Table of Contents

Chapter 1

1. INTRODUCTION

In this Training, I was trained in the field of Machine Learning, which is concerned with the building and deployment of the websites on internet. It concerns with the building up Machine Learning models and deploying them on the Internet.

2. Introduction to Project

The project is a **Caption an Image Generator** which helps the company's website to implement a feature of scanning an image and then generation the suitable caption for the image.

The goal is to help the organization unlock value by cutting complexity and delivering consistent and clear ways of working and also, to provide the best efficiency and help to the company. This needs a high- performing design in the website, balance optimization of machine learning model, operational efficiency .

The project was completed and ultimately deployed over Internet using different technologies on the Heroku WebApp.

3. Scope of the Project

The Machine Learning finds its applications in almost all the important fields today. The scope of this project is as follows:

a) **Web and app based services**: Feature can be implemented in various apps and websites.

b) **Security based services:** Providing security from 3rd party threats andviruses by installing anti malware and anti-virus soft wares.

c) **Online Scanner :** Helping students and others to quickly upload the image they want to scan and generate the correct output for them.

d) **Generate Captions for the Blind :** In the future, it can be modified and be made as such to generate the captions for the surrounding for the blind people by scanning the surrounding and reading aloud the caption.

Chapter 2

4. The Technology Stack Used:

5.1 ML Model

Python
Machine Learning
Pandas
Matplotlib
Tensor Flow

5.2 Deployment

Flask
Heroku
HTML
CSS

5.3 Marketing Tools

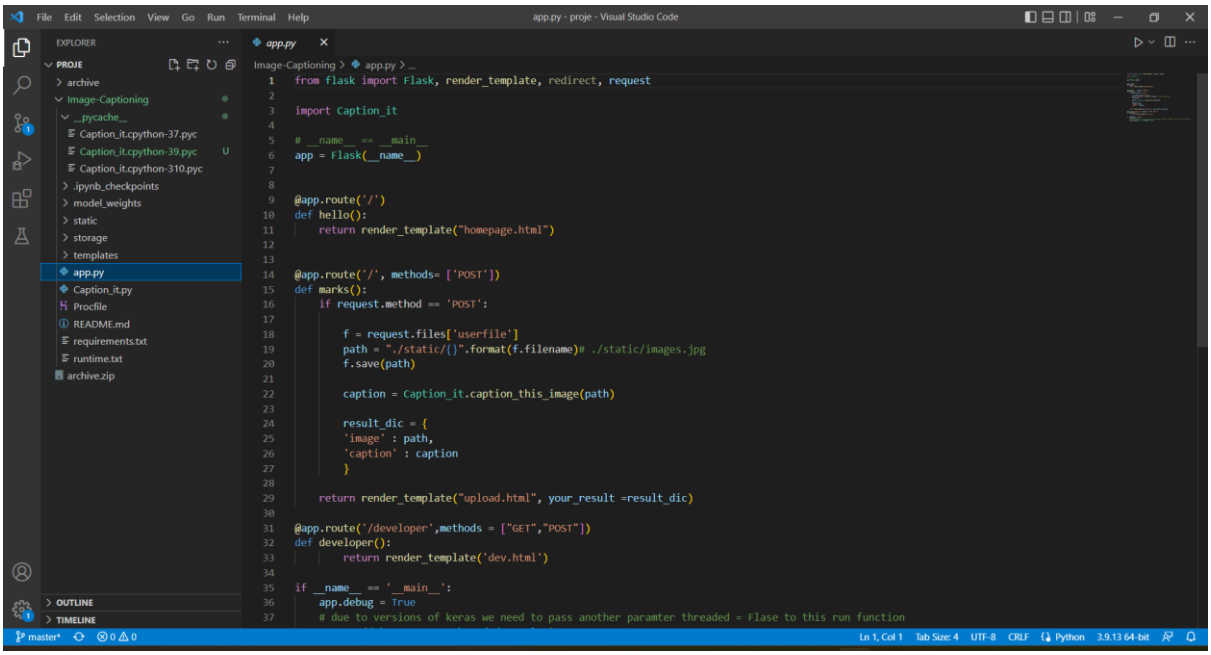
Site deployed on Heroku Web App using the Flask Web Framework.
LINK : https://caption-an-image.herokuapp.com/

