Assignment Number	**3**
Assignment Date	**22 september 2022**
Student Name	**MAD**HESH.V
Student Roll Number	510919106301
Maximum marks	**2 MARKS**

QUESTIONS:

1. Flask-api-main
2. Flask-blog-with-db-main
3. Flask-with-ibm-cloud-object-storage-main
4. Flask-with-ibm-db2-main

Solution:

1. Flask-api-main

```
from flask import Flask, request

app = Flask(__name__)

food_items = {  "1":"rice",
                "2":"beans",
                "3":"yam",
                "4":"plantain",
                "5":"potatoes",
                "6":"wheat"
            }

@app.route("/api")
def index():
    return  "Hello form Flask API Server"

@app.route('/data', methods = ['POST', 'GET'])
def api():
   if request.method == 'GET':
      return food_items

   if request.method =='POST':
       data = request.json
       food_items.update(data)
       return "Data is inserted"

@app.route("/data/<id>", methods=["PUT"])
def update(id):
    data = request.form['item']
    food_items[str(id)]=data
    return "Data updated"

@app.route("/data/<id>", methods=["DELETE"])
def delete(id):
    food_items.pop(str(id))
    return "Data Deleted"
```

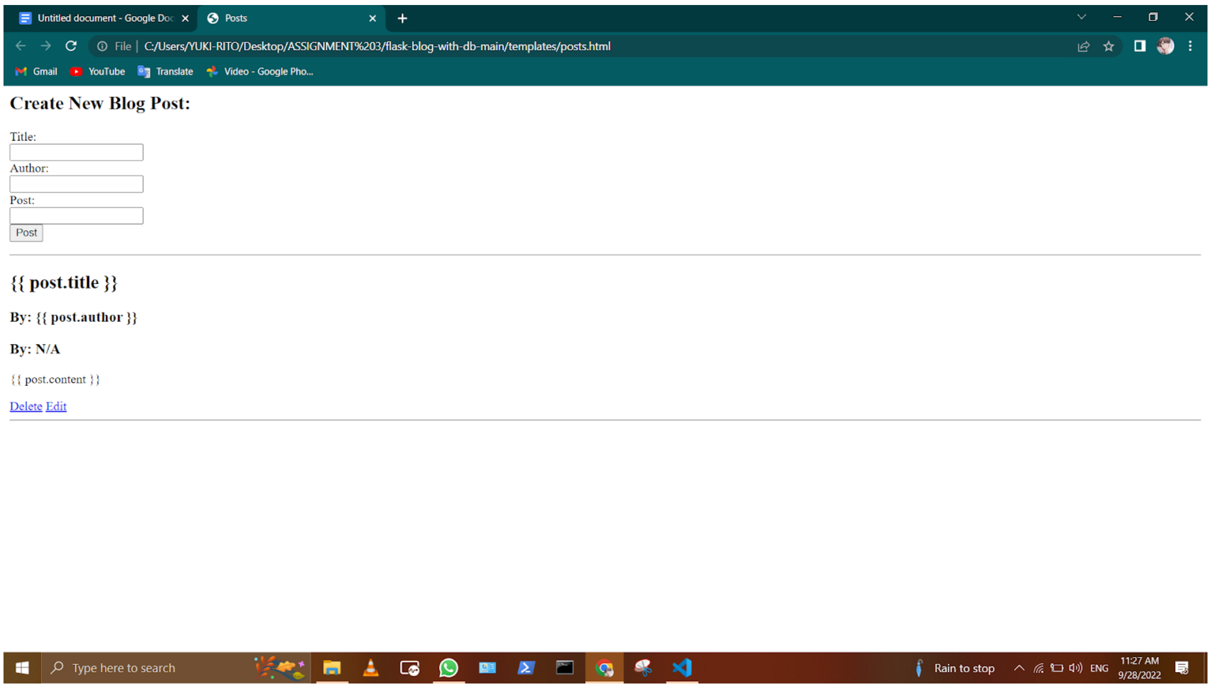
2.Flask-blog-with-db-main

- post.html

```
<title>Posts</title>


    <h2>Create New Blog Post:</h2>
    <form action='/posts' method='POST'>
        Title:  <br>
        <input type='text' name='title' id='title'
class="form-control">
        <br>
        Author: <br>
        <input type='text' name='author' id='author'
class="form-control">
        <br>
        Post:  <br>
        <input type='text' name='content' id='content'
class="form-control">
        <br>
        <input type='submit' value='Post' class="btn btn-success
col-sm-3">
    </form>
    <hr>
```

```
        <h2>{{ post.title }}</h2>


            <h3>By: {{ post.author }}</h3>

            <h3>By: N/A</h3>


        <p>{{ post.content }}</p>
        <a href='/posts/delete/{{post.id}}'>Delete</a>
        <a href='/posts/edit/{{post.id}}'>Edit</a>
        <hr>
```

IMAGES:


Untitled document - Google Doc
Posts
C:/Users/YUKI-RITO/Desktop/ASSIGNMENT%203/flask-blog-with-db-main/templates/posts.html

Create New Blog Post:

Title:

Author:

Post:

