BUILD A FLASK APPLICATION

TEAM ID:PNT2022TMID41505

Step 1: Load the required packages

app.py 1 | camera.py 2 | asl_model.zip

ProjectFiles > Flask > camera.py > ...

```
1  import cv2
2  import numpy as np
3  from tensorflow.keras.models import load_model
4  from tensorflow.keras.preprocessing import image
5
```

Step 2: Initialize graph, load the model, initialize the flask app and load the video graph element is required to work with tensorflow. So, graph element is created explicitly.

app.py 1 | camera.py 2 | Model_Testing.ipynb

ProjectFiles > Training_File > Model_Testing.ipynb > Testing the model

+ Code | + Markdown | Run All | Clear Outputs of All Cells | Restart | Variables | Outline | ... | base (Python 3.9.13)

[18] Python
```
model=load_model('asl_model_84_54.h5')
img=image.load_img(r'D:\Sonal College\7th sem files\IBM\IBM Submissions\Dataset\test_set\D\2.png',target_size=(
```

[19] Python
```
img
```

[20] Python
```
x=image.img_to_array(img)
```

[21] Python
```
x.ndim
```

[22] Python
```
x=np.expand_dims(x,axis=0)
```

[23] Python
```
x.ndim
```

```
4
```

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

```
1/1 [==============================] - 0s 88ms/step
```

[25] Python
```
pred
```

```
array([3], dtype=int64)
```

```
index=['A','B','C','D','E','F','G','H','I']
print(index[pred[0]])
```

Step 3: Configure the home page

app.py 1 | camera.py 2 | Model_Testing.ipynb

ProjectFiles > Flask > app.py > ...

```python
from flask import Flask, Response, render_template
from camera import Video

app = Flask(__name__)
@app.route('/')
def index():
    return render_template('index.html')

def gen(camera):
    while True:
        frame = camera.get_frame()
        yield(b'--frame\r\n'
            b'Content-Type: image/jpeg\r\n\r\n' + frame +
            b'\r\n\r\n')

@app.route('/video_feed')
def video_feed():
    video = Video()
    return Response(gen(video), mimetype='multipart/x-mixed-replace; boundary = frame')


if __name__ == '__main__':
    app.run()
```