**INTERNET TECHNOLOGIES LAB REPORT**

**NAME:** ANURAN CHAKRABORTY

**ROLL NO.:** 20

**CLASS:** BCSE-IV

**SECTION:** A1

**INDEX**

|  |  |  |
| --- | --- | --- |
| **Assn. No.** | **Problem** | **Page No.** |
| 1. | TCP based key-value store | 2-8 |
| 2. | Web socket based key-value Store | 9-15 |
| 3. | Chat Application using Node.js | 16-25 |
| 4. | Apparel Store Using Servlets and JSP | 26-52 |
| 5. | Flight Search Using Servlets and JSP | 53-83 |
| 6. | Mail Application | 84-91 |
| 7. | Flight Search Using Spring | 92-131 |

**ASSIGNMENT NUMBER:** 1

**PROBLEM STATEMENT:**

Implement a TCP-based key-value store. The server implements the key-value store and clients make use of it. The server must accept clients’ connections and serve their requests for ‘get’ and ‘put’ key value pairs. All key-value pairs should be stored by the server only in memory. Keys and values are strings.

The client accepts a variable no of command line arguments where the first argument is the server hostname followed by port no. It should be followed by any sequence of “get <key>” and/or “put <key> <value>”.

./client 192.168.124.5 5555 put city Kolkata put country India get country get city get Institute

India

Kolkata

<blank>

The server should be running on a TCP port. The server should support multiple clients and maintain their key-value stores separately.

Implement authorization so that only few clients having the role “manager” can access other’s key-value stores. A user is assigned the “guest” role by default. The server can upgrade a “guest” user to a “manager” user.

**CODE:**

The assignment has been implemented in python3.6.

**common.py** stores the commonly used functions by server and client

**import** **socket**

portServer=12345

*# Function to create a socket and bind it to a port*

**def** createSocket(port):

s=socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

s.setsockopt(socket.SOL\_SOCKET, socket.SO\_REUSEADDR, 1)

s.bind(('', port))

**return** s

*# Function to receive a connection*

**def** allowConn(s):

s.listen(5)

c, addr=s.accept()

**return** c, addr

*# Function to create a socket and connect to it*

**def** createConn(port,ip=''):

sock=socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

sock.connect((ip,port))

**return** sock

*# Function to send a frame*

**def** send\_frame(frame, c):

*# Send the frame to the other process*

c.send(frame.encode())

**client.py** contains the client-side code

**import** **socket**

**import** **threading**

**import** **common** **as** **co**

**import** **pickle**

**import** **sys**

*# Function to return a dictionary based on the request*

**def** parseArgs(args):

req=[]

i=0

**while** i<(len(args)):

**if**(args[i].lower()=='get'):

**if**(i==len(args)-1 **or** args[i+1].lower()=='put'): *# Error case*

**return** 0,req

**else**:

req.append({'method':'get','key':args[i+1]})

i=i+1

**elif**(args[i].lower()=='put'):

**if**(i==len(args)-2): *# Error case*

**return** 0,req

**else**:

req.append({'method':'put','key':args[i+1],'value':args[i+2]})

i=i+2

**elif**(args[i].lower()=='getother'):

**if**(i==len(args)-2): *# Error case*

**return** 0,req

**else**:

req.append({'method':'getother','key':args[i+2],'username':args[i+1]})

i=i+2

**elif**(args[i].lower()=='upgrade'):

req.append({'method':'upgrade'})

**else**:

**return** 0,req

i=i+1

**return** 1,req

sockClient=co.createConn(port=int(sys.argv[2]),ip=sys.argv[1])

uname=input('Enter a username: ')

sockClient.sendall(uname.encode())

**print**('Usage:')

**print**('get key : To get value corresponding to a key')

**print**('put key value : To insert a value corresponding to a key')

**print**('upgrade : To upgrade user status')

**print**('getother username key : To get value of another user (only allowed if manager)')

**while**(True):

*# Take input*

request=input('>> ')

**if**(request.lower()=='exit'):

**break**

retVal,req=parseArgs(request.split(' '))

**if**(retVal==0):

**print**('Invalid arguments')

**continue**

*# print(req)*

req=pickle.dumps(req)

*# Send the dictionary through socket*

sockClient.sendall(req)

*# Wait for response*

response=sockClient.recv(1024)

response=pickle.loads(response)

**print**(response)

**server.py** contains the server-side code.

**import** **socket**

**import** **threading**

**import** **common** **as** **co**

**import** **pickle**

*# Class to store key value for each client*

**class** **KeyValueClient**:

**def** \_\_init\_\_(self,username):

self.valstore={}

self.mode='guest'

self.username=username

**def** \_change\_mode(self):

self.mode='admin'

**def** \_getValue(self,key):

**if**(key **not** **in** self.valstore):

**return** 'Invalid key'

**return** self.valstore[key]

**def** \_putValue(self,key,value):

self.valstore[key]=value

**return** 'Successful'

*# Function to take action on the requests*

**def** takeAction(self,req):

res=[]

**for** reqs **in** req:

**if**(reqs['method'].lower()=='get'):

res.append(self.\_getValue(reqs['key']))

**elif**(reqs['method'].lower()=='put'):

res.append(self.\_putValue(reqs['key'],reqs['value']))

**elif**(reqs['method'].lower()=='upgrade'):

self.\_change\_mode()

res.append('mode change successfull')

**elif**(reqs['method'].lower()=='getother'):

**if**(self.mode=='guest' **and** self.username!=reqs['username']):

res.append('Access Denied')

**elif**(self.username==reqs['username'] **or** self.mode=='admin'):

**if**(reqs['username'] **in** global\_dict):

res.append(global\_dict[reqs['username']].\_getValue(reqs['key']))

**else**:

res.append('Invalid username')

**return** res

sockServer=co.createSocket(co.portServer)

global\_dict={}

*# Function to service a client*

**def** serviceClient(client, clientAddr):

**while** True:

requestC=clientAddr.recv(1024) *# Receive the request dictionary*

requestC=pickle.loads(requestC)

res=client.takeAction(requestC)

res=pickle.dumps(res)

clientAddr.sendall(res)

**def** allow\_new\_conn():

**while**(True):

*# Wait for a connection*

sockServer.listen(10)

cAddr, addrServer=sockServer.accept()

**print**('Connected to client')

*# Fetch username*

uname=cAddr.recv(1024).decode()

client=KeyValueClient(uname) *# Create client by that username*

global\_dict[uname]=client

*# Start a new thread for the sender*

sendThread=threading.Thread(target=serviceClient, args=[client,cAddr])

sendThread.start()

allow\_new\_conn()

**OUTPUT:**

|  |
| --- |
|  |

On the left two clients are started and on the right the server. The program can handle multiple commands in a single line. The client registers with a username. Then client ‘abc’ asks for the key x. It is not present and hence the output ‘Invalid key’. Client ‘def’ asks for the key of ‘abc’ but at the beginning it is a guest so ‘Access Denied’. Later after ‘upgrade’ it can view.

**ASSIGNMENT NUMBER:** 2

**PROBLEM STATEMENT:**

Implement a key-value store using Websocket. The server implements the key-value store and clients make use of it. The server must accept clients’ connections and serve their requests for ‘get’ and ‘put’ key value pairs. All key-value pairs should be stored by the server only in memory. Keys and values are strings as in Assignment 1. Implement authorization so that only few clients having the role “manager” can access other’s key-value stores. A user is assigned the “guest” role by default. The server can upgrade a “guest” user to a “manager” user.

Submit a report on the comparative analysis of the two assignments especially when both roles of manager and guests are considered.

**CODE:**

The assignment has been implemented in python3.6.

**client.py** contains the client-side code

**import** **asyncio**

**import** **websockets**

**import** **pickle**

**import** **sys**

*# Function to return a dictionary based on the request*

**def** parseArgs(args):

req=[]

i=0

**while** i<(len(args)):

**if**(args[i].lower()=='get'):

**if**(i==len(args)-1 **or** args[i+1].lower()=='put'): *# Error case*

**return** 0,req

**else**:

req.append({'method':'get','key':args[i+1]})

i=i+1

**elif**(args[i].lower()=='put'):

**if**(i==len(args)-2): *# Error case*

**return** 0,req

**else**:

req.append({'method':'put','key':args[i+1],'value':args[i+2]})

i=i+2

**elif**(args[i].lower()=='getother'):

**if**(i==len(args)-2): *# Error case*

**return** 0,req

**else**:

req.append({'method':'getother','key':args[i+2],'username':args[i+1]})

i=i+2

**elif**(args[i].lower()=='upgrade'):

req.append({'method':'upgrade'})

**else**:

**return** 0,req

i=i+1

**return** 1,req

async **def** clientRun():

ws\_url='ws://'+sys.argv[1]+':'+str(sys.argv[2])

*# async with websockets.connect(ws\_url) as sockClient:*

sockClient=await websockets.connect(ws\_url)

*# Accept username*

uname=input('Enter a username: ')

await sockClient.send(uname)

**print**('Usage:')

**print**('get key : To get value corresponding to a key')

**print**('put key value : To insert a value corresponding to a key')

**print**('upgrade : To upgrade user status')

**print**('getother username key : To get value of another user (only allowed if manager)')

**while**(True):

*# Take input*

request=input('>> ')

**if**(request.lower()=='exit'):

**break**

retVal,req=parseArgs(request.split(' '))

**if**(retVal==0):

**print**('Invalid arguments')

**continue**

*# print(req)*

req=pickle.dumps(req)

*# Send the dictionary through socket*

await sockClient.send(req)

*# Wait for response*

**try**:

response=await sockClient.recv()

**except**:

*# Reconnect*

**print**('Reconnecting...')

sockClient=await websockets.connect(ws\_url)

*# response=await sockClient.recv()*

response=pickle.loads(response)

**print**(response)

asyncio.get\_event\_loop().run\_until\_complete(clientRun())

**server.py** contains the server-side code.

**import** **asyncio**

**import** **websockets**

**import** **socket**

**import** **threading**

**import** **pickle**

*# Class to store key value for each client*

**class** **KeyValueClient**:

**def** \_\_init\_\_(self,username):

self.valstore={}

self.mode='guest'

self.username=username

**def** \_change\_mode(self):

self.mode='admin'

**def** \_getValue(self,key):

**if**(key **not** **in** self.valstore):

**return** 'Invalid key'

**return** self.valstore[key]

**def** \_putValue(self,key,value):

self.valstore[key]=value

**return** 'Successful'

*# Function to take action on the requests*

**def** takeAction(self,req):

res=[]

**for** reqs **in** req:

**if**(reqs['method'].lower()=='get'):

res.append(self.\_getValue(reqs['key']))

**elif**(reqs['method'].lower()=='put'):

res.append(self.\_putValue(reqs['key'],reqs['value']))

**elif**(reqs['method'].lower()=='upgrade'):

self.\_change\_mode()

res.append('mode change successfull')

**elif**(reqs['method'].lower()=='getother'):

**if**(self.mode=='guest' **and** self.username!=reqs['username']):

res.append('Access Denied')

**elif**(self.username==reqs['username'] **or** self.mode=='admin'):

**if**(reqs['username'] **in** global\_dict):

res.append(global\_dict[reqs['username']].\_getValue(reqs['key']))

**else**:

res.append('Invalid username')

**return** res

global\_dict={}

*# Function to service a client*

async **def** serviceClient(clientAddr,path):

**print**('Connected to client')

*# Fetch username*

uname=await clientAddr.recv()

client=KeyValueClient(uname) *# Create client by that username*

global\_dict[uname]=client

**while** True:

requestC=await clientAddr.recv() *# Receive the request dictionary*

requestC=pickle.loads(requestC)

res=client.takeAction(requestC)

res=pickle.dumps(res)

**print**('Hi')

await clientAddr.send(res)

start\_server = websockets.server.serve(serviceClient, '', 8765, ping\_timeout=100000, ping\_interval=100000)

asyncio.get\_event\_loop().run\_until\_complete(start\_server)

asyncio.get\_event\_loop().run\_forever()

**OUTPUT:**

|  |
| --- |
|  |

On the left two clients are started and on the right the server. The program can handle multiple commands in a single line. The client registers with a username. Then client ‘abc’ asks for the key x. It is not present and hence the output ‘Invalid key’. Client ‘def’ asks for the key of ‘abc’ but at the beginning it is a guest so ‘Access Denied’. Later after ‘upgrade’ it can view.

**COMPARATIVE ANALYSIS:**

|  |  |
| --- | --- |
| TCP Socket | Websocket |
| For a non-blocking TCP socket it will send data if the size of the data is less than the buffer size. If it is blocking it will wait for the buffer to be full and then send the data. Larger data may be fragmented and transmitted | Websocket can only send data if the data size is less than the buffer size. Websockets do not fragment data. |
| TCP sockets are half duplex i.e. while it receives data from a host it cannot simultaneously send data to the host. | Websockets are full duplex connections which allow simultaneous sending and receiving of data |
| In terms of the coding for interaction with multiple clients using TCP sockets threads needs to be manually created. | Threads need not be manually created and are handled by the library. |

**ASSIGNMENT NUMBER:** 3

**PROBLEM STATEMENT:**

Write a multi-client chat application consisting of both client and server programs. In this chat application simultaneously, several clients can communicate with each other. For this you need a single server program that clients connect to. The client programs send the chat text or image (input) to the server and then the server distributes that message (text or image) to all the other clients. Each client then displays the message sent to it by the server. The server should be able to handle several clients concurrently. It should work fine as clients come and go.

Develop the application using a framework based on Node.JS. How are messages handled concurrently?

Which web application framework(s) did you follow?

Prepare a detailed report of the experiments you have done, and your observations on performance of the system.

**CODE:**

**index.html**

This is the home page of the app.