Post

{{ post.title }}

By: {{ post.author }}

By: N/A

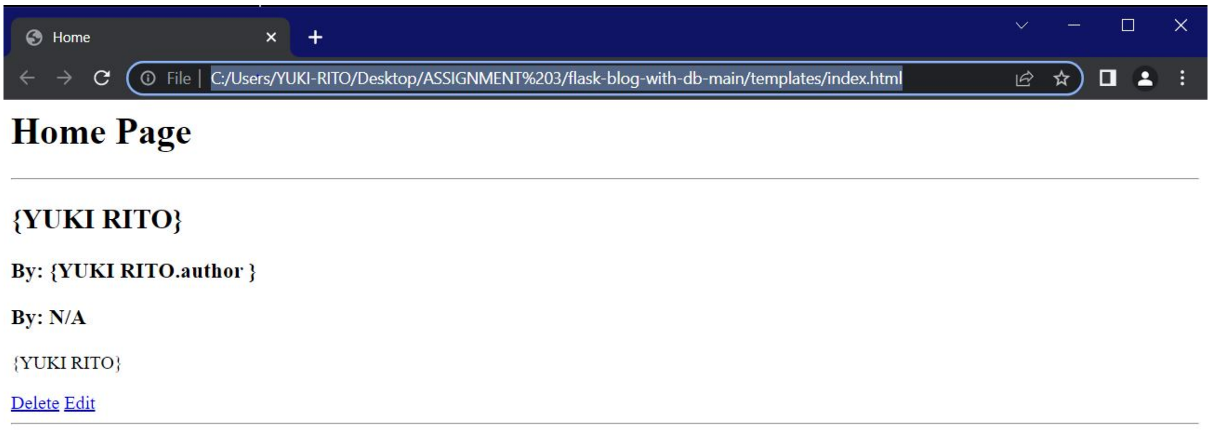
{{ post.content }}

Delete Edit

- Index.html

```
<title>Home</title>

<h1>Home Page</h1>
<hr>
        <h2>{YUKI RITO}</h2>
            <h3>By: { }</h3>
            <h3>By: N/A</h3>

        <p>{YUKI RITO}</p>
        <a href='/posts/delete/{{post.id}}'>Delete</a>
        <a href='/posts/edit/{{post.id}}'>Edit</a>
        <hr>
```

IMAGE:


Home
File | C:/Users/YUKI-RITO/Desktop/ASSIGNMENT%203/flask-blog-with-db-main/templates/index.html

Home Page

{YUKI RITO}

By: {YUKI RITO.author }

By: N/A

{YUKI RITO}

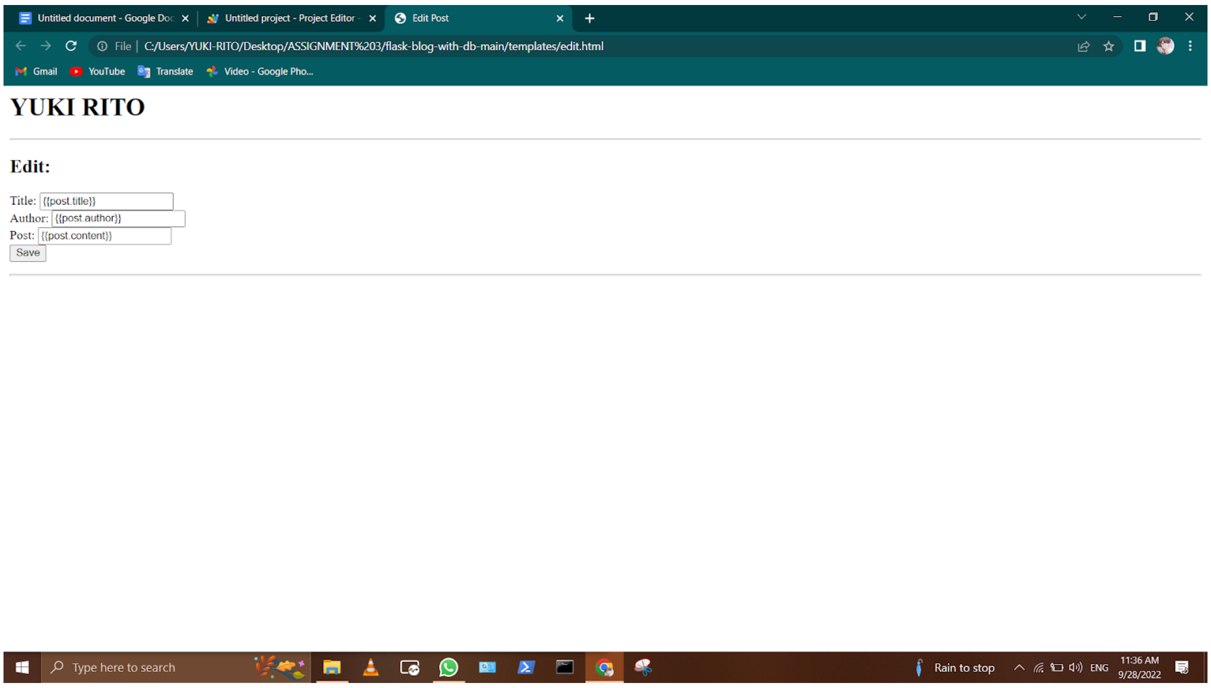
Delete Edit

- EDIT.html

```
<title>Edit Post</title>



<h1>YUKI RITO</h1>

    <hr>
    <h2>Edit:</h2>
    <form action='/posts/edit/{{post.id}}' method='POST'>
```

```
        Title: <input type='text' name='title' id='title'
value="{{post.title}}">
        <br>
        Author: <input type='text' name='author' id='author'
value="{{post.author}}">
        <br>
        Post: <input type='text' name='content' id='content'
value="{{post.content}}">
        <br>
        <input type='submit' value='Save'>
    </form>
    <hr>
```

IMAGE:



Untitled document - Google Doc | Untitled project - Project Editor | Edit Post

C:/Users/YUKI-RITO/Desktop/ASSIGNMENT%203/flask-blog-with-db-main/templates/edit.html

