

Z2RF1WMJ3

JAVA ADVANCE



JANUARY 14, 2019

JAVA ADVANCE PROJECT – JSP, DATABASE & WEB SERVICE PROGRAMMING

Table of Contents

Program details.....	1
Description of the program:	1
Instructions on which OS to use:	1
Type of & version of programs used:.....	1
What do you need to run the program?.....	1
How to run the program?	1
To install GlassFish server:	1
Error handling & user input.	2
Example of handling empty fields.....	2
Example of handling invalid password.....	2
Creating the database.....	3
Creating tables.	4
Web Service	5
Topics.jsp	26
Subject.jsp	30
Profile.jsp	35
Posts.jsp	37
MakePost.jsp	40
Login.jsp.....	42
Join.jsp.....	46
passwordFogotten.jsp	49
deleteAccount.jsp	52
changeEmail.jsp	54
changePassword.jsp	56
changeSecurityQuestion.jsp.....	58
UserNotFound.html	60
InvalidPassword.html.....	61
Screenshots	62
Bibliography & Additional Notes.	69
Bibliography for customizations:	69
Important points:	69

Program details.

Description of the program:

The program is a distributed application, that can be used as a forum for various topics. The interface is a website only. And users are able to register with their personal information. Users also have the ability to create topics (posts), post comments and post replies to other user's comments. Comments & replies are viewable by all users. The web service provides all the functionality & **generates a log file to the desktop.**

Instructions on which OS to use:

Any platform can be used to run the program.

Type of & version of programs used:

NetBeans IDE 8.2 was used on the Windows 10 operating system. Microsoft SQL Server Management Studio v17 was used to develop the database.

What do you need to run the program?

The following resources are needed to run the program.

- Microsoft SQL Server Management Studio. <https://www.microsoft.com/en-us/sql-server/sql-server-downloads>
- Microsoft JDBC Driver 6.2 for SQL server. (**Included in root folder**)
- Java SDK. <https://www.oracle.com/technetwork/java/javase/downloads/index.html>
- NetBeans IDE. <https://netbeans.org/downloads/8.0.2/>
- Glassfish Server (I'm using glassfish server, you can use apache or similar). Will work flawlessly on Glassfish. It is tested on using Glassfish Server. <https://www.oracle.com/technetwork/java/javaee/downloads/index.html>
- A web browser. Firefox is preferred. It is faster.

How to run the program?

To open the program the following are the instructions: (Assuming you are going to run on a newly installed windows OS, if not you can start from step 6).

- 1) Install Java SDK (this will include the jdk).
- 2) Install Microsoft SQL Server Management Studio & SQL Server Express Edition.
- 3) Install NetBeans IDE 8.2 & above is preferred.
- 4) Download Java EE SDK from their website (this contains GlassFish server).
- 5) Download the Microsoft JDBC Driver 6.2 for SQL driver (**included in project dir.**)
- 6) Run NetBeans IDE (Assuming you have GlassFish server installed)
- 7) Click on File > Open Project > browse for project name "**Forum2**"
- 8) Expand **Forum2** in Projects Window > Expand **Web Pages** > Expand **Foxcrypt** > Right click on **Topics.jsp** and click **Run File**.

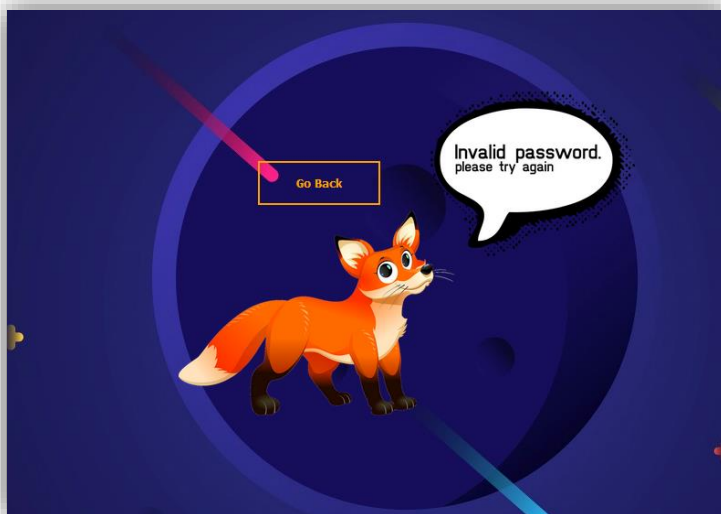
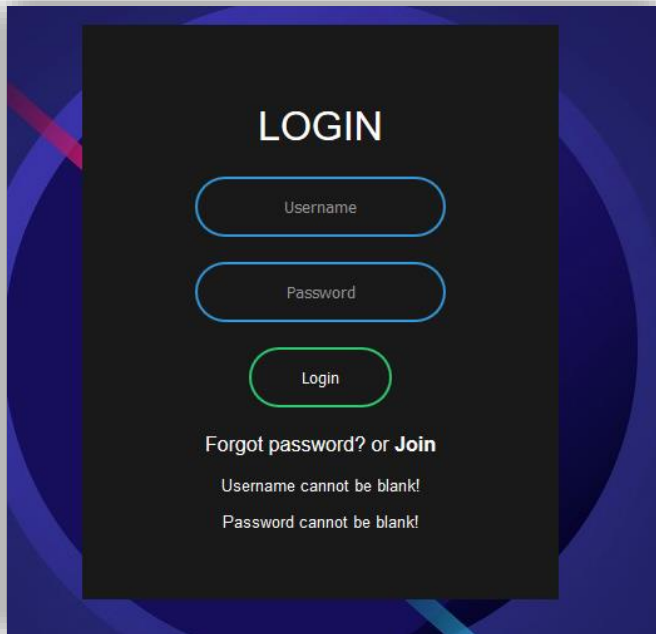
To install GlassFish server:

- 1) Click on Services window.
- 2) Right click on Servers. And click **Add Server**.
- 3) Select **GlassFish Server**. Select Location of GlassFish Server (**located in Java EE folder**) & click Finish.

Error handling & user input.

For error handling, **try...catch** blocks are used & exceptions are caught. Regular expressions is used for email validation. If statements are mostly used to validate input fields. Errors and exceptions are printed to console. Error pages are also used to handle certain errors. And provide the user with alternate options.

Example of handling empty fields.



Example of handling invalid password.

Creating the database.

This SQL query creates a database.

```
1  /*
2  Filename      :      goData
3  Author       :      Aysham Ali Hameed
4  Created      :      12 January 2019
5  Operating System :      Windows 10
6  Version      :      MSSMS v17.8.1
7  Description   :      Creating the database & tables.
8  */
9
10
11  --CREATING DATABASE=====
12  use master
13  go
14
15  --DROPPING EXISTING DATABASE
16  if exists(select name from master.dbo.sysdatabases where name='fDatabase')
17  begin
18      drop database fDatabase
19  end
20  go
21
22  create database fDatabase
23  on primary(
24      name = 'fDatabase_data',
25      filename = 'C:\DB\fDatabase_data.mdf',
26      size = 5mb,
27      filegrowth = 10%
28  )
29
30  log on(
31      name = 'fDatabase_log',
32      filename = 'C:\DB\fDatabase_log.ldf',
33      size = 5mb,
34      filegrowth = 10%
35  )
36  go
37
38
39
40  --CREATING TABLES=====
41  use fDatabase
42  go
43
44  --DROPPING EXISTING TABLES
45  drop table if exists dbo.Replies
46  go
47
48  drop table if exists dbo.Comments
49  go
50
51  drop table if exists dbo.Posts
52  go
53
54  drop table if exists dbo.Users
55  go
56
57  drop table if exists dbo.Topics
58  go
59
60
```

```
1 use fDatabase
2 GO
3
4 DBCC CHECKIDENT (Users, RESEED, 0)
5 GO
6
7 DBCC CHECKIDENT (Topics, RESEED, 0)
8 GO
9
10 DBCC CHECKIDENT (Posts, RESEED, 0)
11 GO
12
13 DBCC CHECKIDENT (Comments, RESEED, 0)
14 GO
15
16 DBCC CHECKIDENT (Replies, RESEED, 0)
17 GO
18
```

SQL query included in root folder. Used to reset identity seed. If needed. (When order of insertion gets messed up)

Creating tables.

Creates tables inside the database.

```
61 --CREATING TABLES
62 create table Users(
63     userID int not null primary key identity,
64     userName varchar(50) not null unique,
65     userEmail varchar(70) not null,
66     userPassword varchar(50) not null,
67     userSQuestion varchar(100) not null,
68     userSAnswer varchar(100) not null
69 )
70 go
71
72 --INSERTING 1 DEFAULT USER
73 insert into Users
74 values ('bruce7','bruce7@gmail.com','password123','Where were your parents born?','Gotham')
75 go
76
77 create table Topics(
78     topicID int not null primary key identity,
79     topicName varchar(50) not null
80 )
81 go
82
83 --INSERTING A NUMBER OF FIX TOPICS/CATEGORIES FOR DISCUSSIONS
84 insert into Topics
85 values ('Social'), ('Apple'), ('Software'), ('Movies'), ('Books'), ('Events'), ('Gaming'), ('Hacking'), ('Coding'), ('Science')
86 go
87
88 create table Posts(
89     postID int not null primary key identity,
90     postSubject varchar(255) not null,
91     postDescription varchar(max) null,
92     postTime varchar(50) not null,
93     postAuthor varchar(50) not null references Users(userName) on delete cascade,
94     topicID int not null references Topics(topicID) on delete cascade
95 )
96 go
97
98
99
100 create table Comments(
101     commentIdentity int not null primary key identity,
102     commentID int not null,
103     commentAuthor varchar(50) not null,
104     commentTime varchar(50) not null,
105     commentDesc varchar(max) not null,
106     unique (commentID, commentAuthor),
107     postID int not null references Posts(postID) on delete cascade
108 )
109 go
110
111 create table Replies(
112     replyID int not null primary key identity,
113     replyDescription varchar(max) not null,
114     replyTime varchar(50) not null,
115     replyAuthor varchar(50) not null,
116     commentIdentity int not null references Comments(commentIdentity) on delete cascade
117     --postID int not null references Posts(postID)
118 )
```

Web Service

Processor (Web service provider). Responsible for all the processing.

```
1  /*
2      Filename      : Processor
3      Author       : Aysham Hameed
4      Created      : Dec 7, 2018, 10:57:16 PM
5      Operating System : Windows 10
6      Version      : NetBeans IDE 8.2
7      Description   : Web Service that does all the functionality.
8  */
9
10
11 package serviceServer; //Package name of webservice class.
12
13 import java.io.*; //Importing IO package.
14 import java.util.Date; //Importing Date class from util package.
15 import java.sql.*; //Importing all sql classes.
16 import java.text.ParseException; //Importing ParseException class to parse date.
17 import java.text.SimpleDateFormat; //Imported to format date.
18 import java.util.Vector; //Importing vector class for storing data.
19 import javax.jws.WebService; //Used for web service class annotation(identification).
20 import javax.jws.WebMethod; //Used for web service method annotation(identification).
21 import javax.jws.WebParam; //Used for web service parameters annotation(identification).
22
23 /**
24  * This is the processor class. It is a web service and acts as a server.
25  * It provides the main functionality to the clients. It implements the
26  * Serializable class because it is also a java bean and contains accessor methods.
27  * @author Aysham Hameed
28  */
29 @WebService(serviceName = "processPosts")
30 public class processor implements Serializable{
31     //DATABASE
32     String driverURL = "com.microsoft.sqlserver.jdbc.SQLServerDriver"; /**driver URL.*/
33     String dbURL = "jdbc:sqlserver://localhost:1433;databaseName=fDatabase"; /**location of database.*/
34     String dbUser = "sa"; /**database username.*/
35     String dbPassword = "123456"; /**database password.*/
36     Connection connection=null; /**Connection object for making the connection.*/
37
38     //POSTS
39     Vector<String> postSubjects = new Vector<String>();
40     Vector<String> postAuthors = new Vector<String>();
41     Vector<Integer> postIDS = new Vector<Integer>();
42
43     //COMMENTS
44     Vector<String> commentDescriptions = new Vector<String>();
45     Vector<String> commentAuthors = new Vector<String>();
46     Vector<String> commentTimes = new Vector<String>();
47     Vector<Integer> commentIDS = new Vector<Integer>();
48     //REPLIES
49     Vector<String> replyDescriptions = new Vector<String>();
50     Vector<String> replyAuthor = new Vector<String>();
51     Vector<String> replyTimes = new Vector<String>();
52
53     String userQuestion = null; /**to store user question in getUserEmail method.*/
54     String userAnswer = null; /**to store user answer in getUserEmail method.*/
55
56     /**
57      * Connects to the database using the {@link #makeConnection} method.
58      */
59     public processor(){
60         makeConnection(); /**calling makeConnection method.
61     }
```

```

64 ▼ /**
65  * Used to create a connection to the database using the driver
66  * URL and database URL consisting of database username & password.
67  * @throws ClassNotFoundException if cannot find driver class.
68  * @throws SQLException if database connection is invalid.
69  */
70 @WebMethod(operationName = "makeConnection")
71 ▼ public void makeConnection() {
72     try{//using try catch block.
73         Class.forName(driverURL);//loading database driver.
74         connection = DriverManager.getConnection(dbURL,dbUser,dbPassword);//creating database connection.
75     } catch (ClassNotFoundException e){//catching class not found exception.
76         System.out.println("ClassNotFoundException :"+e.getMessage());//printing out exception message.
77     }catch (SQLException e){//catching SQLException.
78         System.out.println("SQLException : "+e.getMessage());//printing out exception message.
79     }
80 }
81
82
83 ▼ /**
84  * Used to get the topic ID from database using topic name as a parameter.
85  * @param topicName parameter used. Example = "Social"
86  * @return the ID of the topic. Example = 1
87  * @throws SQLException if there is a query or connection invalid.
88  */
89 @WebMethod(operationName = "getTopicID")
90 ▼ public int getTopicID(@WebParam(name="topicName")String topicName){
91     String query = "select topicID from Topics where topicName='"+topicName+"'";//search query.
92     int id=0;//to store topic ID.
93     try{//using try & catch block
94         Statement statement = connection.createStatement();//creating a statement object.
95         ResultSet records = statement.executeQuery(query);//executing query.
96         while(records.next()){//looping through records.
97             id = records.getInt("topicID");//looking for topicID
98         }
99     } catch (SQLException e){//catching SQLException.
100         System.out.println("SQLException : "+e.getMessage());//printing exception message.
101     }
102     return id;//returning topic ID.
103 }

```



```

106  /**
107   * Used to get to post count for each topic using topic name.
108   * @param topicName Example = "Social"
109   * @return count Example = 54 (Meaning there are 54 posts for the Social topic.
110   * @throws SQLException if there is a query or connection invalid.
111   */
112  @WebMethod(operationName="getPostCount")
113  public int getPostCount(@WebParam(name="topicName")String topicName){
114      int topicID = getTopicID(topicName); //getting topicID from topic name.
115      String query = "select * from Posts where topicID="+topicID; //search query.
116      int count=0; //counter.
117      try{//using try & catch block.
118          Statement statement = connection.createStatement(); //creating statement.
119          ResultSet records = statement.executeQuery(query); //executing query.
120          while(records.next()){ //looping through records.
121              count++; //increasing counter.
122          }
123      } catch (SQLException e){ //catching sql exception.
124          System.out.println("SQLEX : "+e.getMessage()); //displaying exception message.
125      }
126      return count; //returning post count.
127  }
128
129
130  /**
131   * Gets the post data using topic name. Gets posts subjects, authors & ID's.
132   * @param topicName Example = "Social"
133   * @throws SQLException if there is a query or connection invalid.
134   */
135  @WebMethod(operationName="getPostsData")
136  public void getPostsData(@WebParam(name="topicName")String topicName){
137      //REMOVING ALL ELEMENTS FIRST
138      postSubjects.removeAllElements();
139      postAuthors.removeAllElements();
140      postIDS.removeAllElements();
141      //-----
142      String query = "select * from Posts where TopicID="+getTopicID(topicName); //search query.
143      try{//using try & catch block.
144          Statement statement = connection.createStatement(); //creating statement object.
145          ResultSet records = statement.executeQuery(query); //executing query.
146          while(records.next()){ //looping through records.
147              postSubjects.addElement(records.getString("postSubject")); //saving to vector.
148              postAuthors.addElement(records.getString("postAuthor")); //saving to vector.
149              postIDS.addElement(records.getInt("postID")); //saving to vector.
150          }
151      } catch (SQLException e){ //catching sql exception.
152          System.out.println("SQLEX : "+e.getMessage()); //displaying exception message.
153      }
154  }
155

```

```

157  /**
158   * Gets post subject vector.
159   * @return postSubjects, a String vector. Based on topic. Example:
160   * "How much is an iPhone?" or
161   * "What is a good movie to watch?"
162   */
163  @WebMethod(operationName="getSubjects")
164  public Vector<String> getSubjects(){
165      return postSubjects;//reurns post subject vector.
166  }
167
168
169  /**
170   * Gets post author vector.
171   * @return postAuthors, a String vector. Based on topic. Example:
172   * "bruce7" or
173   * "peter1"
174   */
175  @WebMethod(operationName="getAuthors")
176  public Vector<String> getAuthors(){
177      return postAuthors;//returns post author vector.
178  }
179
180
181  /**
182   * Gets post ID vector.
183   * @return postIDS, Based on topic. Example:
184   * 1 or
185   * 3
186   */
187  @WebMethod(operationName="getPostID")
188  public Vector<Integer> getPostID(){
189      return postIDS;//returns post ID vector.
190  }

```

```

193 ▼ /**
194  * Finds the author of the post using postID.
195  * @param postID Example : 3
196  * @return author. Example : "bruce7"
197  * @throws SQLException if there is a query or connection invalid.
198  */
199 @WebMethod(operationName="findPostAuthor")
200 ▼ public String findPostAuthor(@WebParam(name="postID")int postID){
201     String author=""; //to store author name.
202     String query = "select postAuthor from Posts where postID="+postID; //search query.
203 ▼     try{//using try catch block.
204         Statement statement = connection.createStatement();//creating statement object.
205         ResultSet records = statement.executeQuery(query);//executing query.
206         while(records.next()){ //looping through all records in db.
207             author = records.getString("postAuthor");//storing author in variable.
208         }
209     } catch (SQLException e){ //catching sql exception.
210         System.out.println("SQLEX : "+e.getMessage()); //displaying exception message.
211     }
212     return author; //returning author name.
213 }
214
215
216 ▼ /**
217  * Gets the date & time of a post using postID.
218  * @param postID Example : 5
219  * @return Example : 2018-12-11 23:59:59
220  * @throws SQLException if there is a query or connection invalid.
221  */
222 @WebMethod(operationName="getPostDateTime")
223 ▼ public String getPostDateTime(@WebParam(name="postID")int postID){
224     String postDateTime=""; //to store date & time.
225     String query = "select postTime from Posts where postID="+postID; //search query.
226 ▼     try{//to catch exceptions.
227         Statement statement = connection.createStatement();//creating statement object.
228         ResultSet records = statement.executeQuery(query);//executing query.
229         while(records.next()){ //looping through all records in db.
230             postDateTime = records.getString("postTime");//storing date & time in variable.
231         }
232     } catch (SQLException e){ //catching sql exception.
233         System.out.println("SQLEX : "+e.getMessage()); //displaying exception message.
234     }
235     return getDateAndTime(postDateTime); //decoding & returning date & time.
236 }
237