<!DOCTYPE html>

<**html**>

<**head**>

<**title**>Chat App</**title**>

<**style**>

**body**{

**background-color**: #f8f8f8

}

#container{

**width**:700px;

**margin**:0 **auto**;

}

#chatWindow{

**height**:500px;

**overflow-y**: **auto**;

}

#mainWrapper{

**display**:**none**;

}

#chatWrapper{

**float**:**left**;

**border**:1px #ddd **solid**;

**border**-radius:10px;

**background**:#f6f6f6;

**padding**:20px;

}

#messageOther{

**border**:1px #ddd **solid**;

**border**-radius:10px;

**background**: #ADFF2F;

}

#messageMy{

**border**:1px #ddd **solid**;

**border**-radius:10px;

**background**: #32CD32;

}

#userWrapper{

**float**:**left**;

**border**:1px #ddd **solid**;

**border**-radius:10px;

**background**: #f6f6f6;

**padding**:10px;

**margin-left**:20px;

**width**:150px;

**max-height**:200px;

}

#namesWrapper{

**float**:**left**;

**border**:1px #ddd **solid**;

**border**-radius:10px;

**background**: #f6f6f6;

**padding**:10px;

**margin-left**:30px;

*/\*display:none;\*/*

}

**input**{

**height**:30px;

}

</**style**>

</**head**>

<**body**>

<**div** id="container">

<**div** id="namesWrapper">

<**p**>Create Username</**p**>

<**div** id="error"></**div**>

<**form** id="usernameForm">

<**input** type="text" id="username" size="35">

<**input** type="submit" value="Submit">

</**form**>

</**div**>

<**div** id="mainWrapper">

<**h2**>Chat Here</**h2**>

<**div** id="chatWrapper">

<**div** id="chatWindow">

<**form** id="messageForm">

<**input** type="text" id="message" placeholder="Enter your message">

<**input** type="submit" value="Submit"> <**br**>

<**input** type="file" id="fileimage" accept="image/png" name="photo" value="upload">

</**form**>

</**div**>

</**div**>

<**div** id="userWrapper">

<**div** id="users"></**div**>

</**div**>

</**div**>

</**div**>

<**script** src="http://code.jquery.com/jquery-latest.min.js"></**script**>

<**script** src="/socket.io/socket.io.js"></**script**>

<**script**>

$(**function**(){

**var** socket=io.connect();

**var** $messageForm=$('#messageForm');

**var** $message=$('#message');

**var** $chat=$('#chatWindow');

**var** $usernameForm=$("#usernameForm");

**var** $users=$("#users");

**var** $username=$("#username");

**var** $error=$("#error");

**var** uname='';

$usernameForm.submit(**function**(e){

e.preventDefault();

socket.emit('new user',$username.val(),**function**(data){

**if**(data){

$("#namesWrapper").hide();

$("#mainWrapper").show();

uname=$username.val();

}

**else**{

$error.html("Username is taken")

}

});

});

socket.on('usernames',**function**(data){

**var** html="";

**for**(i=0;i<data.length;i++)

{

html+=data[i]+'</br>';

}

$users.html(html);

});

$("#fileimage").click(**function**(){

*//Create an instance of FileReader*

**const** fileReader = **new** FileReader();

**var** fname=''

*//Start reading the file*

fileReader.onload=**function**(){

fname=document.getElementById("fileimage").files[0];

console.log(fname);

**var** dataURL = fileReader.result;

$(**function**() {

socket.emit('img message',{'image':dataURL});

$("#fileimage").val("");

**return** **false**;

});

};

fileReader.readAsDataURL(document.getElementById("fileimage").files[0]);

});

$messageForm.submit(**function**(e){

e.preventDefault();

socket.emit('send message',$message.val());

$message.val('');

});

socket.on('new message',**function**(data){

*//Create a div for messages*

console.log(uname);

**if**(data.user==uname)

{

mess="<div id='messageMy'><strong>"+data.user+":</strong> "+data.msg+"</div><br>";

}

**else**

{

mess="<div id='messageOther'><strong>"+data.user+":</strong> "+data.msg+"</div><br>";

}

$chat.append(mess);

});

socket.on('imageConversionByServer', **function**(data){

console.log(data);

**if**(data.user==uname)

{

mess="<div id='messageMy'><strong>"+data.user+":</strong> <img src=\""+data.img+"\" style=\"width:128px;height:128px;\"/>"+"</div><br>";

}

**else**

{

mess="<div id='messageOther'><strong>"+data.user+":</strong> <img src=\""+data.img+"\" style=\"width:128px;height:128px;\"/>"+"</div><br>";

}

$chat.append(mess);

});

});

</**script**>

</**body**>

</**html**>

**server.js**

This is the code for the server written in node.js

**var** express=require('express'),

app=express(),

server=require('http').createServer(app),

io=require('socket.io').listen(server);

usernames=[];

**var** fs=require('fs');

server.listen(process.env.PORT || 3000)

console.log('Server running');

app.get('/', **function**(req,res){

res.sendFile(\_\_dirname+"/index.html");

});

io.sockets.on('connection',**function**(socket)

{

console.log('Socket connected');

socket.on('new user',**function**(data, callback){

**if**(usernames.indexOf(data)!=-1){

callback(**false**);

}

**else**

{

callback(**true**);

socket.username=data;

usernames.push(socket.username);

updateUserNames();

}

});

**function** updateUserNames(){

io.sockets.emit('usernames',usernames);

}

*//Send message*

socket.on('send message', **function**(data){

io.sockets.emit('new message',{'msg':data,'user':socket.username});

});

socket.on('img message', **function**(msg){

console.log(msg)

**var** base64data = msg.image.replace(/^data:image\/png;base64,/,"");

fs.writeFile('image.png', base64data, 'base64', (err) =>{

**if** (err) **throw** err;

console.log("The file has been saved");

fs.readFile(\_\_dirname+'/image.png', **function**(err, data){

console.log("sending");

io.sockets.emit('imageConversionByServer', {user : socket.username, img : "data:image/png;base64,"+ data.toString("base64")});

});

});

});

socket.on('disconnect', **function**(data){

**if**(!socket.username){

**return**;

}

usernames.splice(usernames.indexOf(socket.username),1);

updateUserNames();

})

});

**OUTPUT:**

|  |
| --- |
|  |
| **Fig. 1** abc’s chat |
|  |
| **Fig. 2** def’s chat |

**DISCUSSIONS:**

1. Frameworks used in this assignment are Node.js, and express.
2. The concurrency is handled by Node.js as it works asynchronously it can automatically handle new users by creating a separate thread for each of them.
3. The overall performance of the application is quite good as it can send files and text pretty quickly over the network.

**ASSIGNMENT NUMBER:** 4

**PROBLEM STATEMENT:**

Design an online apparel store using servlets and jsps. The store keeps records for its items in a database where some items may be discounted and some other items should be displayed as “new arrivals”. A user may search for a specific item. By default, when a user signs in, based on his/her profile (male/female etc.), show him/her preferred set of clothing. Users will be divided into two groups: some users looking for discounted items mainly, some others looking for new arrivals. So, depending on their preference already set in the database, the order of the displayed list would vary. By default, discounted items will be displayed first.

You may apply the concept of “dependency injection” here. Dependency injection (DI) is a technique where one object supplies the dependencies of another object. Basically you have an interface and a number of java beans implementing them. You may use SessionListener or ServletContextListener. The major benefit of DI is loose coupling and ease of use. DI makes classes more cohesive because they have fewer responsibilities.

**FLOW DIAGRAM:**

|  |
| --- |
|  |

**CODE:**

**index.html**

This is the homepage where the login and registration forms are present.

<!DOCTYPE html>

<**html**>

<**head**>

<**meta** charset="ISO-8859-1">

<**title**>Login</**title**>

<**style** type="text/css">

**body**{

**background-color**: #a2a2a2;

}

#login-page{

**width**:360px;

**padding**:10% 0 0;

**margin**:**auto**;

}

#formAll{

**position**: **relative**;

**z-index**: 100;

**background**: #ffffff;

**max-width**: 360px;

**margin**: 0 **auto** 100px;

**padding**: 45px;

**text-align**: **center**;

}

**select**, **input**{

**font-family**: "Times New Roman", **serif**;

**outline**:1;

**background**: #f2f2f2;

**width**: 100%;

**border**: 0;

**margin**: 0 0 15px;

**padding**: 15px;

box-sizing: **border**-box;

**font-size**: 14p;

}

#submit{

**text-transform**: **uppercase**;

**padding**: 15px;

**color**: #FFFFFF;

**background**: #000000;

**cursor**: **pointer**;

}

**.register-Form**{

**display**: **none**;

}

**.forgot-pass-form**{

**display**: **none**;

}

#duplicateUname{

**display**:**none**;

**color**: red;

}

</**style**>

<**script** src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js">

</**script**>

</**head**>

<**body**>

<**div** id="login-page">

<**div** id="form">

<**form** id="loginform formAll" class="loginform" action="login" method="post">

<**input** type="text" name="username" required="" placeholder="Enter your Username"><**br**>

<**input** type="password" name="password" required="" placeholder="Enter your password"><**br**>

<**input** type="submit" id="submit" name="submit" value="Login">

<**p** class="notReg"> Not registered? <**a** class="reg-here" href="#">Register Here </**a**></**p**>

<**p** class="notReg"> Forgot password? <**a** class="forgot-pass" href="#">Click here </**a**></**p**>

</**form**>

<**form** id="reg formAll" class="register-Form" action="register" method="post">

<**input** type="text" name="name" required="" placeholder="Enter your name"><**br**>

<**select** id="gender" name="gender" required="">

<**option** value="" disabled selected>Select your gender</**option**>

<**option** value="male">Male</**option**>

<**option** value="female">Female</**option**>

</**select**><**br**>

<**select** id="producttype" name="prodtype">

<**option** value="" disabled selected>Select your product category</**option**>

<**option** value="discount">Discounted Items</**option**>

<**option** value="newarr">New Arrivals</**option**>

</**select**><**br**>

<**div** id="duplicateUname">Sorry, this username is already taken</**div**>

<**input** type="text" name="username" placeholder="Enter your Username"><**br**>

<**input** type="password" name="password" placeholder="Enter your password"><**br**>

<**input** type="submit" id="submit" name="submit" value="Register">

<**p** class="alreadyReg"> Already registered? <**a** class="login-here" href="#">Login Here </**a**></**p**>

</**form**>

<**form** id="forgot formAll" class="forgot-pass-form" action="forgot" method="post">

<**input** type="text" name="username" placeholder="Enter your username">

<**input** type="password" name="password" placeholder="Enter new password">

<**input** type="submit" name="submit" id="submit" value="Change Password">

</**form**>

</**div**>

</**div**>

</**body**>

<**script** type="text/javascript">

$(document).ready(**function**(){

$('.reg-here').click(**function**(){

$('.loginform').hide();

$('.forgot-pass-form').hide();

$('.register-Form').show();

});

$('.forgot-pass').click(**function**(){

$('.loginform').hide();

$('.forgot-pass-form').show();

$('.register-Form').hide();

})

$('.login-here').click(**function**(){

$('.loginform').show();

$('.forgot-pass-form').hide();

$('.register-Form').hide();

});

});

</**script**>

</**html**>

**dashboard.jsp**

This is the page which will be displayed after the user logs in.

**<%@** page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"**%>**

**<%@**page import="java.sql.ResultSet"**%>**

**<%@**page import="com.shopping.DAO"**%>**

**<%@**page import="com.shopping.User"**%>**

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

**<html>**

**<head>**

**<meta** http-equiv="Content-Type" content="text/html; charset=ISO-8859-1"**>**

**<title>**Dashboard**</title>**

**<style** type="text/css"**>**

ul {

list-style-type: none;

margin: 0;

padding: 2px;

overflow: hidden;

background-color: #333;

height: 40px;

}

li {

padding:1px;

float: left;

color: #ffffff;

font-size: 20px;

text-align: center;

}

.cards{

width:12%;

display: inline-block;

margin:40px;

box-shadow:2px 2px 10px black;

}

.image img{

width:100%;

}

.title{

text-align: center;

}

.price{

text-align: center;

font-size: 20px;

padding:4px;

}

.gender{

text-align: center;

color:#a0a0a0;

font-size: 16px;

padding:4px;

}

.origPrice{

text-align: center;

font-size: 16px;

padding:4px;

text-decoration: line-through;

color: grey;

}

.nameDisp{

float: right;

}

.logout a{

color:white;

}

.ribbon {

position: relative;

left: -5px; top: -5px;

z-index: 1;

overflow: hidden;

width: 75px; height: 75px;

text-align: right;

}

.ribbon span {

font-size: 10px;

font-weight: bold;

color: #FFF;

text-transform: uppercase;

text-align: center;

line-height: 20px;

transform: rotate(-45deg);

-webkit-transform: rotate(-45deg);

width: 100px;

display: block;

background: #79A70A;

background: linear-gradient(#F70505 0%, #8F0808 100%);

box-shadow: 0 3px 10px -5px rgba(0, 0, 0, 1);

position: absolute;

top: 19px; left: -21px;

}

.ribbon span::before {

content: "";

position: absolute; left: 0px; top: 100%;

z-index: -1;

display: block;

border-left: 3px solid #8F0808;

border-right: 3px solid transparent;

border-bottom: 3px solid transparent;

border-top: 3px solid #8F0808;

}

.ribbon span::after {

content: "";

position: absolute; right: 0px; top: 100%;

z-index: -1;

display: block;

border-left: 3px solid transparent;

border-right: 3px solid #8F0808;

border-bottom: 3px solid transparent;

border-top: 3px solid #8F0808;

}

.images img{

width:100%;

}

.error{

font-size: 20px;

text-align: center;

}

.search-bar{

justify-content: center;

align-items: center;

}

.search-field{

margin: auto;

height: 30px;

width: 50%;

padding: 10px;

border-radius: 40px;

}

.search-btn{

height: 40px;

width: 10%;

padding: 10px;

border-radius: 40px;

background-color: grey;

color: white;

text-align: center;

}

.search-btn:hover{

background-color: black;

cursor: pointer;

}

**</style>**

**</head>**

**<body>**

**<%**

response.setHeader("Cache-Control","no-cache, no-store, must-revalidate");

System.out.println((User)session.getAttribute("user"));

*//Session check*

**if**((User)session.getAttribute("user")==**null**)

{

System.out.println("Hi");

response.sendRedirect("index.html");

**return**;

}

**%>**

**<ul>**

**<li** class="logout"**><a** href="**<%=**request.getContextPath()**%>**/logout"**>**Logout**</a></li>**

**<li** class="nameDisp"**>**Hi, **<%=**((User)(session.getAttribute("user"))).getName()**%></li>**

**</ul>**

**<br><br>**

*<!-- Now fetch the data -->*

**<div** class="main"**>**

**<div** class="search-bar"**>**

**<form** method="post"**>**

**<input** type="text" class="search-field" name="search" required="" placeholder="Search..."**>**

**<input** type="submit" class="search-btn" name="submit" value="search"**>**

**</form>**

**</div>**

**<%**

DAO dao=(DAO)session.getAttribute("dao");

ResultSet rs;

**if**(request.getParameter("submit")!=**null**)

rs=dao.getItemsByName(request.getParameter("search"));

**else**

rs=dao.getItems((User)session.getAttribute("user"));

**if**(rs==**null**)

{**%>**

**<div** class="error"**>**

Sorry!! No Items match your search

**</div>**

**<%}**

**else**

**do**

{

System.out.println(rs);

{**%>**

**<div** class="cards"**>**

**<%i**f(rs.getString("itemtype").equals("newarr")){ **%>**

**<div** class="ribbon"**><span>**NEW**</span></div>**

**<%}** **%>**

**<div** class="images"**>**

**<img** src="**<%=**"/ShoppingApp/images/"+rs.getString("name")+".png" **%>**"**/>**

**</div>**

**<div** class="title"**>**

**<h3><%=**rs.getString("name")**%></h3>**

**</div>**

**<div** class="gender"**>**

Gender: **<%=**rs.getString("gender")**%>**

**</div>**

**<%i**f(rs.getString("itemtype").equals("discount"))

{**%>**

**<div** class="origPrice"**>**

Price: Rs. **<%=**rs.getString("price")**%>**

**</div>**

**<%}** **%>**

**<div** class="price"**>**

Price: Rs.

