APPLICATION BUILDING

Date	20 November 2022
Team ID	PNT2022TMID36293
Project Name	VirtualEye - LifeGuard for Swimming Pools to Detect Active Drowning
Maximum Marks	8 Marks

App.py:

```
import os

from cloudant.client import Cloudant
from flask import Flask, flash, redirect, render_template, request, url_for,

Response from werkzeug.utils import secure_filename from detect import detect

UPLOAD_FOLDER = "static/uploads/"
RESULTS_FOLDER = "static/results/"

app = Flask(__name__)
app.secret_key = "secret-key"
app.config["UPLOAD_FOLDER"] = UPLOAD_FOLDER

API_KEY = "I5qBRvqrkDNcwtcPSgqB6bPpg-Mfppv596Iuxy86j2Sc"

USERNAME = "26eb4b40-0ca7-4edd-90be-0c2318c3a564-bluemix"

databaseName = "virtual_eye" client = Cloudant.iam(USERNAME,

API_KEY, connect=True)


@app.route("/") def index(): return
render_template("index.html", static_folder="static")


@app.route("/register", methods=["GET", "POST"])
def register():
   if request.method == "POST":
      # Get the form data
      try:
         email = request.form["email"] password =
         request.form["password"] # Create a database using an
         initialized client my_database =
         client.create_database(databaseName) # Check that
         the database doesn't already exist if
         my_database.exists():
            print(f"'{databaseName}' successfully created.")
         # Create a JSON document
         json_document = {
            "_id": email,
            "email": email,
            "password": password,
            }
            if email in my_database: return render_template("register.html",
                msg="Email already exists")
            else:
                # Create a document using the Database API
                new_document = my_database.create_document(json_document)

                return render_template(
                    "register.html", msg="Account created successfully!"
                )
        except Exception as e:
            return render_template(
                "register.html", msg="Something went wrong! Please try again"
            )
    if request.method == "GET": return
        render_template("register.html")


@app.route("/login", methods=["GET", "POST"])
def login():
    if request.method == "POST": email = request.form["email"] password =
        request.form["password"] my_database = client[databaseName] # Check
        that the database exists if email in my_database and
        my_database[email]["password"] == password:
            return redirect(url_for("predict"))
        else: return render_template("login.html", msg="Invalid
            credentials!")
    if request.method == "GET": return
        render_template("login.html")


@app.route("/predict", methods=["GET", "POST"])
def predict():
    if request.method == "POST":
        if "file" not in request.files:
        flash("No file part") return
        redirect(request.url)
        file = request.files["file"]
        if file.filename == "":
            flash("No video selected for uploading")
            return redirect(request.url)
        else: filename =
            secure_filename(file.filename)
            file.save(os.path.join(app.config["UPLOAD_FOLDER"],
            filename)) return render_template( "predict.html",
                msg="Video uploaded successfully",
                filename=filename,
            )
    if request.method == "GET": return
        render_template("predict.html")
```

```
@app.route("/response/<string:filename>", methods=["GET", "POST"])
def response(filename): print(filename) return Response(
        detect(
            os.path.join(app.config["UPLOAD_FOLDER"], filename),
        ),
        mimetype="multipart/x-mixed-replace; boundary=frame",
    )


@app.route("/logout", methods=["GET"])
def logout(): return
render_template("logout.html")


if __name__ == "__main__":
    app.run(debug=True)
```