```

```

240  /**
241  * Private method used to decode date & time. It also calculates the date
242  * and time and translates it into words.
243  * @param thisdate Example : "2018-12-11 23:59:59"
244  * @return Example : "5 minutes ago"
245  * @throws ParseException
246  */
247  @WebMethod(operationName="getTime")
248  private String getTime(@WebParam(name="thisdate")String thisdate){
249      //Manipulating string to separate time and date.
250      int spacePos = thisdate.indexOf(" "); //findig space position.
251      String date = thisdate.substring(0,spacePos); //storing date.
252      String time = thisdate.replace(date+" ", ""); //storing time.
253
254      //Creating date and time formats for parsing
255      SimpleDateFormat formatDate = new SimpleDateFormat("yyyy-MM-dd"); //date format.
256      SimpleDateFormat formatTime = new SimpleDateFormat("HH:mm:ss"); //time format.
257      String currentDate = formatDate.format(new Date()); //getting current date.
258      String currentTime = formatTime.format(new Date()); //getting current time.
259      long dateDifference=0; //to store date difference.
260      long timeDifference=0; //to store time differences.
261      String output=""; //to store output in words.
262
263      try{//to catch parsing exceptions.
264          //CALCULATING DATE
265          Date parsedDate = formatDate.parse(date); //parsing date.
266          Date parsedCurrentDate = formatDate.parse(currentDate); //parsing current date.
267          dateDifference = parsedCurrentDate.getTime()- parsedDate.getTime(); //getting date difference.
268
269          //CALCULATING TIME
270          Date parsedTime = formatTime.parse(time); //parsing time.
271          Date parsedCurrentTime = formatTime.parse(currentTime); //parsing current time.
272          timeDifference = parsedCurrentTime.getTime()-parsedTime.getTime(); //getting time differences.
273      } catch (ParseException e){ //catching parse exception.
274          System.out.println("Parse Exception: "+e.toString()); //displaying exception message.
275      }
276
277      //1000MS = 1 SEC | 60SEC = 1 MIN | 60MIN = 1 HOUR | 24HOUR = 1 DAY
278      dateDifference = dateDifference/(1000*60*60*24); //difference in days for example= 45 days
279      if(dateDifference<=1){ //so if its less than a day
280          timeDifference = timeDifference/(1000*60); //gets time in minutes
281          if(timeDifference>=60){ //if more than or 60 minutes then convert to hours
282              timeDifference = timeDifference/60; //example 70 minutes divide by 60
283              output = timeDifference + " Hours ago"; //displaying output.
284          } else if(timeDifference<60){ //if less than 60 minutes
285              output = timeDifference + " Minutes ago"; //displaying output.
286          }
287      }
288      else
289      if(dateDifference>1 && dateDifference<7){ //its been longer than a day & less than a week
290          output = dateDifference + " Days ago"; //displaying output.
291      }
292      else
293      if(dateDifference>=7){ //if its been longer than a week.
294          dateDifference = dateDifference/7; // then get the difference in weeks, so 45/7 therefore 6 weeks
295          if(dateDifference<2){ //if is been 1 week.
296              output = dateDifference + " Week ago"; //display week.
297          }
298          else if(dateDifference>=2 && dateDifference<=4){ //if more than a week.
299              output = dateDifference+" Weeks ago"; //display weeks.
300          }
301          else {
302              dateDifference = dateDifference/4; //then get difference in months, so 6/4 (because 4 weeks in a month aprox)
303              if(dateDifference<2){ //if less than 2 months.
304                  output = dateDifference + " Month ago"; //display month.
305              } else { //if more than or equal to 2 months
306                  output = dateDifference + " Months ago"; //display months.
307              }
308          }
309      }
310      return output; //return date or time as words.

```

```

313  /**
314  * Gets the description of a post using postID.
315  * @param postID Example : 2
316  * @return Example : "Can you please help me with this?"
317  * @throws SQLException if there is a query or connection invalid.
318  */
319  @WebMethod(operationName="getPostDescription")
320  public String getPostDescription(@WebParam(name="postID")int postID){
321      String description="";//used to store description of post.
322      String query = "select postDescription from Posts where postID="+postID;//search query.
323      try{//using try & catch block to catch exception.
324          Statement statement = connection.createStatement();//creating statement object.
325          ResultSet records = statement.executeQuery(query);//executing query.
326          while(records.next()){//looping through records.
327              description = records.getString("postDescription");//storing description into variable.
328          }
329      } catch (SQLException e){//catching SQL Exception
330          System.out.println("SQLLEX : "+e.getMessage());//displaying exception message.
331      }
332      return description;//returns post description.
333  }
334
335
336  /**
337  * Gets the title of the post using the topicName and postID.
338  * @param topicName Example : "Apple"
339  * @param postID Example : 1
340  * @return Example : "What is a good iPhone?"
341  * @throws SQLException if there is a query or connection invalid.
342  */
343  @WebMethod(operationName="getPostTitle")
344  public String getPostTitle(@WebParam(name="topicName")String topicName,@WebParam(name="postID")int postID){
345      String query = "select * from Posts where TopicID="+getTopicID(topicName)+" and postID="+postID;//search query.
346      String title="";//to store the post title.
347      try{//using try & catch block
348          Statement statement = connection.createStatement();//creating statement object.
349          ResultSet records = statement.executeQuery(query);//executing query.
350          while(records.next()){//looping through each record.
351              title = records.getString("postSubject");//locating and assigning post title.
352          }
353      } catch (SQLException e){//catching SQL exception.
354          System.out.println("SQLLEX : "+e.getMessage());//displaying exception message.
355      }
356      return title;//returning postTitle(subject)
357  }
358

```



```

360  /**
361  * Gets the total count of comments.
362  * @return Example : 5
363  * @throws SQLException if there is a query or connection invalid.
364  */
365  @WebMethod(operationName="getTotalCommentCount")
366  public int getTotalCommentCount(){
367      String query = "select * from Comments";//search query.
368      int count=0;//counter.
369      try{//using try & catch block.
370          Statement statement = connection.createStatement();//creating statement object.
371          ResultSet records = statement.executeQuery(query);//executing query.
372          while(records.next()){//looping through records.
373              count++;//incrementing counter.
374          }
375      } catch (SQLException e){//catching exception.
376          System.out.println("SQLEX : "+e.getMessage());//displaying exception message.
377      }
378      return count;//returning comment count.
379  }
380
381
382  /**
383  * Gets the count of comments of a specific post using postID
384  * @param postID Example : 2
385  * @return Example : 5
386  * @throws SQLException if there is a query or connection invalid.
387  */
388  @WebMethod(operationName="getCommentCount")
389  public int getCommentCount(@WebParam(name="postID")int postID){
390      String query = "select * from Comments where postID="+postID;//search query.
391      int count=0;//counter.
392      try{//try & catch block.
393          Statement statement = connection.createStatement();//creating statement object.
394          ResultSet records = statement.executeQuery(query);//executing query.
395          while(records.next()){//looping through records.
396              count++;//incrementing count.
397          }
398      } catch (SQLException e){//catching exception.
399          System.out.println("SQLEX : "+e.getMessage());//displaying exception message.
400      }
401      return count;//returning comment count.
402  }

```

```

405  /**
406   * Gets the comment data from database and saves to related vectors.
407   * @param postID Example : 2
408   * @throws SQLException if there is a query or connection invalid.
409   */
410  @WebMethod(operationName="getCommentsData")
411  public void getCommentsData(@WebParam(name="postID")int postID){
412      //REMOVING ELEMENTS FIRST-----
413      commentDescriptions.removeAllElements();
414      commentAuthors.removeAllElements();
415      commentTimes.removeAllElements();
416      commentIDS.removeAllElements();
417      //-----
418
419      String query = "select * from Comments where postID="+postID;//search query.
420
421      try{//try & catch block.
422          Statement statement = connection.createStatement();//creating statement object.
423          ResultSet records = statement.executeQuery(query);//executing query.
424          while(records.next()){//looping through records.
425              commentDescriptions.addElement(records.getString("commentDesc"));//saving to vector.
426              commentAuthors.addElement(records.getString("commentAuthor"));//saving to vector.
427              commentTimes.addElement(getDateTime(records.getString("commentTime")));//saving to vector.
428              commentIDS.addElement(records.getInt("CommentID"));//saving to vector.
429          }
430      } catch (SQLException e){//catching SQL exception.
431          System.out.println("SQLEX : "+e.getMessage());//displaying exception message.
432      }
433  }
434
435
436
437  /**
438   * Gets the comment description vector.
439   * @return commentDescriptions
440   */
441  @WebMethod(operationName="getCommentDescription")
442  public Vector<String> getCommentDescription(){
443      return commentDescriptions;//returning comment description vector.
444  }
445
446
447  /**
448   * Gets the comment author vector.
449   * @return commentAuthors
450   */
451  @WebMethod(operationName="getCommentAuthors")
452  public Vector<String> getCommentAuthors(){
453      return commentAuthors;//returns comment author vector.
454  }
455
456
457  /**
458   * Gets the comment time vector.
459   * @return commentTimes
460   */
461  @WebMethod(operationName="getCommentTimes")
462  public Vector<String> getCommentTimes(){
463      return commentTimes;//returns comment times vector.
464  }

```

```

467 ▾ /**
468  * Gets the comment ID vector.
469  * @return commentIDS
470  */
471 @WebMethod(operationName="getCommentIDs")
472 public Vector<Integer> getCommentIDs(){
473     return commentIDS;//returns comment ID vector.
474 }
475
476
477 ▾ /**
478  * Gets the comment identity from database which is a composite key.
479  * @param commentID Example : 4
480  * @param commentAuthor Example : "bruce7"
481  * @param postID Example : "2"
482  * @return Example : 1
483  */
484 @WebMethod(operationName="findCommentIdentity")
485 ▾ public int findCommentIdentity(@WebParam(name="commentID")int commentID,@WebParam(name="commentAuthor") String commentAuthor, @WebParam(name="postID")int postID){
486     String query = "select * from Comments where commentID="+commentID+" and commentAuthor='"+commentAuthor+"' and postID="+postID;//search query.
487     int id = 0;//to store ID
488     try{//try & catch block.
489         Statement statement = connection.createStatement();//creating statement object.
490         ResultSet records = statement.executeQuery(query);//executing query.
491         while(records.next()){//Looping through records.
492             id = records.getInt("commentID");//finding and storing id.
493         }
494     } catch (SQLException e){//catching SQL exception.
495         System.out.println("SQLEX : "+e.getMessage());//displaying exception.
496     }
497     return id; //returning ID
498 }
499
500
501 ▾ /**
502  * Gets the reply count using comment identity (composite key).
503  * @param commentIdentity Example : 1
504  * @return Example : 9
505  */
506 @WebMethod(operationName="getReplyCount")
507 ▾ public int getReplyCount(@WebParam(name="commentIdentity")int commentIdentity){
508     String query = "select * from Replies where commentIdentity="+commentIdentity;//search query.
509     int count=0;//counter variable.
510     try{//try & catch block.
511         Statement statement = connection.createStatement();//creating statement object.
512         ResultSet records = statement.executeQuery(query);//executing query.
513         while(records.next()){//Looping through records.
514             count++;//incrementing counter.
515         }
516     } catch (SQLException e){//catching SQL exception.
517         System.out.println("SQLEX : "+e.getMessage());//displaying exception message.
518     }
519     return count;//returning count.
520 }
521

```



```

523 ▼ /**
524  * Gets the reply data from database and assigns to related vectors.
525  * @param commentIdentity Example : 1
526  */
527 @WebMethod(operationName="getReplyData")
528 ▼ public void getReplyData(@WebParam(name="commentIdentity")int commentIdentity){
529     //REMOVING ELEMENTS FIRST-----
530     replyDescriptions.removeAllElements();
531     replyAuthor.removeAllElements();
532     replyTimes.removeAllElements();
533     //-----
534     String query = "select * from Replies where commentIdentity="+commentIdentity;//search query.
535 ▼ try{//try & catch block
536         Statement statement = connection.createStatement();//creating statement object
537         ResultSet records = statement.executeQuery(query);//executing query
538 ▼ while(records.next()){//looping through records
539             replyDescriptions.addElement(records.getString("replyDescription"));//saving to vector.
540             replyAuthor.addElement(records.getString("replyAuthor"));//saving to vector.
541             replyTimes.addElement(getDateTime(records.getString("replyTime")));//saving to vector.
542         }
543     } catch (SQLException e){//catching SQL exception
544         System.out.println("SQLLEX : "+e.getMessage());//displaying exception message
545     }
546 }
547
548
549 ▼ /**
550  * Gets the reply description vector.
551  * @return replyDescriptions
552  */
553 @WebMethod(operationName="getReplyDescriptions")
554 public Vector<String> getReplyDescriptions(){
555     return replyDescriptions;//returning reply description vector.
556 }
557
558
559 ▼ /**
560  * Gets the reply author vector.
561  * @return replyAuthor
562  */
563 @WebMethod(operationName="getReplyAuthors")
564 public Vector<String> getReplyAuthors(){
565     return replyAuthor;//returning reply author vector.
566 }
567
568
569 ▼ /**
570  * Gets the reply time vector.
571  * @return replyTimes
572  */
573 @WebMethod(operationName="getReplyTimes")
574 public Vector<String> getReplyTimes(){
575     return replyTimes;//returns reply times vector.
576 }

```

```

579  /**
580   * Private method, used to current generate time & date.
581   * @return Example "2019-01-01 12:32:12"
582   */
583  private String generateTime(){
584      SimpleDateFormat formatDate = new SimpleDateFormat("yyyy-MM-dd");//date format
585      SimpleDateFormat formatTime = new SimpleDateFormat("HH:mm:ss");//time format
586      String currentDate = formatDate.format(new Date());//storing new current date
587      String currentTime = formatTime.format(new Date());//storing new current time
588      return currentDate+" "+currentTime;//returning date & time
589  }
590
591
592  /**
593   * Used to add a reply.
594   * @param text Example : "This is a reply"
595   * @param author Example : "bruce7"
596   * @param commentIdentity Example : 1
597   * @throws SQLException if there is a query or connection invalid.
598   */
599  @WebMethod(operationName="addReply")
600  public void addReply(@WebParam(name="text")String text,@WebParam(name="author")String author,@WebParam(name="commentIdentity")int commentIdentity){
601      String replyTime = generateTime();//current reply time.
602
603      try{//try & catch block
604
605          String query = "insert into Replies (replyDescription, replyTime, replyAuthor, commentIdentity) "
606              + "values(?,?,?,?)";//insert query
607          //using prepared statement
608          PreparedStatement pst = connection.prepareStatement(query);
609          pst.setString(1, text);//text(description)
610          pst.setString(2, replyTime);//time
611          pst.setString(3, author);//author
612          pst.setInt(4, commentIdentity);//comment identity
613          pst.executeUpdate();
614
615          String log_out_REPLY = "Reply added by : "+author+" at : "+replyTime;//building string for log
616          writeToLog(log_out_REPLY);//writing to log
617      } catch (SQLException e){//catching SQL exception
618          System.out.println("SQLE : "+e.getMessage());//displaying exception message.
619      }
620  }
621  }

```

```

623 /**
624  * Used to add a post.
625  * @param subject Example : "How much is an iPhone?"
626  * @param text Example : "Can someone please tell me the price of an iPhone....."
627  * @param author Example : "peter1"
628  * @param topicID Example : 2 (topicName = Apple)
629  * @throws SQLException if there is a query or connection invalid.
630  */
631 @WebMethod(operationName="addPost")
632 public void addPost(@WebParam(name="subject")String subject,@WebParam(name="text") String text,@WebParam(name="author") String author,@WebParam(name="topicID") int topicID){
633     //postID, postSubject, postDescription, postTime, postAuthor, topicID
634     String postTime = generateTime();//current post time
635     try{//try & catch block
636         String query = "insert into Posts (postSubject,postDescription,postTime,postAuthor,topicID)"
637             + "values(?,?,?,?,?)"; //query
638
639         //Using prepared statement to insert into table
640         PreparedStatement pst = connection.prepareStatement(query);
641         pst.setString(1, subject);//subject
642         pst.setString(2, text);//text
643         pst.setString(3, postTime);//time
644         pst.setString(4, author);//author
645         pst.setInt(5, topicID);//topic id
646         pst.executeUpdate();//executing update
647
648         String log_out_POST = "Post added by : "+author+" at : "+postTime;//building string for LOG
649         writeToLog(log_out_POST);//writing to LOG
650     } catch (SQLException e){//catching SQL Exception
651         System.out.println("SQLE : "+e.getMessage());//displaying exception message
652     }
653 }
654
655 }
656
657
658 /**
659  * Private method used to write to LOG. (Stores on desktop)
660  * @param action Example : "Post added by : peter1 at : 2019-01-01 12:32:12"
661  * @throws IOException
662  */
663 private void writeToLog(String action){
664     String p = System.getProperty("user.home");//gets user path.
665     p = p.replace("\\", "/")+"/Desktop/log.txt";//replaces slashes to append to path.
666     try{//using try & catch block
667         PrintWriter out = new PrintWriter(new BufferedWriter(new FileWriter(p, true)), true);//creating file.
668         out.println(action);//writing to file.
669     } catch (IOException e) {//catching IOException.
670         System.out.println("Writing Error : " + e.toString());//displaying exception message.
671     }
672     System.out.println(action);//displaying to console the string written to LOG
673     System.out.println(p);//displaying to console path of LOG
674 }

```

```

675
676
677  /**
678   * Used to add a comment.
679   * @param commentAuthor Example : "bruce7"
680   * @param commentDesc Example : "Thank you!"
681   * @param postID Example : 2
682   */
683  @WebMethod(operationName="addComment")
684  public void addComment(@WebParam(name="commentAuthor")String commentAuthor,@WebParam(name="commentDesc") String commentDesc,@WebParam(name="postID") int postID){
685      int commentID =getTotalCommentCount()+1;//gets the comment id from previous count.
686      String commentTime = generateTime();//current comment time
687
688      try{//using try & catch block
689          String query = "insert into Comments (commentID, commentAuthor, commentTime, commentDesc, postID) "
690              + "values (?,?,?,?,?)";//insert query
691          //using prepared statement
692          PreparedStatement pst = connection.prepareStatement(query);
693          pst.setInt(1, commentID);//comment id
694          pst.setString(2, commentAuthor);//comment author
695          pst.setString(3, commentTime);//comment time
696          pst.setString(4, commentDesc);//comment description/text
697          pst.setInt(5, postID);//post id
698          pst.executeUpdate();//executing insert statement
699
700          String log_out_COMMENT = "Comment added by : "+commentAuthor+" at : "+commentTime;//building string for LOG
701          writeToLog(log_out_COMMENT);//writing to LOG
702      } catch (SQLException e){//catching SQL Exception
703          System.out.println("SQLE : "+e.getMessage());//displaying exception message
704      }
705  }