**<%**

int disc=Integer.parseInt(rs.getString("discount"));

int orig=Integer.parseInt(rs.getString("price"));

int newprice=orig-(int)(orig\*disc/100.0);

out.print(newprice);

**%>**

**</div>**

**</div>**

**<%}**

}

**while**(rs.next());

**%>**

**</div>**

**</body>**

**</html>**

**User.java**

The class which describes a user. It stores the user details in it.

**package** **com.shopping**;

**public** **class** **User** {

**private** String name,uname,gender,choice,password;

**public** User(String name,String uname,String gender,String choice, String password)

{

**this**.name=name;

**this**.uname=uname;

**this**.gender=gender;

**this**.choice=choice;

**this**.password=password;

}

**public** String getName() {

**return** name;

}

**public** String getUname() {

**return** uname;

}

**public** String getGender() {

**return** gender;

}

**public** String getChoice() {

**return** choice;

}

**public** String getPassword() {

**return** password;

}

}

**RegisterServlet.java**

This is the servlet responsible for registering an User in the database.

**package** **com.shopping**;

**import** **java.io.IOException**;

**import** **java.io.PrintWriter**;

**import** **java.sql.ResultSet**;

**import** **java.sql.SQLException**;

**import** **javax.servlet.RequestDispatcher**;

**import** **javax.servlet.ServletConfig**;

**import** **javax.servlet.ServletException**;

**import** **javax.servlet.annotation.WebServlet**;

**import** **javax.servlet.http.HttpServlet**;

**import** **javax.servlet.http.HttpServletRequest**;

**import** **javax.servlet.http.HttpServletResponse**;

*/\*\**

*\* Servlet implementation class RegisterServlet*

*\*/*

@WebServlet(name = "register", urlPatterns = { "/register" })

**public** **class** **RegisterServlet** **extends** HttpServlet {

**private** **static** **final** long serialVersionUID = 1L;

**private** String dburl,dbuname,dbpass;

*/\*\**

*\* @see HttpServlet#HttpServlet()*

*\*/*

**public** RegisterServlet() {

**super**();

*// TODO Auto-generated constructor stub*

}

*/\*\**

*\* @see Servlet#init(ServletConfig)*

*\*/*

**public** void init(ServletConfig config) **throws** ServletException {

*// TODO Auto-generated method stub*

**super**.init(config);

dburl=getServletContext().getInitParameter("dburl");

dbuname=getServletContext().getInitParameter("dbuname");

dbpass=getServletContext().getInitParameter("dbpass");

System.out.println(dburl);

}

*/\*\**

*\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)*

*\*/*

**protected** void doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

*// TODO Auto-generated method stub*

String name=request.getParameter("name");

String gender=request.getParameter("gender");

String prodtype=request.getParameter("prodtype");

String uname=request.getParameter("username");

String password=request.getParameter("password");

DAO dao=**new** DAO(dburl,dbuname,dbpass);

User u=**new** User(name,uname,gender,prodtype,password);

PrintWriter out = response.getWriter();

**try** {

**if**(dao.checkUname(uname))

{

out.println("<script type='text/javascript'>");

*// out.println("document.getElementById('duplicateUname').show()");*

out.println("alert('Duplicate Username');");

out.println("</script>");

RequestDispatcher rd=request.getRequestDispatcher("index.html");

rd.include(request, response);

**return**;

}

}

**catch** (Exception e1) {

*// TODO Auto-generated catch block*

e1.printStackTrace();

}

**try**{

dao.register(u);

}

**catch**(Exception e)

{

e.printStackTrace();

}

response.sendRedirect("index.html");

}

}

**LoginServlet.java**

This servlet takes the username and password, checks whether it exists in the database and accordingly allows the user to log in or displays an error message for wrong credentials.

**package** **com.shopping**;

**import** **java.io.IOException**;

**import** **java.io.PrintWriter**;

**import** **java.sql.ResultSet**;

**import** **java.sql.SQLException**;

**import** **javax.servlet.RequestDispatcher**;

**import** **javax.servlet.ServletException**;

**import** **javax.servlet.annotation.WebServlet**;

**import** **javax.servlet.http.HttpServlet**;

**import** **javax.servlet.http.HttpServletRequest**;

**import** **javax.servlet.http.HttpServletResponse**;

**import** **javax.servlet.http.HttpSession**;

*/\*\**

*\* Servlet implementation class LoginServlet*

*\*/*

@WebServlet(name = "login", urlPatterns = { "/login" })

**public** **class** **LoginServlet** **extends** HttpServlet {

**private** **static** **final** long serialVersionUID = 1L;

**private** String dburl,dbuname,dbpass;

*/\*\**

*\* @see HttpServlet#HttpServlet()*

*\*/*

**public** LoginServlet() {

**super**();

*// TODO Auto-generated constructor stub*

}

**public** void init()

{

dburl=getServletContext().getInitParameter("dburl");

dbuname=getServletContext().getInitParameter("dbuname");

dbpass=getServletContext().getInitParameter("dbpass");

}

*/\*\**

*\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)*

*\*/*

**protected** void doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

*// TODO Auto-generated method stub*

String uname=request.getParameter("username");

String password=request.getParameter("password");

PrintWriter out = response.getWriter();

DAO dao=**new** DAO(dburl,dbuname,dbpass);

ResultSet rs=**null**;

**try**

{

rs = dao.loginCheck(uname, password);

}

**catch** (ClassNotFoundException | SQLException e1) {

*// TODO Auto-generated catch block*

e1.printStackTrace();

}

**if**(rs!=**null**)

{

User u=**null**;

*//Set the session*

**try**

{

u = **new** User(rs.getString("name"), rs.getString("uname"), rs.getString("gender"), rs.getString("choice"), rs.getString("password"));

}

**catch** (SQLException e)

{

*// TODO Auto-generated catch block*

e.printStackTrace();

}

HttpSession session=request.getSession();

session.setAttribute("user", u);

session.setAttribute("dao", dao);

*//Then redirect to dashboard*

response.sendRedirect("dashboard.jsp");

}

**else**

{

out.println("<script type='text/javascript'>");

*// out.println("document.getElementById('duplicateUname').show()");*

out.println("alert('Wrong Login Credentials');");

out.println("</script>");

RequestDispatcher rd=request.getRequestDispatcher("index.html");

rd.include(request, response);

**return**;

}

}

}

**LogoutServlet.java**

This servlet is responsible for logging out a user. It mainly invalidates the session variable thus logging the user out.

**package** **com.shopping**;

**import** **java.io.IOException**;

**import** **javax.servlet.ServletException**;

**import** **javax.servlet.annotation.WebServlet**;

**import** **javax.servlet.http.HttpServlet**;

**import** **javax.servlet.http.HttpServletRequest**;

**import** **javax.servlet.http.HttpServletResponse**;

**import** **javax.servlet.http.HttpSession**;

*/\*\**

*\* Servlet implementation class LogoutServlet*

*\*/*

@WebServlet(name = "logout", urlPatterns = { "/logout" })

**public** **class** **LogoutServlet** **extends** HttpServlet {

**private** **static** **final** long serialVersionUID = 1L;

*/\*\**

*\* @see HttpServlet#HttpServlet()*

*\*/*

**public** LogoutServlet() {

**super**();

*// TODO Auto-generated constructor stub*

}

*/\*\**

*\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)*

*\*/*

**protected** void doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

*// TODO Auto-generated method stub*

*// Empty the session variable and redirect*

HttpSession session=request.getSession();

session.removeAttribute("user");

session.removeAttribute("dao");

session.invalidate();

response.sendRedirect("index.html");

}

}

**ForgotPassServlet.java**

This servlet resets the password for a user.

**package** **com.shopping**;

**import** **java.io.IOException**;

**import** **java.io.PrintWriter**;

**import** **javax.servlet.RequestDispatcher**;

**import** **javax.servlet.ServletConfig**;

**import** **javax.servlet.ServletException**;

**import** **javax.servlet.annotation.WebServlet**;

**import** **javax.servlet.http.HttpServlet**;

**import** **javax.servlet.http.HttpServletRequest**;

**import** **javax.servlet.http.HttpServletResponse**;

*/\*\**

*\* Servlet implementation class ForgotPassServlet*

*\*/*

@WebServlet(name = "forgot", urlPatterns = { "/forgot" })

**public** **class** **ForgotPassServlet** **extends** HttpServlet {

**private** **static** **final** long serialVersionUID = 1L;

**private** String dburl,dbuname,dbpass;

*/\*\**

*\* @see HttpServlet#HttpServlet()*

*\*/*

**public** ForgotPassServlet() {

**super**();

*// TODO Auto-generated constructor stub*

}

*/\*\**

*\* @see Servlet#init(ServletConfig)*

*\*/*

**public** void init(ServletConfig config) **throws** ServletException {

*// TODO Auto-generated method stub*

**super**.init(config);

dburl=getServletContext().getInitParameter("dburl");

dbuname=getServletContext().getInitParameter("dbuname");

dbpass=getServletContext().getInitParameter("dbpass");

System.out.println(dburl);

}

*/\*\**

*\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)*

*\*/*

**protected** void doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

*// TODO Auto-generated method stub*

String uname=request.getParameter("username");

String password=request.getParameter("password");

DAO dao=**new** DAO(dburl,dbuname,dbpass);

PrintWriter out = response.getWriter();

*//Check if username in database*

**try** {

**if**(dao.checkUname(uname))

{

dao.updatePass(uname, password);

response.sendRedirect("index.html");

}

**else**

{

out.println("<script type='text/javascript'>");

out.println("alert('Sorry!! Not a registered user');");

out.println("</script>");

RequestDispatcher rd=request.getRequestDispatcher("index.html");

rd.include(request, response);

**return**;

}

}

**catch** (Exception e1) {

*// TODO Auto-generated catch block*

e1.printStackTrace();

}

}

}

**DAO.java**

This Java bean is responsible for interacting with the database. It takes care of all database operations- connecting with the database, querying the database and returning the required data to the servlets.

**package** **com.shopping**;

**import** **java.sql.\***;

**public** **class** **DAO** {

**private** Connection con;

**private** Statement stmt;

**private** String url,uname,password;

**public** DAO(String url, String uname, String password)

{

**this**.url=url;

**this**.uname=uname;

**this**.password=password;

}

*//Function to establish the connection*

**private** void setConnection() **throws** ClassNotFoundException, SQLException

{

Class.forName("com.mysql.jdbc.Driver");

con=DriverManager.getConnection(url,uname,password);

stmt=con.createStatement();

}

*//Function to close the connection*

**private** void closeConn() **throws** SQLException

{

con.close();

}

*//Function for login*

**public** ResultSet loginCheck(String username, String password) **throws** ClassNotFoundException, SQLException

{

setConnection();

String query="select \* from logininfo where uname=\'"+username+"' and password=\'"+password+"'";

ResultSet rs=stmt.executeQuery(query);

**if**(rs.next())

{

**return** rs; *//Means username password exists*

}

closeConn();

**return** **null**;

}

*//Function for checking duplicate username*

**public** boolean checkUname(String username) **throws** ClassNotFoundException, SQLException

{

setConnection();

String query="select \* from logininfo where uname=\'"+username+"'";

ResultSet rs=stmt.executeQuery(query);

**if**(rs.next())

{

**return** **true**; *//Means username password exists*

}

closeConn();

**return** **false**;

}

*//Function for registering*

**public** void register(User u) **throws** ClassNotFoundException, SQLException

{

setConnection();

String query="insert into logininfo values('"+u.getName()+"','"+u.getGender()+"','"+u.getUname()+"','"+u.getPassword()+"','"+u.getChoice()+"')";

int res=stmt.executeUpdate(query);

closeConn();

}

*//Function for fetching data from items table*

**public** ResultSet getItems(User u) **throws** ClassNotFoundException, SQLException

{

setConnection();

String query="select \* from items where gender='"+u.getGender()+"'";

**if**(u.getChoice().equals("newarr"))

query+=" order by itemtype desc";

**else**

query+=" order by itemtype asc, discount desc";

System.out.println(query);

ResultSet rs=stmt.executeQuery(query);

*// while(rs.next())*

*// System.out.println(rs.getString("name"));*

*//*

*// System.out.println(rs);*

**if**(rs.next())

{

**return** rs;

}

closeConn();

**return** **null**;

}

*//Function for fetching data from items table*

**public** ResultSet getItemsByName(String name) **throws** ClassNotFoundException, SQLException

{

setConnection();

String query="select \* from items where name='"+name+"'";

System.out.println(query);

ResultSet rs=stmt.executeQuery(query);

*// while(rs.next())*

*// System.out.println(rs.getString("name"));*

*//*

*// System.out.println(rs);*

**if**(rs.next())

{

**return** rs;

}

closeConn();

**return** **null**;

}

**public** void updatePass(String username, String pass) **throws** ClassNotFoundException, SQLException

{

setConnection();

String query="update logininfo set password='"+pass+"' where uname='"+username+"'";

int res=stmt.executeUpdate(query);

closeConn();

}

}

**Deployment Descriptor: web.xml**

<?xml version="1.0" encoding="UTF-8"?>

**<web-app** xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd" id="WebApp\_ID" version="3.1"**>**

**<display-name>**ShoppingApp**</display-name>**

**<context-param>**

**<param-name>**dburl**</param-name>**

**<param-value>**jdbc:mysql://localhost:3306/shopping**</param-value>**

**</context-param>**

**<context-param>**

**<param-name>**dbuname**</param-name>**

**<param-value>**root**</param-value>**

**</context-param>**

**<context-param>**

**<param-name>**dbpass**</param-name>**

**<param-value></param-value>**

**</context-param>**

**</web-app>**

**DATABASE SCHEMA:**

|  |
| --- |
|  |
| **items Table** |
|  |
| **logininfo Table** |

**OUTPUT:**

|  |
| --- |
|  |
| **Fig. 1** User Registration process |
|  |
| **Fig. 2** User Login process |
|  |
| **Fig. 3** Dashboard After Login |
|  |
| **Fig. 4** After Searching Blue Shirt |

**ASSIGNMENT NUMBER:** 5

**PROBLEM STATEMENT:**

Implement a web application for “Travel Thru Air” using servlets to support the following two use cases

1. A list of current special deals must appear on the home page. Each special deal must display the departure city, the arrival city, and the cost. These special deals are set up by the marketing department and change during the day, so it can’t be static. Special deals are only good for a limited amount of time. 2. A user may search for flights, given a departure city, time and an arrival city. The results must display the departure city, the arrival city, the total cost, and how many legs the flight will have.