YUKI RITO

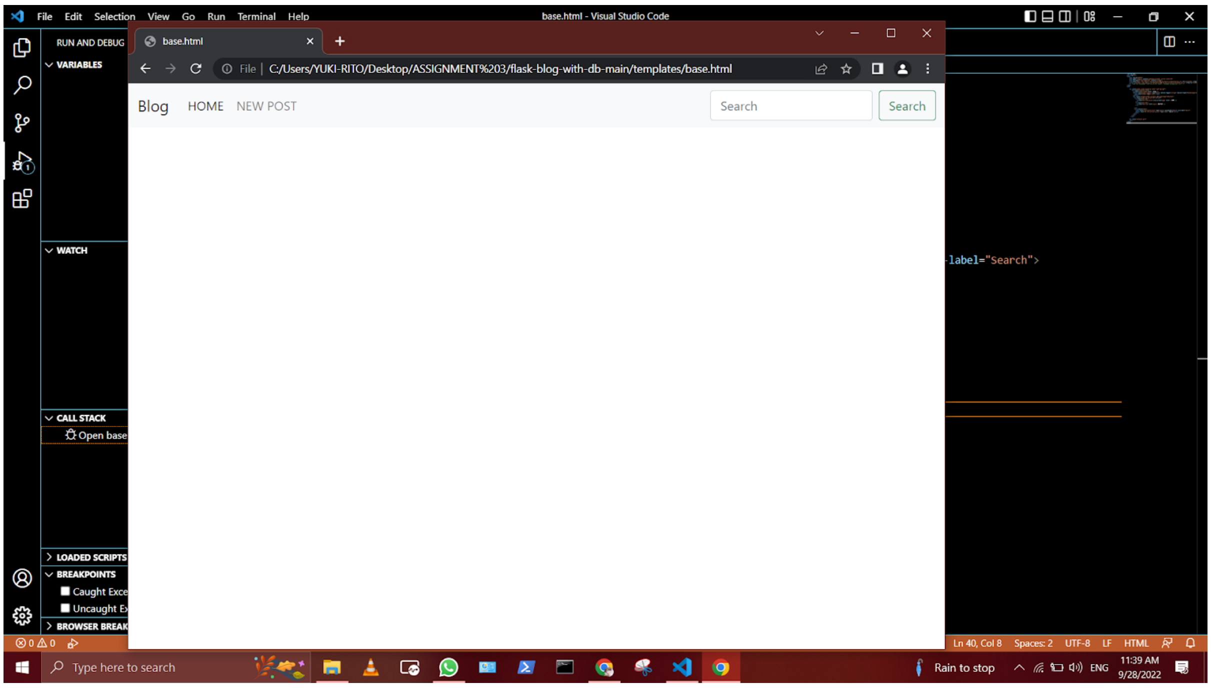
Edit:

Title: {{post.title}}
Author: {{post.author}}
Post: {{post.content}}
Save

- Base.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.m
in.css" rel="stylesheet"
integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65Vohh
puuCOmLASjC" crossorigin="anonymous">
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bun
dle.min.js"
integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcX
n/tWtIaxVXM" crossorigin="anonymous"></script>
    <!-- <link rel="stylesheet" href="{{ url_for('static',
filename='css/main.css') }}"> -->
</head>
<body>

    <nav class="navbar navbar-expand-lg navbar-light bg-light">
        <div class="container-fluid">
          <a class="navbar-brand" href="/">Blog</a>
          <button class="navbar-toggler" type="button"
data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent"
aria-controls="navbarSupportedContent" aria-expanded="false"
aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse"
id="navbarSupportedContent">
            <ul class="navbar-nav me-auto mb-2 mb-lg-0">
              <li class="nav-item">
                <a class="nav-link active" aria-current="page"
href="/">HOME</a>
              </li>
              <li class="nav-item">
                <a class="nav-link" href="/posts">NEW POST</a>
              </li>

            </ul>
            <form class="d-flex">
              <input class="form-control me-2" type="search"
placeholder="Search" aria-label="Search">
              <button class="btn btn-outline-success"
type="submit">Search</button>
            </form>
          </div>
        </div>
      </nav>

    <div class="container py-5">
    </div>
</body>
</html>
```

IMAGE:



- App.py

```
from flask import Flask, render_template, request, redirect
from flask_sqlalchemy import SQLAlchemy
from datetime import datetime

app = Flask(__name__)
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///posts.db'
db = SQLAlchemy(app)

class BlogPost(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    title = db.Column(db.String(100), nullable=False)
    content = db.Column(db.Text, nullable=False)
    author = db.Column(db.String(20), nullable=False, default='N/A')
    date_posted = db.Column(db.DateTime, nullable=False,
default=datetime.utcnow)

    def __repr__(self):
```

```python
        return 'Blog post ' + str(self.id)

@app.route('/')
def index():
    all_posts = BlogPost.query.order_by(BlogPost.date_posted).all()
    return render_template('index.html', posts=all_posts)

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

    if request.method == 'POST':
        post_title = request.form['title']
        post_content = request.form['content']
        post_author = request.form['author']
        new_post = BlogPost(title=post_title, content=post_content,
author=post_author)
        db.session.add(new_post)
        db.session.commit()
        return redirect('/posts')
    else:
        all_posts = BlogPost.query.order_by(BlogPost.date_posted).all()
        return render_template('posts.html', posts=all_posts)

@app.route('/posts/delete/<int:id>')
def delete(id):
    post = BlogPost.query.get_or_404(id)
    db.session.delete(post)
    db.session.commit()
    return redirect('/posts')

@app.route('/posts/edit/<int:id>', methods=['GET', 'POST'])
def edit(id):

    post = BlogPost.query.get_or_404(id)

    if request.method == 'POST':
        post.title = request.form['title']
        post.author = request.form['author']
        post.content = request.form['content']
        db.session.commit()
        return redirect('/posts')
    else:
        return render_template('edit.html', post=post)
```

```


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

3.Flask-with-ibm-cloud-object-storage-main

- App.py

```
from flask import Flask,redirect,url_for,render_template,request
import ibm_boto3
from ibm_botocore.client import Config, ClientError

COS_ENDPOINT="https://s3.jp-tok.cloud-object-storage.appdomain.cl
oud"
COS_API_KEY_ID=" "
COS_INSTANCE_CRN=""




# Create resource
https://s3.ap.cloud-object-storage.appdomain.cloud
cos = ibm_boto3.resource("s3",
    ibm_api_key_id=COS_API_KEY_ID,
    ibm_service_instance_id=COS_INSTANCE_CRN,
    config=Config(signature_version="oauth"),
    endpoint_url=COS_ENDPOINT
)

app=Flask(__name__)




def get_item(bucket_name, item_name):
    print("Retrieving item from bucket: {0}, key:
{1}".format(bucket_name, item_name))
    try:
        file = cos.Object(bucket_name, item_name).get()

        print("File Contents: {0}".format(file["Body"].read()))
    except ClientError as be:
        print("CLIENT ERROR: {0}\n".format(be))
    except Exception as e:
        print("Unable to retrieve file contents: {0}".format(e))
```