```

```

707  /**
708  * Checks if the user exists.
709  * @param username Example : "bruce7"
710  * @param password Example : "1234567"
711  * @return code Example : 1
712  * MEANINGS OF CODE:
713  * 1 = user found & password correct
714  * 2 = user found & password incorrect
715  * -1 = user not found
716  * @throws SQLException
717  */
718  @WebMethod(operationName="doesUserExist")
719  public int doesUserExist(@WebParam(name="username")String username,@WebParam(name="password")String password){
720      String query = "select * from Users";//search query
721      boolean userFound = false;//flag variable for username
722      boolean passFound = false;//flag variable for password
723      int code = 0;//to store output code
724      try{//using try & catch block
725          Statement statement = connection.createStatement();//creating statement object
726          ResultSet records = statement.executeQuery(query);//executing query
727          while(records.next()){//looping through records
728              if(records.getString("userName").equals(username)){//if username is found
729                  userFound = true;//set username to be true
730              }
731
732              //is username found and password is also found
733              if(userFound==true && records.getString("userPassword").equals(password)){
734                  passFound = true;//set password to be true
735              }
736          }
737          catch (SQLException e){//catch SQL exception
738              System.out.println("SQLEX : "+e.getMessage());//display sql exception
739          }
740
741          if(userFound==true && passFound==true){//is username and password are true
742              code = 1;//set output code to 1
743          } else if(userFound==false){//else if username is false
744              code = -1;//set output code to -1
745          } else if(userFound==true && passFound==false){//else if username is true & password is false
746              code = 2; //set output code to 2
747          }
748          return code;//return output code.
749      }

```

```

750
751  /**
752  * Used to add a user to database.
753  * @param username Example : "bruce7"
754  * @param email Example : "bruce7@gmail.com"
755  * @param password Example : "bruce12345"
756  * @param question Example : "Where did your parents meet?"
757  * @param answer Example : "London"
758  * @return returns output code
759  * MEANINGS OF OUTPUT CODE:
760  * 0 = neutral
761  * 1 = user can be added (everything is valid)
762  * 2 = user already exists
763  * @throws SQLException
764  */
765  @WebMethod(operationName="addUser")
766  public int addUser(@WebParam(name="username")String username,@WebParam(name="email") String email,
767      @WebParam(name="password")String password,@WebParam(name="q") String question, @WebParam(name="a")String answer){
768      String addingTime = generateTime();//current adding time
769      int message = 0;//output code
770      if(doesUserExist(username, password)==-1){//if user does not exist
771          message = 1;//set message to 1
772          //username, email, password, question, answer
773          String query = "insert into Users values "
774              + "("+"'+username+'','"+email+'','"+password+'','"+question+'','"+answer+'")";//insert query
775          int rowsadded;//rows added
776          try{//using try & catch block
777              Statement statement = connection.createStatement();//creating statement object
778              rowsadded = statement.executeUpdate(query);//executing query
779              String log_out_ADDUSER = "User added by : "+Server+" at : "+addingTime;//building log string
780              writeToLog(log_out_ADDUSER);//writing to log
781          } catch (SQLException e){//catching sql exception
782              System.out.println("SQLE : "+e.getMessage());//displaying exception message
783          }
784      }
785      } else{//else if user already exists
786          message = 2;//set message(output code) to 2
787      }
788
789      return message;//return output code(message)
790  }

```

```

793  /**
794   * Used to change email of a user.
795   * @param username Example : "bruce7"
796   * @param newEmail Example : "bruce12@gmail.com"
797   * @throws SQLException
798   */
799  @WebMethod(operationName="changeEmail")
800  public void changeEmail(@WebParam(name="username")String username, @WebParam(name="newEmail")String newEmail){
801      String addingTime = generateTime();//current adding time
802      int records;//record to be update
803      String query = "update Users set userEmail='"+newEmail+"' where userName='"+username+"'";//update query
804      try{//using try & catch block
805          Statement statement = connection.createStatement();//creating statment object
806          records = statement.executeUpdate(query);//executing query
807          String log_out_EMAILCHANGED = "Email changed by : "+username+" at : "+addingTime;//building LOG String
808          writeToLog(log_out_EMAILCHANGED);//writing to LOG
809      } catch (SQLException e){//catching SQL Exception
810          System.out.println("SQLException : "+e.toString());//displaying exception message
811      }
812  }
813
814
815  /**
816   * Used to change user password.
817   * @param username Example : "bruce7"
818   * @param password Example : "newpassword123!"
819   * @throws SQLException
820   */
821  @WebMethod(operationName="changePassword")
822  public void changePassword(@WebParam(name="username")String username,@WebParam(name="password") String password){
823      String addingTime = generateTime();//current adding time
824      int records;//current record
825      String query = "update Users set userPassword='"+password+"' where userName='"+username+"'";//update query
826      try{//using try & catch block
827          Statement statement = connection.createStatement();//creating statement object
828          records = statement.executeUpdate(query);//executing query
829      } catch (SQLException e){//catching SQL Exception
830          System.out.println("SQLException : "+e.toString());//displaying exception message
831      }
832      String log_out_PASSCHANGED = "Password changed by : "+username+" at : "+addingTime;//building LOG string
833      writeToLog(log_out_PASSCHANGED);//writing to LOG
834  }
835

```



```

837 ▾ /**
838      * Gets user email using their username
839      * @param username Example : "bruce7"
840      * @return email Example : "bruce7@gmail.com"
841      * @throws SQLException
842      */
843      @WebMethod(operationName="getUserEmail")
844 ▾ public String getUserEmail(@WebParam(name="username")String username){
845          String query = "select * from Users where username='"+username+"'"; //search query
846          String email=""; //to store email
847 ▾      try{//using try & catch block
848          Statement statement = connection.createStatement();//creating statement object
849          ResultSet records= statement.executeQuery(query);//executing query
850 ▾          while (records.next()){ //looping through records
851              email = records.getString("userEmail");//storing user email
852              userQuestion = records.getString("userSQuestion");//storing user security question
853              userAnswer = records.getString("userSAnswer");//storing user security answer
854          }
855          } catch (SQLException e){ //catching SQL exception
856              System.out.println("SQLException : "+e.toString()); //displaying exception message
857          }
858          return email; //returning user email.
859      }
860
861
862 ▾ /**
863      * Gets user security question
864      * @return Example : "Where were your parents born?"
865      */
866      @WebMethod(operationName="getUserQ")
867      public String getUserQ(){
868          return userQuestion; //returns user security question
869      }
870
871
872 ▾ /**
873      * Gets user security question answer
874      * @return Example : "London"
875      */
876      @WebMethod(operationName="getUserA")
877      public String getUserA(){
878          return userAnswer; //returns user security answer
879      }
880

```



```

882  /**
883   * Updates user security question and answer.
884   * @param username Example : "bruce7"
885   * @param question Example : "Where were you born?"
886   * @param answer Example : "Kenya"
887   * @throws SQLException
888   */
889   @WebMethod(operationName="updateSQA")
890   public void updateSQA(@WebParam(name="username")String username,@WebParam(name="question") String question,@WebParam(name="answer") String answer){
891       String addingTime = generateTime(); //current adding time
892       String query = "update Users set userSQuestion='"+question+"', userSAnswer='"+answer+"' where userName='"+username+"'"; //update query
893       int rows=0; //rows to be updated
894       try{
895           Statement statement = connection.createStatement(); //creating statement object
896           rows = statement.executeUpdate(query); //executing query
897       } catch (SQLException e){ //catching SQL exception
898           System.out.println("SQLEX : "+e.toString()); //displaying exception message
899       }
900       String log_out_SUPDATE = "Security question updated by : "+username+" at : "+addingTime; //building log string
901       writeToLog(log_out_SUPDATE); //writng to LOG
902   }
903
904
905  /**
906   * Counts and returns number of replies posted by a user.
907   * @param username Example : "bruce7"
908   * @return Example : 2
909   * @throws SQLException
910   */
911   @WebMethod(operationName="countReplies")
912   public int countReplies(@WebParam(name="username")String username){
913       int replies = 0; //counter variable
914       String query = "select * from Replies where replyAuthor='"+username+"'"; //search query
915       try{//using try & catch block
916           Statement statement = connection.createStatement(); //creating statement object
917           ResultSet records= statement.executeQuery(query); //executing query
918           while (records.next()){ //looping through records
919               replies++; //incrementing counter
920           }
921       } catch (SQLException e){ //catching SQLException
922           System.out.println("SQLException : "+e.toString()); //displayinh sql exception message
923       }
924       return replies; //returns number of replies
925   }

```

```

928 ▼ /**
929  * Counts and returns number of comments posted by a user.
930  * @param username Example : "peter1"
931  * @return Example : 4
932  * @throws SQLException
933  */
934  @WebMethod(operationName="countComments")
935 ▼ public int countComments(@WebParam(name="username")String username){
936      int comments = 0; //counter variable
937      String query = "select * from Comments where commentAuthor='"+username+"'"; //search query
938 ▼ try{//try & catch block
939          Statement statement = connection.createStatement();//creating statement object
940          ResultSet records= statement.executeQuery(query); //executing query
941          while (records.next()){ //looping through records
942              comments++; //incrementing comments
943          }
944      } catch (SQLException e){ //catching SQLException
945          System.out.println("SQLException : "+e.toString()); //displaying exception message
946      }
947      return comments; //returning number of comments posted by a user.
948  }
949
950
951 ▼ /**
952  * Counts and returns number of posts posted by a user.
953  * @param username Example : "bruce7"
954  * @return Example : 7
955  */
956  @WebMethod(operationName="countPosts")
957 ▼ public int countPosts(@WebParam(name="username")String username){
958      int posts = 0; //counter variable
959      String query = "select * from Posts where postAuthor='"+username+"'"; //search query
960 ▼ try{//using try & catch block
961          Statement statement = connection.createStatement();//creating statement object
962          ResultSet records= statement.executeQuery(query); //executing query
963          while (records.next()){ //looping through records
964              posts++; //incrementing posts counter
965          }
966      } catch (SQLException e){ //catching SQL Exception
967          System.out.println("SQLException : "+e.toString()); //displaying exception message
968      }
969      return posts; //returning number of posts posted by a user.
970  }

```

```

973  /**
974   * Resets user password.
975   * @param username Example : "bruce7"
976   * @param email Example : "bruce7@yahoo.com"
977   * @param question Example : "Where were your parents born?"
978   * @param answer Example : "London"
979   * @param newPassword Example : "brucecanfly"
980   * @return output code.
981   * OUTPUT CODE MEANINGS:
982   * 1 = everything is correct
983   * 0 = neutral
984   * -1 = user does not exist
985   * -2 = email does not match
986   * -3 = question does not match
987   * -4 = answer does not match
988   * @throws SQLException
989   */
990  @WebMethod(operationName="resetPassword")
991  public int resetPassword(@WebParam(name="username")String username,@WebParam(name="email") String email,
992                          @WebParam(name="question")String question,@WebParam(name="answer") String answer){
993      String addingTime = generateTime(); //gets current adding time & date
994      int code = 0; //Neutral
995
996      //Checking if username exists
997      if(doesUserExist(username, "null")==2){//if user exists
998          if(getUserEmail(username).equals(email)){//if email matches
999              if(getUserQ().equals(question)){//if question matches
1000                  if(getUserA().equals(answer)){//if answer matches
1001                      code = 1;//assign 1 to code
1002                  }else {code = -4;} //answer does not match
1003              } else {code = -3;} //question does not match
1004          } else {code = -2;} //email does not match
1005      } else {code = -1;} //user does not exist
1006
1007      return code;//returning OUTPUT code.
1008  }

```

```

1011  /**
1012   * Deletes user account/profile.
1013   * @param username Example : "bruce7"
1014   * @param password Example : "brucecanfly"
1015   * @return returns OUTPUT CODE:
1016   * OUTPUT CODE MEANINGS:
1017   * 0 - neutral
1018   * 1 - user exists
1019   * 2 - user does not exists
1020   */
1021  @WebMethod(operationName="deleteAccount")
1022  public int deleteAccount(@WebParam(name="username")String username,@WebParam(name="password") String password){
1023      String addingTime = generateTime();//
1024      int code = 0;//output code
1025      int records;//to execute update query
1026
1027      if(doesUserExist(username, password)==1){//if user exists
1028          code = 1;//assign 1 to code
1029      }else if(doesUserExist(username, password)==2){//else if not
1030          code = 2;//assign 2 to code
1031      }
1032
1033      if(code==1){//if user exists
1034          String query = "delete from Users where userName='"+username+"' and userPassword='"+password+"'";//delete query
1035          try{//try & catch block for SQL Exception
1036              Statement statement = connection.createStatement();//creating statement obeject
1037              records = statement.executeUpdate(query);//executing query
1038          } catch (SQLException e){//catching SQL Exception
1039              System.out.println("SQLException : "+e.toString()); //displaying exception message
1040          }
1041          String log_out_ACCDEL = "Account deleted by : "+username+" at : "+addingTime;//building LOG string
1042          writeToLog(log_out_ACCDEL);//writing to LOG
1043      }
1044      return code;//returning output CODE.
1045  }
1046
1047
1048
1049
1050

```

Topics.jsp

Used to display the topics.

```
1 <%--
2   Filename      : Topics
3   Author       : Aysham Hameed
4   Created      : Dec 7, 2018, 10:57:16 PM
5   Operating System : Windows 10
6   Version      : NetBeans IDE 8.2
7   Description   : Used to display the topics.
8 --%>
9
10 <html><!--HTML open tag-->
11 <head><!--head open tag-->
12   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"> <!--specify the type of document it is-->
13   <title>Topics</title><!--Topics as title of page-->
14   <link rel="stylesheet" href="CSS/headerStyle.css"><!--using headerStyle sheet for customizations-->
15   <link rel="stylesheet" href="CSS/topicsStyle.css"><!--using topicsStyle sheet for customizations-->
16 </head><!--head close tag-->
17
18 <body><!--body open tag-->
19
20   <!--HEADER & NAVIGATION BAR-->
21   <header><!--header open tag-->
22     <div class="container"><!--div tag-->
23       <a href="Topics.jsp"><!--link to home page-->
24         <!--displaying Logo-->
25       </a><!--link home page close tag-->
26
27       <nav><!--nav open tag-->
28         <ul><!--unordered list open tag-->
29           <li><a href="Topics.jsp">Home</a></li><!--Home button-->
30           <%
31             if(session.getAttribute("usernameSession")!=null){<!--If session is created
32               out.println("<li><a href='\"Topics.jsp?logout=true\"'>");<!--create hyperlink to homepage
33               String user = (String)session.getAttribute("usernameSession");<!--get username
34               out.println("Logout, "+user+"</a></li>");<!--display Logout button in navigation bar
35               out.println("<li><a href='\"Profile.jsp\"'>Profile</a></li>");<!--display Profile button in navigation bar
36             }else{<!--else if user session is not created then
37               session.setAttribute("pageName", "Topics");<!--create a session for page name(see Login page)
38               out.println("<li><a href='\"Login.jsp\"'>Login</a></li>");<!--display Login button
39             }
40           <%>
41         </ul><!--unordered list close tag-->
42       </nav><!--nav close tag-->
43     </div><!--div close tag-->
44   </header><!--header close tag-->
45
46   <!--MAIN HEADING-->
47   <div style="overflow-x: hidden;"><!--div open tag-->
48   <h1 id="mainHeading"><!--assigning id to main heading-->
49     Welcome to Fox forums!<!--main heading-->
50   </h1><!--main h1 heading close tag-->
51   </div><!--main heading div close tag-->
52
53   <!--LOGOUT BUTTON FUNCTIONALITY-->
54   <%
55     String logOUT = request.getParameter("logout")==null?"0":
56     request.getParameter("logout");//getting logout parameter from page url
57     if(logOUT.equals("true")){<!--if logout is true then
58       session.invalidate();<!--invalidate user session
59       response.sendRedirect("Topics.jsp");//and go to homepage
60     }
61   <%>
```

```

63 <!--INSTANTIATING WEB SERVICE-->
64 <%
65     serviceClient.ProcessPosts service = new serviceClient.ProcessPosts();//creating instance of client
66     serviceClient.Processor port = service.getProcessorPort();//creating instance of server
67 %>
68
69 <!--IMAGE TILES(10 FIXED TOPICS/CATEGORIES)-->
70 <div class="xop-section"><!--div open tag image tiles-->
71 <ul class="xop-grid"><!--unordered lists open tag image tiles-->
72
73 <!--TOPIC1-->
74 <li><!--topic 1 open-->
75 <a href="Posts.jsp?topic=Social" style="text-decoration: none;"><!--creating a custom link for Social-->
76 <div class="xop-box xop-img-1"><!--div open topic 1-->
77 <h3>Social</h3><!--topic 1 label-->
78 <p>Lets talk about things,<!--topic 1 description--><b
79     style="color: orange"> posts : <% out.println(port.getPostCount("Social"));%></b></p><!--number of posts -->
80 </div><!--div close topic 1-->
81 </a><!--topic 1 close link-->
82 </li><!--topic 1 close-->
83
84 <!--TOPIC2-->
85 <li><!--topic 2 open-->
86 <a href="Posts.jsp?topic=Apple" style="text-decoration: none;"><!--creating a custom link for Apple-->
87 <div class="xop-box xop-img-2"><!--div open topic 2-->
88 <h3>Apple</h3><!--topic 2 label-->
89 <p>All things Apple,<!--topic 2 description--><b
90     style="color: orange"> posts : <% out.println(port.getPostCount("Apple"));%></b></p><!--number of posts -->
91 </div><!--div close topic 2-->
92 </a><!--topic 2 close link-->
93 </li><!--topic 2 close-->
94
95 <!--TOPIC3-->
96 <li><!--topic 3 open-->
97 <a href="Posts.jsp?topic=Software" style="text-decoration: none;"><!--creating a custom link for Software-->
98 <div class="xop-box xop-img-3"><!--div open topic 3-->
99 <h3>Software</h3><!--topic 3 label-->
100 <p>Programs & programs,<!--topic 3 description--><b
101     style="color: orange"> posts : <% out.println(port.getPostCount("Software"));%></b></p><!--number of posts -->
102 </div><!--div close topic 3-->
103 </a><!--topic 3 close link-->
104 </li><!--topic 3 close-->
105
106 <!--TOPIC4-->
107 <li><!--topic 4 open-->
108 <a href="Posts.jsp?topic=Movies" style="text-decoration: none;"><!--creating a custom link for Movies-->
109 <div class="xop-box xop-img-4"><!--div open topic 4-->
110 <h3>Movies</h3><!--topic 4 label-->
111 <p>Everyone likes movies!,<!--topic 4 description--><b
112     style="color: orange"> posts : <% out.println(port.getPostCount("Movies"));%></b></p><!--number of posts -->
113 </div><!--div close topic 4-->
114 </a><!--topic 4 close link-->
115 </li><!--topic 4 open-->
116

