
                        '_id': x[1],
                        'name': x[0],
                        'psw': x[2]
                    }
                    print(data)
                    query = {'_id': {'Seq': data['_id']}}
        docs = my_database.get_query_result(query)
print(docs)          print(len(docs.all()))              if
(len(docs.all()) == 0):
                        url = my_database.create_document(data)
                return render_template('goback.html',
data="Register, please login using your details")          else:
```

```
                return render_template('goback.html',
            data="You are already a member, please login
            using your
            details")


 @app.route("/userlog", methods=['GET', 'POST'])  def
userlog():          if request.method == 'POST':

            user = request.form['_id']              passw =
request.form['psw']
                print(user, passw)



                query = {'_id': {'$eq': user}}
        docs = my_database.get_query_result(query)
print(docs)          print(len(docs.all()))          if
(len(docs.all()) == 0):
                    return
render_template('goback.html',   pred="The username is not
found.")          else:
                    if ((user == docs[0][0]['_id'] and passw
            == docs[0][0]['psw'])):



                    return
 render_template("userhome.html")
else:
                    return
            render_template('goback.html',data="user name
            and password
            incorrect")
```

```
@app.route("/predict", methods=['GET', 'POST'])
def predict():
    if request.method == 'POST':

        file = request.files['fileupload']
        file.save('static/Out/Test.jpg')

        import warnings
        warnings.filterwarnings('ignore')

        import tensorflow as tf
        classifierLoad = tf.keras.models.load_model('body.h5')

        import numpy as np
        from keras.preprocessing import image

        test_image = image.load_img('static/Out/Test.jpg', target_size=(200, 200))
        img1 = cv2.imread('static/Out/Test.jpg')
        #test_image = image.img_to_array(test_image)
        test_image = np.expand_dims(test_image, axis=0)
        result = classifierLoad.predict(test_image)

        result1 = ''

        if result[0][0] == 1:
```

```
result1 = "front"

elif result[0][1] == 1:

result1 = "rear"

elif result[0][2] == 1:
result1 = "side"

file = request.files['fileupload1']
file.save('static/Out/Test1.jpg')

import warnings
warnings.filterwarnings('ignore')

import tensorflow as tf
classifierLoad =
tf.keras.models.load_model('level.h5')

import numpy as np
from keras.preprocessing import image

test_image =
```

```
                image.load_img('static/Out/Test1.jpg',
                target_size=(200,
                200))
        img1 = cv2.imread('static/Out/Test1.jpg')               #
test_image = image.img_to_array(test_image)
test_image = np.expand_dims(test_image, axis=0)
                    result = classifierLoad.predict(test_image)


                    result2 = "

                                                    if result[0][0] == 1:

            result2 = "minor"



                                                elif result[0][1] == 1:

                result2 = "moderate"

                                                elif result[0][2] == 1:
                        result2 = "severe"




                    if (result1 == "front" and result2 == "minor"):
                        value = "3000 - 5000 INR"
                    elif (result1 == "front" and result2 ==
                "moderate"):
            value = "6000 8000 INR"             elif (result1 ==
"front" and result2 == "severe"):
                        value = "9000 11000 INR"

                elif (result1 == "rear" and result2 == "minor"):
                        value = "4000 - 6000 INR"
```

```
            elif (result1 == "rear" and result2 ==
        "moderate"):
                value = "7000 9000 INR"

        elif (result1 == "rear" and result2 == "severe"):
                value = "11000 - 13000 INR"

        elif (result1 == "side" and result2 == "minor"):
                value = "6000 - 8000 INR"


            elif (result1 == "side" and result2 ==
        "moderate"):
                value = "9000 - 11000 INR"

        elif (result1 == "side" and result2 == "severe"):
                value = "12000 - 15000 INR"

                                                    else:
                value = "16000 - 50000 INR"




            return render_template('userhome.html',
        prediction=value)




    if __name__ == '__main__':     app.run(debug=True,
use_reloader=True)
```