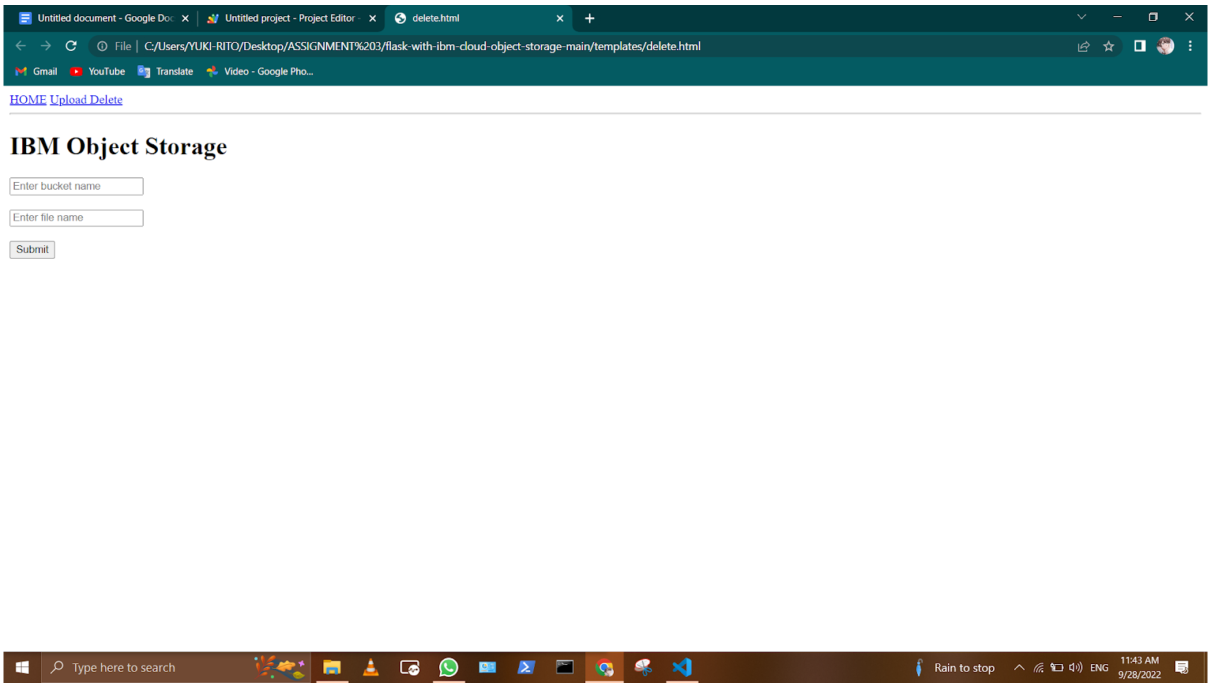
```python
def get_bucket_contents(bucket_name):
    print("Retrieving bucket contents from:
{0}".format(bucket_name))
    try:
        files = cos.Bucket(bucket_name).objects.all()
        files_names = []
        for file in files:
            files_names.append(file.key)
            print("Item: {0} ({1} bytes).".format(file.key,
file.size))
        return files_names
    except ClientError as be:
        print("CLIENT ERROR: {0}\n".format(be))
    except Exception as e:
        print("Unable to retrieve bucket contents:
{0}".format(e))


def delete_item(bucket_name, object_name):
    try:
        cos.delete_object(Bucket=bucket_name, Key=object_name)
        print("Item: {0} deleted!\n".format(object_name))
    except ClientError as be:
        print("CLIENT ERROR: {0}\n".format(be))
    except Exception as e:
        print("Unable to delete object: {0}".format(e))



def multi_part_upload(bucket_name, item_name, file_path):
    try:
        print("Starting file transfer for {0} to bucket:
{1}\n".format(item_name, bucket_name))
        # set 5 MB chunks
        part_size = 1024 * 1024 * 5

        # set threadhold to 15 MB
        file_threshold = 1024 * 1024 * 15

        # set the transfer threshold and chunk size
        transfer_config = ibm_boto3.s3.transfer.TransferConfig(
            multipart_threshold=file_threshold,
            multipart_chunksize=part_size
        )

        # the upload_fileobj method will automatically execute a
multi-part upload
        # in 5 MB chunks for all files over 15 MB
        with open(file_path, "rb") as file_data:
            cos.Object(bucket_name, item_name).upload_fileobj(
                Fileobj=file_data,
                Config=transfer_config
            )

        print("Transfer for {0} Complete!\n".format(item_name))
    except ClientError as be:
        print("CLIENT ERROR: {0}\n".format(be))
    except Exception as e:
        print("Unable to complete multi-part upload:
{0}".format(e))


@app.route('/')
def index():
    files = get_bucket_contents('flaskapp123')
    return render_template('index.html', files = files)

@app.route('/deletefile', methods = ['GET', 'POST'])
def deletefile():
   if request.method == 'POST':
       bucket=request.form['bucket']
       name_file=request.form['filename']

       delete_item(bucket,name_file)
       return 'file deleted successfully'

   if request.method == 'GET':
       return render_template('delete.html')


@app.route('/uploader', methods = ['GET', 'POST'])
def upload():
   if request.method == 'POST':
       bucket=request.form['bucket']
```

```
        name_file=request.form['filename']
        f = request.files['file']
        multi_part_upload(bucket,name_file,f.filename)
        return 'file uploaded successfully <a href="/">GO to
Home</a>'


    if request.method == 'GET':
        return render_template('upload.html')

if __name__=='__main__':
    app.run(host='0.0.0.0',port=8080,debug=True)
```

- DELETE.html

```
<html>
   <body>

    <a href="/">HOME</a>
    <a href="/uploader">Upload </a>
    <a href="/deletefile">Delete </a>
    <br><hr>

<h1>IBM Object Storage</h1>

      <form action = "/deletefile" method = "POST" >

         <input type = "text" placeholder="Enter bucket name" name =
"bucket" />
         <br>
         <br>
         <input type = "text" placeholder="Enter file name" name =
"filename" />
         <br>
         <br>
         <input type = "submit"/>
      </form>
   </body>
</html>
```

IMAGE:


C:/Users/YUKI-RITO/Desktop/ASSIGNMENT%203/flask-with-ibm-cloud-object-storage-main/templates/delete.html