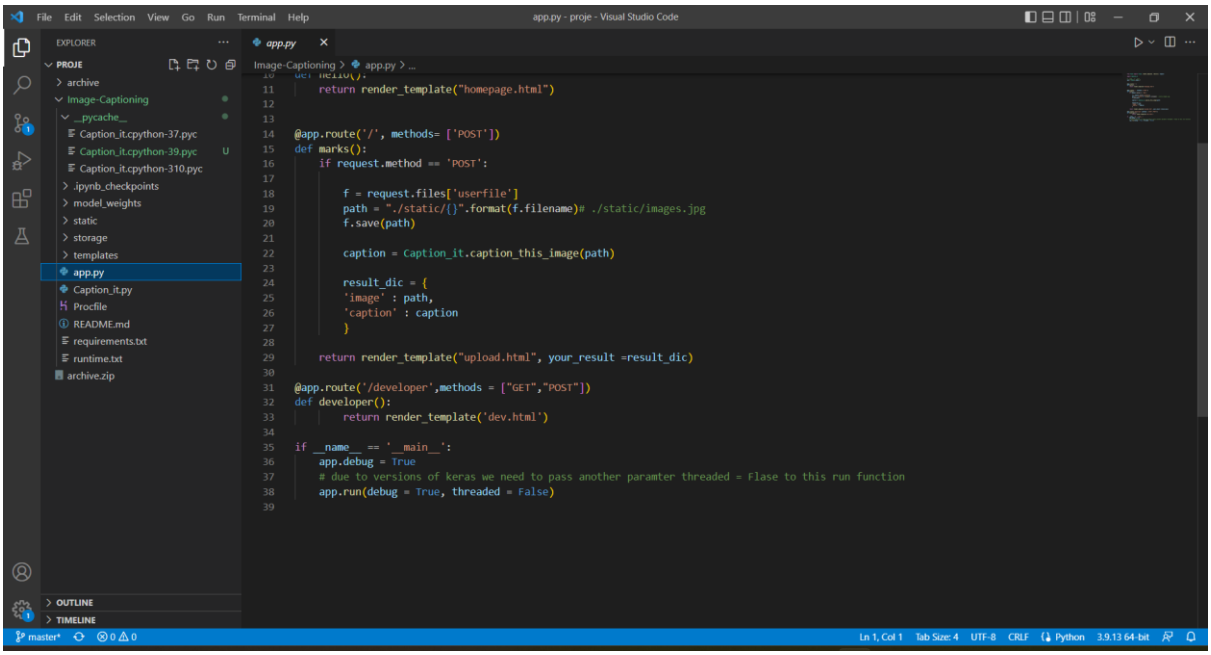
Chapter 3

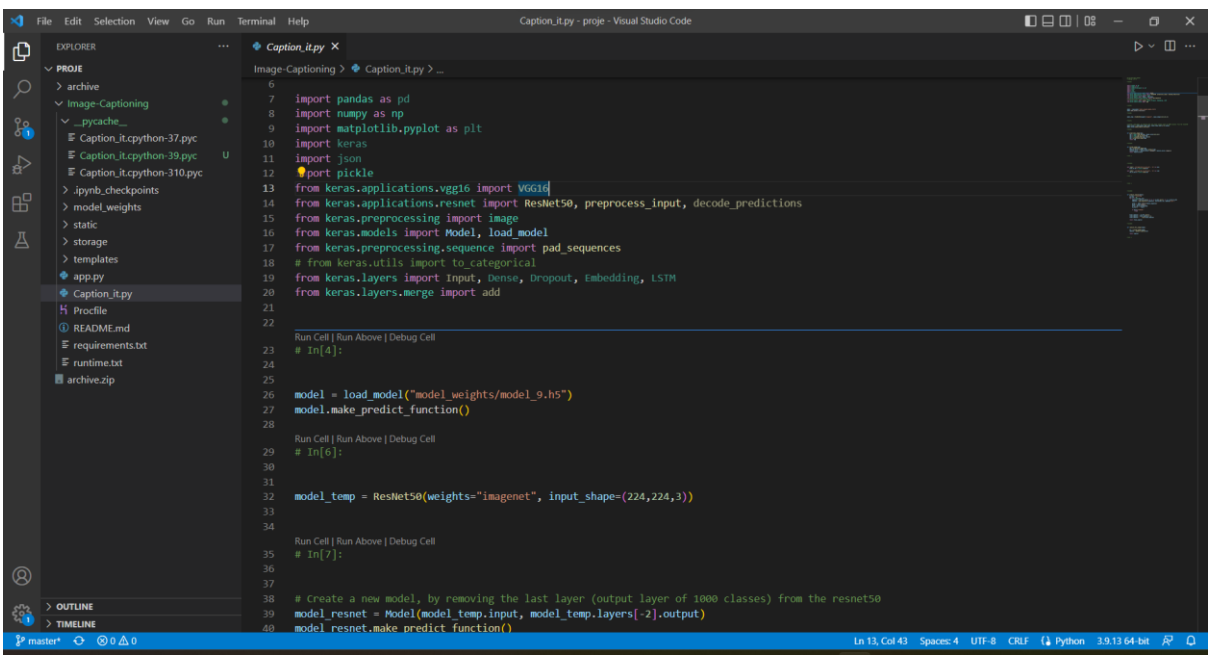
1. Project details and Outcomes

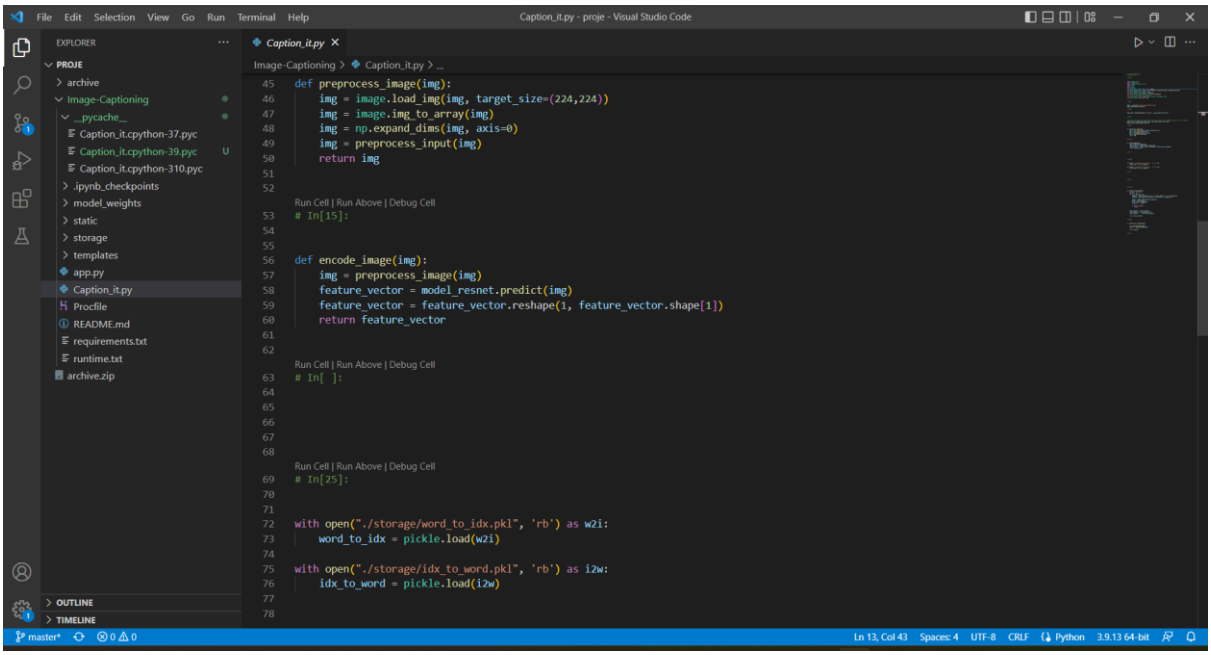
1.1 SOURCE CODE

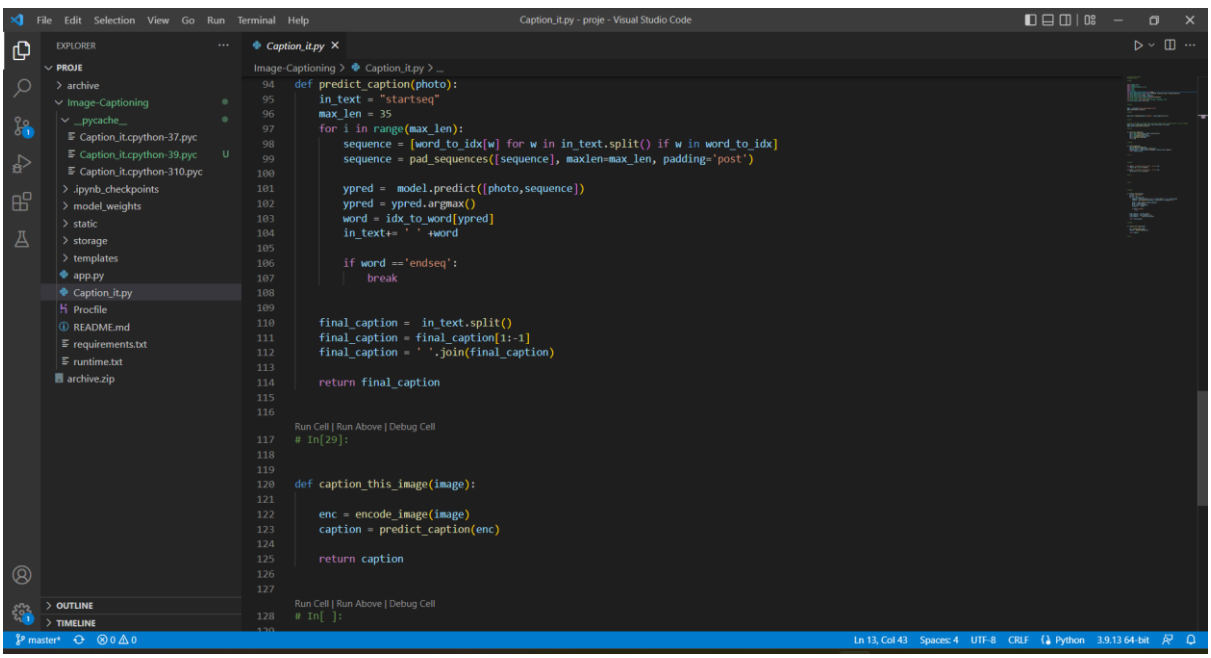
FLASK FOR DEPLOYMENT




app.py - proje - Visual Studio Code
Image-Captioning > app.py > ...
return render_template("homepage.html")
@app.route('/', methods= ['POST'])
def marks():
if request.method == 'POST':
f = request.files['userfile']
path = "./static/{}".format(f.filename)# ./static/images.jpg
f.save(path)
caption = Caption_it.caption_this_image(path)
result_dic = {
'image' : path,
'caption' : caption
}
return render_template("upload.html", your_result =result_dic)