State and explain why and where you have used design patterns.

**FLOW DIAGRAM:**

|  |
| --- |
|  |

**CODE:**

**index.jsp**

This is the homepage where the user can search for flights and the special deals are shown.

**<%@** page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"**%>**

**<%@** page import="java.sql.\*" **%>**

**<%@** page import="travel.DAO" **%>**

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

**<html>**

**<head>**

**<link** rel="stylesheet" href="style.css"**>**

**<meta** http-equiv="Content-Type" content="text/html; charset=ISO-8859-1"**>**

*<!-- Latest compiled and minified CSS -->*

**<link** rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css"**>**

*<!-- jQuery library -->*

**<script** src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"**></script>**

*<!-- Latest compiled JavaScript -->*

**<script** src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"**></script>**

**<style>**

.title-bar{

width: 100%;

color: white;

padding: 10px;

font-family: serif;

font-size: 100px;

text-align: center;

background-color: #1abc9c;

}

.error{

text-align: center;

font-size: 30px;

font-family: serif;

}

.travel-form{

margin: auto;

align-content: center;

display: block;

text-align: center;

background-color: #DBDADA;

padding: 20px;

}

.travel-form form{

display: inline-block;

}

.travel-form select, input{

height: 60px;

width: 300px;

}

.search{

background-color: #000faa;

color: white;

font-style: strong;

font-size: large;

font-weight: 300;

font-family: monospace;

border-radius: 100px;

border: solid;

background-image: linear-gradient(to bottom right, #6666ff, #9999ff, #0066ff);

}

.special-deals{

width: 100%;

padding: 10px;

font-family: serif;

font-size: 50px;

text-align: center;

color: white;

font-family: cursive;

background-image: linear-gradient(to bottom right, #0636ff, #0699ff, #f016ff);

}

.offer{

width: 100%;

color: white;

padding: 10px;

text-align: center;

background-color: #1abc9c;

background-image: radial-gradient(red, yellow);

}

.normal-text{

font-size: 25px;

font-family: serif;

}

.discount{

font-size: 70px;

}

.source{

font-size: 60px;

}

/\* Hide the images by default \*/

.slide {

display: none;

}

/\* Next & previous buttons \*/

.prev, .next {

cursor: pointer;

position: absolute;

top: 50%;

width: auto;

margin-top: -75px;

padding: 16px;

color: white;

font-weight: bold;

font-size: 18px;

transition: 0.6s ease;

border-radius: 0 3px 3px 0;

user-select: none;

}

/\* Position the "next button" to the right \*/

.next {

right: 0;

border-radius: 3px 0 0 3px;

}

/\* On hover, add a black background color with a little bit see-through \*/

.prev:hover, .next:hover {

background-color: rgba(0,0,0,0.8);

}

**</style>**

**<script** type="text/javascript"**>**

function timeDiff(target) {

function z(n) {return (n**<10**? '0' : '') + n;}

var timeDiff = target - (new Date());

var hours = timeDiff / 3.6e6 | 0;

var minutes = timeDiff % 3.6e6 / 6e4 | 0;

var seconds = timeDiff % 6e4 / 1e3 | 0;

if (hours<0 || minutes<0 || seconds<0) {

document.getElementById('divBody').style.display='none';

document.getElementById('divExpired').style.display='';

return '<b**>**EXPIRED**</b>**';

}

else {

return '**<b>**' + z(hours) + '**</b>** h **<b>**' + z(minutes) + '**</b>** m **<b>**' + z(seconds) + '**</b>** s';

}

}

var addFunctionOnWindowLoad = function(callback){

if(window.addEventListener){

window.addEventListener('load',callback,false);

}else{

window.attachEvent('onload',callback);

}

}

**</script>**

**<title>**Home**</title>**

**</head>**

**<body>**

*<!-- Title bar -->*

**<div** class="title-bar"**>**

**<h1>** TravelThruAir **</h1>**

**</div>**

**<%**

String url,user,pass;

url="jdbc:mysql://localhost:3306/travel";

System.out.println(url);

user="root";

pass="";

DAO dao=**new** DAO(url,user,pass);

ResultSet rs;

rs=dao.getCities();

**%>**

*<!-- Form -->*

**<div** class="travel-form"**>**

**<form** action="search" id="travel-form" method="post"**>**

**<select** class="src" id="src" name="src"**>**

**<%**

**do**

{**%>**

**<option** value="**<%=**rs.getString("code")**%>**"**><%=**rs.getString("code")**%>**-**<%=**rs.getString("city")**%>** **</option>**

**<%}**

**while**(rs.next());

rs=dao.getCities();

**%>**

**</select>**

**<select** class="dest" id="dest" name="dest"**>**

**<%**

**do**

{**%>**

**<option** value="**<%=**rs.getString("code")**%>**"**><%=**rs.getString("code")**%>**-**<%=**rs.getString("city")**%>** **</option>**

**<%}**

**while**(rs.next());

**%>**

**</select>**

**<input** type="date" id="date" name="date" required=""**>**

**<input** class="search" type="button" value="SEARCH" onclick="validate\_and\_submit()"**>**

**</form>**

**</div>**

**<div** class="special-deals"**>**

SPECIAL DEALS

**</div>**

*<!-- Offers -->*

**<div** class="offers-container"**>**

**<%**

rs=dao.getOffers();

**if**(rs==**null**)

{**%>**

**<div** class="error"**>**No offers right now**</div>**

**<%** }

**else**

{

**do**

{**%>**

**<div** class="offer slide"**>**

*<!-- START TIME:* **<%=**rs.getString("start\_time") **%>**

*END TIME:* **<%=**rs.getString("end\_time") **%>** *-->*

**<div** class="normal-text"**>**Flat**</div>**

**<div** class="discount"**><strong><%=**rs.getString("discount") **%>**%**</strong>** off**</div>**

**<div** class="normal-text"**>**on flights**</div>**

**<div** class="source"**><strong><%=**rs.getString("src") **%>** **</strong>** **<span** style='font-size:75px;'**>&#8594;</span>** **<strong><%=**rs.getString("dest") **%>** **</strong></div>**

**<div** class="normal-text"**><div** id="countdown**<%=**rs.getString("id")**%>**"**></div>**

**<script** language="javaScript"**>**

function doCountDown**<%=**rs.getString("id")**%>**(target) {

document.getElementById('countdown**<%=**rs.getString("id")**%>**').innerHTML = '**<span** style=\"color:white\"**><b>**EXPIRES IN**</b></span>**: ' + timeDiff(target);

var lag = 1020 - (new Date() % 100);

setTimeout(function(){doCountDown**<%=**rs.getString("id")**%>**(target);}, lag);

}

var x**<%=**rs.getString("id")**%>** = function doStart**<%=**rs.getString("id")**%>**() {

//Insert Expiration Date from mySQL into t var

var t="**<%=**rs.getString("end\_time")**%>**".split(/[- :]/);

doCountDown**<%=**rs.getString("id")**%>**(new Date(t[0],t[1]-1,t[2],t[3],t[4],t[5]));

}

addFunctionOnWindowLoad(x**<%=**rs.getString("id")**%>**);

**</script>**

**</div>**

**</div>**

*<!-- Next and previous buttons -->*

**<a** class="prev" onclick="plusSlides(-1)"**>&#10094;</a>**

**<a** class="next" onclick="plusSlides(1)"**>&#10095;</a>**

**<%}**

**while**(rs.next());

}

**%>**

**</div>**

**</body>**

**<script** type="text/javascript"**>**

var today = new Date().toISOString().split('T')[0];

document.getElementsByName("date")[0].setAttribute('min', today);

//Script for slideshow

var slideIndex = 1;

showSlides(slideIndex);

showSlidesAuto();

// Next/previous controls

function plusSlides(n) {

showSlides(slideIndex += n);

}

// Thumbnail image controls

function currentSlide(n) {

showSlides(slideIndex = n);

}

function showSlides(n) {

var i;

var slides = document.getElementsByClassName("slide");

console.log(slides.length);

if (n > slides.length) {slideIndex = 1}

if (n **< 1**) {slideIndex = slides.length}

for (i = 0; i < slides.length; i++) {

slides[i].style.display = "none";

}

slides[slideIndex-1].style.display = "block";

}

function showSlidesAuto() {

var i;

var slides = document.getElementsByClassName("slide");

for (i = 0; i < slides.length; i++) {

slides[i].style.display = "none";

}

slideIndex++;

if (slideIndex **>** slides.length) {slideIndex = 1}

slides[slideIndex-1].style.display = "block";

setTimeout(showSlidesAuto, 5000); // Change image every 2 seconds

}

function validate\_and\_submit(){

var src=document.getElementById("src").value;

var dest=document.getElementById("dest").value;

if(src===dest)

{

//Error

alert("Source and destination city cannot be same");

}

else

document.getElementById("travel-form").submit();

}

**</script>**

**</html>**

**flights.jsp**

This is the page which will show the list of all flights depending on the source destination and the date chosen. It also shows any available discount on flights being shown. The flights can also be filtered by company or Non-stop or more stops.

**<%@** page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"**%>**

**<%@** page import="java.sql.\*" **%>**

**<%@** page import="travel.DAO" **%>**

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

**<html>**

**<head>**

**<meta** http-equiv="Content-Type" content="text/html; charset=ISO-8859-1"**>**

*<!-- Latest compiled and minified CSS -->*

**<link** rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css"**>**

*<!-- jQuery library -->*

**<script** src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"**></script>**

*<!-- Latest compiled JavaScript -->*

**<script** src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"**></script>**

**<style>**

.flight{

width: 100%;

text-align: center;

margin: 20px;

padding: 5px;

border: solid;

}

table{

width: 100%;

}

th {

background-color: #4CAF50;

color: white;

font-size: 15px;

}

th, td {

padding: 15px;

text-align: center;

border-bottom: 1px solid #ddd;

}

tr:nth-child(even) {background-color: #f2f2f2;}

.company{

font-size: 20px;

font-family: serif;

}

.stops{

font-size: 20px;

font-family: serif;

}

.time{

font-size: 25px;

font-family: sans-serif;

font-weight: 100;

}

.city{

font-size: 20px;

font-family: serif;

}

.actual-price{

font-size: 27px;

font-family: sans-serif;

}

.actual-price-discount{

font-size: 23px;

font-family: sans-serif;

text-decoration: line-through;

color: grey;

}

.price-discount{

color: green;

font-size: 25px;

}

.num-stops{

color:blue;

}

.error{

text-align: center;

font-size: 30px;

font-family: serif;

}

.duration{

font-size: 27px;

font-family: sans-serif;

}

.stop-list{

color: grey;

}

.travel-form{

margin: auto;

align-content: center;

display: block;

text-align: center;

background-color: #DBDADA;

padding: 20px;

}

.travel-form form{

display: inline-block;

}

.travel-form select, input{

height: 60px;

width: 300px;

}

**</style>**

**<title>**Flights**</title>**

**</head>**

**<body>**

**<%**

String url,user,pass;

url="jdbc:mysql://localhost:3306/travel";

System.out.println(url);

user="root";

pass="";

DAO dao=**new** DAO(url,user,pass);

ResultSet rs;

rs=dao.getCompanies();

String day=(String)request.getAttribute("day");

String src=(String)request.getAttribute("src");

String dest=(String)request.getAttribute("dest");

ResultSet rs2;

**%>**

**<a** href="index.jsp"**>**HOME**</a>**

*<!-- Form for filtering -->*

**<div** class="travel-form"**>**

**<form** action="filter" id="filter-form" method="post"**>**

**<input** type="hidden" name="day" value="**<%=**day**%>**"**>**

**<input** type="hidden" name="src" value="**<%=**src**%>**"**>**

**<input** type="hidden" name="dest" value="**<%=**dest**%>**"**>**

**<select** class="company" id="company" name="company"**>**

**<option** value="all"**>**All**</option>**

**<%**

**do**

{**%>**

**<option** value="**<%=**rs.getString("company")**%>**"**><%=**rs.getString("company")**%>** **</option>**

**<%}**

**while**(rs.next());

**%>**

**</select>**

**<select** class="stops" id="stops" name="stops"**>**

**<option** value="all"**>**All**</option>**

**<option** value="0"**>**Non-Stop**</option>**

**</select>**

**<input** class="search" type="button" value="SEARCH" onclick="validate\_and\_submit()"**>**

**</form>**

**</div>**

**<div** class="list"**>**

**<%**

rs=**null**;

rs=(ResultSet)request.getAttribute("flights");

**if**(rs==**null**)

{**%>**

**<div** class="error"**>**Sorry!!! No flights found**</div>**

**<%** }

**else**

{**%>**

**<table>**

**<tr>**

**<th>**Flight**</th>**

**<th>**Departure**</th>**

**<th>**Duration**</th>**

**<th>**Arrival**</th>**

**<th>**Price**</th>**

**</tr>**

**<%**

**do**

{

String disc= rs.getString("discount");