HOME Upload Delete

IBM Object Storage

Enter bucket name

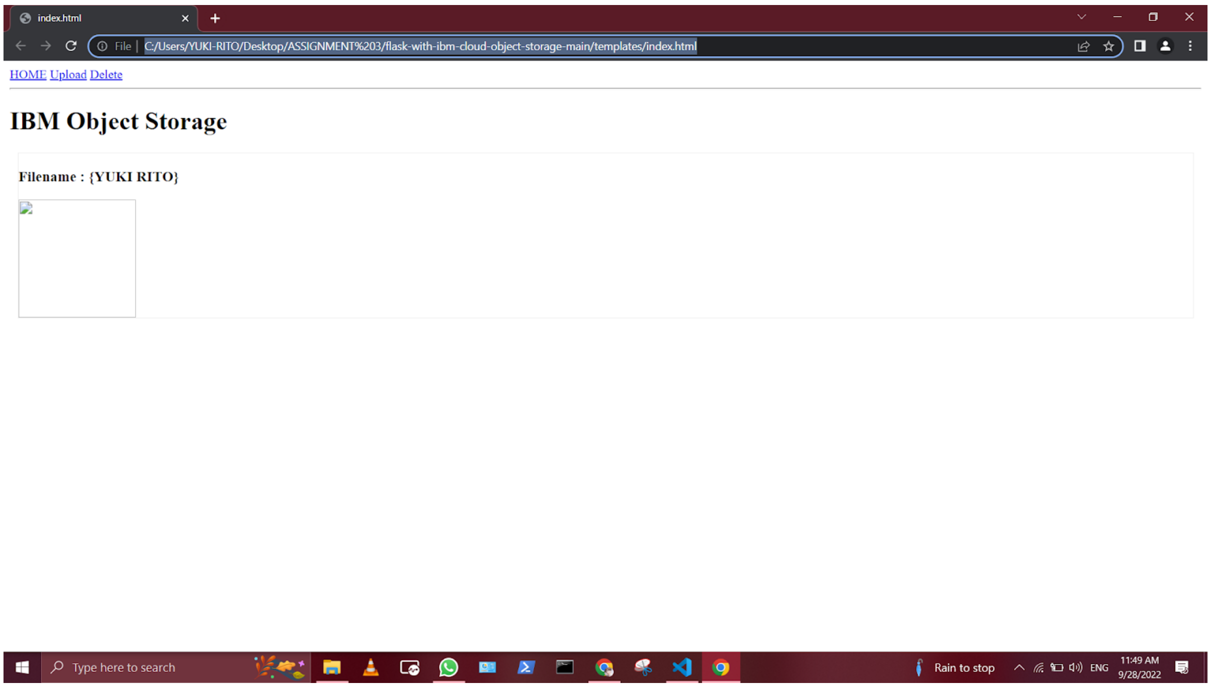
Enter file name

Submit

- INDEX.HTML

```
<a href="/">HOME</a>
<a href="/uploader">Upload</a>
<a href="/deletefile">Delete</a>
<br><hr>
<h1>IBM Object Storage</h1>


<!doctype html>
<html>
   <body>

         <div style="border: 1px solid #EFEFEF;margin:10px;">
            <h3>Filename : {YUKI RITO} </h3>
            <img
src="https://flaskapp123.s3.jp-tok.cloud-object-storage.appdomain.cloud
/{{row}}" width="150px"></td>
         </div>

   </body>
</html>
```

Image:



index.html

C:/Users/YUKI-RITO/Desktop/ASSIGNMENT%203/flask-with-ibm-cloud-object-storage-main/templates/index.html

HOME Upload Delete

IBM Object Storage

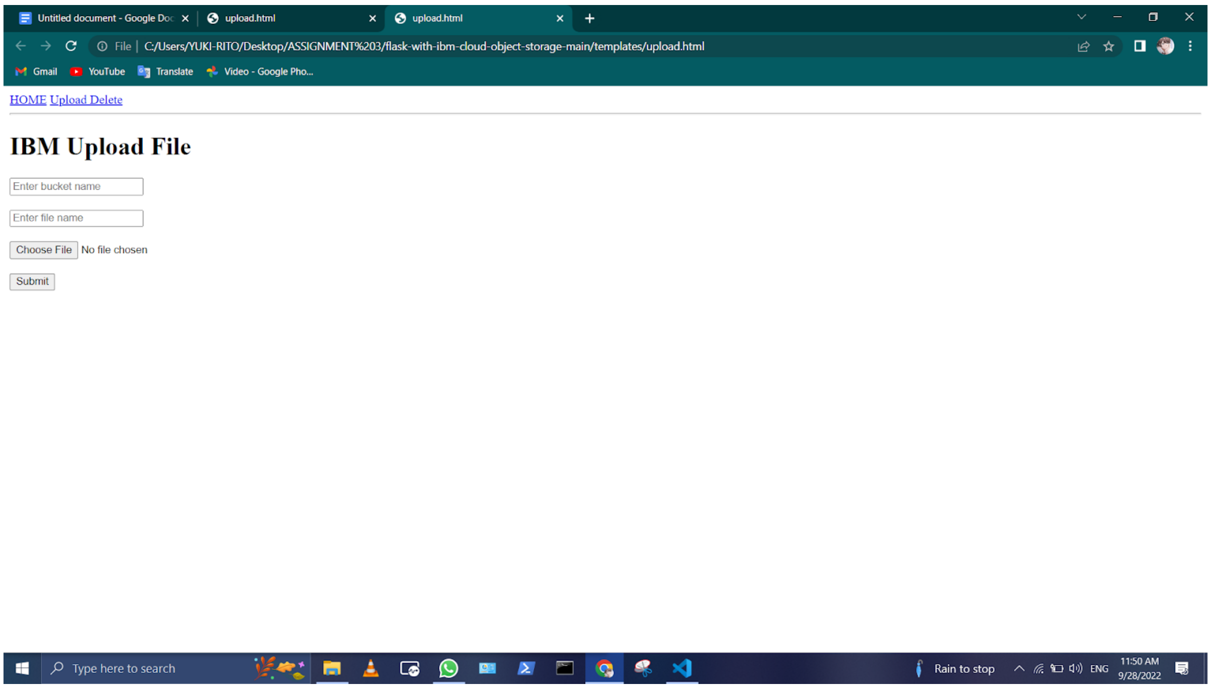
Filename : {YUKI RITO}

- UPLOAD .HTML

```
<html>
   <body>


<a href="/">HOME</a>
<a href="/uploader">Upload </a>
<a href="/deletefile">Delete </a>
<br><hr>

<h1>IBM Upload File</h1>

      <form action = "/uploader" method = "POST"
         enctype = "multipart/form-data">
         <input type = "text" placeholder="Enter bucket name" name =
"bucket" />
         <br>
         <br>
         <input type = "text" placeholder="Enter file name" name =
"filename" />
         <br>
         <br>
         <input type = "file" name = "file" />
         <br>
```

```
            <br>
            <input type = "submit"/>
        </form>
    </body>
</html>
```