@app.route('/developer',methods = ["GET","POST"])
def developer():
return render_template('dev.html')
if __name__ == '__main__':
app.debug = True
# due to versions of keras we need to pass another paramter threaded = Flase to this run function
app.run(debug = True, threaded = False)


Caption_it.py - proje - Visual Studio Code
Image-Captioning > Caption_it.py > ...
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import keras
import json
import pickle
from keras.applications.vgg16 import VGG16
from keras.applications.resnet import ResNet50, preprocess_input, decode_predictions
from keras.preprocessing import image
from keras.models import Model, load_model
from keras.preprocessing.sequence import pad_sequences
# from keras.utils import to_categorical
from keras.layers import Input, Dense, Dropout, Embedding, LSTM
from keras.layers.merge import add
Run Cell | Run Above | Debug Cell
# In[4]:
model = load_model("model_weights/model_9.h5")
model.make_predict_function()
Run Cell | Run Above | Debug Cell
# In[6]:
model_temp = ResNet50(weights="imagenet", input_shape=(224,224,3))
Run Cell | Run Above | Debug Cell
# In[7]:
# Create a new model, by removing the last layer (output layer of 1000 classes) from the resnet50
model_resnet = Model(model_temp.input, model_temp.layers[-2].output)
model_resnet.make_predict_function()