String act\_price= rs.getString("price");

int discount=0;

int a\_price=Integer.parseInt(act\_price);

int d\_price=0;

**if**(disc!=**null**)

{

discount=Integer.parseInt(disc);

d\_price=(int)((double)a\_price\*(100.0-(double)discount)/100.0);

}

String disc\_price=Integer.toString(d\_price);

**%>**

**<tr>**

**<td>**

**<div** class="company"**>**

**<strong><%=**rs.getString("company")**%></strong><br>**

**</div>**

**<%=**rs.getString("flight\_num")**%>**

**</td>**

**<td>**

**<div** class="time"**>**

**<%**

String dept\_time=rs.getString("dept\_time");

dept\_time=dept\_time.substring(0,dept\_time.lastIndexOf(':'));

**%>**

**<strong><%=**dept\_time**%></strong><br>**

**</div>**

**<div** class="city"**>**

**<%=**rs.getString("src\_city")**%>**

**</div>**

**</td>**

**<td>**

**<div** class="duration"**>**

**<%**

**if**(rs.getString("dur\_hrs").compareTo("0")!=0)

{**%>**

**<strong><%=**rs.getString("dur\_hrs")**%>**h **</strong>**

**<%}**

**if**(rs.getString("dur\_min").compareTo("0")!=0)

{**%>**

**<strong><%=**rs.getString("dur\_min")**%>**m **</strong>**

**<%}%>**

**</div>**

**<div** class="num-stops"**>**

**<%**

**if**(rs.getString("num\_stops").compareTo("0")==0)

{**%>**

Non stop

**<%}**

**else**

{**%>**

**<%=**rs.getString("num\_stops")**%>** stop(s)

**<div** class="stop-list"**>**

**<%**

rs2=dao.getStops(rs.getString("id"));

**do**{

**if**(Integer.parseInt(rs2.getString("stop\_no"))<Integer.parseInt(rs.getString("num\_stops"))){**%>**

**<%=**rs2.getString("dest\_city")**%><span** style='font-size:15px;'**>&#8594;</span>**

**<%}else**

{**%>**

**<%=**rs2.getString("dest\_city")**%>**

**<%}**}

**while**(rs2.next());

**%>**

**</div>**

**<%}%>**

**</div>**

**</td>**

**<td>**

**<div** class="time"**>**

**<%**

String arr\_time=rs.getString("arr\_time");

arr\_time=arr\_time.substring(0,arr\_time.lastIndexOf(':'));

**%>**

**<strong><%=**arr\_time**%></strong><br>**

**</div>**

**<div** class="city"**>**

**<%=**rs.getString("dest\_city")**%>**

**</div>**

**</td>**

**<%**

**if**(disc==**null**)

{

**%>**

**<td>**

**<div** class="actual-price"**>**

**<strong>&#x20b9;<%=**rs.getString("price")**%></strong>**

**</div>**

**</td>**

**<%** } **else** { **%>**

**<td>**

**<div** class="actual-price-discount"**>**

**&#x20b9;<%=**rs.getString("price")**%>**

**</div>**

**<div** class="actual-price price-discount"**>**

**<strong>&#x20b9;<%=**disc\_price**%></strong>**

**</div>**

**</td>**

**<%}%>**

**</tr>**

**<%}**

**while**(rs.next());

}**%>**

**</table>**

**</div>**

**</body>**

**<script** type="text/javascript"**>**

function validate\_and\_submit(){

// var src=document.getElementById("src").value;

// var dest=document.getElementById("dest").value;

// if(src===dest)

// {

// //Error

// alert("Source and destination city cannot be same");

// }

// else

document.getElementById("filter-form").submit();

}

**</script>**

**</html>**

**SearchFlights.java**

This servlet is responsible for fetching the flights based on source destination and date.

**package** **travel**;

**import** **java.io.IOException**;

**import** **java.sql.ResultSet**;

**import** **java.text.DateFormat**;

**import** **java.text.ParseException**;

**import** **java.text.SimpleDateFormat**;

**import** **java.util.Date**;

**import** **javax.servlet.RequestDispatcher**;

**import** **javax.servlet.ServletException**;

**import** **javax.servlet.annotation.WebServlet**;

**import** **javax.servlet.http.HttpServlet**;

**import** **javax.servlet.http.HttpServletRequest**;

**import** **javax.servlet.http.HttpServletResponse**;

*/\*\**

*\* Servlet implementation class SearchFlights*

*\*/*

@WebServlet(name = "search", urlPatterns = { "/search" })

**public** **class** **SearchFlights** **extends** HttpServlet {

**private** **static** **final** long serialVersionUID = 1L;

**private** String dburl,dbuname,dbpass;

*/\*\**

*\* @see HttpServlet#HttpServlet()*

*\*/*

**public** SearchFlights() {

**super**();

*// TODO Auto-generated constructor stub*

}

**public** void init()

{

dburl=getServletContext().getInitParameter("dburl");

dbuname=getServletContext().getInitParameter("dbuname");

dbpass=getServletContext().getInitParameter("dbpass");

}

*/\*\**

*\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)*

*\*/*

**protected** void doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

*// TODO Auto-generated method stub*

response.getWriter().append("Served at: ").append(request.getContextPath());

}

*/\*\**

*\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)*

*\*/*

**protected** void doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

*// TODO Auto-generated method stub*

*//Get the parameters*

String src=request.getParameter("src");

String dest=request.getParameter("dest");

String date=request.getParameter("date");

System.out.println(date);

DAO dao=**new** DAO(dburl,dbuname,dbpass);

ResultSet rs=**null**;

*// get day of week from date string*

String day;

SimpleDateFormat format1=**new** SimpleDateFormat("yyyy-MM-dd");

Date dt1=**null**;

**try**

{

dt1 = (Date) format1.parse(date);

}

**catch** (ParseException e1) {

*// TODO Auto-generated catch block*

e1.printStackTrace();

}

DateFormat format2=**new** SimpleDateFormat("EEEE");

day=format2.format(dt1);

System.out.println(day);

**try**

{

rs=dao.getFlights(src,dest,day);

}

**catch** (Exception e)

{

*// TODO Auto-generated catch block*

e.printStackTrace();

}

request.setAttribute("flights",rs);

request.setAttribute("day",day);

request.setAttribute("src",src);

request.setAttribute("dest",dest);

*//Then redirect to dashboard*

request.getRequestDispatcher("flights.jsp").forward(request, response);

}

}

**FilterFlights.java**

This is the servlet for filtering the flights on different criteria like company and number of stops.

**package** **travel**;

**import** **java.io.IOException**;

**import** **java.sql.ResultSet**;

**import** **java.text.DateFormat**;

**import** **java.text.ParseException**;

**import** **java.text.SimpleDateFormat**;

**import** **java.util.Date**;

**import** **javax.servlet.RequestDispatcher**;

**import** **javax.servlet.ServletException**;

**import** **javax.servlet.annotation.WebServlet**;

**import** **javax.servlet.http.HttpServlet**;

**import** **javax.servlet.http.HttpServletRequest**;

**import** **javax.servlet.http.HttpServletResponse**;

*/\*\**

*\* Servlet implementation class SearchFlights*

*\*/*

@WebServlet(name = "filter", urlPatterns = { "/filter" })

**public** **class** **FilterFlights** **extends** HttpServlet {

**private** **static** **final** long serialVersionUID = 1L;

**private** String dburl,dbuname,dbpass;

*/\*\**

*\* @see HttpServlet#HttpServlet()*

*\*/*

**public** FilterFlights() {

**super**();

*// TODO Auto-generated constructor stub*

}

**public** void init()

{

dburl=getServletContext().getInitParameter("dburl");

dbuname=getServletContext().getInitParameter("dbuname");

dbpass=getServletContext().getInitParameter("dbpass");

}

*/\*\**

*\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)*

*\*/*

**protected** void doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

*// TODO Auto-generated method stub*

response.getWriter().append("Served at: ").append(request.getContextPath());

}

*/\*\**

*\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)*

*\*/*

**protected** void doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

*// TODO Auto-generated method stub*

*//Get the parameters*

String src=request.getParameter("src");

String dest=request.getParameter("dest");

String day=request.getParameter("day");

String company=request.getParameter("company");

String stops=request.getParameter("stops");

DAO dao=**new** DAO(dburl,dbuname,dbpass);

ResultSet rs=**null**;

System.out.println(day);

**try**

{

rs=dao.filterFlights(src,dest,company,stops,day);

}

**catch** (Exception e)

{

*// TODO Auto-generated catch block*

e.printStackTrace();

}

request.setAttribute("flights",rs);

request.setAttribute("day",day);

request.setAttribute("src",src);

request.setAttribute("dest",dest);

*//Then redirect to dashboard*

request.getRequestDispatcher("flights.jsp").forward(request, response);;

}

}

**DAO.java**

This Java bean is responsible for interacting with the database. It takes care of all database operations- connecting with the database, querying the database and returning the required data to the servlets.

**package** **travel**;

**import** **java.sql.\***;

**public** **class** **DAO** {

**private** Connection con;

**private** Statement stmt;

**private** String url,uname,password;

**public** DAO(String url, String uname, String password)

{

**this**.url=url;

**this**.uname=uname;

**this**.password=password;

}

*//Function to establish the connection*

**private** void setConnection() **throws** ClassNotFoundException, SQLException

{

Class.forName("com.mysql.jdbc.Driver");

con=DriverManager.getConnection(url,uname,password);

stmt=con.createStatement();

}

*//Function to close the connection*

**private** void closeConn() **throws** SQLException

{

con.close();

}

*//Function for getting cities*

**public** ResultSet getCities() **throws** ClassNotFoundException, SQLException

{

setConnection();

String query="select \* from cities";

ResultSet rs=stmt.executeQuery(query);

**if**(rs.next())

{

**return** rs;

}

closeConn();

**return** **null**;

}

*//Function for getting flights*

**public** ResultSet getFlights(String src, String dest, String day) **throws** ClassNotFoundException, SQLException

{

setConnection();

String query1="(select \* from flights where src\_city=\'"+src+"\' and dest\_city=\'"+dest+"\' and day=\'"+day+"\')";

String query2="(select \* from offers where CURRENT\_TIMESTAMP between start\_time and end\_time)";

String query="select \* from "+query1+" f left join "+query2+" o on f.src\_city=o.src and f.dest\_city=o.dest";

System.out.println(query);

ResultSet rs=stmt.executeQuery(query);

**if**(rs.next())

{

**return** rs;

}

closeConn();

**return** **null**;

}

*//Function for getting flights*

**public** ResultSet getOffers() **throws** ClassNotFoundException, SQLException

{

setConnection();

String query="select \* from offers where CURRENT\_TIMESTAMP between offers.start\_time and offers.end\_time";

System.out.println(query);

ResultSet rs=stmt.executeQuery(query);

**if**(rs.next())

{

**return** rs;

}

closeConn();

**return** **null**;

}

*//Function for getting flights*

**public** ResultSet getCompanies() **throws** ClassNotFoundException, SQLException

{

setConnection();

String query="select distinct(company) as company from flights";

System.out.println(query);

ResultSet rs=stmt.executeQuery(query);

**if**(rs.next())

{

**return** rs;

}

closeConn();

**return** **null**;

}

*//Function for getting flights*

**public** ResultSet filterFlights(String src, String dest, String company, String stops, String day) **throws** ClassNotFoundException, SQLException

{

setConnection();

String query1="select \* from flights where src\_city=\'"+src+"\' and dest\_city=\'"+dest+"\' and day=\'"+day+"\'";

**if**(company.compareToIgnoreCase("all")!=0)

query1+=" and company='"+company+"'";

**if**(stops.compareToIgnoreCase("all")!=0)

query1+=" and num\_stops='"+stops+"'";

String query2="(select \* from offers where CURRENT\_TIMESTAMP between start\_time and end\_time)";

String query="select \* from ("+query1+") f left join "+query2+" o on f.src\_city=o.src and f.dest\_city=o.dest";

System.out.println(query);

ResultSet rs=stmt.executeQuery(query);

**if**(rs.next())

{

**return** rs;

}

closeConn();

**return** **null**;

}

*//Function for getting flights*

**public** ResultSet getStops(String flight\_id) **throws** ClassNotFoundException, SQLException

{

setConnection();

String query="select \* from flight\_stops where flight\_id='"+flight\_id+"' order by stop\_no";

System.out.println(query);

ResultSet rs=stmt.executeQuery(query);

**if**(rs.next())

{

**return** rs;

}

closeConn();

**return** **null**;

}

}

**Deployment Descriptor: web.xml**

<?xml version="1.0" encoding="UTF-8"?>

**<web-app** xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd" id="WebApp\_ID" version="3.1"**>**

**<display-name>**travel**</display-name>**

**<context-param>**

**<param-name>**dburl**</param-name>**

**<param-value>**jdbc:mysql://localhost:3306/travel**</param-value>**

**</context-param>**

**<context-param>**

**<param-name>**dbuname**</param-name>**

**<param-value>**root**</param-value>**

**</context-param>**

**<context-param>**

**<param-name>**dbpass**</param-name>**

**<param-value></param-value>**

**</context-param>**

**</web-app>**

**DATABASE SCHEMA**

|  |
| --- |
|  |
| **cities table** |
|  |
| **flights table** |
|  |
| **flight\_stops table** |
|  |
| **offers table** |

**OUTPUT:**

|  |
| --- |
|  |
| **Fig 1.** Home page |
|  |
| **Fig 2.** Flight search page |
|  |
| **Fig 3.** Filtered flights |

**DISCUSSION:**

1. Design patterns have been used to develop the application.
2. The main design pattern used here is Data Access Object or DAO. A separate class has been designed which interacts with the database and fetches data from it. The servlets only create objects of the DAO class and calls its respective methods. No servlet directly interacts with the database. This allows loose coupling. In case there is a change in database query or the database platform there is no need to change the codes in the servlets. For every database platform a separate DAO can be defined and worked with.
3. The various components of the design pattern can be classified as :

• **Business Object:** The jsp pages, i.e., frontend

• **Data Acess Object:** The DAO class

• **Transfer Object:** The Servlet classes

• **Data Source:** 'travel' database

**ASSIGNMENT NUMBER:** 6

**PROBLEM STATEMENT**

Design a Mail server and client program that implements SMTP and POP3 protocols for sending and receiving emails. An acknowledgement is sent to the sender whenever the recipient views the email. Recipient should find email notifications whenever (s)he logs in. Two possible use cases are shown above. For the details of POP3 and SMTP protocols you can refer to RFC 1939 (http://tools.ietf.org/html/rfc1939) and RFC 821 (http://tools.ietf.org/html/rfc821), respectively

Observe corresponding packet flow using Wireshark and report on traffic analysis.