```

```

117 <!--TOPIC5-->
118 <li><!--topic 5 open-->
119     <a href="Posts.jsp?topic=Books" style="text-decoration: none;"><!--creating a custom link for Books-->
120     <div class="xop-box xop-img-5"><!--div open topic 5-->
121         <h3>Books</h3><!--topic 5 label-->
122         <p>Lets read together,<!--topic 5 description--><b
123             style="color: orange"> posts : <% out.println(port.getPostCount("Books"));%></b></p><!--number of posts -->
124     </div><!--div close topic 5-->
125     </a><!--topic 5 close link-->
126 </li><!--topic 5 close-->
127
128 <!--TOPIC6-->
129 <li><!--topic 6 open-->
130     <a href="Posts.jsp?topic=Events" style="text-decoration: none;"><!--creating a custom link for Events-->
131     <div class="xop-box xop-img-6"><!--div open topic 6-->
132         <h3>Events</h3><!--topic 6 label-->
133         <p>For all events discussion,<!--topic 6 description--><b
134             style="color: orange"> posts : <% out.println(port.getPostCount("Events"));%></b></p><!--number of posts -->
135     </div><!--div close topic 6-->
136     </a><!--topic 6 close link-->
137 </li><!--topic 6 close-->
138
139 <!--TOPIC7-->
140 <li><!--topic 7 open-->
141     <a href="Posts.jsp?topic=Gaming" style="text-decoration: none;"><!--creating a custom link for Gaming-->
142     <div class="xop-box xop-img-7"><!--div open topic 7-->
143         <h3>Gaming</h3><!--topic 7 label-->
144         <p>Achievement unlocked?,<!--topic 7 description--><b
145             style="color: orange"> posts : <% out.println(port.getPostCount("Gaming"));%></b></p><!--number of posts -->
146     </div><!--div close topic 7-->
147     </a><!--topic 7 close link-->
148 </li><!--topic 7 close-->
149
150 <!--TOPIC8-->
151 <li><!--topic 8 open-->
152     <a href="Posts.jsp?topic=Hacking" style="text-decoration: none;"><!--creating a custom link for Hacking-->
153     <div class="xop-box xop-img-8"><!--div open topic 8-->
154         <h3>Hacking</h3><!--topic 8 label-->
155         <p>Hacking make you feel cool,<!--topic 8 description--><b
156             style="color: orange"> posts : <% out.println(port.getPostCount("Hacking"));%></b></p><!--number of posts -->
157     </div><!--div close topic 8-->
158     </a><!--topic 8 close link-->
159 </li><!--topic 8 close-->
160
161 <!--TOPIC9-->
162 <li><!--topic 9 open-->
163     <a href="Posts.jsp?topic=Coding" style="text-decoration: none;"><!--creating a custom link for Coding-->
164     <div class="xop-box xop-img-9"><!--div open topic 9-->
165         <h3>Coding</h3><!--topic 9 label-->
166         <p>Source code, problems & more,<!--topic 9 description--><b
167             style="color: orange"> posts : <% out.println(port.getPostCount("Coding"));%></b></p><!--number of posts -->
168     </div><!--div close topic 9-->
169     </a><!--topic 9 close link-->
170 </li><!--topic 9 close-->
171

```

```

172         <!--TOPIC10-->
173         <li><!--topic 10 open-->
174             <a href="Posts.jsp?topic=Science" style="text-decoration: none;"><!--creating a custom link for Science-->
175             <div class="xop-box xop-img-10"><!--div open topic 10-->
176                 <h3>Science</h3><!--topic 10 label-->
177                 <p>Science is fascinating,<!--topic 10 description--><b
178                     style="color: orange"> posts : <% out.println(port.getPostCount("Science"));%></b></p><!--number of posts -->
179             </div><!--div close topic 10-->
180             </a> <!--topic 10 close link-->
181         </li><!--topic 10 close-->
182
183         </ul><!--Unordered list close-->
184     </div><!--div close tag image tiles-->
185
186 </body><!--body close tag-->
187 </html><!--HTML close tag-->
188

```


Subject.jsp

Displays subject (a specific post & its, comments & replies).

```
1  <%--
2      Filename      : Subject
3      Author       : Aysham Hameed
4      Created      : Dec 7, 2018, 10:57:16 PM
5      Operating System : Windows 10
6      Version      : NetBeans IDE 8.2
7      Description   : Displays subject.
8  --%>
9
10
11 <html><!--HTML open tag-->
12 <head><!--head open tag-->
13     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"><!--identify type of doc-->
14     <title>Subject</title><!--page title-->
15     <!--Using multiple style sheets-->
16     <link rel="stylesheet" href="CSS/headerStyle.css"><!--headerStyle sheet-->
17     <link rel="stylesheet" href="CSS/topicsStyle.css"><!--topicsStyle sheet-->
18     <link rel="stylesheet" href="CSS/subjectStyle.css"><!--subjectStyle sheet-->
19 </head><!--head close tag-->
20
21 <body><!--body open tag-->
22
23     <!--HEADER & NAVIGATION BAR-->
24     <header><!--header open tag-->
25         <div class="container"><!--div tag-->
26             <a href="Topics.jsp"><!--link to home page-->
27                 <!--displaying logo-->
28             </a><!--link home page close tag-->
29
30             <nav><!--nav open tag-->
31                 <ul><!--unordered list open tag-->
32                     <li><a href="Topics.jsp">Home</a></li><!--Home button-->
33                     <%
34                         if(session.getAttribute("usernameSession")!=null){//If session is created
35                             out.println("<li><a href='\"Topics.jsp?logout=true\"'>");//create hyperlink to homepage
36                             String user = (String)session.getAttribute("usernameSession");//get username
37                             out.println("Logout, "+user+"</a></li>");//display Logout button in navigation bar
38                             out.println("<li><a href='\"Profile.jsp\">Profile</a></li>");//display Profile button in navigation bar
39                         }else{//else if user session is not created then
40                             session.setAttribute("pageName", "Topics");//create a session for page name(see Login page)
41                             out.println("<li><a href='\"Login.jsp\">Login</a></li>");//display Login button
42                         }
43                     <%>
44                 </ul><!--unordered list close tag-->
45             </nav><!--nav close tag-->
46         </div><!--div close tag-->
47     </header><!--header close tag-->
48
49
50
```



```

51 <!--INSTANTIATING WEB SERVICE-->
52 <%
53     serviceClient.ProcessPosts service = new serviceClient.ProcessPosts();//creating instance of client
54     serviceClient.Processor port = service.getProcessorPort();//creating instance of server
55
56     String postid = (String) request.getParameter("postID");//Getting postID from Posts.jsp session object
57     String user="";//used to get login name
58 %>
59
60 <!-- <div SUBJECT HEADING-->
61 <div style="overflow-x: hidden;"><!--div open tag-->
62 <h1 class="subjectHeading"><!--assigning id to subject heading-->
63 <%
64     //Getting topic name first from session created Posts page
65     String topicName = (String) session.getAttribute("topicChosen");
66     //Then getting the post title based on topic and post id
67     String postTitle = port.getPostTitle(topicName, Integer.parseInt(postid));
68     //Printing post title
69     out.println(postTitle);
70
71 %>
72 </h1><!--subject h1 heading close tag-->
73 </div><!--subject heading div close tag-->
74
75
76
77 <!--SUBJECT DESCRIPTION-->
78 <div class="subjectDescription"><!--subject description div open-->
79 <p><!--paragraph tag open-->
80 <b>posted by : <%//post author
81     out.println(port.findPostAuthor(Integer.parseInt(postid)));//displaying post author
82     %> , <% out.println(port.getPostDateTime(Integer.parseInt(postid)));//displaying post date+time %></b> <br>
83     <% out.println(port.getPostDescription(Integer.parseInt(postid)));//displaying post description %>
84 </p><!--paragraph tag close-->
85 </div><!--subject description div close-->
86
87 <!--DISPLAYING COMMENTS-->
88 <div class="subjectDescription"><!--div container open-->
89 <table><!--table tag open-->
90 <%
91     //Calling this method to get the comment data from database.
92     port.getCommentsData(Integer.parseInt(postid));
93     //How many comments related to that post id
94     int commentCount = port.getCommentCount(Integer.parseInt(postid));
95     //So it can be accessed outside loop.
96     int commentIdentity=0;
97     //Displaying comment data
98     for (int x=0; x<commentCount; x++){
99         out.println("<tr >");//table row
100         out.println("<td id=\"commentBox\">");//table data with id

```

```

101 String commentAuthor = port.getCommentAuthors().get(x); //storing comment author
102 String commentDescription = port.getCommentDescription().get(x); //storing comment description
103 String commentTime = port.getCommentTimes().get(x); //storing comment times
104 int commentID = port.getCommentIDs().get(x); //storing comment ids
105
106 //comment identity (using composite keys)
107 commentIdentity=port.findCommentIdentity(commentID,commentAuthor, Integer.parseInt(postid));
108 out.println("<b>"+commentAuthor+", <i>"+commentTime+"</i></b> <br>"); //displaying in table
109 out.println("<p id=\"cdes\">"+commentDescription+"</p><br>"); //displaying in table
110
111 //appending comment id to page url AND%
112 <!--CREATING REPLY FORM IN COMMENT SECTION-->
113 <form action=<%out.println("Subject.jsp?postID="+postid+"&commentID="+commentIdentity+"&button=breply");%> method="post"><!--form open-->
114     <textarea name="replyText" cols="80" rows="1" wrap="soft" id="replyTextArea">Click here to type!</textarea><!--text area to type-->
115     <input type="submit" value="Reply" id="replyButton"/><!--reply button-->
116 </form><!--form close--><%
117
118 out.println("</b>"); //closing bold tag
119 out.println("</td>"); //closing table data tag
120 //-----
121
122 //Getting reply data for every single comment
123 port.getReplyData(commentIdentity);
124
125 //reply count of every comment
126 int replies = port.getReplyCount(port.findCommentIdentity(commentID, commentAuthor, Integer.parseInt(postid)));
127
128 //DISPLAYING REPLIES=====
129 for (int i=0; i<replies; i++){
130     out.println("<table id=\"replyBox\">"); //creating a table open
131     out.println("<tr>"); //table row open
132     out.println("<td >"); //table data open
133     String replyDescriptions = port.getReplyDescriptions().get(i); //getting reply desc
134     String replyAuthor = port.getReplyAuthors().get(i); //getting reply author
135     String replyTime = port.getReplyTimes().get(i); //getting reply time
136     out.println("<b>"+replyAuthor+", <i>"+replyTime+"</i></b> <br>"); //displaying in table
137     out.println(replyDescriptions); //displaying in table
138     out.println("</td>"); //table data close
139     out.println("</tr>"); //table row close
140     out.println("<table>"); //table close
141     } //===== reply loop end
142
143 out.println("</tr>"); //table row end
144 } //===== comment loop end
145
146 //IF REPLY BUTTON IS CLICKED.
147 String replyB = request.getParameter("button")==null ? "0":request.getParameter("button"); //gets paramater from url
148 if("POST".equals(request.getMethod()) && replyB.equals("breply")){ //if it is clicked then
149
150

```

```

151
152         if(request.getSession().getAttribute("usernameSession") != null){//if user is logged in
153         user = (String) session.getAttribute("usernameSession");//get username
154
155         String REPLY_AUTHOR = user;//assign username to reply author
156
157         String COMMENT_IDENTITY = (request.getParameter("commentID")==null)? "0":
158         request.getParameter("commentID");//get comment identity from url
159
160         String REPLY_TEXT = (request.getParameter("replyText")==null)? " " :
161         request.getParameter("replyText");//get reply text from url
162
163
164         //posting (saving to database)
165         port.addReply(REPLY_TEXT, REPLY_AUTHOR, Integer.parseInt(COMMENT_IDENTITY));
166         //redirecting page to correct destination
167         response.sendRedirect("Subject.jsp?postID="+postid);
168
169     }else { // else if not logged in
170         session.setAttribute("pageName", "Reply");//create pageName session to Reply (see Login page)
171         response.sendRedirect("Login.jsp");//go to login page
172     }
173 }
174
175 %>
176
177 </table><!--table close-->
178 </div><!--div close-->
179
180 <!--COMMENT AREA-->
181 <div id="tarea"><!--div open-->
182 <form action=<%out.println("Subject.jsp?postID="+postid+"&button=bcomment");%> method="POST"><!--form open-->
183 <textarea name="text" cols="5" rows="5" wrap="hard" autofocus>Type something...</textarea><!--text area open & close-->
184 <input type="submit" value="Comment" id="commentButton"/><!--input-->
185 </form><!--form close-->
186 </div><!--div close-->
187
188
189 <%
190
191 //gets type of button pressed from url
192 String click = request.getParameter("button")==null ? "0" : request.getParameter("button");
193 //IF COMMENT BUTTON IS CLICKED
194 if("POST".equals(request.getMethod()) && click.equals("bcomment")){
195     //First, check if user is logged in or not.
196     //if user is logged in then
197     if(request.getSession().getAttribute("usernameSession") != null){
198         user = (String) session.getAttribute("usernameSession");
199
200         String COMMENT_AUTHOR = user;//assign comment author to username

```

```

201
202 String COMMENT_DESC = (request.getParameter("text")==null)? "0":
203 request.getParameter("text");//get comment description from text area
204
205 String POST_ID = (request.getParameter("postID")==null)? " ":
206 request.getParameter("postID");//get comment post id from url
207
208
209 //out.println("<h1>"+REPLY_AUTHOR+" "+COMMENT_IDENTITY+" "+REPLY_TEXT+"</h1>");
210 //add comment to database
211 port.addComment(COMMENT_AUTHOR, COMMENT_DESC, Integer.parseInt(POST_ID));
212
213 //Refreshing the page
214 response.sendRedirect("Subject.jsp?postID="+postid);
215 }else {//if user is logged in then
216     session.setAttribute("pageName", "Comments");//create pageName session to Reply (see Login page)
217     response.sendRedirect("Login.jsp");//go to login page
218 }
219 }
220 %>
221
222 </body><!--end body tag-->
223 </html><!--end HTML tag-->
224

```

Profile.jsp

Used to display user profile.

```
1 <%--
2     Filename       : Profile
3     Author        : Aysham Hameed
4     Created       : Dec 7, 2018, 10:57:16 PM
5     Operating System : Windows 10
6     Version       : NetBeans IDE 8.2
7     Description    : Used to diplay user profile.
8 --%>
9
10 <%@page contentType="text/html" pageEncoding="UTF-8"%><!--type of JSP content-->
11 <html><!--HTML open tag-->
12 <head><!--head open tag-->
13     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"> <!--specify the type of document it is-->
14     <title>Profile</title><!--title of page-->
15     <!--USING CSS STYLESHEETS-->
16     <link rel="stylesheet" href="CSS/headerStyle.css"><!--sheet1-->
17     <link rel="stylesheet" href="CSS/profileStyle.css"><!--sheet2-->
18 </head><!--head close-->
19
20 <body><!--body open-->
21
22     <!--HEADER & NAVIGATION BAR-->
23     <!--HEADER & NAVIGATION BAR-->
24     <header><!--header open-->
25         <div class="container"><!--div open-->
26             <a href="Topics.jsp"><!--Link to homepage-->
27                 <!--Logo image-->
28             </a><!--Link close-->
29
30         <nav><!--nav open-->
31             <ul><!--unordered list open-->
32                 <li><a href="Topics.jsp">Home</a></li><!--home button-->
33
34                 <% //Instantiating web service
35                     serviceClient.ProcessPosts service = new serviceClient.ProcessPosts();
36                     serviceClient.Processor port = service.getProcessorPort();
37
38                     //If user is Logged in
39                     if(session.getAttribute("usernameSession")!=null){
40                         //create logout link
41                         out.println("<li><a href='\"Topics.jsp?logout=true\"'>");
42                         //get username
43                         String user = (String)session.getAttribute("usernameSession");
44                         //display logout button and username together
45                         out.println("Logout, "+user+"</a></li>");
46                         //display profile button
47                         out.println("<li><a href='\"#\">Profile</a></li>");
48                     }else{//else if user is not logged in
49                         //create pageName session (see Login page)
50                         session.setAttribute("pageName", "Topics");
```

```

51 //display login button
52 out.println("<li><a href=\"Login.jsp\">Login</a></li>");
53 }
54 %>
55 </ul><!--unordered list close-->
56 </nav><!--nav close-->
57 </div><!--div close-->
58 </header><!--header close-->
59
60 <!--getting username-->
61 <% String username = (String) session.getAttribute("usernameSession");%>
62
63 <div class="profilebox"><!--div open (profile box)-->
64 <!--just temp image-->
65 <h1><% out.println(username); %></h1><!--displaying username-->
66 <h5><% out.println(port.getUserEmail(username)); %> </h5><!--displaying email-->
67 <p>Posted : <% out.println(port.countPosts(username)); %> times</p><!--displaying num posts-->
68 <p>Commented : <% out.println(port.countComments(username)); %> times</p><!--displaying num comments-->
69 <p>Replied : <% out.println(port.countReplies(username)); %> times</p><!--displaying num replies-->
70 <hr><!--horizontal line-->
71 <p><b>Security Question :</b> <% out.println(port.getUserQ()); %></p><!--displaying security question-->
72 <p><b>Security Answer :</b> <% out.println(port.getUserA()); %></p><!--displaying answer question-->
73 <hr><!--horizontal-->
74 <!--link to change email page-->
75 <p><b><a href="changeEmail.jsp">Change email</a></b></p>
76 <!--link to change password page-->
77 <p><b><a href="changePassword.jsp">Change password</a></b></p>
78 <!--link to change security question page-->
79 <p><b><a href="changeSecurityQuestion.jsp">Change security question</a></b></p>
80 <hr><!--horizontal line-->
81 <!--link to delete account page-->
82 <p><b><a href="deleteAccount.jsp">Delete account</a></b></p>
83 </div><!--div close-->
84
85 </body><!--body close-->
86 </html><!--html close-->
87

```

Posts.jsp

Used to display relevant posts.

```
1 <%--
2     Filename           : Posts
3     Author            : Aysham Hameed
4     Created           : Dec 7, 2018, 10:57:16 PM
5     Operating System  : Windows 10
6     Version           : NetBeans IDE 8.2
7     Description       : Used to display relevant post.
8 --%>
9
10 <html><!--html open-->
11 <head><!--head open-->
12     <!--meta data for html page-->
13     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
14     <title>Posts</title><!--page title-->
15     <!--Using multiple style sheets-->
16     <link rel="stylesheet" href="CSS/headerStyle.css"><!--sheet 1-->
17     <link rel="stylesheet" href="CSS/topicsStyle.css"><!--sheet 2-->
18     <link rel="stylesheet" href="CSS/tableStyle.css"><!--sheet 3-->
19 </head><!--head close-->
20
21 <body><!--body open-->
22
23     <!--HEADER & NAVIGATION BAR-->
24     <header><!--header open-->
25         <div class="container"><!--div open-->
26             <a href="Topics.jsp"><!--Link to homepage-->
27                 <!--Logo image-->
28             </a><!--Link close-->
29
30             <nav><!--nav open-->
31                 <ul><!--unordered list open-->
32                     <li><a href="Topics.jsp">Home</a></li><!--home button-->
33
34                     <% //Instantiating web service
35                         serviceClient.ProcessPosts service = new serviceClient.ProcessPosts();
36                         serviceClient.Processor port = service.getProcessorPort();
37
38                         //If user is logged in
39                         if(session.getAttribute("usernameSession")!=null){
40                             //create logout link
41                             out.println("<li><a href='\"Topics.jsp?logout=true\"'>");
42                             //get username
43                             String user = (String)session.getAttribute("usernameSession");
44                             //display logout button and username together
45                             out.println("Logout, "+user+"</a></li>");
46                             //display profile button
47                             out.println("<li><a href='\"Profile.jsp\"'>Profile</a></li>");
48                         }else{//else if user is not logged in
49                             //create pageName session (see Login page)
50                             session.setAttribute("pageName", "Topics");
```



```

51         //display login button
52         out.println("<li><a href=\"Login.jsp\">Login</a></li>");
53     }
54     %>
55     </ul><!--unordered list close-->
56     </nav><!--nav close-->
57     </div><!--div close-->
58     </header><!--header close-->
59
60
61     <!--LOG OUT FUNCTIONALITY-->
62     <%
63         //first get logout parameter from url
64         String logOUT = request.getParameter("logout")==null?"0":request.getParameter("logout");
65         if(logOUT.equals("true")){ //if logout equals true
66             session.invalidate(); //then delete user session
67             response.sendRedirect("Topics.jsp"); //go to home page
68         }
69     %>
70
71
72     <!--MAIN HEADING-->
73     <div style="overflow-x: hidden;"><!--div open-->
74     <h1 id="mainHeading"><!--heading open-->
75         Forum Topic : <!--forum topic-->
76     <!--displaying forum topic
77         String sTopic = request.getParameter("topic");//getting topic from url
78         out.println(sTopic); //printing to console (testing)
79         session.setAttribute("topicChosen", sTopic); //creating & saving to session
80     %>
81     </h1><!--heading close-->
82     </div><!--div close-->
83
84
85
86     <!--GOES TO LOGIN PAGE-->
87     <%
88         //If there is user logged in (user session created)
89         if (request.getSession().getAttribute("usernameSession") != null) {
90             //Then check that the user session matches the username
91             if(session.getAttribute("usernameSession").equals(session.getAttribute("usernameSession"))){
92                 //create make post button
93                 out.println("<div>");
94                 out.println("<h2 id=\"postButton\"><a href=\"MakePost.jsp\">Post</a></h2>");
95                 out.println("</div>");
96             }
97         } else{
98
99             //If there is no user logged in then go to the login page.
100             out.println("<div>");//open div

