Wondershare
PDFelement

FINAL CODE

TEAM ID	PNT2022TMID52124
PROJECT NAME	Digital Naturalist - AI Enabled Tool For Biodiversity Researchers

PYTHON CODE (app.py):

Wondershare
PDFelement

```
from
__future__
import
division,
print_func
tion
              import os
              import numpy as np
              import tensorflow as tf
              from flask import Flask, redirect,
              render_template, request
              from keras.applications.inception_v3
              import preprocess_input
              from keras.models import model_from_json
              from werkzeug.utils import secure_filename
```

Wondershare
PDFelement

```
global graph
graph=tf.compat.v1.get_default_graph()
#this list is used to log the predictions
in the server console
predictions = ["Corpse Flower",
               "Great Indian Bustard",
               "Lady's slipper orchid",
               "Pangolin",
               "Spoon Billed Sandpiper",
               "Seneca White Deer"
              ]
#this list contains the link to the
predicted species
found = [

"https://en.wikipedia.org/wiki/Amorphophal
lus_titanum",

"https://en.wikipedia.org/wiki/Great_India
n_bustard",

"https://en.wikipedia.org/wiki/Cypripedioi
deae",

"https://en.wikipedia.org/wiki/Pangolin",
```

Wondershare
PDFelement

Wondershare
PDFelement

```
"https://en.wikipedia.org/wiki/Spoon-
billed_sandpiper",

"https://en.wikipedia.org/wiki/Seneca_whit
e_deer",
        ]
app =
Flask(__name__,template_folder="Templates"
)
@app.route('/index')
def pop():
    return render_template('index.html')
@app.route('/',methods=['GET','POST'])
def index():
    # Home Page
    return render_template("welcome.html")
@app.route('/predict', methods=['GET',
'POST'])
def upload():
    if request.method == 'GET':
        return ("<h6 style=\"font-
face:\"Courier New\";\">No GET request
herd.....</h6 >")
    if request.method == 'POST':
        # Fetching the uploaded image from
the post request using the id
```

Wondershare
PDFelement

Wondershare
PDFelement

```
'uploadedimg'
        f = request.files['uploadedimg']
        basepath =
os.path.dirname(__file__)
        #Securing the file by creating a
path in local storage
        file_path = os.path.join(basepath,
'uploads', secure_filename(f.filename))
        #Saving the uploaded image locally
        f.save(file_path)
        #loading the locally saved image
        img =
tf.keras.utils.load_img(file_path,
target_size=(224, 224))
        #converting the loaded image to
image array
        x =
tf.keras.utils.img_to_array(img)
        x = preprocess_input(x)
        # Converting the prepreocessed
image to numpy array
        inp = np.array([x])
        with graph.as_default():
            #loading the saved model from
training
            json_file =
```

Wondershare
PDFelement

Wondershare PDFelement

```
open('DigitalNaturalist.json', 'r')
            loaded_model_json =
json_file.read()
            json_file.close()
            loaded_model =
model_from_json(loaded_model_json)
            #adding weights to the trained
model

loaded_model.load_weights("DigitalNaturali
st.h5")
            #predecting the image
            preds =
np.argmax(loaded_model.predict(inp),axis=1
)
            #logs are printed to the
console
            print("Predicted the Species "
+ str(predictions[preds[0]]))
        text = found[preds[0]]
        return redirect(text)
if __name__ == '__main__':
    #Threads enabled so multiple users can
request simultaneously
    #debug is turned off, turn on during
development to debug the errors
```

Wondershare PDFelement

```
    #application is binded to port 8000
    app.run(threaded =
True,debug=True,port="8000")
```

Wondershare PDFelement