**CODE:**

**index.jsp**

This is the homepage that shows the inbox of the user

**<%@** page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"**%>**

**<%@** page import="com.mailapp.FetchMail" **%>**

**<%@** page import="javax.mail.\*" **%>**

<!DOCTYPE html>

**<html>**

**<head>**

**<meta** charset="ISO-8859-1"**>**

**<title>**Inbox**</title>**

**</head>**

**<body>**

**<%**

ServletContext context = pageContext.getServletContext();

String from=(String)request.getServletContext().getInitParameter("fromMail");

String password=(String)request.getServletContext().getInitParameter("password");

FetchMail fm=**new** FetchMail("pop.gmail.com","pop3",from,password);

Message messages[]=fm.fetchMails();

**%>**

**<a** href="SendMail.html"**>**Compose Mail**</a>**

**<h1>**INBOX**</h1>**

**<div** class="table-mess"**>**

**<table>**

**<tr>**

**<th>**From**</th>**

**<th>**Subject**</th>**

**<th>**Message**</th>**

**</tr>**

**<%**

**for** (int i = 0; i < messages.length; i++)

{

Message message = messages[i];

**%>**

**<tr>**

**<td><%=**message.getFrom()[0]**%></td>**

**<td><%=**message.getSubject()**%></td>**

**<td><%=**message.getContent().toString()**%></td>**

**</tr>**

**<%** }

**%>**

**</table>**

**</div>**

**</body>**

**</html>**

**SendMail.html**

This is the form for composing a mail.

<!DOCTYPE html>

<**html**>

<**head**>

<**meta** charset="ISO-8859-1">

<**title**>Compose Mail</**title**>

</**head**>

<**body**>

<**h2**>Compose Mail</**h2**>

<**div** class="mail-form">

<**form** id="email-form" action="send" method=post>

To: <**input** type="email" name="recipient" required=""><**br**>

Sub: <**input** type="text" name="subject"><**br**>

Message: <**input** type="text" name="message" required=""><**br**>

<**input** type="button" value="Send Mail" onclick="validate\_and\_submit()">

</**form**>

</**div**>

</**body**>

<**script** type="text/javascript">

**function** validate\_and\_submit(){

*//May insert validation here*

document.getElementById("email-form").submit();

}

</**script**>

</**html**>

**FetchMail.java**

This servlet is responsible for fetching the email from the inbox of the user.

**package** **com.mailapp**;

**import** **java.util.Properties**;

**import** **javax.mail.Folder**;

**import** **javax.mail.Message**;

**import** **javax.mail.PasswordAuthentication**;

**import** **javax.mail.Session**;

**import** **javax.mail.Store**;

**public** **class** **FetchMail**

{

**private** String host, storeType, user, password, from;

**public** FetchMail(String host, String storeType, String user, String password)

{

**this**.host=host;

**this**.storeType=storeType;

**this**.user=user;

**this**.password=password;

**this**.from=from;

}

*//Function to fetch the mail*

**public** Session getSession()

{

*//create properties field*

Properties properties = **new** Properties();

properties.put("mail.pop3.host", host);

properties.put("mail.pop3.port", "995");

properties.put("mail.pop3.starttls.enable", "true");

*//get Session*

Session session = Session.getDefaultInstance(properties);

**return** session;

}

**public** Message[] fetchMails()

{

Session emailSession=getSession();

Message[] messages = **null**;

**try**

{

*//create the POP3 store object and connect with the pop server*

Store store = emailSession.getStore("pop3s");

store.connect(host, user, password);

*//create the folder object and open it*

Folder emailFolder = store.getFolder("INBOX");

emailFolder.open(Folder.READ\_ONLY);

*// retrieve the messages from the folder in an array and print it*

messages = emailFolder.getMessages();

}

**catch**(Exception e)

{

e.printStackTrace();

}

**return** messages;

}

}

**SendServlet.java**

This servlet is responsible for sending an email.

**package** **com.mailapp**;

**import** **java.io.IOException**;

**import** **java.util.Properties**;

**import** **javax.mail.\***;

**import** **javax.mail.internet.\***;

**import** **javax.servlet.ServletException**;

**import** **javax.servlet.annotation.WebServlet**;

**import** **javax.servlet.http.HttpServlet**;

**import** **javax.servlet.http.HttpServletRequest**;

**import** **javax.servlet.http.HttpServletResponse**;

*/\*\**

*\* Servlet implementation class SendServlet*

*\*/*

@WebServlet(name = "send", urlPatterns = { "/send" })

**public** **class** **SendServlet** **extends** HttpServlet {

**private** **static** **final** long serialVersionUID = 1L;

*/\*\**

*\* Default constructor.*

*\*/*

**public** SendServlet() {

*// TODO Auto-generated constructor stub*

}

*/\*\**

*\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)*

*\*/*

**protected** void doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

*// TODO Auto-generated method stub*

response.getWriter().append("Served at: ").append(request.getContextPath());

}

*/\*\**

*\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)*

*\*/*

**protected** void doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

*// TODO Auto-generated method stub*

doGet(request, response);

String sendTo=(String)request.getParameter("recipient");

String sub=(String)request.getParameter("subject");

String mess=(String)request.getParameter("message");

String from=(String)request.getServletContext().getInitParameter("fromMail");

String password=(String)request.getServletContext().getInitParameter("password");

System.out.println(sendTo);

*//Send the mail*

sendMail(from, password, sendTo, sub, mess);

response.sendRedirect("index.jsp");

}

**private** void sendMail(String from, String password, String sendTo, String sub, String mess) {

*// TODO Auto-generated method stub*

*//Get properties object*

Properties props = **new** Properties();

props.put("mail.smtp.host", "smtp.gmail.com");

props.put("mail.smtp.socketFactory.port", "465");

props.put("mail.smtp.socketFactory.class","javax.net.ssl.SSLSocketFactory");

props.put("mail.smtp.auth", "true");

props.put("mail.smtp.port", "465");

*//get Session*

Session session2 = Session.getInstance(props,

**new** javax.mail.Authenticator() {

**protected** PasswordAuthentication getPasswordAuthentication() {

**return** **new** PasswordAuthentication(from,password);

}

});

**try**

{

MimeMessage message = **new** MimeMessage(session2);

message.addRecipient(Message.RecipientType.TO,**new** InternetAddress(sendTo));

message.setSubject(sub);

message.setText(mess);

*//send message*

Transport.send(message);

System.out.println("message sent successfully");

}

**catch** (Exception e)

{

e.printStackTrace();

}

}

}

**OUTPUT:**

|  |
| --- |
|  |
| **Fig 1.** Home page showing the inbox |
|  |
| **Fig 2.** Form for composing mail |

**ASSIGNMENT NUMBER:** 7

**PROBLEM STATEMENT:**

Implement a web application for “Travel Thru Air” based on Spring MVC framework to support any of the following two use cases

1. A list of current special deals must appear on the home page. Each special deal must display the departure city, the arrival city, and the cost. These special deals are set up by the marketing department and change during the day, so it can’t be static. Special deals are only good for a limited amount of time. 2. A user may search for flights, given a departure city, time and an arrival city. The results must display the departure city, the arrival city, the total cost, and how many legs the flight will have.

Provide a login controller for user login facility. The solution should reflect each layer of the spring framework.

**FLOW DIAGRAM:**

|  |
| --- |
|  |

**CODE:**

**index.jsp**

This is the login page where the user can register and login to search for flights.

**<%@** page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"**%>**

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

**<html>**

**<head>**

**<meta** charset="ISO-8859-1"**>**

**<title>**Login**</title>**

**<style** type="text/css"**>**

body{

background-color: #a2a2a2;

}

#login-page{

width:360px;

padding:10% 0 0;

margin:auto;

}

#formAll{

position: relative;

z-index: 100;

background: #ffffff;

max-width: 360px;

margin: 0 auto 100px;

padding: 45px;

text-align: center;

}

select, input{

font-family: "Times New Roman", serif;

outline:1;

background: #f2f2f2;

width: 100%;

border: 0;

margin: 0 0 15px;

padding: 15px;

box-sizing: border-box;

font-size: 14p;

}

#submit{

text-transform: uppercase;

padding: 15px;

color: #FFFFFF;

background: #000000;

cursor: pointer;

}

.register-Form{

display: none;

}

.forgot-pass-form{

display: none;

}

#duplicateUname{

display:none;

color: red;

}

**</style>**

**<script** src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"**>**

**</script>**

**</head>**

**<body>**

**<div** id="login-page"**>**

**<div** id="form"**>**

**<form** id="loginform formAll" class="loginform" action="login" method="post"**>**

**<input** type="text" name="username" required="" placeholder="Enter your Username"**><br>**

**<input** type="password" name="password" required="" placeholder="Enter your password"**><br>**

**<input** type="submit" id="submit" name="submit" value="Login"**>**

**<p** class="notReg"**>** Not registered? **<a** class="reg-here" href="#"**>**Register Here **</a></p>**

**</form>**

**<form** id="reg formAll" class="register-Form" action="register" method="post"**>**

**<input** type="text" name="name" required="" placeholder="Enter your name"**><br>**

**<div** id="duplicateUname"**>**Sorry, this username is already taken**</div>**

**<input** type="text" name="username" placeholder="Enter your Username"**><br>**

**<input** type="password" name="password" placeholder="Enter your password"**><br>**

**<input** type="submit" id="submit" name="submit" value="Register"**>**

**<p** class="alreadyReg"**>** Already registered? **<a** class="login-here" href="#"**>**Login Here **</a></p>**

**</form>**

**<form** id="forgot formAll" class="forgot-pass-form" action="forgot" method="post"**>**

**<input** type="text" name="username" placeholder="Enter your username"**>**

**<input** type="password" name="password" placeholder="Enter new password"**>**

**<input** type="submit" name="submit" id="submit" value="Change Password"**>**

**</form>**

**</div>**

**</div>**

**</body>**

**<script** type="text/javascript"**>**

$(document).ready(function(){

$('.reg-here').click(function(){

$('.loginform').hide();

$('.forgot-pass-form').hide();

$('.register-Form').show();

});

$('.forgot-pass').click(function(){

$('.loginform').hide();

$('.forgot-pass-form').show();

$('.register-Form').hide();

})

$('.login-here').click(function(){

$('.loginform').show();

$('.forgot-pass-form').hide();

$('.register-Form').hide();

});

});

**</script>**

**</html>**

**searchFlight.jsp**

This is the page where the user can search for flights and the special deals are shown after the user has logged in.

**<%@** page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"**%>**

**<%@** page import="java.sql.\*" **%>**

**<%@** page import="com.flight.DAO" **%>**

**<%@** page import="com.flight.User" **%>**

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

**<html>**

**<head>**

**<link** rel="stylesheet" href="style.css"**>**

**<meta** http-equiv="Content-Type" content="text/html; charset=ISO-8859-1"**>**

*<!-- Latest compiled and minified CSS -->*

**<link** rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css"**>**

*<!-- jQuery library -->*

**<script** src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"**></script>**

*<!-- Latest compiled JavaScript -->*

**<script** src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"**></script>**

**<style>**

.title-bar{

width: 100%;

color: white;

padding: 10px;

font-family: serif;

font-size: 100px;

text-align: center;

background-color: #1abc9c;

}

.error{

text-align: center;

font-size: 30px;

font-family: serif;

}

.travel-form{

margin: auto;

align-content: center;

display: block;

text-align: center;

background-color: #DBDADA;

padding: 20px;

}

.travel-form form{

display: inline-block;

}

.travel-form select, input{

height: 60px;

width: 300px;

}

.search{

background-color: #000faa;

color: white;

font-style: strong;

font-size: large;

font-weight: 300;

font-family: monospace;

border-radius: 100px;

border: solid;

background-image: linear-gradient(to bottom right, #6666ff, #9999ff, #0066ff);

}

.special-deals{

width: 100%;

padding: 10px;

font-family: serif;

font-size: 50px;

text-align: center;

color: white;

font-family: cursive;

background-image: linear-gradient(to bottom right, #0636ff, #0699ff, #f016ff);

}

.offer{

width: 100%;

color: white;

padding: 10px;

text-align: center;

background-color: #1abc9c;

background-image: radial-gradient(red, yellow);

}

.normal-text{

font-size: 25px;

font-family: serif;

}

.discount{

font-size: 70px;

}

.source{

font-size: 60px;

}

/\* Hide the images by default \*/

.slide {

display: none;

}

/\* Next & previous buttons \*/

.prev, .next {

cursor: pointer;

position: absolute;

top: 50%;

width: auto;

margin-top: -75px;

padding: 16px;

color: white;

font-weight: bold;

font-size: 18px;

transition: 0.6s ease;

border-radius: 0 3px 3px 0;

user-select: none;

}

/\* Position the "next button" to the right \*/

.next {

right: 0;

border-radius: 3px 0 0 3px;

}

/\* On hover, add a black background color with a little bit see-through \*/

.prev:hover, .next:hover {

background-color: rgba(0,0,0,0.8);

}

**</style>**

**<script** type="text/javascript"**>**

function timeDiff(target) {

function z(n) {return (n**<10**? '0' : '') + n;}

var timeDiff = target - (new Date());

var hours = timeDiff / 3.6e6 | 0;

var minutes = timeDiff % 3.6e6 / 6e4 | 0;

var seconds = timeDiff % 6e4 / 1e3 | 0;

if (hours<0 || minutes<0 || seconds<0) {

document.getElementById('divBody').style.display='none';

document.getElementById('divExpired').style.display='';

return '<b**>**EXPIRED**</b>**';

}

else {

return '**<b>**' + z(hours) + '**</b>** h **<b>**' + z(minutes) + '**</b>** m **<b>**' + z(seconds) + '**</b>** s';

}

}

var addFunctionOnWindowLoad = function(callback){

if(window.addEventListener){

window.addEventListener('load',callback,false);

}else{

window.attachEvent('onload',callback);

}

}

**</script>**

**<title>**Home**</title>**

**</head>**

**<body>**

*<!-- Title bar -->*

**<div** class="title-bar"**>**

**<h1>** TravelThruAir **</h1>**

**</div>**

**<%**

response.setHeader("Cache-Control","no-cache, no-store, must-revalidate");