File Edit Selection View Go Run Terminal Help
Caption_it.py - proje - Visual Studio Code
EXPLORER
PROJE
archive
Image-Captioning
__pycache__
Caption_it.cpython-37.pyc
Caption_it.cpython-39.pyc
Caption_it.cpython-310.pyc
.ipynb_checkpoints
model_weights
static
storage
templates
app.py
Caption_it.py
Procfile
README.md
requirements.txt
runtime.txt
archive.zip
OUTLINE
TIMELINE
Caption_it.py
Image-Captioning > Caption_it.py > ...
def preprocess_image(img):
img = image.load_img(img, target_size=(224,224))
img = image.img_to_array(img)
img = np.expand_dims(img, axis=0)
img = preprocess_input(img)
return img
Run Cell | Run Above | Debug Cell
# In[15]:
def encode_image(img):
img = preprocess_image(img)
feature_vector = model_resnet.predict(img)
feature_vector = feature_vector.reshape(1, feature_vector.shape[1])
return feature_vector
Run Cell | Run Above | Debug Cell
In[ ]:
Run Cell | Run Above | Debug Cell
In[25]:
with open("./storage/word_to_idx.pkl", 'rb') as w2i:
word_to_idx = pickle.load(w2i)
with open("./storage/idx_to_word.pkl", 'rb') as i2w:
idx_to_word = pickle.load(i2w)
master*
Ln 13, Col 43
Spaces: 4
UTF-8
CRLF
Python
3.9.13 64-bit


File Edit Selection View Go Run Terminal Help
Caption_it.py - proje - Visual Studio Code
EXPLORER
PROJE
archive
Image-Captioning
__pycache__
Caption_it.cpython-37.pyc
Caption_it.cpython-39.pyc
Caption_it.cpython-310.pyc
.ipynb_checkpoints
model_weights
static
storage
templates
app.py
Caption_it.py
Procfile
README.md
requirements.txt
runtime.txt
archive.zip
OUTLINE
TIMELINE
Caption_it.py
Image-Captioning > Caption_it.py > ...
def predict_caption(photo):
in_text = "startseq"
max_len = 35
for i in range(max_len):
sequence = [word_to_idx[w] for w in in_text.split() if w in word_to_idx]
sequence = pad_sequences([sequence], maxlen=max_len, padding='post')
ypred = model.predict([photo,sequence])
ypred = ypred.argmax()
word = idx_to_word[ypred]
in_text+= ' ' +word
if word =='endseq':
break
final_caption = in_text.split()
final_caption = final_caption[1:-1]
final_caption = ' '.join(final_caption)
return final_caption
Run Cell | Run Above | Debug Cell
# In[29]:
def caption_this_image(image):
enc = encode_image(image)
caption = predict_caption(enc)
return caption
Run Cell | Run Above | Debug Cell
In[ ]:
master*
Ln 13, Col 43
Spaces: 4
UTF-8
CRLF
Python
3.9.13 64-bit