IMAGE:



Untitled document - Google Doc
upload.html
upload.html
File | C:/Users/YUKI-RITO/Desktop/ASSIGNMENT%203/flask-with-ibm-cloud-object-storage-main/templates/upload.html
Gmail
YouTube
Translate
Video - Google Pho...

HOME Upload Delete

IBM Upload File

Enter bucket name

Enter file name

Choose File No file chosen

Submit

Type here to search
Rain to stop
ENG
11:50 AM
9/28/2022

4. Flask-with-ibm-db2-main
   - APP.py

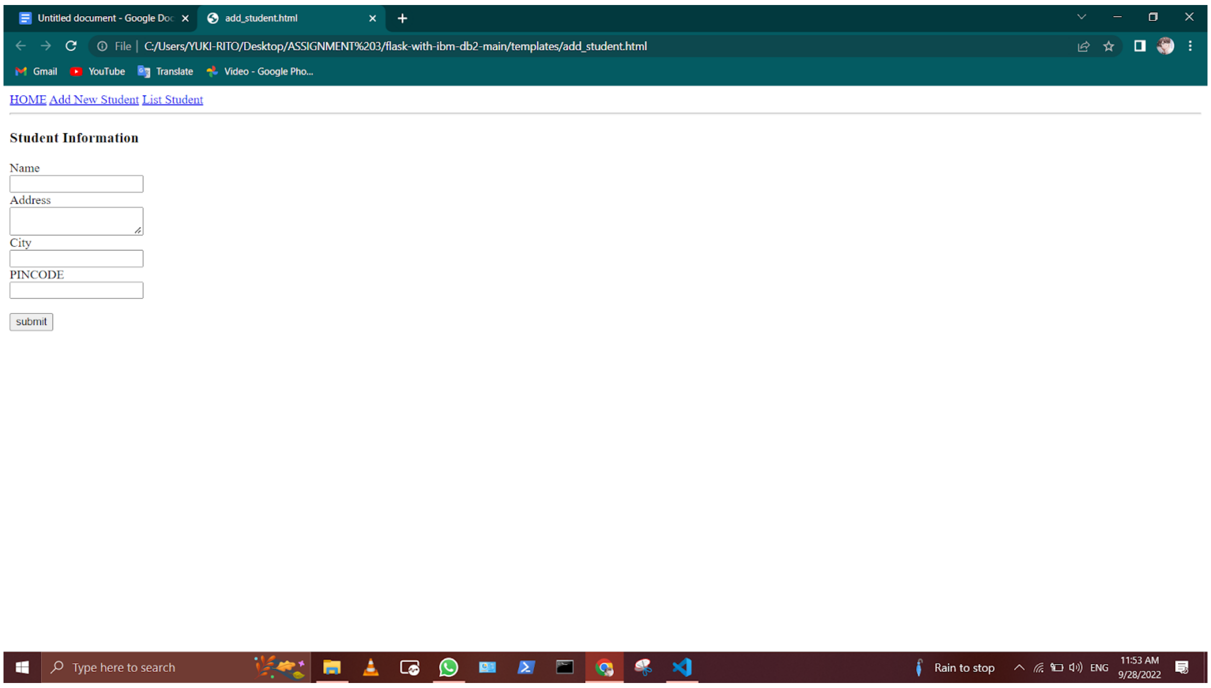
```
from turtle import st
from flask import Flask, render_template, request, redirect, url_for,
session
from markupsafe import escape

import ibm_db
conn =
ibm_db.connect("DATABASE=<databasename>;HOSTNAME=<your-hostname>;PORT=<
portnumber>;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;
UID=<username>;PWD=<password>",'','')

app = Flask(__name__)



@app.route('/')
```

```
def home():
  return render_template('home.html')

@app.route('/addstudent')
def new_student():
  return render_template('add_student.html')

@app.route('/addrec',methods = ['POST', 'GET'])
def addrec():
  if request.method == 'POST':

    name = request.form['name']
    address = request.form['address']
    city = request.form['city']
    pin = request.form['pin']

    sql = "SELECT * FROM students WHERE name =?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,name)
    ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)

    if account:
      return render_template('list.html', msg="You are already a
member, please login using your details")
    else:
      insert_sql = "INSERT INTO students VALUES (?,?,?,?)"
      prep_stmt = ibm_db.prepare(conn, insert_sql)
      ibm_db.bind_param(prep_stmt, 1, name)
      ibm_db.bind_param(prep_stmt, 2, address)
      ibm_db.bind_param(prep_stmt, 3, city)
      ibm_db.bind_param(prep_stmt, 4, pin)
      ibm_db.execute(prep_stmt)

    return render_template('home.html', msg="Student Data saved
successfuly..")

@app.route('/list')
def list():
  students = []
  sql = "SELECT * FROM Students"
  stmt = ibm_db.exec_immediate(conn, sql)
  dictionary = ibm_db.fetch_both(stmt)
  while dictionary != False:
    # print ("The Name is : ",  dictionary)
    students.append(dictionary)
    dictionary = ibm_db.fetch_both(stmt)

  if students:
    return render_template("list.html", students = students)

@app.route('/delete/<name>')
def delete(name):
  sql = f"SELECT * FROM Students WHERE name='{escape(name)}'"
  print(sql)
  stmt = ibm_db.exec_immediate(conn, sql)
  student = ibm_db.fetch_row(stmt)
  print ("The Name is : ",  student)
  if student:
    sql = f"DELETE FROM Students WHERE name='{escape(name)}'"
    print(sql)
    stmt = ibm_db.exec_immediate(conn, sql)

    students = []
    sql = "SELECT * FROM Students"
    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary != False:
      students.append(dictionary)
      dictionary = ibm_db.fetch_both(stmt)
    if students:
      return render_template("list.html", students = students,
msg="Delete successfully")



  # # while student != False:
  # #   print ("The Name is : ",  student)

  # print(student)
  return "success..."

# @app.route('/posts/edit/<int:id>', methods=['GET', 'POST'])
# def edit(id):

#     post = BlogPost.query.get_or_404(id)
#     if request.method == 'POST':
#         post.title = request.form['title']
#         post.author = request.form['author']
#         post.content = request.form['content']
#         db.session.commit()
#         return redirect('/posts')
#     else:
#         return render_template('edit.html', post=post)
```