```



```

101 //create post button with link to login page
102 out.println("<h2 id=\"postButton\"><a href=\"Login.jsp\">Post</a></h2>");
103 out.println("</div>");//div close
104 //create pageName session and set to Posts (see Login.jsp)
105 session.setAttribute("pageName","Posts");
106
107 }
108 %>
109
110
111
112
113 <!--TABLE-->
114 <center><!--center open tag-->
115 <div style="overflow-x: auto;" id="box"><!--div open-->
116 <table><!--table open-->
117
118 <!--DISPLAYING ALL POSTS BASED ON TOPIC IN TABLE-->
119 <%
120 //gettin topic first
121 port.getPostsData(sTopic);
122 //counting number of posts
123 int posts = port.getPostCount(sTopic);
124 //if number of posts is less than 1 then
125 if(posts<1){
126 //display topic is empty!
127 out.println("<h2 style=\"color:white;\">This topic is empty!</h2>");
128 } else { //else
129
130 //create table layout
131 out.println("<tr>");//table row open
132 out.println("<th>Subject</th>");//subject heading
133 out.println("<th>Posted by</th>");//posted by heading
134 out.println("</tr>");//table row close
135
136
137 for(int x=0; x<posts; x++){
138 out.println("<tr>");//table row open
139 //getting post name
140 String postName = port.getSubjects().get(x);
141 System.out.println(postName);
142 //getting post author
143 String postAuthor = port.getAuthors().get(x);
144 //getting post ID (converting int to String)
145 String postID = String.valueOf(port.getPostID().get(x));
146
147 //Creating a post url to get postID parameter
148 String postURL = "?postID="+postID;
149
150 //Creating a session for postURL
151 session.setAttribute("postPAGE_URL", postURL);
152
153 //Displaying all clickable posts in table
154 out.println("<td><a href=\"Subject.jsp\"+postURL+\"\" >");
155 out.println(postName);//name of the post
156 out.println("</a></td>");//can make href
157
158 //Displaying post author columns
159 out.println("<td>"+postAuthor+"</td>");//displaying author
160 out.println("</tr>");//table row end
161 }
162 }
163
164 %>
165
166 </table><!--table close-->
167 </div><!--div close-->
168 </center><!--center close-->
169
170 </body><!--body close-->
171 </html><!--html close-->
172

```

MakePost.jsp

Allows user to make a post.

```
1 <!--
2     Filename      : Make Post
3     Author       : Aysham Hameed
4     Created      : Dec 7, 2018, 10:57:16 PM
5     Operating System : Windows 10
6     Version      : NetBeans IDE 8.2
7     Description   : Allows user to make a post.
8 --%>
9
10 <!--specifying the type of jsp content-->
11 <%@page contentType="text/html" pageEncoding="UTF-8"%>
12 <html><!--html open-->
13 <head><!--head open-->
14     <title>Make Post</title><!--page title-->
15     <!--using multiple style sheets-->
16     <link rel="stylesheet" href="CSS/topicsStyle.css"><!--sheet 1-->
17     <link rel="stylesheet" href="CSS/headerStyle.css"><!--sheet 2-->
18     <link rel="stylesheet" href="CSS/postPageStyle.css"><!--sheet 3-->
19 </head><!--head close-->
20
21
22 <body><!--body open-->
23
24     <!--HEADER & NAVIGATION BAR-->
25     <header><!--header open-->
26         <div class="container"><!--div open-->
27             <a href="Topics.jsp"><!--link to home page-->
28                 <!--page logo-->
29             </a><!--link close-->
30
31         <nav><!--nav open-->
32             <ul><!--unordered list open-->
33                 <li><a href="Topics.jsp">Home</a></li><!--home button-->
34
35                 <% //Instantiating web service
36                     serviceClient.ProcessPosts service = new serviceClient.ProcessPosts();
37                     serviceClient.Processor port = service.getProcessorPort();
38
39                     //If user is logged in
40                     if(session.getAttribute("usernameSession")!=null){
41                         //create logout link
42                         out.println("<li><a href='\"Topics.jsp?logout=true\"'>");
43                         //get username
44                         String user = (String)session.getAttribute("usernameSession");
45                         //display logout button and username together
46                         out.println("Logout, "+user+"</a></li>");
47                         //display profile button
48                         out.println("<li><a href='\"Profile.jsp\">Profile</a></li>");
49                     }else{//else if user is not logged in
50                         //create pageName session (see Login page)
```

```

51         session.setAttribute("pageName", "Topics");
52         //display Login button
53         out.println("<li><a href=\"Login.jsp\">Login</a></li>");
54     }
55 }
56
57     </ul><!--unordered list close-->
58     </nav><!--nav close-->
59     </div><!--div close-->
60 </header><!--header close-->
61
62 <!--FORM-->
63 <form class="postForm" action="MakePost.jsp" method="post"><!--form open-->
64     <h1>Create a post</h1><!--heading in form-->
65     <input type="text" name="postSubject" placeholder="Title" /><!--title field-->
66     <textarea name="postText">Text...</textarea><!--text area-->
67     <input type="submit" value="Post" /><!--submit button-->
68
69
70 <!--JSP FUNCTIONALITY-->
71 <%
72     //if post button is clicked
73     if("POST".equals(request.getMethod())){
74         // subject, text, author, topicID
75         //get subject from form
76         String subject = request.getParameter("postSubject")==null? "Title" : request.getParameter("postSubject");
77
78         //get text from form
79         String text = request.getParameter("postText")==null? "Text" : request.getParameter("postText");
80         //get username (author) from user session
81         String author = (String) session.getAttribute("usernameSession")==null? "Author" : (String) session.getAttribute("usernameSession");
82         //get topic id from user session
83         int topicID = port.getTopicID((String)session.getAttribute("topicChosen"));
84
85         if(!subject.equals("") && !text.equals("")){ //post/add post to database
86             port.addPost(subject, text, author, topicID);
87             //go to posts while still have chosen topic
88             String redirectURL = "Posts.jsp?topic="+ (String)session.getAttribute("topicChosen");
89             response.sendRedirect(redirectURL); //go to redirect url
90         }else{
91             out.println("<h4 style=\"color:white; font-size:11; text-align:center;\">Missing information.</h4>");
92         }
93     }
94 }
95
96 %>
97 </form><!--form close-->
98 </body><!--body close-->
99 </html><!--html close-->
100

```

Login.jsp

Login JSP file, used for user login.

```
1 <%--
2   Filename      : Login
3   Author       : Aysham Hameed
4   Created      : Dec 7, 2018, 10:36:16 PM
5   Operating System : Windows 10
6   Version      : NetBeans IDE 8.2
7   Description   : Login JSP file, used for user login.
8 --%>
9
10
11 <html><!--html open-->
12 <head><!--head open-->
13   <!--meta data for html document-->
14   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
15   <title>Login</title><!--title of page-->
16   <!--using multiple style sheets-->
17   <link rel="stylesheet" href="CSS/headerStyle.css"><!--sheet 1-->
18   <link rel="stylesheet" href="CSS/loginStyle.css"><!--sheet 2-->
19 </head><!--head close-->
20
21 <body><!--body open-->
22   <!--HEADER & NAVIGATION BAR-->
23   <header><!--header open-->
24     <div class="container"><!--div open-->
25       <a href="Topics.jsp"><!--link to home page-->
26         <!--page logo-->
27       </a><!--link close-->
28
29       <nav><!--nav open-->
30         <ul><!--unordered list open-->
31           <li><a href="Topics.jsp">Home</a></li><!--home page button-->
32         </ul><!--unordered list close-->
33       </nav><!--nav close-->
34
35     </div><!--div close-->
36   </header><!--header close-->
37
38
39   <!--LOGIN FORM-->
40   <form class="loginForm" action="Login.jsp" method="post"><!--form open-->
41     <h1>Login</h1><!--heading inside form-->
42     <input type="text" name="userName" placeholder="Username"><!--username field-->
43     <input type="password" name="userPassword" placeholder="Password"><!--password field-->
44     <input type="submit" value="Login"><!--login button-->
45     <!--FORGOT OR SIGN UP OPTIONS-->
46     <p style="color: white"><!--style -->
47       <!--link to forgot password page-->
48       <a href="passwordForgotten.jsp" style="text-decoration: none; color: white">Forgot password?</a> or
49       <!--link to join page-->
50       <a href="Join.jsp" style="text-decoration: none; color: white"><b>Join</b></a>
```

```

51      </p> <!--paragraph end-->
52
53
54
55      <!--JSP FUNCTIONALITY-->
56      <%
57          String username = null;//to store username
58          String password = null;//to store password
59          ##VALIDATION-1 [Checks if the username or password is null or not,
60          //if are null then get their values.]
61
62          ##VALIDATION-2 [Checks to see if the username & passwords match
63          //in the database then redirect the user to the Topics page.]
64          //Gets username from login form. Check #VALIDATION-1
65
66          boolean usernameValid = false;//flag variable for username field
67          boolean passwordValid = false;//flag variable for password field
68          //getting username from form
69          username = request.getParameter("userName")==null ? "" : request.getParameter("userName");
70          //getting password from form
71          password = request.getParameter("userPassword")==null ? "" : request.getParameter("userPassword");
72
73          //if username is not blank and lenght is greater than 5 characters
74          if(!username.equals("") && username.length()>5){
75              usernameValid = true;//then make usernameValid true
76          }
77
78          //if password is not blank and length is greater than 7 characters
79          if(!password.equals("") && password.length()>7){
80              passwordValid = true;//then make passwordValid true
81          }
82
83          //PROVIDING VERY SPECIFIC ERRORS TO USER
84          String usernameError = "";//to make username error message
85          String passwordError = "";//to make password error message
86
87          if(username.equals("")){//if username is blank
88              usernameError = "Username cannot be blank!";//display error
89          }
90
91          if(password.equals("")){//if password is blank
92              passwordError = "Password cannot be blank!";//display error
93          }
94
95          if(!username.equals("")){//if username is not blank but
96              if(username.length()<5){//username length is less than 5
97                  //then display error message
98                  usernameError = "Username incorrect, must be at least 6 characters long!";
99              }
100          }

```

```

101
102 ▼ if(!password.equals("")){//if password is not blank but
103 ▼ if(password.length()<7){//password length is less than 5
104 //then display error message
105 passwordError = "Password incorrect, must be at least 8 characters long!";
106 }
107 }
108
109
110
111
112 String ptopic = null;
113 //instantiating web service
114 serviceClient.ProcessPosts service = new serviceClient.ProcessPosts();
115 serviceClient.Processor port = service.getProcessorPort();
116 //1 = exists
117 //-1 = does not exist
118 //2 = wrong password
119
120 //If Login button is pressed
121 ▼ if("POST".equals(request.getMethod())){
122 //if username field & password field is valid then
123 ▼ if(usernameValid && passwordValid){
124 ▼ //If username and password matches
125 ▼ if(port.doesUserExist(username, password)==1){
126 //Stores username in session
127 session.setAttribute("usernameSession", username);
128
129 //decide which page to go to
130 String pageName = (String) session.getAttribute("pageName")==null? "0" :
131 (String) session.getAttribute("pageName");
132
133 //So if you want to login directly like on homepage
134 if(pageName.equals("Topics.jsp")){
135 response.sendRedirect("Topics.jsp");
136 }
137
138 //If page name = Comments, go to comments page
139 ▼ if(pageName.equals("Comments")){
140 //Creating a page url to redirect to after the user logs in
141 String pageURL = "Subject.jsp"+(String)session.getAttribute("postPAGE_URL");
142 response.sendRedirect(pageURL);
143 }
144
145 //If page name = Posts, go to posts page
146 ▼ if(pageName.equals("Posts")){
147 String pageURL = "Posts.jsp?topic="+(String)session.getAttribute("topicChosen");
148 response.sendRedirect(pageURL);
149 }
150

```

```

151 //If page name = Reply, go to reply page
152 if(pageName.equals("Reply")){
153     String pageURL = "Subject.jsp" + (String)session.getAttribute("postPAGE_URL") + "&user=" + username;
154     response.sendRedirect(pageURL);
155 }
156
157 //If page name = Topics, got to topics page
158 if(pageName.equals("Topics")){
159     String pageURL = "Topics.jsp?";
160     response.sendRedirect(pageURL);
161 }
162
163 }
164 else
165 //else if user does not exist
166 if(port.doesUserExist(username, password)==-1){
167     response.sendRedirect("UserNotFound.html");//got to error page
168 }
169 else
170 //else if password is wrong
171 if(port.doesUserExist(username, password)==2){
172     response.sendRedirect("InvalidPassword.html");// go to error page
173 }
174 }else {
175     //else if username field and password field are not valid then
176     //display username error
177     out.println("<p style=\"color:white; font-size:10pt; text-align:center; \>"+usernameError+"</p>");
178     //display password errors
179     out.println("<p style=\"color:white; font-size:10pt; text-align:center; \>"+passwordError+"</p>");
180 }
181
182 }
183
184 %>
185 </form><!--form close-->
186
187 </body><!--body close-->
188 </html><!--html close-->
189

```


Join.jsp

Used to register a user.

```
1 <%--
2     Filename      : Join
3     Author       : Aysham Hameed
4     Created      : Dec 9, 2018, 12:43:16 PM
5     Operating System : Windows 10
6     Version      : NetBeans IDE 8.2
7     Description   : Used to register a user.
8 --%>
9
10 <%@page import="java.util.regex.Pattern"%><!--import regex to verify email-->
11 <html><!--html open-->
12 <head><!--head open-->
13     <title>Join</title><!--page title-->
14     <!--using multiple style sheets-->
15     <link rel="stylesheet" href="CSS/headerStyle.css"><!--sheet 1-->
16     <link rel="stylesheet" href="CSS/joinStyle.css"><!--sheet 2-->
17 </head><!--head close-->
18
19
20 <body><!--body open-->
21
22     <!--HEADER & NAVIGATION BAR-->
23     <header><!--header open-->
24         <div class="container"><!--div open-->
25             <a href="Topics.jsp"><!--link to home page-->
26                 <!--page Logo-->
27             </a><!--link close-->
28
29             <nav><!--nav open-->
30                 <ul><!--unordered list-->
31                     <li><a href="Topics.jsp">Home</a></li><!--home button link-->
32                     <li><a href="Login.jsp">Login</a></li><!--Login button link-->
33                 </ul><!--unordered list close-->
34             </nav><!--nav close-->
35
36         </div><!--div close-->
37     </header><!--header close-->
38
39
40
41
42
43     <!--JOIN FORM-->
44     <form class="joinForm" action="Join.jsp" method="post"><!--form open-->
45         <h1>JOIN</h1><!--heading inside form-->
46         <!--username field-->
47         <input type="text" name="userName" placeholder="Username">
48         <!--password field-->
49         <input type="password" name="userPassword1" placeholder="Password">
50     </form>
```

```

51 <!--confirm password field-->
52 <input type="password" name="userPassword2" placeholder="Confirm Password">
53 <!--email field-->
54 <input type="text" name="userEmail" placeholder="Email">
55 <!--combo box field-->
56 ▼ <select id="cbox" name="userSQuestion"><!--options-->
57 ▼   <option selected="true" disabled>Security Questions</option>
58     <option value="Where were your parents born?">Where were your parents born?</option><!--option1-->
59     <option value="What is the name of your pet?">What is the name of your pet?</option><!--option2-->
60     <option value="How old were you when you had a cell phone?">How old were you when you had a cell phone?</option><!--option3-->
61 </select><!--options close-->
62 <!--answer field-->
63 <input type="text" name="userSAnswer" placeholder="Answer">
64 <!--join button-->
65 <input type="submit" value="Join" >
66
67 <!--JSP FUNCTIONALITY-->
68 ▼ <% //Instantiating web service
69     serviceClient.ProcessPosts service = new serviceClient.ProcessPosts();
70     serviceClient.Processor port = service.getProcessorPort();
71
72
73     //If Join button is pressed
74 ▼     if("POST".equals(request.getMethod())){
75         //to build error string
76         String errorMessage = "";
77
78         //USERNAME VALIDATION
79         String username = request.getParameter("userName");//getting username from form
80 ▼         if(username.length()>5){//checking username length
81             boolean userNameValid = Pattern.matches("\\w+", username);//verifying username
82 ▼             if(userNameValid==true){//if username is valid
83                 String password1 = request.getParameter("userPassword1");//get password 1
84
85                 //PASSWORD VALIDATION
86 ▼                 if(password1.length()>7){//if password 1 length is greater than 7 then
87                     //get password 2 (confirmation)
88                     String password2 = request.getParameter("userPassword2")==null? "":request.getParameter("userPassword2");
89                     if(password1.equals(password2)){ //if both passwords match (confirmed)
90 ▼                         String email = request.getParameter("userEmail");//get user email from form
91
92                     //EMAIL VALIDATION
93 ▼                     if(!email.equals("")){//if email is not blank then
94                         //check email format
95                         boolean emailValid = Pattern.matches("[\\w]+@[\\w]+[.][\\w]+", email);
96 ▼                         if(emailValid==true){//if email is valid
97                             //get security question
98                             String question = request.getParameter("userSQuestion")==null ? "":request.getParameter("userSQuestion");
99
100                             //QUESTION VALIDATION