1.2 WEBSITE

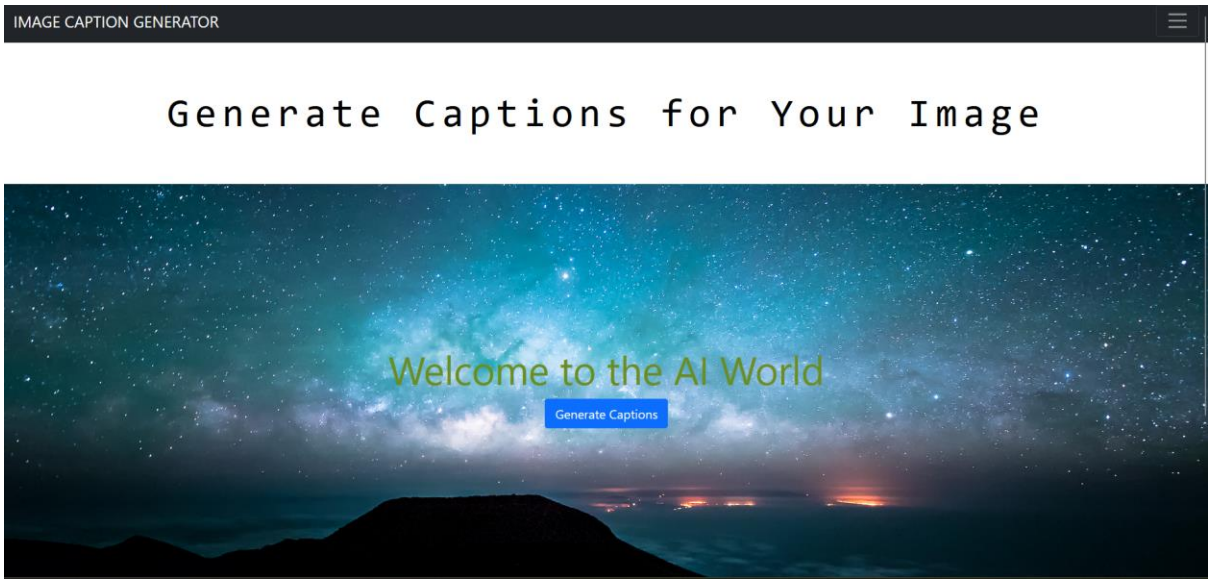
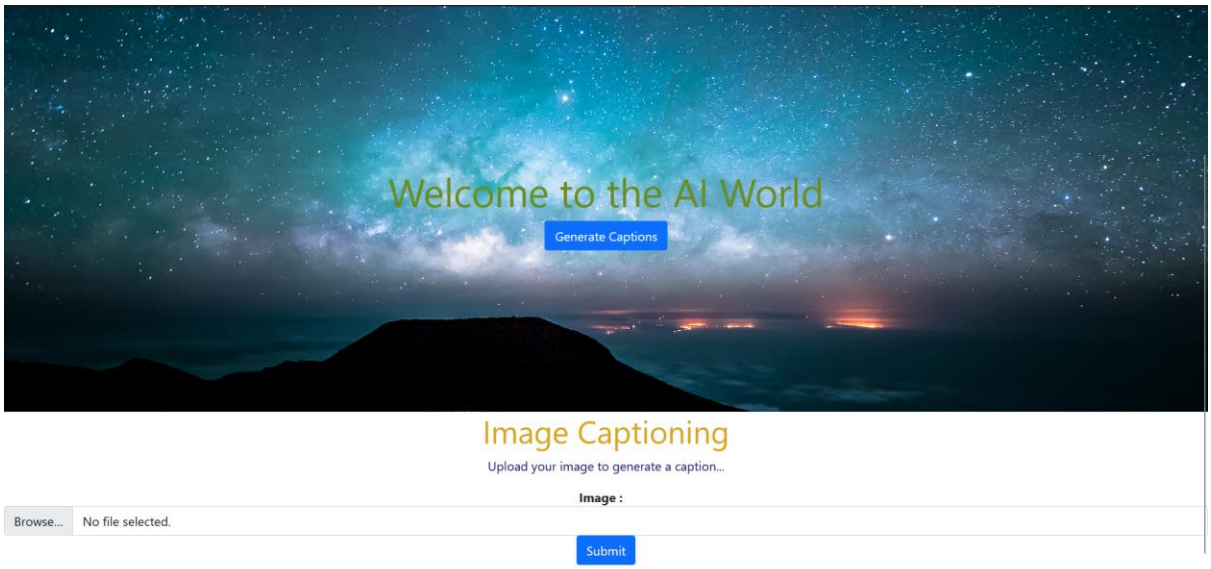


IMAGE CAPTION GENERATOR
Generate Captions for Your Image
Welcome to the AI World
Generate Captions


Welcome to the AI World
Generate Captions
Image Captioning
Upload your image to generate a caption...
Image :
Browse...
No file selected.
Submit