- ADD_STUDENT .HTML

```
<a href="/">HOME</a>
<a href="/addstudent">Add New Student</a>
<a href="/list">List Student</a>
<hr>


<form action = "{{ url_for('addrec') }}" method = "POST">
    <h3>Student Information</h3>
    Name<br>
    <input type = "text" name="name" /></br>

    Address<br>
    <textarea name="address" ></textarea><br>

    City<br>
    <input type = "text" name="city" /><br>

    PINCODE<br>
    <input type = "text" name="pin" /><br><br>
    <input type = "submit" value = "submit" /><br>
 </form>
```

IMAGE:


Untitled document - Google Doc
add_student.html
File | C:/Users/YUKI-RITO/Desktop/ASSIGNMENT%203/flask-with-ibm-db2-main/templates/add_student.html
Gmail
YouTube
Translate
Video - Google Pho...
Type here to search
Rain to stop
ENG
11:53 AM
9/28/2022

HOME Add New Student List Student

Student Information

Name

Address

City

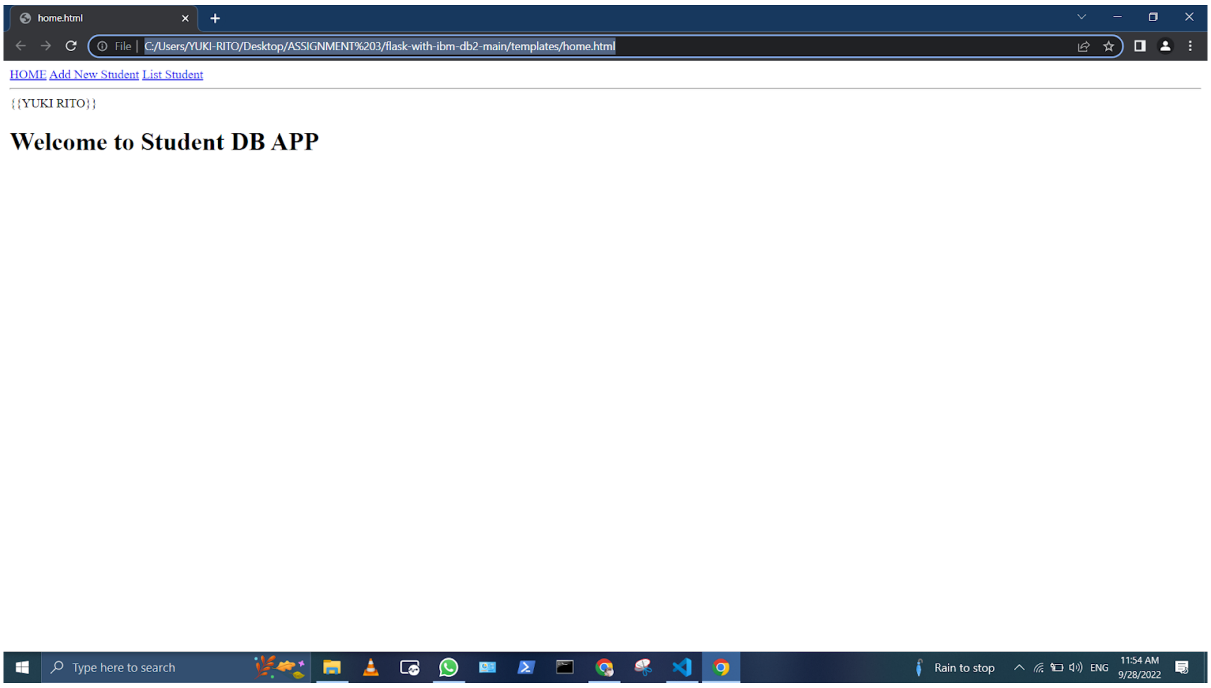
PINCODE

submit

- HOME.HTML

```
<a href="/">HOME</a>
<a href="/addstudent">Add New Student</a>
<a href="/list">List Student</a>
<hr>

{{YUKI RITO}}

<h1>Welcome to Student DB APP</h1>
```

IMAGE:


home.html
File | C:/Users/YUKI-RITO/Desktop/ASSIGNMENT%203/flask-with-ibm-db2-main/templates/home.html
Type here to search
Rain to stop
ENG
11:54 AM
9/28/2022