```

```

101 ▼ if(!question.equals("")){//if question is not blank
102 //get security answer
103 String answer = request.getParameter("userSAnswer");
104
105 //ANSWER VALIDATION
106 ▼ if(!answer.equals("")){//if answer is not blank
107
108 //CHECKING IF USER DETAILS ARE CORRECT
109 ▼ if((port.addUser(username, email,password1, question, answer))==1){
110 //THEN ADD TO DATABASE
111 port.addUser(username, email , password1, question, answer);
112 response.sendRedirect("Login.jsp");//go to login page
113
114 }else errorMessage = "User already exists!";//build error message
115
116 }else errorMessage = "Please entry a proper answer.");//build error message
117
118 }else errorMessage = "Please choose a Security question.");//build error message
119
120 }else errorMessage = "Email is invalid.");//build error message
121
122 }else errorMessage = "Email cannot be blank.");//build error message
123
124 }else errorMessage = "Passwords dont match.");//build error message
125
126 }else errorMessage = "Password must at least be 8 characters long.");//build error message
127
128 } else errorMessage = "Username can only contain letters and numbers.");//build error message
129
130 } else errorMessage = "Username must at least be 6 characters long.");//build error message
131
132 //DISPLAY ERROR MESSAGE!
133 out.println("<p style=\"color:white; font-size:11; text-align:center;\"><b>"+errorMessage+"<b><p>");
134 }
135 %>
136 </form><!--form close-->
137
138 </body><!--body close-->
139 </html><!--html close-->
140

```

passwordFogotten.jsp

Used to reset forgotten password.

```
1 <%--
2     Filename      : passwordForgotten
3     Author       : Aysham Hameed
4     Created      : Dec 7, 2018, 10:57:16 PM
5     Operating System : Windows 10
6     Version      : NetBeans IDE 8.2
7     Description   : Used to reset forgotten password.
8 --%>
9
10 <%@page contentType="text/html" pageEncoding="UTF-8"%><!--specifies the content of JSP.-->
11
12 <html><!--html open-->
13 <head><!--head open-->
14     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"><!--meta data of html page-->
15     <title>Forgotten Password</title> <!--title of page-->
16     <link rel="stylesheet" href="CSS/headerStyle.css"> <!--using headerStyle sheet-->
17     <link rel="stylesheet" href="CSS/profileStyle.css"><!--using profileStyle sheet-->
18 </head><!--head close-->
19
20 <body><!--body open-->
21     <!--HEADER & NAVIGATION BAR-->
22     <header><!--header open-->
23         <div class="container"><!--div open-->
24             <a href="Topics.jsp"><!--Link to home page-->
25                  <!--website Logo-->
26             </a><!--Link close-->
27
28             <nav><!--nav open-->
29                 <ul><!--unordered list open-->
30                     <li><a href="Topics.jsp">Home</a></li><!--Link to home page-->
31                 </ul><!--unordered list close-->
32             </nav><!--nav close-->
33         </div><!--div close-->
34     </header><!--header close-->
35
36
37     <!--FORM-->
38     <div class="passwordForgottenBOX"><!--div passwordForgottenBOX open-->
39         <form action="passwordForgotten.jsp" method="POST"><!--form open-->
40             <h1>Reset Password</h1><!--heading inside form-->
41             <!--username field inside form-->
42             <input type="text" name="username" placeholder="Enter Username"/>
43             <!--password field inside form-->
44             <input type="text" name="email" placeholder="Enter Email"/>
45             <!--combo box inside form-->
46             <select id="cbbox" name="userSQuestion"><!--select open-->
47                 <option selected="true" disabled>Choose from these</option><!--instruction-->
48                 <option value="Where were your parents born?">Where were your parents born?</option><!--option1-->
49                 <option value="What is the name of your pet?">What is the name of your pet?</option><!--option2-->
50                 <option value="How old were you when you had a cell phone?">How old were you when you had a cell phone?</option><!--option3-->
```

```

51     </select><!--select end-->
52     <!--answer field inside form-->
53     <input type="text" name="userSAnswer" placeholder="Answer">
54     <input type="submit" value="Reset"/><!--reset button inside form-->
55 </form><!--form end-->
56
57 <!--JSP CODE-->
58 <% //instantiating web service.
59 serviceClient.ProcessPosts service = new serviceClient.ProcessPosts();
60 serviceClient.Processor port = service.getProcessorPort();
61
62
63 if("POST".equals(request.getMethod())){//if reset button is clicked
64     String username = request.getParameter("username");//get username from form
65     String email = request.getParameter("email");//get email from form
66     String question = request.getParameter("userSQuestion");//get question from form
67     String answer = request.getParameter("userSAnswer");//get answer from form
68
69     //DOING VALIDATION ON FIELDS.
70     boolean valid = false;//flag boolean variable
71     String errorMessage = "";//to build output string
72
73     if(!username.equals("")){//if password is not null
74         if(!email.equals("")){//if email is not null
75             if(!question.equals("")){//if question is not null
76                 if(!answer.equals("")){//if answer is not null
77                     valid = true;//then all fields are valid
78                 }else {errorMessage = "Answer cannot be blank!";}//display error.
79             }else {errorMessage = "Please choose a question";}//display error.
80         } else {errorMessage = "Please enter your email!";}//display error.
81     } else {errorMessage = "Please enter your username!";}//display error.
82
83
84     if(valid==true){//if all fields are valid then
85         //check if all the information exists in database.
86         int outputCode = port.resetPassword(username, email, question, answer);
87         /* OUTPUT CODE MEANINGS:
88         /* 1 = everything is correct
89         /* 0 = neutral
90         /* -1 = user does not exist
91         /* -2 = email does not match
92         /* -3 = question does not match
93         /* -4 = answer does not match
94         String outputMessage = "";
95
96         if(outputCode==1){//if everything is correct in DB
97             //create user session
98             session.setAttribute("usernameSession", username);
99             //redirect page to change password
100             response.sendRedirect("changePassword.jsp");//display error.

```

```

101 }else if(outputCode==1){//check code meaning
102     outputMessage = "User does not exist!";//display error.
103 }else if(outputCode==2){//check code meaning
104     outputMessage = "Email does not match!";//display error.
105 }else if(outputCode==3){//check code meaning
106     outputMessage = "Question does not match!";//display error.
107 }else if(outputCode==4){//check code meaning
108     outputMessage = "Wrong answer!";//display error.
109 }
110 //display output message.
111 out.println("<p style=\"color:white; font-weight:bold; \>"+outputMessage+"</p>");
112
113 } else{//if fields are not valid then display error message
114     out.println("<p style=\"color:white; font-weight:bold; \>"+errorMessage+"</p>");
115 }
116
117 }
118 %>
119
120 </div> <!--div close-->
121 </body><!--body close-->
122 </html><!--html close-->
123

```

deleteAccount.jsp

Used to delete account.

```
1 <%--
2     Filename      : deleteAccount
3     Author       : Aysham Hameed
4     Created      : Dec 7, 2018, 10:57:16 PM
5     Operating System : Windows 10
6     Version      : NetBeans IDE 8.2
7     Description   : Used to delete account.
8 --%>
9
10 <%@page contentType="text/html" pageEncoding="UTF-8"%><!--specifies the content of JSP.--%>
11
12 <html> <!--html open-->
13 <head><!--head open-->
14     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"><!--meta data of html page-->
15     <title>Delete Account</title> <!--title of page-->
16     <link rel="stylesheet" href="CSS/headerStyle.css"> <!--using headerStyle sheet-->
17     <link rel="stylesheet" href="CSS/profileStyle.css"><!--using profileStyle sheet-->
18 </head><!--head close-->
19
20 <body><!--body open-->
21
22     <!--HEADER & NAVIGATION BAR-->
23     <header><!--header open-->
24         <div class="container"><!--div open-->
25             <a href="Topics.jsp"><!--link to homepage-->
26                 <!--Logo image-->
27             </a><!--link close-->
28
29             <nav><!--nav open-->
30                 <ul><!--unordered List open-->
31                     <li><a href="Topics.jsp">Home</a></li><!--home button-->
32
33
34                 <% //Instantiating web service
35                     serviceClient.ProcessPosts service = new serviceClient.ProcessPosts();
36                     serviceClient.Processor port = service.getProcessorPort();
37
38                     //If user is logged in
39                     if(session.getAttribute("usernameSession")!=null){
40                         //create logout link
41                         out.println("<li><a href='\"Topics.jsp?logout=true\"'>");
42                         //get username
43                         String user = (String)session.getAttribute("usernameSession");
44                         //display logout button and username together
45                         out.println("Logout, "+user+"</a></li>");
46                         //display profile button
47                         out.println("<li><a href='\"#\">Profile</a></li>");
48                     }else{//else if user is not logged in
49                         //create pageName session (see Login page)
50                         session.setAttribute("pageName", "Topics");
```



```

51 //display login button
52 out.println("<li><a href=\"Login.jsp\">Login</a></li>");
53 }
54 %>
55 </ul><!--unordered list close-->
56 </nav><!--nav close-->
57 </div><!--div close-->
58 </header><!--header close-->
59
60
61
62 <!--DELETE FORM-->
63 <div class="deleteBOX"> <!--div open-->
64 <form action="deleteAccount.jsp" method="POST"> <!--form open-->
65 <h1>Delete Account</h1> <!--heading-->
66 <input type="password" name="password" placeholder="Enter password"/> <!--password field-->
67 <input type="submit" value="Delete"/> <!--delete button-->
68 </form> <!--form close-->
69
70 <%
71 //if delete button on form was pressed
72 if("POST".equals(request.getMethod())){
73 //get username from session
74 String username = (String)session.getAttribute("usernameSession");
75 //get password from form
76 String password = request.getParameter("password");
77 boolean fieldValid = false; //flag variable
78 String errorMessage = ""; //to build output string
79
80 if(!password.equals("")){ //if password is not blank
81 if(password.length()>5){ //if password is longer than 5 chars
82 fieldValid = true; //then fieldValue is valid
83 }else{ //else display error
84 errorMessage = "Incorrect password!";
85 }
86 }else{ //else display error
87 errorMessage = "Enter password to confirm deletion!";
88 }
89
90
91 if(fieldValid==true){ //if field value is valid
92 String outputMessage = ""; //to build output string
93 //get result from deleteAccount method
94 int deleteResult = port.deleteAccount(username, password);
95
96 /* OUTPUT CODE MEANINGS:
97 /* 0 - neutral
98 /* 1 - user exists
99 /* 2 - user does not exists
100

```

```

101 if(deleteResult==1){ //if everything is correct
102 session.invalidate(); //then delete user session
103 response.sendRedirect("Topics.jsp"); //and go to homepage
104 }else if(deleteResult==2){ //else user does not exist
105 outputMessage = "Incorrect password!"; //build message
106 //display message
107 out.println("<p style=\"color:white; font-weight:bold; \">"+outputMessage+"</p>");
108 }
109 }else{
110 //else if field value is invalid then display error message
111 out.println("<p style=\"color:white; font-weight:bold; \">"+errorMessage+"</p>");
112 }
113 }
114 %>
115
116 </div><!--div end-->
117 </body><!--body end-->
118 </html><!--html end-->
119
120

```

changeEmail.jsp

Used to change user email.

```
1  <!--
2      Filename      : changeEmail
3      Author       : Aysham Hameed
4      Created      : Dec 7, 2018, 10:57:16 PM
5      Operating System : Windows 10
6      Version      : NetBeans IDE 8.2
7      Description   : Used to change user email.
8  -->
9
10 <%@page import="java.util.regex.Pattern"%><!--import regex-->
11 <%@page contentType="text/html" pageEncoding="UTF-8"%><!--specifies the content of JSP.-->
12
13 <html> <!--html open-->
14 <head><!--head open-->
15     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"><!--meta data of html page-->
16     <title>Update Email</title> <!--title of page-->
17     <link rel="stylesheet" href="CSS/headerStyle.css"> <!--using headerStyle sheet-->
18     <link rel="stylesheet" href="CSS/profileStyle.css"><!--using profileStyle sheet-->
19 </head><!--head close-->
20
21 <body><!--body open-->
22     <!--HEADER & NAVIGATION BAR-->
23 <header><!--header open-->
24     <div class="container"><!--div open-->
25         <a href="Topics.jsp"><!--link to homepage-->
26             <!--Logo image-->
27         </a><!--link close-->
28
29     <nav><!--nav open-->
30     <ul><!--unordered list open-->
31         <li><a href="Topics.jsp">Home</a></li><!--home button-->
32
33     <% //Instantiating web service
34         serviceClient.ProcessPosts service = new serviceClient.ProcessPosts();
35         serviceClient.Processor port = service.getProcessorPort();
36
37         //If user is logged in
38     if(session.getAttribute("usernameSession")!=null){
39         //create logout link
40         out.println("<li><a href=\"Topics.jsp?logout=true\">");
41         //get username
42         String user = (String)session.getAttribute("usernameSession");
43         //display logout button and username together
44         out.println("Logout, "+user+"</a></li>");
45         //display profile button
46         out.println("<li><a href=\"#\">Profile</a></li>");
47     }else{ //else if user is not logged in
48         //create pageName session (see login page)
49         session.setAttribute("pageName", "Topics");
50         //display login button
```

```

51         out.println("<li><a href=\"Login.jsp\">Login</a></li>");
52     }
53     %>
54     </ul><!--unordered list close-->
55     </nav><!--nav close-->
56     </div><!--div close-->
57     </header><!--header close-->
58
59
60     <!--FORM-->
61     <div class="emailChangeBOX"><!--div open-->
62     <form action="changeEmail.jsp" method="POST"><!--form open-->
63         <h1>Change Email</h1><!--heading-->
64         <!--displaying current email-->
65         <p style="color:white;"><% out.println(port.getUserEmail((String)session.getAttribute("usernameSession"))); %></p>
66         <input type="text" name="newEmail" placeholder="New Email"/><!--new email field-->
67         <input type="submit" value="Change"/><!--change button-->
68     </form><!--form close-->
69
70     <!--JSP FUNCTIONALITY-->
71     <%
72         //if change button is clicked
73         if("POST".equals(request.getMethod())){
74             //get email from form
75             String email = request.getParameter("newEmail");
76             String errorMessage="";//used to build string
77
78             if(!email.equals("")){//if email field is not blank
79                 //check for email format
80                 boolean emailValid = Pattern.matches("[\\w]+@[\\w]+.[\\w]+", email);
81                 //if email is valid
82                 if(emailValid==true){
83                     //get username from user session
84                     String username = (String) session.getAttribute("usernameSession");
85                     //change email in database
86                     port.changeEmail(username, email);
87                     //go to profile page
88                     response.sendRedirect("Profile.jsp");
89                 } else {//else provide error message
90                     errorMessage = "Invalid email";
91                     //display error message
92                     out.println("<p style=\"color:white; font-weight:bold; \\>"+errorMessage+"</p>");
93                 }
94             } else {//else provide error message
95                 errorMessage = "Please enter an email";
96                 //display error message
97                 out.println("<p style=\"color:white; font-weight:bold; \\>"+errorMessage+"</p>");
98             }
99         }
100     }
101     %>
102
103
104     </div><!--div close-->
105     </body><!--body close-->
106     </html><!--html close-->
107

```

changePassword.jsp

Used to change user password.

```
1 <%--
2   Filename      : changePassword
3   Author       : Aysham Hameed
4   Created      : Dec 7, 2018, 10:57:16 PM
5   Operating System : Windows 10
6   Version      : NetBeans IDE 8.2
7   Description   : Used to change user password.
8 --%>
9
10 <%@page contentType="text/html" pageEncoding="UTF-8"%><!--specifies the content of JSP.--%>
11
12 <html> <!--html open-->
13   <head><!--head open-->
14     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"><!--meta data of html page-->
15     <title>Update Password</title> <!--title of page-->
16     <link rel="stylesheet" href="CSS/headerStyle.css"> <!--using headerStyle sheet-->
17     <link rel="stylesheet" href="CSS/profileStyle.css"><!--using profileStyle sheet-->
18   </head><!--head close-->
19
20   <body><!--body open-->
21     <!--HEADER & NAVIGATION BAR-->
22     <header><!--header open-->
23       <div class="container"><!--div open-->
24         <a href="Topics.jsp"><!--link to homepage-->
25         <!--Logo image-->
26       </a><!--link close-->
27
28       <nav><!--nav open-->
29         <ul><!--unordered list open-->
30           <li><a href="Topics.jsp">Home</a></li><!--home button-->
31
32           <% //Instantiating web service
33             serviceClient.ProcessPosts service = new serviceClient.ProcessPosts();
34             serviceClient.Processor port = service.getProcessorPort();
35
36             //If user is logged in
37             if(session.getAttribute("usernameSession")!=null){
38               //create logout link
39               out.println("<li><a href='\"Topics.jsp?logout=true\">");
40               //get username
41               String user = (String)session.getAttribute("usernameSession");
42               //display logout button and username together
43               out.println("Logout, "+user+"</a></li>");
44               //display profile button
45               out.println("<li><a href='\"#\">Profile</a></li>");
46             }else{//else if user is not logged in
47               //create pageName session (see login page)
48               session.setAttribute("pageName", "Topics");
49               //display login button
50               out.println("<li><a href='\"Login.jsp\">Login</a></li>");
```

```

51     }
52     %>
53     </ul><!--unordered list close-->
54     </nav><!--nav close-->
55     </div><!--div close-->
56 </header><!--header close-->
57
58
59 <!--FORM-->
60 <div class="passwordChangeBOX"><!--div open-->
61     <form action="changePassword.jsp" method="POST"><!--form open-->
62         <h1>New Password</h1><!--heading-->
63         <input type="password" name="newPassword" placeholder="New Password"/><!--password field-->
64         <input type="submit" value="Change"/><!--change button-->
65     </form><!--form close-->
66
67 <!--JSP FUNCTIONALITY-->
68 <% //dont need to check if user session is created or not
69     //because can only change password once logged in.
70     if("POST".equals(request.getMethod())){ //if change button is clicked
71         String passwordNEW = request.getParameter("newPassword");//get new password
72         String errorMessage="";//to build error string
73
74         if(!passwordNEW.equals("")){ //if new password field is not blank
75             if(passwordNEW.length()>5){ //if password lenght is longer than 5 char
76                 //get username from usersession
77                 String username = (String) session.getAttribute("usernameSession");
78                 //change password in database
79                 port.changePassword(username, passwordNEW);
80                 //go to profile page
81                 response.sendRedirect("Profile.jsp");
82             } else { //display error if less than 5 char
83                 errorMessage = "Password must at least be 6 characters long";
84                 //displaying error
85                 out.println("<p style='color:white; font-weight:bold; \>'"+errorMessage+"</p>");
86             }
87
88         } else { //display error if password is blank
89             errorMessage = "Password cannot be blank!";
90             //displaying error
91             out.println("<p style='color:white; font-weight:bold; \>'"+errorMessage+"</p>");
92         }
93     }
94     %>
95
96 </div><!--div end-->
97
98 </body><!--body end-->
99 </html><!--html end-->

