APP.PY

|  |
| --- |
| import flask |
|  | from cloudant.client import Cloudant |
|  |  |
|  | import cv2 |
|  |  |
|  | client = Cloudant.iam("58eb6520-c4d6-4d65-8e8e-3d671b1977b7-bluemix","yvkDniBmXeMn4iQ7HWKNaMgku7u-yb3I-EWAqxpxXKjs",connect=True) |
|  | my\_database = client.create\_database("database-aarthi") |
|  |  |
|  |  |
|  | app = flask.Flask(\_\_name\_\_) |
|  | app.config.from\_object(\_\_name\_\_) |
|  | app.config['SECRET\_KEY'] = '7d441f27d441f27567d441f2b6176a' |
|  |  |
|  |  |
|  |  |
|  | @app.route("/") |
|  | def homepage(): |
|  |  |
|  | return flask.render\_template('index.html') |
|  |  |
|  |  |
|  |  |
|  | @app.route("/userhome") |
|  | def userhome(): |
|  |  |
|  | return flask.render\_template('userhome.html') |
|  | @app.route("/addamount") |
|  |  |
|  | @app.route("/NewUser") |
|  | def NewUser(): |
|  |  |
|  | return flask.render\_template('NewUser.html') |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | @app.route("/user") |
|  | def user(): |
|  |  |
|  | return flask.render\_template('user.html') |
|  |  |
|  |  |
|  | @app.route("/newuse",methods=['GET','POST']) |
|  | def newuse(): |
|  | if flask.request.method == 'POST':# |
|  |  |
|  | x = [x for x in flask.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 flask.render\_template('goback.html', data="Register, please login using your details") |
|  | else: |
|  | return flask.render\_template('goback.html', data="You are already a member, please login using your details") |
|  |  |
|  | @app.route("/userlog", methods=['GET', 'POST']) |
|  | def userlog(): |
|  | if flask.request.method == 'POST': |
|  |  |
|  | user = flask.request.form['\_id'] |
|  | passw = flask.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 flask.render\_template('goback.html', pred="The username is not found.") |
|  | else: |
|  | if ((user == docs[0][0]['\_id'] and passw == docs[0][0]['psw'])): |
|  |  |
|  | return flask.render\_template("userhome.html") |
|  | else: |
|  | return flask.render\_template('goback.html', data="user name and password incorrect") |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | @app.route("/predict", methods=['GET', 'POST']) |
|  | def predict(): |
|  | if flask.request.method == 'POST': |
|  |  |
|  |  |
|  | file = flask.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 = flask.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 flask.render\_template('userhome.html', prediction=value) |
|  |  |
|  |  |
|  |  |
|  | if \_\_name\_\_ == '\_\_main\_\_': |
|  | app.run(debug=True, use\_reloader=True) |
| import flask |
|  | from cloudant.client import Cloudant |
|  |  |
|  | import cv2 |
|  |  |
|  | client = Cloudant.iam("58eb6520-c4d6-4d65-8e8e-3d671b1977b7-bluemix","yvkDniBmXeMn4iQ7HWKNaMgku7u-yb3I-EWAqxpxXKjs",connect=True) |
|  | my\_database = client.create\_database("database-aarthi") |
|  |  |
|  |  |
|  | app = flask.Flask(\_\_name\_\_) |
|  | app.config.from\_object(\_\_name\_\_) |
|  | app.config['SECRET\_KEY'] = '7d441f27d441f27567d441f2b6176a' |
|  |  |
|  |  |
|  |  |
|  | @app.route("/") |
|  | def homepage(): |
|  |  |
|  | return flask.render\_template('index.html') |
|  |  |
|  |  |
|  |  |
|  | @app.route("/userhome") |
|  | def userhome(): |
|  |  |
|  | return flask.render\_template('userhome.html') |
|  | @app.route("/addamount") |
|  |  |
|  | @app.route("/NewUser") |
|  | def NewUser(): |
|  |  |
|  | return flask.render\_template('NewUser.html') |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | @app.route("/user") |
|  | def user(): |
|  |  |
|  | return flask.render\_template('user.html') |
|  |  |
|  |  |
|  | @app.route("/newuse",methods=['GET','POST']) |
|  | def newuse(): |
|  | if flask.request.method == 'POST':# |
|  |  |
|  | x = [x for x in flask.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 flask.render\_template('goback.html', data="Register, please login using your details") |
|  | else: |
|  | return flask.render\_template('goback.html', data="You are already a member, please login using your details") |
|  |  |
|  | @app.route("/userlog", methods=['GET', 'POST']) |
|  | def userlog(): |
|  | if flask.request.method == 'POST': |
|  |  |
|  | user = flask.request.form['\_id'] |
|  | passw = flask.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 flask.render\_template('goback.html', pred="The username is not found.") |
|  | else: |
|  | if ((user == docs[0][0]['\_id'] and passw == docs[0][0]['psw'])): |
|  |  |
|  | return flask.render\_template("userhome.html") |
|  | else: |
|  | return flask.render\_template('goback.html', data="user name and password incorrect") |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | @app.route("/predict", methods=['GET', 'POST']) |
|  | def predict(): |
|  | if flask.request.method == 'POST': |
|  |  |
|  |  |
|  | file = flask.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 = flask.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 flask.render\_template('userhome.html', prediction=value) |
|  |  |
|  |  |
|  |  |
|  | if \_\_name\_\_ == '\_\_main\_\_': |
|  | app.run(debug=True, use\_reloader=True) |