HOME Add New Student List Student

{{YUKI RITO}}

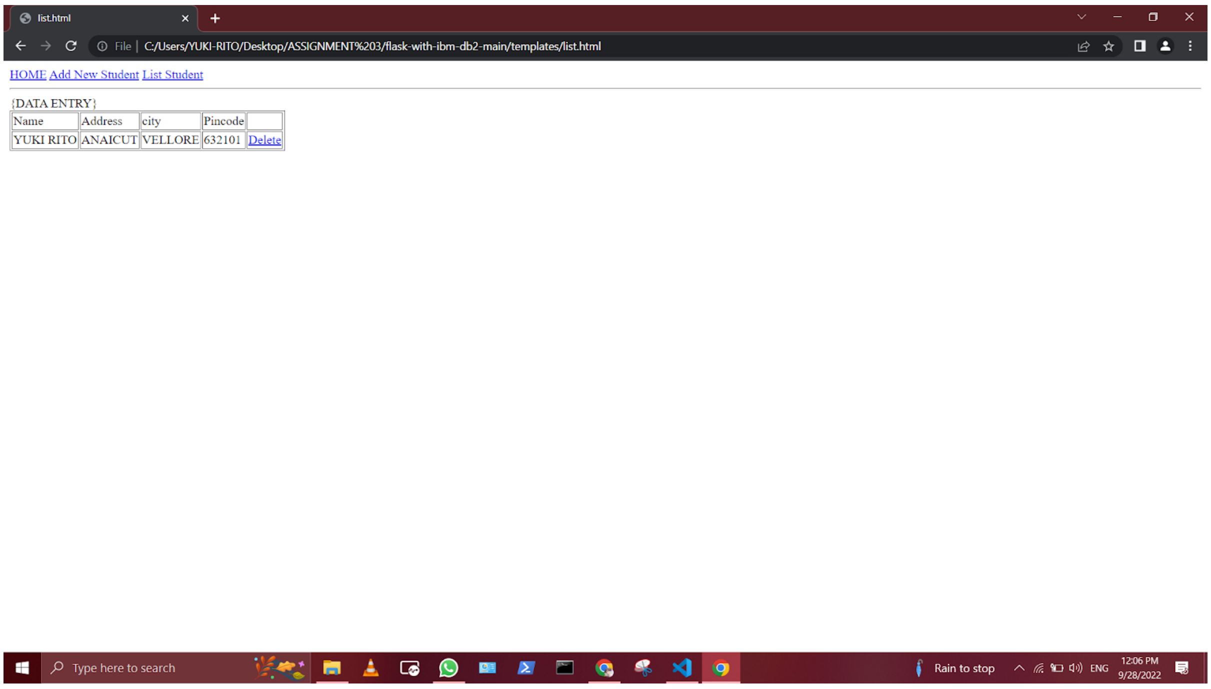
Welcome to Student DB APP

- List.html

```
<!doctype html>
<html>
   <body>

    <a href="/">HOME</a>
    <a href="/addstudent">Add New Student</a>
    <a href="/list">List Student</a>
    <br><hr>

    {DATA ENTRY}

      <table border = 1>
         <thead>
            <td>Name</td>
            <td>Address</td>
            <td>city</td>
            <td>Pincode</td>
            <td></td>
         </thead>

            <tr>
               <td>YUKI RITO</td>
               <td>ANAICUT</td>
```

```
                    <td>VELLORE</td>
                    <td>632101</td>
                    <td><a href="/delete/{{row['NAME']}}">Delete</a></td>
                </tr>

        </table>


    </body>
</html>
```

Image:



list.html

File | C:/Users/YUKI-RITO/Desktop/ASSIGNMENT%203/flask-with-ibm-db2-main/templates/list.html

HOME Add New Student List Student

{DATA ENTRY}

Name	Address	city	Pincode	
YUKI RITO	ANAICUT	VELLORE	632101	Delete

Type here to search

Rain to stop

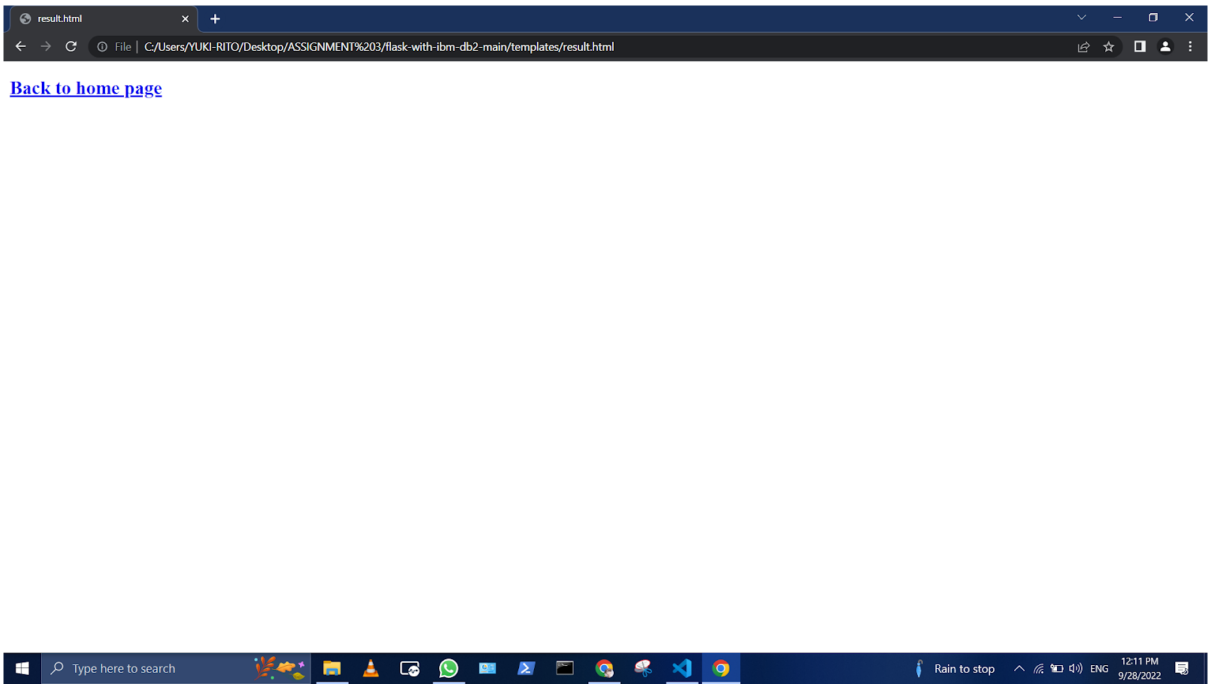
ENG

12:06 PM
9/28/2022

- RESULT.html

```
<!doctype html>
<html>
    <body>

        <h2><a href = "\">Back to home page</a></h2>
    </body>
</html>
```

IMAGE:


result.html
File
C:/Users/YUKI-RITO/Desktop/ASSIGNMENT%203/flask-with-ibm-db2-main/templates/result.html
Type here to search
Rain to stop
ENG
12:11 PM
9/28/2022

Back to home page