```

changeSecurityQuestion.jsp

Used to change user security question.

```
1 <!--
2     Filename      : changeSecurityQuestion
3     Author       : Aysham Hameed
4     Created      : Dec 7, 2018, 10:57:16 PM
5     Operating System : Windows 10
6     Version      : NetBeans IDE 8.2
7     Description   : Used to change user security question.
8 --%>
9
10 <%@page contentType="text/html" pageEncoding="UTF-8"%><!--specifies the content of JSP.--%>
11
12 <html> <!--html open-->
13 <head><!--head open-->
14     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"><!--meta data of html page-->
15     <title>Update Security Question</title> <!--title of page-->
16     <link rel="stylesheet" href="CSS/headerStyle.css"> <!--using headerStyle sheet-->
17     <link rel="stylesheet" href="CSS/profileStyle.css"><!--using profileStyle sheet-->
18 </head><!--head close-->
19
20 <body><!--body open-->
21
22     <!--HEADER & NAVIGATION BAR-->
23     <header><!--header open-->
24         <div class="container"><!--div open-->
25             <a href="Topics.jsp"><!--link to homepage-->
26                 <!--Logo image-->
27             </a><!--link close-->
28
29             <nav><!--nav open-->
30                 <ul><!--unordered list open-->
31                     <li><a href="Topics.jsp">Home</a></li><!--home button-->
32
33
34                 <% //Instantiating web service
35                     serviceClient.ProcessPosts service = new serviceClient.ProcessPosts();
36                     serviceClient.Processor port = service.getProcessorPort();
37
38                     //If user is logged in
39                     if(session.getAttribute("usernameSession")!=null){
40                         //create logout link
41                         out.println("<li><a href=\"Topics.jsp?logout=true\">");
42                         //get username
43                         String user = (String)session.getAttribute("usernameSession");
44                         //display logout button and username together
45                         out.println("Logout, "+user+"</a></li>");
46                         //display profile button
47                         out.println("<li><a href=\"#\">Profile</a></li>");
48                     }else{//else if user is not logged in
49                         //create pageName session (see login page)
50                         session.setAttribute("pageName", "Topics");
51                     }
52                 <%
53             </nav><!--nav close-->
54         </div><!--div close-->
55     </header><!--header close-->
56
57     <!--Main Content-->
58     <div class="container"><!--div open-->
59         <div class="row"><!--row open-->
60             <div class="col-md-12"><!--col open-->
61                 <div class="panel panel-default"><!--panel open-->
62                     <div class="panel-heading"><!--panel heading open-->
63                         <h3>Update Security Question</h3><!--panel heading close-->
64                     </div><!--panel heading close-->
65                     <div class="panel-body"><!--panel body open-->
66                         <div class="form-group"><!--form group open-->
67                             <input type="text" value="New Security Question"/><!--form group close-->
68                         </div><!--form group close-->
69                         <div class="form-group"><!--form group open-->
70                             <input type="text" value="Confirm New Security Question"/><!--form group close-->
71                         </div><!--form group close-->
72                         <div class="form-group"><!--form group open-->
73                             <input type="text" value="Old Security Question"/><!--form group close-->
74                         </div><!--form group close-->
75                         <div class="form-group"><!--form group open-->
76                             <input type="text" value="Confirm Old Security Question"/><!--form group close-->
77                         </div><!--form group close-->
78                         <div class="form-group"><!--form group open-->
79                             <input type="button" value="Update Security Question"/><!--form group close-->
80                         </div><!--form group close-->
81                     </div><!--panel body close-->
82                 </div><!--panel close-->
83             </div><!--col close-->
84         </div><!--row close-->
85     </div><!--div close-->
86
87 </body><!--body close-->
88 </html><!--html close-->
```



```

51 //display login button
52 out.println("<li><a href=\"Login.jsp\">Login</a></li>");
53 }
54 %>
55 </ul><!--unordered list close-->
56 </nav><!--nav close-->
57 </div><!--div close-->
58 </header><!--header close-->
59 <!--FORM-->
60 <div class="SQChangeBOX"><!--div open-->
61 <form action="changeSecurityQuestion.jsp" method="POST"><!--form open-->
62 <h1>Change Security Question</h1><!--heading inside form-->
63 <!--displaying current chosen question-->
64 <p style="color: white; text-align: left;"><b>Current Question :</b> <% out.println(port.getUserQ()); %></p>
65 <!--displaying current chosen answers-->
66 <p style="color: white; text-align: left;"><b>Current Answer :</b> <% out.println(port.getUserA()); %></p>
67
68 <hr><!--horizontal line-->
69 <select id="cbox" name="userSQuestion"><!--select box(combo box)-->
70 <option selected="true" disabled>Choose from these</option><!--instruction inside select tag-->
71 <option value="Where were your parents born?">Where were your parents born?</option><!--option1-->
72 <option value="What is the name of your pet?">What is the name of your pet?</option><!--option2-->
73 <option value="How old were you when you had a cell phone?">How old were you when you had a cell phone?</option><!--option3-->
74 </select><!--select end-->
75 <input type="text" name="userSAnswer" placeholder="Answer"><!--answer field-->
76 <input type="submit" value="Change"/><!--change button-->
77 </form><!--form end-->
78
79 <!--JSP FUNTIONALITY-->
80 <%
81 //if change button is clicked
82 if("POST".equals(request.getMethod())){
83 //get username from user session
84 String username = (String) session.getAttribute("usernameSession");
85 //get user question from form
86 String question = request.getParameter("userSQuestion")==null ? "":request.getParameter("userSQuestion");
87 //get user answer from form
88 String answer = request.getParameter("userSAnswer");
89
90 String errorMessage = "";//to build error message
91
92 if(!question.equals("")){if question is not blank
93 if(!answer.equals("")){if answer field is not blank
94 //then update security question and answer
95 port.updateSQA(username, question, answer);
96 response.sendRedirect("Profile.jsp");//reditect page to Profile
97 }else {if else display error message.
98 errorMessage = "Answer cannot be blank!";
99 }
100 }else {if else display error message.
101 errorMessage = "Please choose a question!";
102 }
103
104 //displaying error message
105 out.println("<p style=\"color:white; font-weight:bold; \">"+errorMessage+"</p>");
106 }
107 %>
108 </div><!--div end-->
109 </body><!--body end-->
110 </html><!--html end-->

```


UserNotFound.html

Displays error message to user, if no user is found.

```
1  <!--
2      Filename      : UserNotFound
3      Author       : Aysham Hameed
4      Created      : Dec 7, 2018, 12:40:16 PM
5      Operating System : Windows 10
6      Version      : NetBeans IDE 8.2
7      Description   : Display error message to user, if no user is found.
8  -->
9
10 <html><!--html open-->
11 <head><!--head open-->
12     <title>User Not Found</title><!--page title-->
13     <!--using CSS STYLE SHEETS-->
14     <link rel="stylesheet" href="CSS/userNotFoundStyle.css" ><!--sheet1-->
15     <link rel="stylesheet" href="CSS/headerStyle.css"><!--sheet2-->
16     <link rel="stylesheet" href="CSS/joinStyle.css"><!--sheet3-->
17 </head><!--head close-->
18
19 <body><!--body open-->
20
21     <!--HEADER & NAVIGATION BAR-->
22 <header><!--header open-->
23     <div class="container"><!--div open-->
24         <a href="Topics.jsp"><!--link to home page-->
25             <!--Logo on page-->
26         </a><!--link close-->
27
28         <nav><!--nav open-->
29             <ul><!--unordered list open-->
30                 <li><a href="Topics.jsp">Home</a></li><!--home button-->
31             </ul><!--unordered list close-->
32         </nav><!--nav open-->
33     </div><!--div close-->
34 </header><!--header close-->
35
36     <!--fox image displayed on page-->
37     
38     <!--go back button-->
39     <a href="Login.jsp" > <button id="goBack">Go Back</button></a>
40
41 </body><!--body close-->
42 </html><!--html close-->
43
```

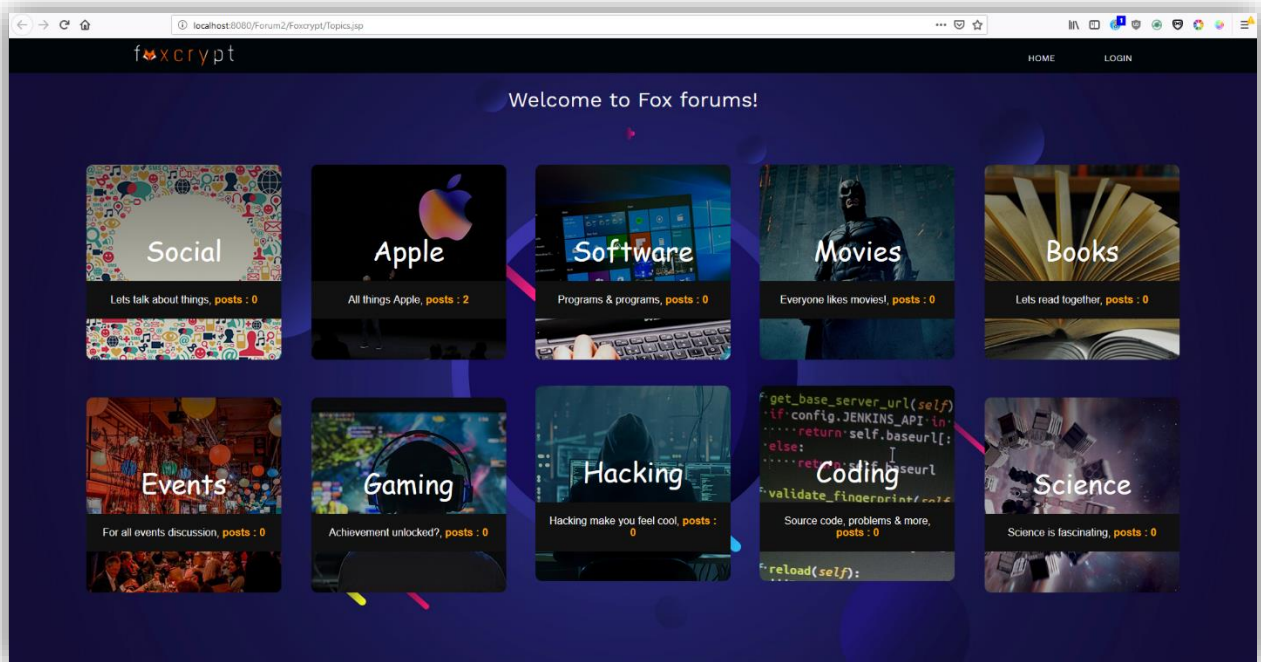
InvalidPassword.html

Displays error message to user, if password is wrong.

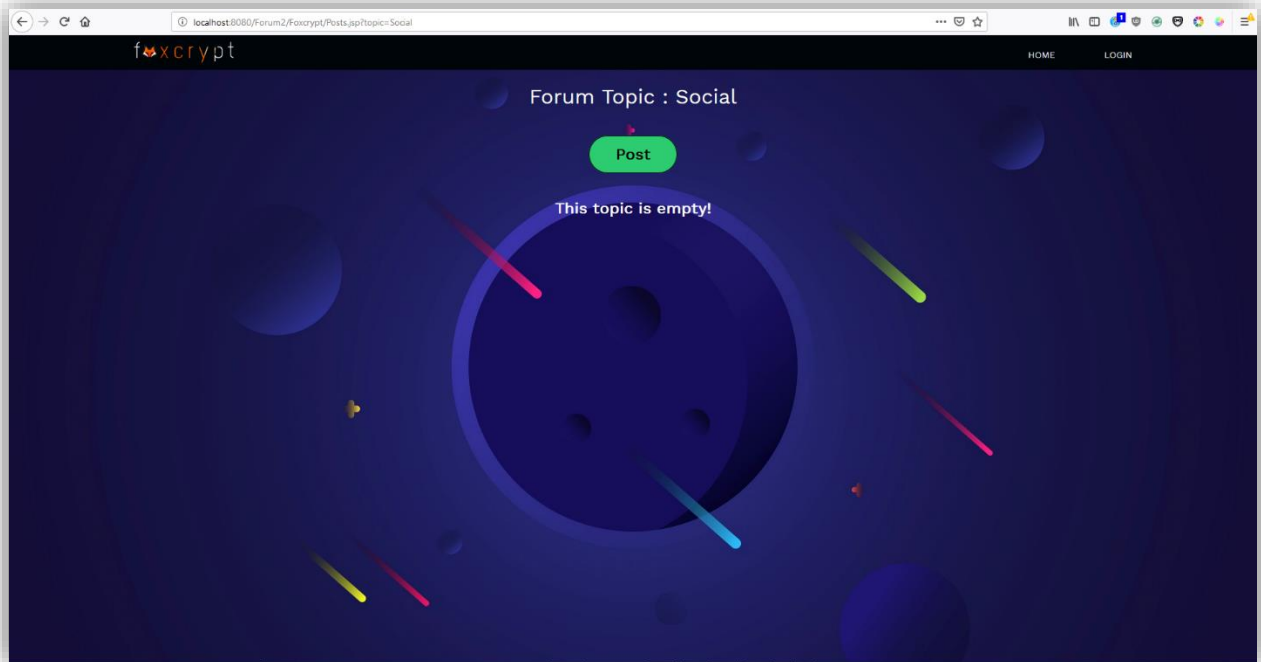
```
1  <!--
2      Filename      : InvalidPassword
3      Author       : Aysham Hameed
4      Created      : Dec 7, 2018, 12:40:16 PM
5      Operating System : Windows 10
6      Version      : NetBeans IDE 8.2
7      Description   : Display error message to user, if no password is invalid.
8  -->
9
10 <html><!--html open-->
11 <head><!--head open-->
12     <title>Invalid Password</title><!--page title-->
13     <!--using CSS STYLE SHEETS-->
14     <link rel="stylesheet" href="CSS/userNotFoundStyle.css" ><!--sheet1-->
15     <link rel="stylesheet" href="CSS/headerStyle.css"><!--sheet2-->
16     <link rel="stylesheet" href="CSS/joinStyle.css"><!--sheet3-->
17 </head><!--head close-->
18
19 <body><!--body open-->
20
21     <!--HEADER & NAVIGATION BAR-->
22     <header><!--header open-->
23         <div class="container"><!--div open-->
24             <a href="Topics.jsp"><!--link to home page-->
25                 <!--Logo on page-->
26             </a><!--link close-->
27
28             <nav><!--nav open-->
29                 <ul><!--unordered list open-->
30                     <li><a href="Topics.jsp">Home</a></li><!--home button-->
31                 </ul><!--unordered list close-->
32             </nav><!--nav open-->
33         </div><!--div close-->
34     </header><!--header close-->
35
36     <!--fox image displayed on page-->
37     
38     <!--go back button-->
39     <a href="Login.jsp" > <button id="goBack">Go Back</button></a>
40
41 </body><!--body close-->
42 </html><!--html close-->
43
```

Screenshots

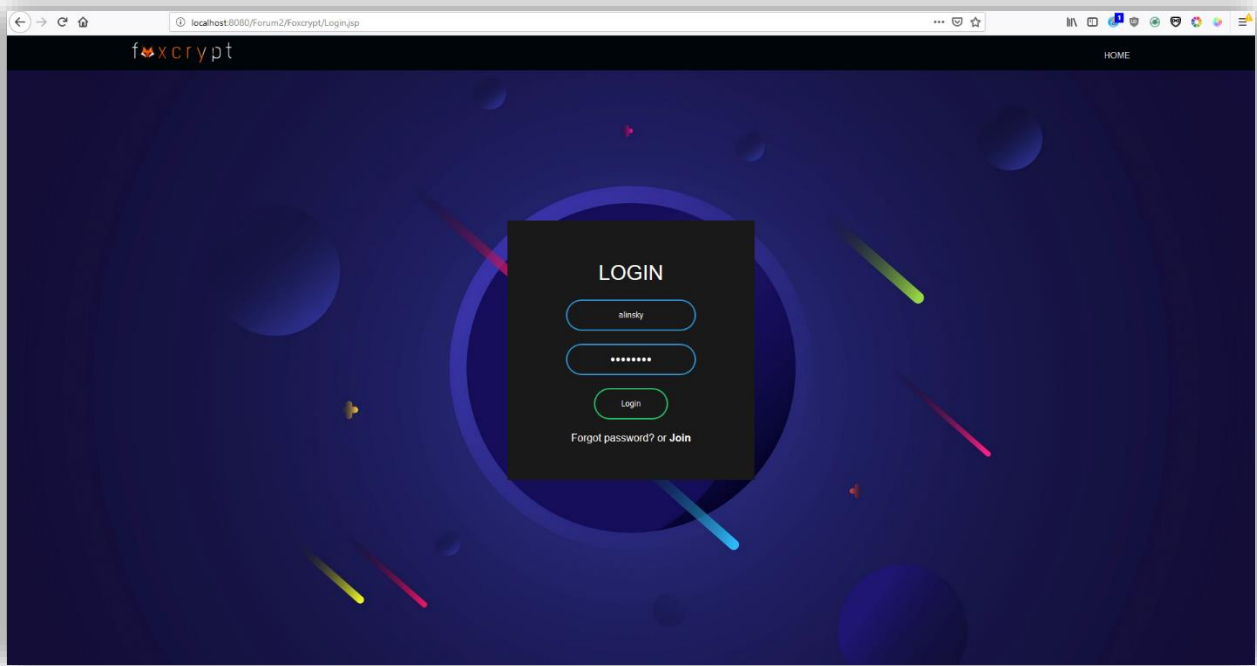
This is the main page (Homepage) and it contains all the categories/to fixed topics of the forum. (Cannot be changed by user)



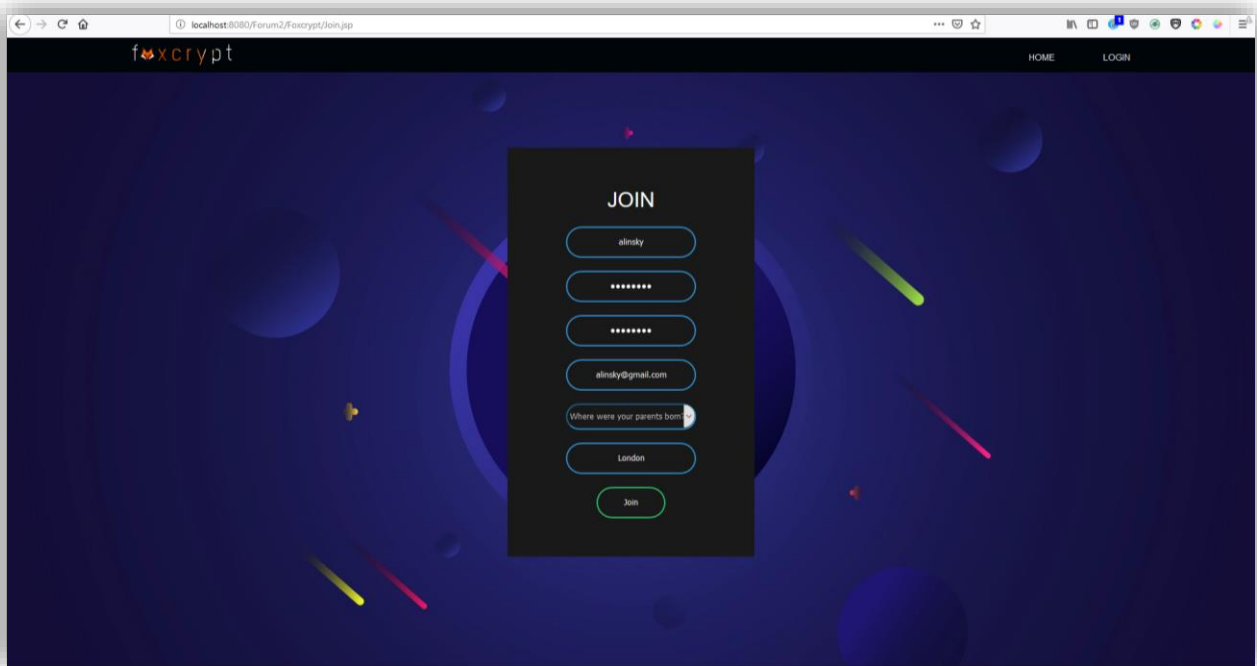
Posts page. Where the user creates a post. But need to login first in order to create it. But can view the posts posted by other users.