System.out.println((User)session.getAttribute("user"));

*//Session check*

**if**(((User)session.getAttribute("user")).getUname()==**null**)

{

System.out.println("Hi");

response.sendRedirect("index.jsp");

**return**;

}

**%>**

**<%**

String url,user,pass;

url="jdbc:mysql://localhost:3306/travel";

System.out.println(url);

user="root";

pass="";

DAO dao=**new** DAO(url,user,pass);

ResultSet rs;

rs=dao.getCities();

**%>**

**<ul>**

**<li** class="logout"**><a** href="index.jsp"**>**Home**</a></li>**

**<li** class="logout"**><a** href="**<%=**request.getContextPath()**%>**/logout"**>**Logout**</a></li>**

**<li** class="nameDisp"**>**Hi, **<%=**((User)(session.getAttribute("user"))).getName()**%></li>**

**</ul>**

*<!-- Form -->*

**<div** class="travel-form"**>**

**<form** action="search" id="travel-form" method="post"**>**

**<select** class="src" id="src" name="src"**>**

**<%**

**do**

{**%>**

**<option** value="**<%=**rs.getString("code")**%>**"**><%=**rs.getString("code")**%>**-**<%=**rs.getString("city")**%>** **</option>**

**<%}**

**while**(rs.next());

rs=dao.getCities();

**%>**

**</select>**

**<select** class="dest" id="dest" name="dest"**>**

**<%**

**do**

{**%>**

**<option** value="**<%=**rs.getString("code")**%>**"**><%=**rs.getString("code")**%>**-**<%=**rs.getString("city")**%>** **</option>**

**<%}**

**while**(rs.next());

**%>**

**</select>**

**<input** type="date" id="date" name="date" required=""**>**

**<input** class="search" type="button" value="SEARCH" onclick="validate\_and\_submit()"**>**

**</form>**

**</div>**

**<div** class="special-deals"**>**

SPECIAL DEALS

**</div>**

*<!-- Offers -->*

**<div** class="offers-container"**>**

**<%**

rs=dao.getOffers();

**if**(rs==**null**)

{**%>**

**<div** class="error"**>**No offers right now**</div>**

**<%** }

**else**

{

**do**

{**%>**

**<div** class="offer slide"**>**

*<!-- START TIME:* **<%=**rs.getString("start\_time") **%>**

*END TIME:* **<%=**rs.getString("end\_time") **%>** *-->*

**<div** class="normal-text"**>**Flat**</div>**

**<div** class="discount"**><strong><%=**rs.getString("discount") **%>**%**</strong>** off**</div>**

**<div** class="normal-text"**>**on flights**</div>**

**<div** class="source"**><strong><%=**rs.getString("src") **%>** **</strong>** **<span** style='font-size:75px;'**>&#8594;</span>** **<strong><%=**rs.getString("dest") **%>** **</strong></div>**

**<div** class="normal-text"**><div** id="countdown**<%=**rs.getString("id")**%>**"**></div>**

**<script** language="javaScript"**>**

function doCountDown**<%=**rs.getString("id")**%>**(target) {

document.getElementById('countdown**<%=**rs.getString("id")**%>**').innerHTML = '**<span** style=\"color:white\"**><b>**EXPIRES IN**</b></span>**: ' + timeDiff(target);

var lag = 1020 - (new Date() % 100);

setTimeout(function(){doCountDown**<%=**rs.getString("id")**%>**(target);}, lag);

}

var x**<%=**rs.getString("id")**%>** = function doStart**<%=**rs.getString("id")**%>**() {

//Insert Expiration Date from mySQL into t var

var t="**<%=**rs.getString("end\_time")**%>**".split(/[- :]/);

doCountDown**<%=**rs.getString("id")**%>**(new Date(t[0],t[1]-1,t[2],t[3],t[4],t[5]));

}

addFunctionOnWindowLoad(x**<%=**rs.getString("id")**%>**);

**</script>**

**</div>**

**</div>**

*<!-- Next and previous buttons -->*

**<a** class="prev" onclick="plusSlides(-1)"**>&#10094;</a>**

**<a** class="next" onclick="plusSlides(1)"**>&#10095;</a>**

**<%}**

**while**(rs.next());

}

**%>**

**</div>**

**</body>**

**<script** type="text/javascript"**>**

var today = new Date().toISOString().split('T')[0];

document.getElementsByName("date")[0].setAttribute('min', today);

//Script for slideshow

var slideIndex = 1;

showSlides(slideIndex);

showSlidesAuto();

// Next/previous controls

function plusSlides(n) {

showSlides(slideIndex += n);

}

// Thumbnail image controls

function currentSlide(n) {

showSlides(slideIndex = n);

}

function showSlides(n) {

var i;

var slides = document.getElementsByClassName("slide");

console.log(slides.length);

if (n > slides.length) {slideIndex = 1}

if (n **< 1**) {slideIndex = slides.length}

for (i = 0; i < slides.length; i++) {

slides[i].style.display = "none";

}

slides[slideIndex-1].style.display = "block";

}

function showSlidesAuto() {

var i;

var slides = document.getElementsByClassName("slide");

for (i = 0; i < slides.length; i++) {

slides[i].style.display = "none";

}

slideIndex++;

if (slideIndex **>** slides.length) {slideIndex = 1}

slides[slideIndex-1].style.display = "block";

setTimeout(showSlidesAuto, 5000); // Change image every 2 seconds

}

function validate\_and\_submit(){

var src=document.getElementById("src").value;

var dest=document.getElementById("dest").value;

if(src===dest)

{

//Error

alert("Source and destination city cannot be same");

}

else

document.getElementById("travel-form").submit();

}

**</script>**

**</html>**

**flights.jsp**

This is the page which will show the list of all flights depending on the source destination and the date chosen. It also shows any available discount on flights being shown. The flights can also be filtered by company or Non-stop or more stops.

**<%@** page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"**%>**

**<%@** page import="java.sql.\*" **%>**

**<%@** page import="com.flight.\*" **%>**

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

**<html>**

**<head>**

**<meta** http-equiv="Content-Type" content="text/html; charset=ISO-8859-1"**>**

*<!-- Latest compiled and minified CSS -->*

**<link** rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css"**>**

*<!-- jQuery library -->*

**<script** src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"**></script>**

*<!-- Latest compiled JavaScript -->*

**<script** src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"**></script>**

**<style>**

.flight{

width: 100%;

text-align: center;

margin: 20px;

padding: 5px;

border: solid;

}

table{

width: 100%;

}

th {

background-color: #4CAF50;

color: white;

font-size: 15px;

}

th, td {

padding: 15px;

text-align: center;

border-bottom: 1px solid #ddd;

}

tr:nth-child(even) {background-color: #f2f2f2;}

.company{

font-size: 20px;

font-family: serif;

}

.stops{

font-size: 20px;

font-family: serif;

}

.time{

font-size: 25px;

font-family: sans-serif;

font-weight: 100;

}

.city{

font-size: 20px;

font-family: serif;

}

.actual-price{

font-size: 27px;

font-family: sans-serif;

}

.actual-price-discount{

font-size: 23px;

font-family: sans-serif;

text-decoration: line-through;

color: grey;

}

.price-discount{

color: green;

font-size: 25px;

}

.num-stops{

color:blue;

}

.error{

text-align: center;

font-size: 30px;

font-family: serif;

}

.duration{

font-size: 27px;

font-family: sans-serif;

}

.stop-list{

color: grey;

}

.travel-form{

margin: auto;

align-content: center;

display: block;

text-align: center;

background-color: #DBDADA;

padding: 20px;

}

.travel-form form{

display: inline-block;

}

.travel-form select, input{

height: 60px;

width: 300px;

}

**</style>**

**<title>**Flights**</title>**

**</head>**

**<body>**

**<%**

response.setHeader("Cache-Control","no-cache, no-store, must-revalidate");

System.out.println((User)session.getAttribute("user"));

*//Session check*

**if**(((User)session.getAttribute("user")).getUname()==**null**)

{

System.out.println("Hi");

response.sendRedirect("index.jsp");

**return**;

}

**%>**

**<%**

String url,user,pass;

url="jdbc:mysql://localhost:3306/travel";

System.out.println(url);

user="root";

pass="";

DAO dao=**new** DAO(url,user,pass);

ResultSet rs;

rs=dao.getCompanies();

String day=(String)request.getAttribute("day");

String src=(String)request.getAttribute("src");

String dest=(String)request.getAttribute("dest");

ResultSet rs2;

**%>**

**<ul>**

**<li** class="logout"**><a** href="index.jsp"**>**Home**</a></li>**

**<li** class="logout"**><a** href="**<%=**request.getContextPath()**%>**/logout"**>**Logout**</a></li>**

**<li** class="logout"**>**Hi, **<%=**((User)(session.getAttribute("user"))).getName()**%></li>**

**</ul>**

*<!-- Form for filtering -->*

**<div** class="travel-form"**>**

**<form** action="filter" id="filter-form" method="post"**>**

**<input** type="hidden" name="day" value="**<%=**day**%>**"**>**

**<input** type="hidden" name="src" value="**<%=**src**%>**"**>**

**<input** type="hidden" name="dest" value="**<%=**dest**%>**"**>**

**<select** class="company" id="company" name="company"**>**

**<option** value="all"**>**All**</option>**

**<%**

**do**

{**%>**

**<option** value="**<%=**rs.getString("company")**%>**"**><%=**rs.getString("company")**%>** **</option>**

**<%}**

**while**(rs.next());

**%>**

**</select>**

**<select** class="stops" id="stops" name="stops"**>**

**<option** value="all"**>**All**</option>**

**<option** value="0"**>**Non-Stop**</option>**

**</select>**

**<input** class="search" type="button" value="SEARCH" onclick="validate\_and\_submit()"**>**

**</form>**

**</div>**

**<div** class="list"**>**

**<%**

rs=**null**;

rs=(ResultSet)request.getAttribute("flights");

**if**(rs==**null**)

{**%>**

**<div** class="error"**>**Sorry!!! No flights found**</div>**

**<%** }

**else**

{**%>**

**<table>**

**<tr>**

**<th>**Flight**</th>**

**<th>**Departure**</th>**

**<th>**Duration**</th>**

**<th>**Arrival**</th>**

**<th>**Price**</th>**

**</tr>**

**<%**

**do**

{

String disc= rs.getString("discount");

String act\_price= rs.getString("price");

int discount=0;

int a\_price=Integer.parseInt(act\_price);

int d\_price=0;

**if**(disc!=**null**)

{

discount=Integer.parseInt(disc);

d\_price=(int)((double)a\_price\*(100.0-(double)discount)/100.0);

}

String disc\_price=Integer.toString(d\_price);

**%>**

**<tr>**

**<td>**

**<div** class="company"**>**

**<strong><%=**rs.getString("company")**%></strong><br>**

**</div>**

**<%=**rs.getString("flight\_num")**%>**

**</td>**

**<td>**

**<div** class="time"**>**

**<%**

String dept\_time=rs.getString("dept\_time");

dept\_time=dept\_time.substring(0,dept\_time.lastIndexOf(':'));

**%>**

**<strong><%=**dept\_time**%></strong><br>**

**</div>**

**<div** class="city"**>**

**<%=**rs.getString("src\_city")**%>**

**</div>**

**</td>**

**<td>**

**<div** class="duration"**>**

**<%**

**if**(rs.getString("dur\_hrs").compareTo("0")!=0)

{**%>**

**<strong><%=**rs.getString("dur\_hrs")**%>**h **</strong>**

**<%}**

**if**(rs.getString("dur\_min").compareTo("0")!=0)

{**%>**

**<strong><%=**rs.getString("dur\_min")**%>**m **</strong>**

**<%}%>**

**</div>**

**<div** class="num-stops"**>**

**<%**

**if**(rs.getString("num\_stops").compareTo("0")==0)

{**%>**

Non stop

**<%}**

**else**

{**%>**

**<%=**rs.getString("num\_stops")**%>** stop(s)

**<div** class="stop-list"**>**

**<%**

rs2=dao.getStops(rs.getString("id"));

**do**{

**if**(Integer.parseInt(rs2.getString("stop\_no"))<Integer.parseInt(rs.getString("num\_stops"))){**%>**

**<%=**rs2.getString("dest\_city")**%><span** style='font-size:15px;'**>&#8594;</span>**

**<%}else**

{**%>**

**<%=**rs2.getString("dest\_city")**%>**

**<%}**}

**while**(rs2.next());

**%>**

**</div>**

**<%}%>**

**</div>**

**</td>**

**<td>**

**<div** class="time"**>**

**<%**

String arr\_time=rs.getString("arr\_time");

arr\_time=arr\_time.substring(0,arr\_time.lastIndexOf(':'));

**%>**

**<strong><%=**arr\_time**%></strong><br>**

**</div>**

**<div** class="city"**>**

**<%=**rs.getString("dest\_city")**%>**

**</div>**

**</td>**

**<%**

**if**(disc==**null**)

{

**%>**

**<td>**

**<div** class="actual-price"**>**

**<strong>&#x20b9;<%=**rs.getString("price")**%></strong>**

**</div>**

**</td>**

**<%** } **else** { **%>**

**<td>**

**<div** class="actual-price-discount"**>**

**&#x20b9;<%=**rs.getString("price")**%>**

**</div>**

**<div** class="actual-price price-discount"**>**

**<strong>&#x20b9;<%=**disc\_price**%></strong>**

**</div>**

**</td>**

**<%}%>**

**</tr>**

**<%}**

**while**(rs.next());

}**%>**

**</table>**

**</div>**

**</body>**

**<script** type="text/javascript"**>**

function validate\_and\_submit(){

// var src=document.getElementById("src").value;

// var dest=document.getElementById("dest").value;

// if(src===dest)

// {

// //Error

// alert("Source and destination city cannot be same");

// }

// else

document.getElementById("filter-form").submit();

}

**</script>**

**</html>**

**error.jsp**

This is the page where any error related to login and registration are shown.

**<%@** page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"**%>**

<!DOCTYPE html>

**<html>**

**<head>**

**<meta** charset="ISO-8859-1"**>**

**<title>**Insert title here**</title>**

**</head>**

**<body>**

**<%**

String str=(String)request.getAttribute("msg");

System.out.println(str);

**%>**

**<h1><%=**str **%></h1>**

**</body>**

**</html>**

**User.java**

This is the Java class that stores the user details.

