Python Code:

import os

from datetime import datetime

from flask import Flask, flash, request, redirect, url\_for, render\_template, send\_from\_directory

from flask\_sqlalchemy import SQLAlchemy

from flask\_migrate import Migrate

from flask\_login import login\_required, login\_user, LoginManager, logout\_user, UserMixin, current\_user

from werkzeug.security import check\_password\_hash, generate\_password\_hash

from werkzeug.utils import secure\_filename

UPLOAD\_FOLDER = '/home/HassanSherien/mysite/Shape\_Image'

ALLOWED\_EXTENSIONS = {'png', 'jpg', 'jpeg'}

#app.config['MAX\_CONTENT\_LENGTH'] = 16 \* 1024 \* 1024

app = Flask(\_\_name\_\_)

app.config['UPLOAD\_FOLDER'] = UPLOAD\_FOLDER

#app.config["DEBUG"] = True

SQLALCHEMY\_DATABASE\_URI = "mysql+mysqlconnector://{username}:{password}@{hostname}/{databasename}".format(

username="HassanSherien",

password="Ilovepython1234",

hostname="HassanSherien.mysql.pythonanywhere-services.com",

databasename="HassanSherien$comments",

)

app.config["SQLALCHEMY\_DATABASE\_URI"] = SQLALCHEMY\_DATABASE\_URI

app.config["SQLALCHEMY\_POOL\_RECYCLE"] = 299

app.config["SQLALCHEMY\_TRACK\_MODIFICATIONS"] = False

db = SQLAlchemy(app)

migrate = Migrate(app, db)

app.secret\_key = "How much wood would a woodchuck chuck if a woodchuck could chuck wood?"

login\_manager = LoginManager()

login\_manager.init\_app(app)

class User(UserMixin, db.Model):

\_\_tablename\_\_ = "users"

id = db.Column(db.Integer, primary\_key=True)

username = db.Column(db.String(128))

password\_hash = db.Column(db.String(128))

def check\_password(self, password):

return check\_password\_hash(self.password\_hash, password)

def get\_id(self):

return self.username

@login\_manager.user\_loader

def load\_user(user\_id):

return User.query.filter\_by(username=user\_id).first()

class Comment(db.Model):

\_\_tablename\_\_ = "comments"

id = db.Column(db.Integer, primary\_key=True)

content = db.Column(db.String(4096))

posted = db.Column(db.DateTime, default=datetime.now)

commenter\_id = db.Column(db.Integer, db.ForeignKey('users.id'), nullable=True)

commenter = db.relationship('User', foreign\_keys=commenter\_id)

shape\_name= db.Column(db.String(128), db.ForeignKey('shape.shape\_name'))

class Shape(db.Model):

\_\_tablename\_\_ = "shape"

shape\_name = db.Column((db.Integer), primary\_key=True)

shape\_vertices = db.Column(db.Float, nullable=False)

shape\_file\_name = db.Column(db.String(4096))

shape\_feedback =db.Column(db.String(4096))

#what is commented below is how to classify integers of shape\_name primary key as a string which are the shape\_names

#\_\_tablename\_\_ = "shape"

#shape\_name = db.Column((db.Integer), primary\_key=True)

#shape\_size = db.Column(db.Float, nullable=False)

#shape\_colorname = db.Column(db.String(4096))

#shape\_color = db.Column(db.Float, nullable=False)

#class Shape(db.Model):

#def \_\_init\_\_(self):

#if shape\_name == 1:

#print("Circle")

#elif shape\_name == 2:

#print("Square")

#elif shape\_name == 3:

#print("Triangle")

#else:

#print("Other Undefined Shape")

#THIS SECTION IS where THE UPLOADING PHOTOS SECTIONS STARTS

def allowed\_file(filename):

return '.' in filename and \

filename.rsplit('.', 1)[1].lower() in ALLOWED\_EXTENSIONS

@app.route('/', methods=['GET', 'POST'])

def upload\_file():

if request.method == 'POST':

# check if the post request has the file part

if 'file' not in request.files:

flash('No file part')

return redirect(request.url)

file = request.files['file']

# if user does not select file, browser also

# submit a empty part without filename

if file.filename == '':

flash('No selected file')

return redirect(request.url)

if file and allowed\_file(file.filename):

filename = secure\_filename(file.filename)

file.save(os.path.join(app.config['UPLOAD\_FOLDER'], filename))

return redirect(url\_for('uploaded\_file',filename=filename))

elif request.method == "GET":

return render\_template("overview\_main\_page.html", comments=Comment.query.all())

return

@app.route('/uploads/<filename>') #, #methods=["GET", "POST"])

def uploaded\_file(filename):

return send\_from\_directory(app.config['UPLOAD\_FOLDER'],

filename)

#THIS IS WHERE THE UPLOADING PHOTOS SECTION ENDS

@app.route("/", methods=["GET", "POST"])

def index():

if request.method == "GET":

return render\_template("overview\_main\_page.html", comments=Comment.query.all())

comment = Comment(content=request.form["contents"], commenter=current\_user)

if not current\_user.is\_authenticated:

return redirect(url\_for('index'))

db.session.add(comment)

db.session.commit()

return redirect(url\_for('index'))

#BELOW LETS ANY USER SIGN UP-START; needs to be fixed (url does not show)

app.route("/signup/", methods=["GET", "POST"])

def signup\_page():

if request.method =="GET":

return render\_template("signup\_page.html", error=False)

#user = User(password\_hash = generate\_password\_hash(request.form["password"]), username=request.form["username"] )

#db.session.add(user)

#db.session.commit()

#return redirect(url\_for('login'))

#THIS IS WHERE SIGN UP ENDS

@app.route("/login/", methods=["GET", "POST"])

def login():

if request.method == "GET":

return render\_template("login\_page.html", error=False)

user = load\_user(request.form["username"])

if user is None:

return render\_template("login\_page.html", error=True)

if not user.check\_password(request.form["password"]):

return render\_template("login\_page.html", error=True)

login\_user(user)

return redirect(url\_for('index'))

@app.route("/logout/")

@login\_required

def logout():

logout\_user()

return redirect(url\_for('index'))