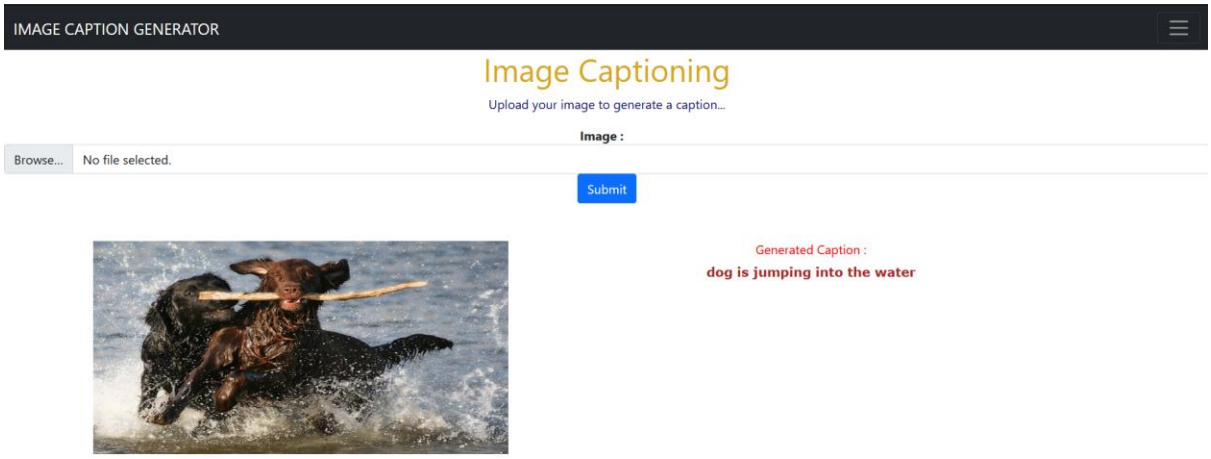


IMAGE CAPTION GENERATOR
Image Captioning
Upload your image to generate a caption...
Image :
Browse...
No file selected.
Submit
Generated Caption :
dog is jumping into the water

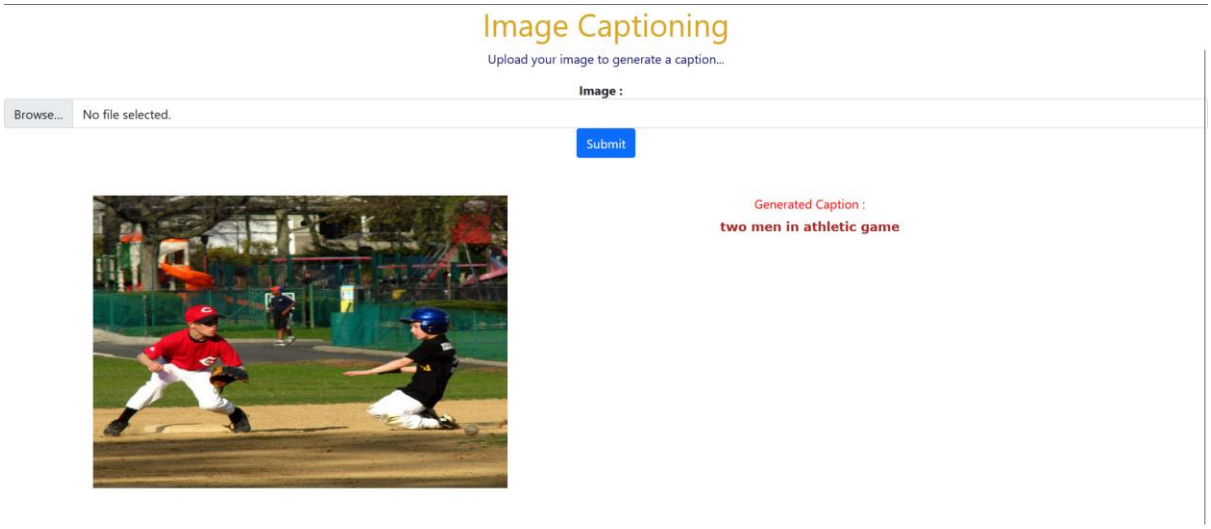


Image Captioning
Upload your image to generate a caption...
Image :
Browse...
No file selected.
Submit
Generated Caption :
two men in athletic game

