

**Assignment -2**  
Dashboard

Assignment Date	19 <sup>th</sup> October 2022
Student Name	K VELMURUGAN
Student Roll Number	620119104108
Maximum Marks	2 Marks

**Index page:**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="{{ url_for('static', filename='style.css')}}">
  <title>Home</title>
</head>
<body>

  <h2 class="home">Home Page</h2>

  <div class="space">
    <a class="text" href="/about"> About </a>
    <a class="text" href="/signin"> Log In </a>
    <a class="text" href="/signup"> Sign Up </a>
  </div>

</body>
</html>
```

**Sign in page:**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css"
rel="stylesheet"
```

```

    integrity="sha384-
iYQeCzEYFbKjA/T2uDLTpkwGzCiq6soy8tYal1GyVh/UjpbCx/TYkiZhlZB6+fzT"
crossorigin="anonymous">
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css')}}">
    <title>Sign In</title>
</head>
<body>

    <div class="space">
        <a class="text" href="/"> Home </a>
        <a class="text" href="/about"> About </a>
        <a class="text" href="/signup"> Sign Up </a>
    </div>
    <h2 class="home">Log In</h2>

    <div class="signin">
        <form>
            <div class="form-group">
                <label for="exampleInputEmail1">Email</label>
                <input type="email" class="form-control" id="exampleInputEmail1" aria-
describedby="emailHelp" placeholder="Enter email">
            </div> <br>

            <div class="form-group">
                <label for="exampleInputEmail1">password</label>
                <input type="password" class="form-control" id="exampleInputEmail1"
aria-describedby="emailHelp" placeholder="Enter password">
            </div>
            <br>

            <button type="submit" class="btn btn-primary">Submit</button>
        </form>
    </div>

</body>
</html>

```

### Sign up Page:

```

<!DOCTYPE html>
<html lang="en">
<head>

```

```

<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css"
rel="stylesheet"
integrity="sha384-
iYQeCzEYFbKjA/T2uDLTpkwGzCiq6soy8tYal1GyVh/UjpbCx/TYkiZhlZB6+fzT"
crossorigin="anonymous">
<link rel="stylesheet" href="{{ url_for('static', filename='style.css')}}">
<title>Sign Up</title>
</head>
<body>

<div class="space">
<a class="text" href="/"> Home </a>
<a class="text" href="/about"> About </a>
<a class="text" href="/signin"> Log In </a>
</div>

<h2 class="home">Sign Up</h2>

<div class="signup">
<form>
<div class="form-group">
<label for="Name">Name</label>
<input class="form-control" type="text" name="Name" placeholder="Enter
Name">
</div><br>

<div class="form-group">
<label for="exampleInputEmail1">Email</label>
<input type="email" class="form-control" id="exampleInputEmail1" aria-
describedby="emailHelp" placeholder="Enter email">
</div><br>

<div class="form-group">
<label for="exampleInputEmail1">password</label>
<input type="password" class="form-control" id="exampleInputEmail1"
aria-describedby="emailHelp" placeholder="Enter password">
</div><br>

```

```

    <div class="form-group">
      <label for="exampleInputEmail1">Re - password</label>
      <input type="password" class="form-control" id="exampleInputEmail1"
aria-describedby="emailHelp" placeholder="Enter password">
    </div><br>

    <button type="submit" class="btn btn-primary">Submit</button>
  </form>

</div>

```

```

</body>
</html>

```

### Style code :

```

@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@100;200;300;400;5
00;600;700&display=swap');
*{
  font-family: 'Poppins', sans-serif;
  top: 0;
  margin: 0;
}
body{
  background-
image:url("https://cdn.discordapp.com/attachments/999321523577966613/1023
800785215176865/img.png") ;
}
.home{
  font-size: 25px;
  text-align: center;
  margin-top: 10px;
  color: black;
}

.text{
  text-decoration: none;
  font-size: 20px;
  color: rgb(128, 42, 207);
  margin-left: 10px;
  margin-right: 10px;
}

```

```

        margin-top: 10px;
    }
    p{
        margin-left: 25px;
        margin-top: 10px;
        color:rgb(122, 2, 42);
        font-size: 20px;
    }
    .signin, .signup{
        height: 50vh;
        width: 50vw;
        margin-top: 10px;
        margin-left: 25px;
    }
    .space{
        margin-top:15px;
        text-align: right;
        margin-right: 10px;
    }

```

### **App python :**

```

from flask import Flask
from flask import render_template

app = Flask(__name__)

@app.route('/')
def main():
    return render_template('index.html')

@app.route('/about')
def about():
    return render_template('about.html')

@app.route('/signin')
def signin():
    return render_template('signin.html')

@app.route('/signup')
def signup():
    return render_template('signup.html')

```

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

### Source file:

```
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.cloud" #  
Current list available at https://control.cloud-object-  
storage.cloud.ibm.com/v2/endpoints  
COS_API_KEY_ID = "xVWVt8tjo5lL5eY-jii6RAZu4IAfMrRQY2uaD0_-9LZj"  
COS_INSTANCE_CRN = "crn:v1:bluemix:public:iam-  
identity::a/37641ef4661f44a7bfab12af5b4ba6fe::serviceid:ServiceId-dd1803b4-  
6727-46a8-b09c-1e921e8211c6"  
  
# Create resource  
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))  
  
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 threshold 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
        )

        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('jeeva-bucket1')

    return render_template('index.html', files = files)

@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'

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

@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')

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