**package** **com.flight**;

**public** **class** **User** {

**private** String name,uname,password;

**public** User(String name,String uname, String password)

{

**this**.name=name;

**this**.uname=uname;

**this**.password=password;

}

**public** String getName() {

**return** name;

}

**public** String getUname() {

**return** uname;

}

**public** String getPassword() {

**return** password;

}

}

**LoginController.java**

This is a Spring Controller responsible for the registration of users and the login.

**package** **com.flight**;

**import** **org.springframework.beans.factory.annotation.Value**;

**import** **org.springframework.stereotype.Controller**;

**import** **org.springframework.web.bind.annotation.RequestMapping**;

**import** **org.springframework.web.bind.annotation.RequestMethod**;

**import** **org.springframework.web.bind.annotation.RequestParam**;

**import** **org.springframework.web.bind.annotation.SessionAttributes**;

**import** **org.springframework.web.servlet.ModelAndView**;

**import** **java.sql.\***;

@Controller

@SessionAttributes("user")

**public** **class** **LoginController** {

@Value("jdbc:mysql://localhost:3306/travel")

String dburl;

@Value("root")

String dbuname;

@Value("")

String dbpass;

@RequestMapping(value="/login", method = RequestMethod.POST)

**public** ModelAndView Login(@RequestParam("username") String uname, @RequestParam("password") String password)

{

DAO dao=**new** DAO(dburl,dbuname,dbpass);

ResultSet rs=**null**;

**try**

{

rs = dao.loginCheck(uname, password);

}

**catch** (ClassNotFoundException | SQLException e1) {

*// TODO Auto-generated catch block*

e1.printStackTrace();

}

ModelAndView modelAndView = **new** ModelAndView();

**if**(rs!=**null**)

{

User u=**null**;

*//Set the session*

**try**

{

u = **new** User(rs.getString("name"), rs.getString("uname"), rs.getString("password"));

}

**catch** (SQLException e)

{

*// TODO Auto-generated catch block*

e.printStackTrace();

}

modelAndView.addObject("user", u);

modelAndView.setViewName("searchFlight.jsp");

**return** modelAndView;

}

**else**

{

System.out.println("Error"+rs);

String msg=**new** String("sorry!! Wrong credentials");

System.out.println(msg);

modelAndView.addObject("msg", msg);

modelAndView.setViewName("error.jsp");

**return** modelAndView;

}

}

@RequestMapping(value="/register", method = RequestMethod.POST)

**public** ModelAndView Register(@RequestParam("name") String name, @RequestParam("username") String uname, @RequestParam("password") String password)

{

DAO dao=**new** DAO(dburl,dbuname,dbpass);

User u=**new** User(name,uname,password);

**try** {

**if**(dao.checkUname(uname))

{

ModelAndView modelAndView = **new** ModelAndView();

String msg="sorry!! Duplicate username";

modelAndView.addObject("msg", msg);

modelAndView.setViewName("error.jsp");

**return** modelAndView;

}

}

**catch** (Exception e1) {

*// TODO Auto-generated catch block*

e1.printStackTrace();

}

**try**{

dao.register(u);

}

**catch**(Exception e)

{

e.printStackTrace();

}

ModelAndView modelAndView = **new** ModelAndView();

modelAndView.setViewName("searchFlight.jsp");

**return** modelAndView;

}

@RequestMapping(value="/logout", method = RequestMethod.GET)

**public** ModelAndView Logout()

{

User u=**new** User(**null**,**null**,**null**);

ModelAndView modelAndView = **new** ModelAndView();

modelAndView.addObject("user", u);

modelAndView.setViewName("index.jsp");

**return** modelAndView;

}

}

**SearchFlights.java**

This is the Controller responsible for the flight search and flight filtering

**package** **com.flight**;

**import** **org.springframework.beans.factory.annotation.Value**;

**import** **org.springframework.stereotype.Controller**;

**import** **org.springframework.web.bind.annotation.RequestMapping**;

**import** **org.springframework.web.bind.annotation.RequestMethod**;

**import** **org.springframework.web.bind.annotation.RequestParam**;

**import** **org.springframework.web.servlet.ModelAndView**;

**import** **java.sql.\***;

**import** **java.text.SimpleDateFormat**;

**import** **java.util.Date**;

**import** **java.text.DateFormat**;

@Controller

**public** **class** **SearchFlights**

{

@Value("jdbc:mysql://localhost:3306/travel")

String dburl;

@Value("root")

String dbuname;

@Value("")

String dbpass;

@RequestMapping(value="/search", method = RequestMethod.POST)

**public** ModelAndView getFlights(@RequestParam("src") String src, @RequestParam("dest") String dest, @RequestParam("date") String date)

{

System.out.println(dburl);

DAO dao=**new** DAO(dburl,dbuname,dbpass);

ResultSet rs=**null**;

*// get day of week from date string*

String day;

SimpleDateFormat format1=**new** SimpleDateFormat("yyyy-MM-dd");

Date dt1=**null**;

**try**

{

dt1 = (Date) format1.parse(date);

}

**catch** (Exception e1) {

*// TODO Auto-generated catch block*

e1.printStackTrace();

}

DateFormat format2=**new** SimpleDateFormat("EEEE");

day=format2.format(dt1);

System.out.println(day);

**try**

{

rs=dao.getFlights(src,dest,day);

}

**catch** (Exception e)

{

*// TODO Auto-generated catch block*

e.printStackTrace();

}

ModelAndView mv=**new** ModelAndView();

mv.setViewName("flights.jsp");

mv.addObject("flights",rs);

mv.addObject("day",day);

mv.addObject("src",src);

mv.addObject("dest",dest);

**return** mv;

}

@RequestMapping(value="/filter", method = RequestMethod.POST)

**public** ModelAndView filter(@RequestParam("src") String src, @RequestParam("dest") String dest, @RequestParam("day") String day, @RequestParam("company") String company, @RequestParam("stops") String stops)

{

System.out.println(dburl);

DAO dao=**new** DAO(dburl,dbuname,dbpass);

ResultSet rs=**null**;

System.out.println(day);

**try**

{

rs=dao.filterFlights(src,dest,company,stops,day);

}

**catch** (Exception e)

{

*// TODO Auto-generated catch block*

e.printStackTrace();

}

ModelAndView mv=**new** ModelAndView();

mv.setViewName("flights.jsp");

mv.addObject("flights",rs);

mv.addObject("day",day);

mv.addObject("src",src);

mv.addObject("dest",dest);

**return** mv;

}

}

**DAO.java**

This Java bean is responsible for interacting with the database. It takes care of all database operations- connecting with the database, querying the database and returning the required data to the controllers.

**package** **com.flight**;

**import** **java.sql.\***;

**public** **class** **DAO** {

**private** Connection con;

**private** Statement stmt;

**private** String url,uname,password;

**public** DAO(String url, String uname, String password)

{

**this**.url=url;

**this**.uname=uname;

**this**.password=password;

}

*//Function to establish the connection*

**private** void setConnection() **throws** ClassNotFoundException, SQLException

{

Class.forName("com.mysql.jdbc.Driver");

con=DriverManager.getConnection(url,uname,password);

stmt=con.createStatement();

}

*//Function to close the connection*

**private** void closeConn() **throws** SQLException

{

con.close();

}

*//Function for getting cities*

**public** ResultSet getCities() **throws** ClassNotFoundException, SQLException

{

setConnection();

String query="select \* from cities";

ResultSet rs=stmt.executeQuery(query);

**if**(rs.next())

{

**return** rs;

}

closeConn();

**return** **null**;

}

*//Function for getting flights*

**public** ResultSet getFlights(String src, String dest, String day) **throws** ClassNotFoundException, SQLException

{

setConnection();

String query1="(select \* from flights where src\_city=\'"+src+"\' and dest\_city=\'"+dest+"\' and day=\'"+day+"\')";

String query2="(select \* from offers where CURRENT\_TIMESTAMP between start\_time and end\_time)";

String query="select \* from "+query1+" f left join "+query2+" o on f.src\_city=o.src and f.dest\_city=o.dest";

System.out.println(query);

ResultSet rs=stmt.executeQuery(query);

**if**(rs.next())

{

**return** rs;

}

closeConn();

**return** **null**;

}

*//Function for getting flights*

**public** ResultSet getOffers() **throws** ClassNotFoundException, SQLException

{

setConnection();

String query="select \* from offers where CURRENT\_TIMESTAMP between offers.start\_time and offers.end\_time";

System.out.println(query);

ResultSet rs=stmt.executeQuery(query);

**if**(rs.next())

{

**return** rs;

}

closeConn();

**return** **null**;

}

*//Function for getting flights*

**public** ResultSet getCompanies() **throws** ClassNotFoundException, SQLException

{

setConnection();

String query="select distinct(company) as company from flights";

System.out.println(query);

ResultSet rs=stmt.executeQuery(query);

**if**(rs.next())

{

**return** rs;

}

closeConn();

**return** **null**;

}

*//Function for getting flights*

**public** ResultSet filterFlights(String src, String dest, String company, String stops, String day) **throws** ClassNotFoundException, SQLException

{

setConnection();

String query1="select \* from flights where src\_city=\'"+src+"\' and dest\_city=\'"+dest+"\' and day=\'"+day+"\'";

**if**(company.compareToIgnoreCase("all")!=0)

query1+=" and company='"+company+"'";

**if**(stops.compareToIgnoreCase("all")!=0)

query1+=" and num\_stops='"+stops+"'";

String query2="(select \* from offers where CURRENT\_TIMESTAMP between start\_time and end\_time)";

String query="select \* from ("+query1+") f left join "+query2+" o on f.src\_city=o.src and f.dest\_city=o.dest";

System.out.println(query);

ResultSet rs=stmt.executeQuery(query);

**if**(rs.next())

{

**return** rs;

}

closeConn();

**return** **null**;

}

*//Function for getting flights*

**public** ResultSet getStops(String flight\_id) **throws** ClassNotFoundException, SQLException

{

setConnection();

String query="select \* from flight\_stops where flight\_id='"+flight\_id+"' order by stop\_no";

System.out.println(query);

ResultSet rs=stmt.executeQuery(query);

**if**(rs.next())

{

**return** rs;

}

closeConn();

**return** **null**;

}

*//Function for login*

**public** ResultSet loginCheck(String username, String password) **throws** ClassNotFoundException, SQLException

{

setConnection();

String query="select \* from logininfo where uname=\'"+username+"' and password=\'"+password+"'";

System.out.println(query);

ResultSet rs=stmt.executeQuery(query);

**if**(rs.next())

{

**return** rs; *//Means username password exists*

}

closeConn();

**return** **null**;

}

*//Function for checking duplicate username*

**public** boolean checkUname(String username) **throws** ClassNotFoundException, SQLException

{

setConnection();

String query="select \* from logininfo where uname=\'"+username+"'";

System.out.println(query);

ResultSet rs=stmt.executeQuery(query);

**if**(rs.next())

{

**return** **true**; *//Means username password exists*

}

closeConn();

**return** **false**;

}

*//Function for registering*

**public** void register(User u) **throws** ClassNotFoundException, SQLException

{

setConnection();

String query="insert into logininfo values('"+u.getName()+"','"+u.getUname()+"','"+u.getPassword()+"')";

System.out.println(query);

int res=stmt.executeUpdate(query);

closeConn();

}

}

**Deployment Descriptor: web.xml**

<!DOCTYPE web-app PUBLIC

"-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"

"http://java.sun.com/dtd/web-app\_2\_3.dtd" >

**<web-app>**

**<display-name>**Archetype Created Web Application**</display-name>**

**<context-param>**

**<param-name>**dburl**</param-name>**

**<param-value>**jdbc:mysql://localhost:3306/travel**</param-value>**

**</context-param>**

**<context-param>**

**<param-name>**dbuname**</param-name>**

**<param-value>**root**</param-value>**

**</context-param>**

**<context-param>**

**<param-name>**dbpass**</param-name>**

**<param-value></param-value>**

**</context-param>**

**<servlet>**

**<servlet-name>**flight**</servlet-name>**

**<servlet-class>**org.springframework.web.servlet.DispatcherServlet**</servlet-class>**

**<load-on-startup>**1**</load-on-startup>**

**</servlet>**

**<servlet-mapping>**

**<servlet-name>**flight**</servlet-name>**

**<url-pattern>**/**</url-pattern>**

**</servlet-mapping>**

**</web-app>**

**flight-servlet.xml**

<?xml version="1.0" encoding="UTF-8"?>

**<beans** xmlns="http://www.springframework.org/schema/beans"

xmlns:ctx="http://www.springframework.org/schema/context"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:mvc="http://www.springframework.org/schema/mvc"

xsi:schemaLocation="http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans-2.5.xsd

http://www.springframework.org/schema/mvc

http://www.springframework.org/schema/mvc/spring-mvc-3.0.xsd

http://www.springframework.org/schema/context

http://www.springframework.org/schema/context/spring-context-2.5.xsd "**>**

**<ctx:annotation-config></ctx:annotation-config>**

**<ctx:component-scan** base-package="com.flight"**></ctx:component-scan>**

**</beans>**

**DATABASE SCHEMA**

|  |
| --- |
|  |
| **cities table** |
|  |
| **flights table** |
|  |
| **flight\_stops table** |
|  |
| **offers table** |
|  |
| **logininfo table** |

**OUTPUT:**

|  |
| --- |
|  |
| **Fig. 1** User Registration process |
|  |
| **Fig. 2** User Login process |
|  |
| **Fig 3.** Home page |
|  |
| **Fig 4.** Flight search page |
|  |
| **Fig 5.** Filtered flights |

**DISCUSSION:**

It can be observed from the outputs that the application works in accordance with the given problem statement. The application has a signup-login-logout framework. To maintain session instead of using HttpSession as we did in JSP-Servlet applications, the @SessionAttribute annotation of Spring Framework has been used. Session is invalidated on logout so that users cannot access items page without logging in. HttpRequest and HttpResponse objects are substituted by @RequestParam annotation and ModelAndView object in the controllers.

On succcessful login, the website displays flights according to the

choices entered by the user. It also allows the user to enter preferences and

shows list of flights accordingly. Instead of using various servlets for login-signup-logout, the Spring framework helps us to keep one controller for user validation and

registration. Also, instead of updating servlet and servlet-mapping tags in the web.xml file, the @RequestMapping annotation is used. This keeps the deployment task simple for the programmer.