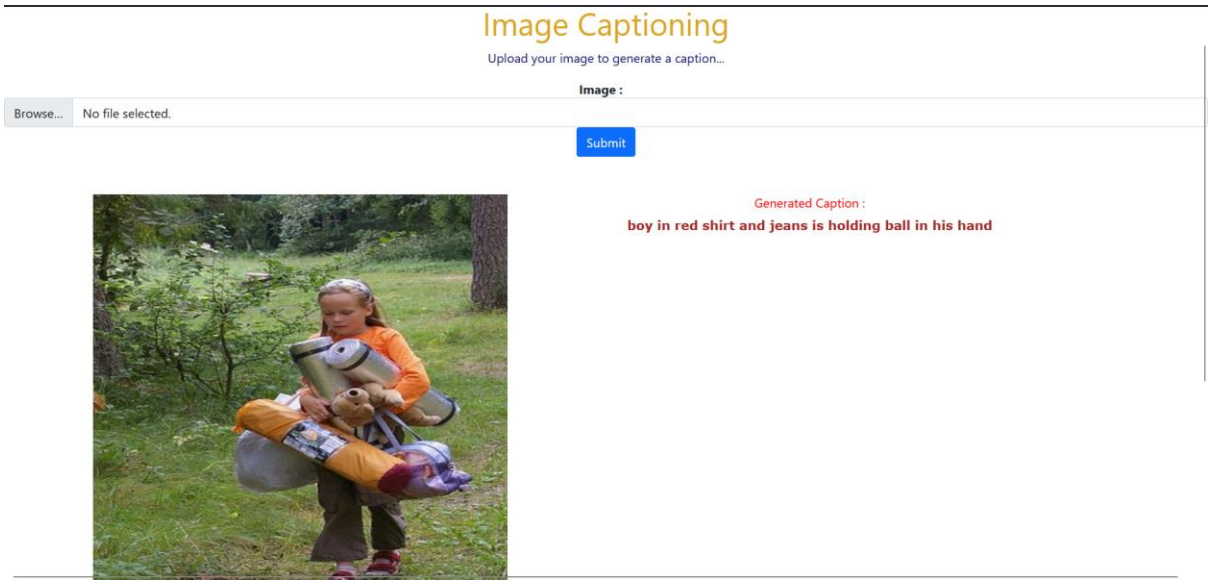


Image Captioning
Upload your image to generate a caption...
Image :
Browse...
No file selected.
Submit
Generated Caption :
boy in red shirt and jeans is holding ball in his hand

Chapter 4

Conclusion:

The project “Caption an Image” has been developed as per the requirement specification. It has been developed in Machine Learning with the use of libraries like TensorFlow, Pandas etc. and technologies such as LSTM, Regression etc. The complete system is thoroughly tested with the availability of data and throughput reports which areprepared manually.

These are found to be more accurate because of availability of information from various levels. This design is so flexible that any new modules can be incorporated easily.

This model and feature is thus ready to be implemented on the website of the company and gives an astonishing accuracy.

Chapter 5

Future Scope:

1) **Freelancing :** We can take it up as a profession as it is increasingly becoming popular as a career option. It is increasingly becoming famous as it earns good returns.

2) **Visually Impaired :** This project can be used for the visually impaired people as it can generate captions for them by clicking or understanding the surrounding images and help them know their surrounding better.

3) **Social Media :** Using this model in Social Media, we can help it in various different social media sites and can be used to implement new features such as new filters, also if a tourist wants to know new things they can just scan the item and will get the generated caption and research on that a lot better. It can be used in a daily life event very easily and effectively.

4) **NLP Applications :** Furthermore, this model can also be used in various applications of Natural Learning Processing, like in the case of Digital Image Processing, it can be used to research and train the big models.

Bibliography:

https://www.w3schools.com/

https://www.heroku.com/

https://stackoverflow.com/

https://pandas.pydata.org/docs/

https://www.tensorflow.org/api_docs