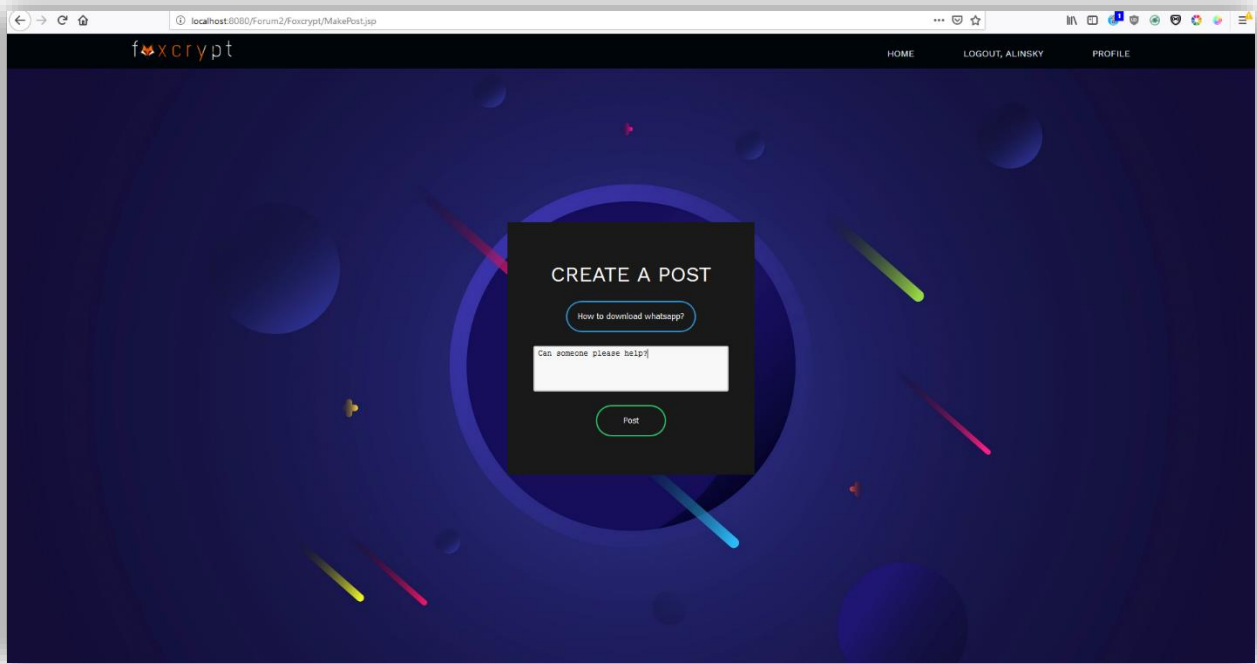
The Login page is where the user logs in with their username and password in order to Post, Comment and Reply and even to make changes to their Profile.



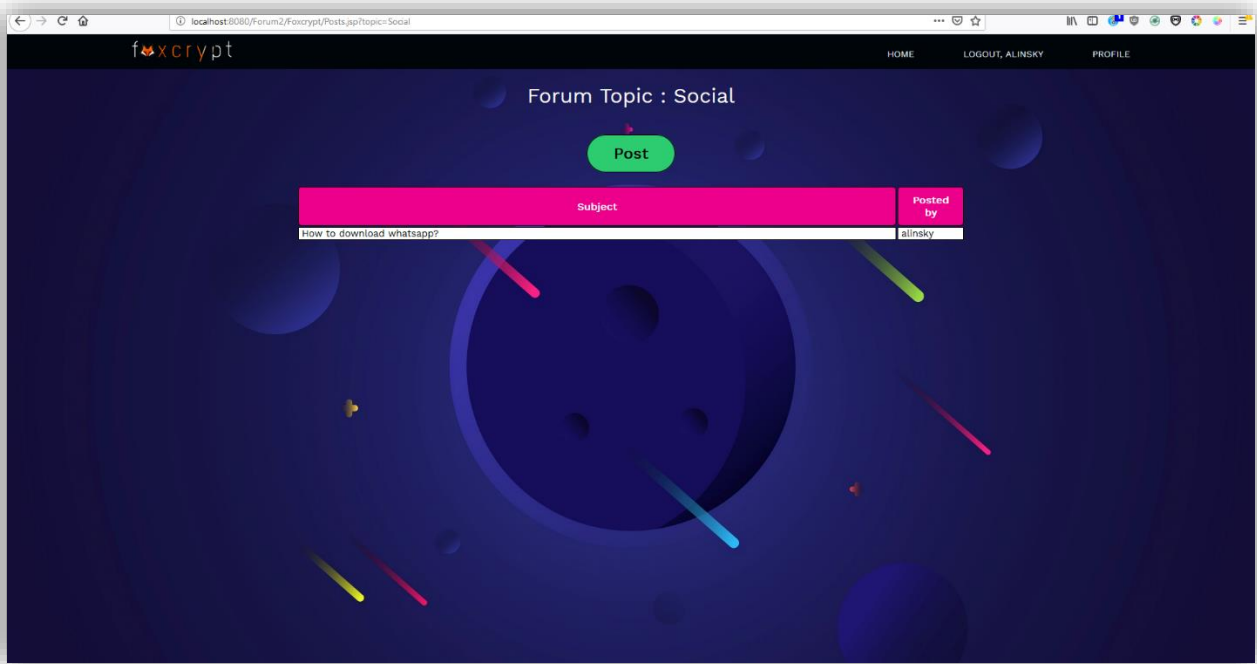
The Join page allows users to Register. And after registering redirects the user to the Login page. This page also performs validation on the input fields.



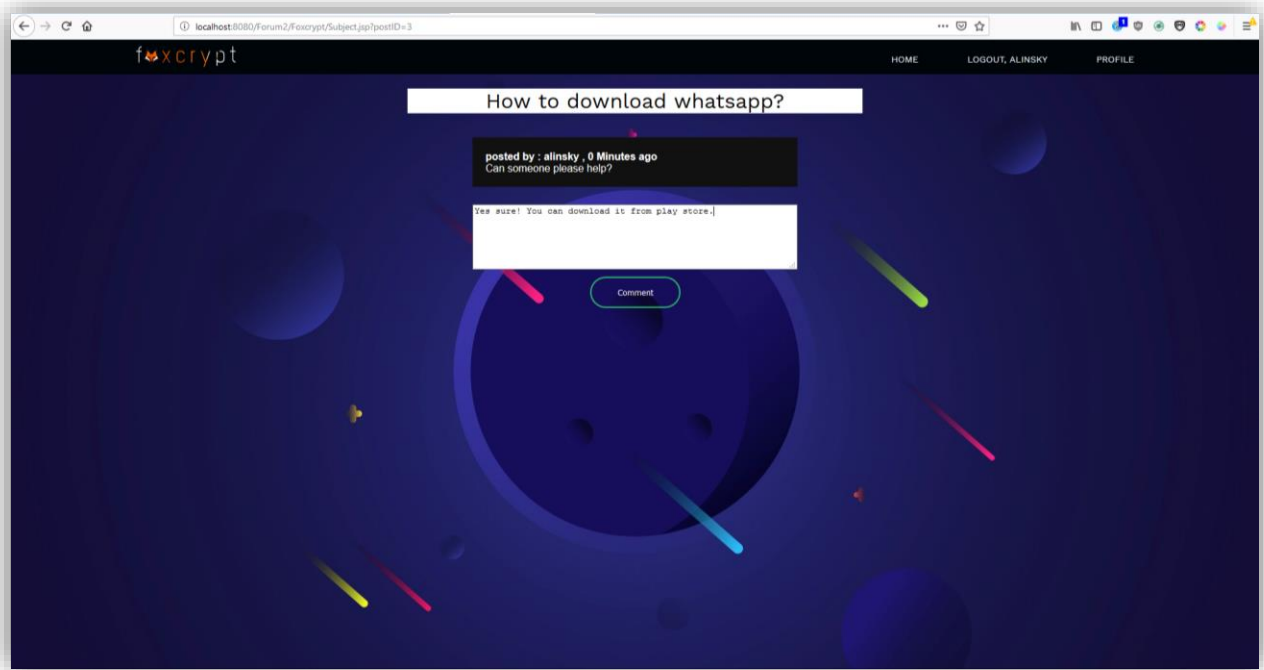
This page allows the user to create a new Post. Input fields contain error handling.



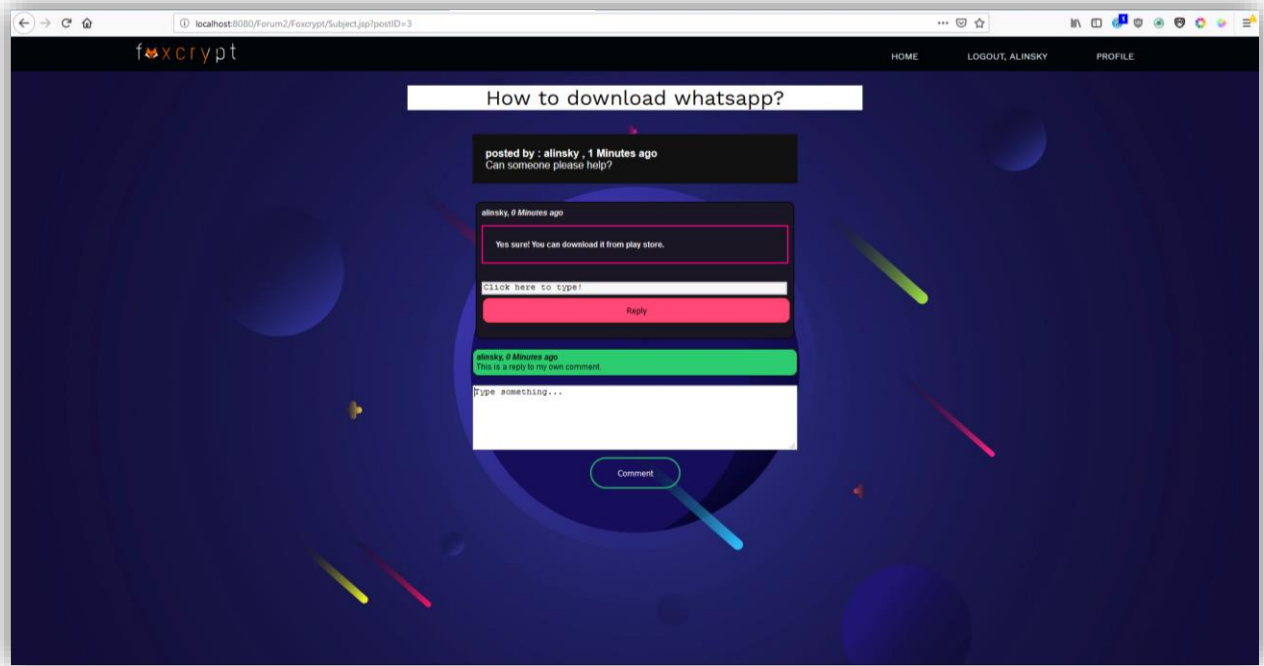
The user is able to view any post he/she clicks on.



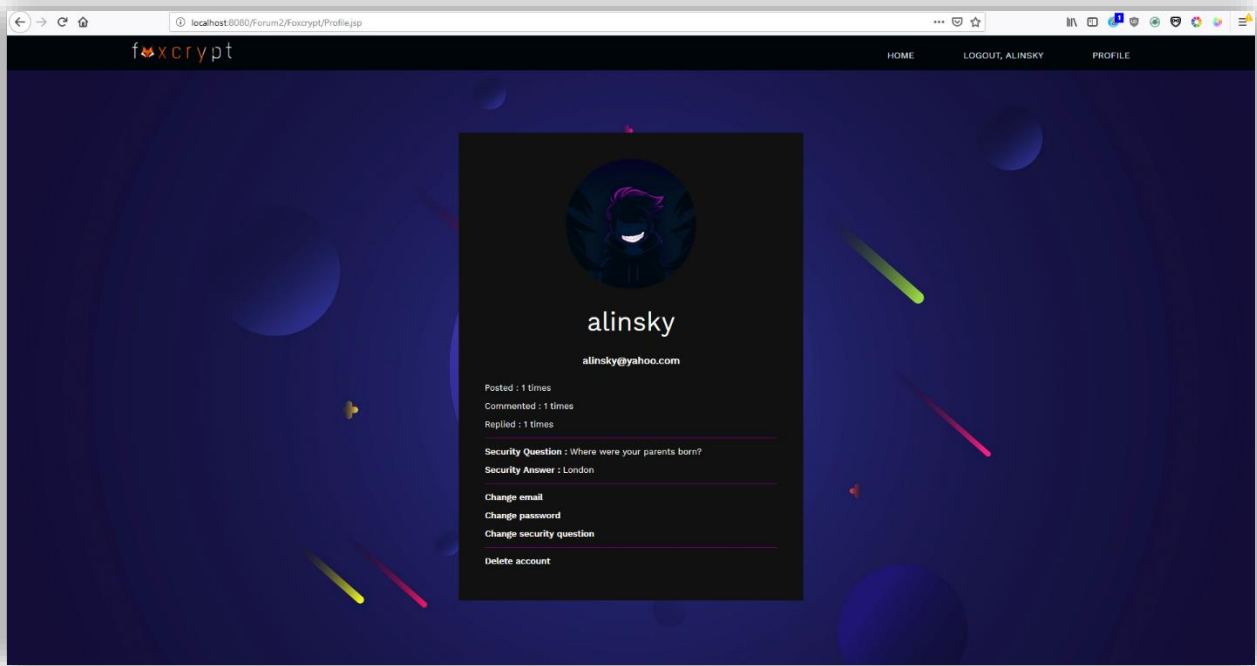
This page displays information about the selected post. Information including Comments, Replies, Times & Descriptions.



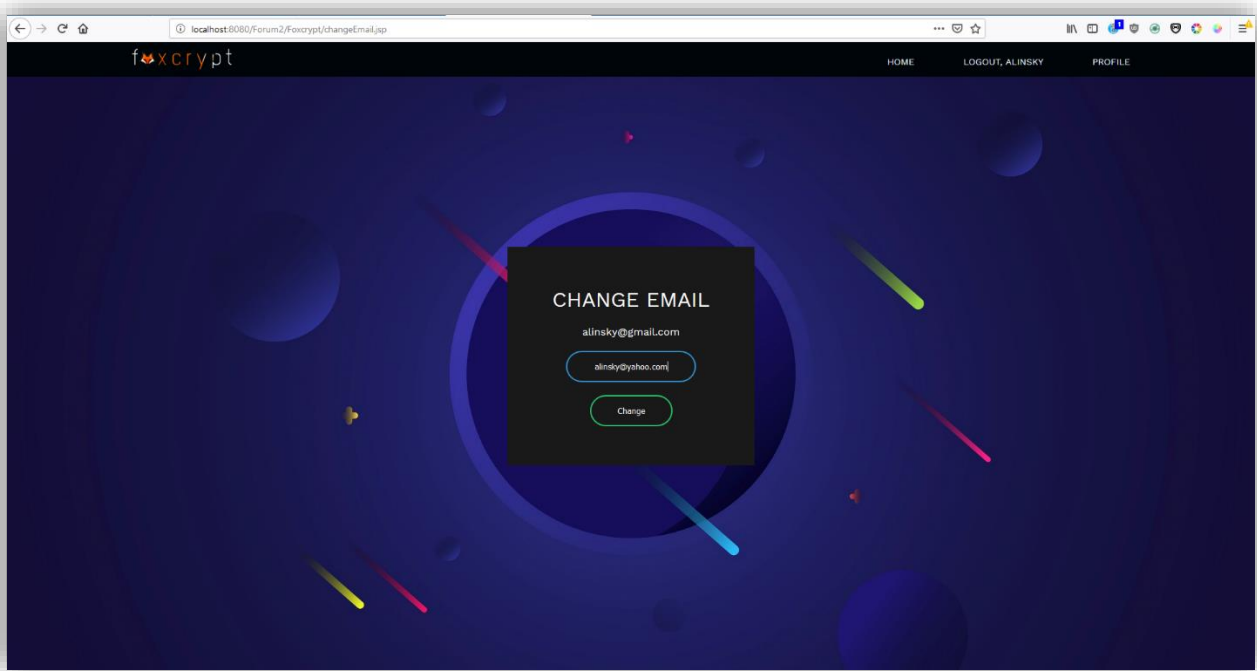
The user is able to Reply & Comment. If user is not logged in then he/she is redirected to the Login page. To first Login.



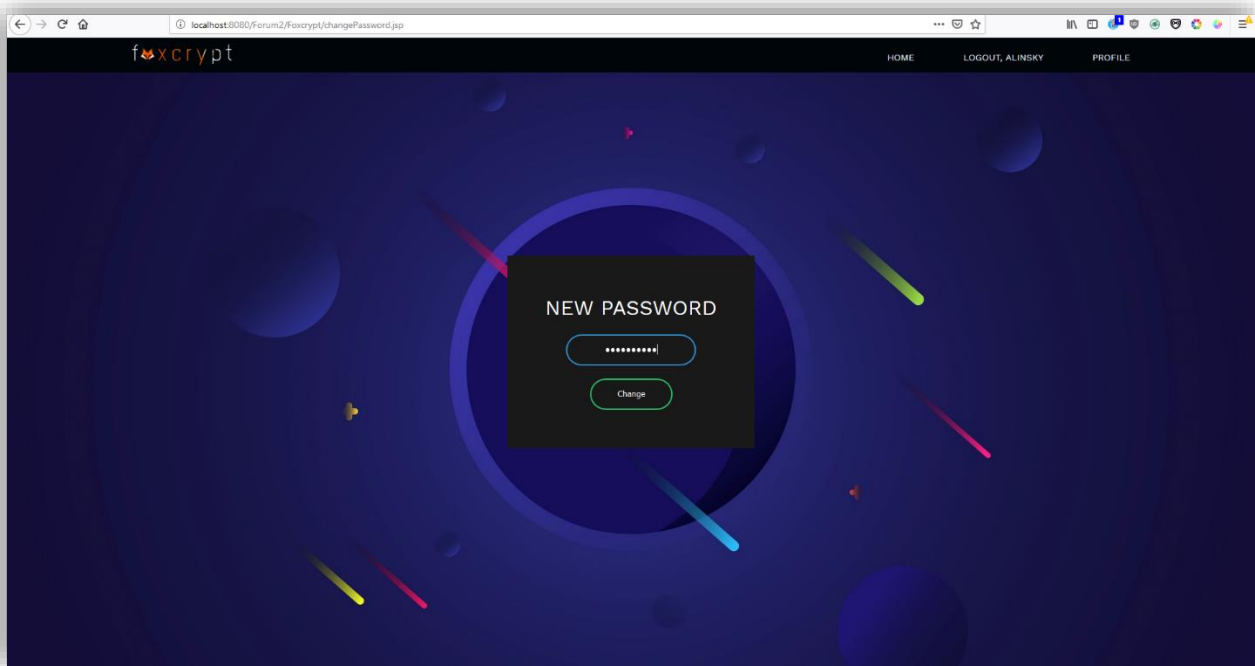
This page provides user with their information & options to change their personal information and also contains option to delete user profile.



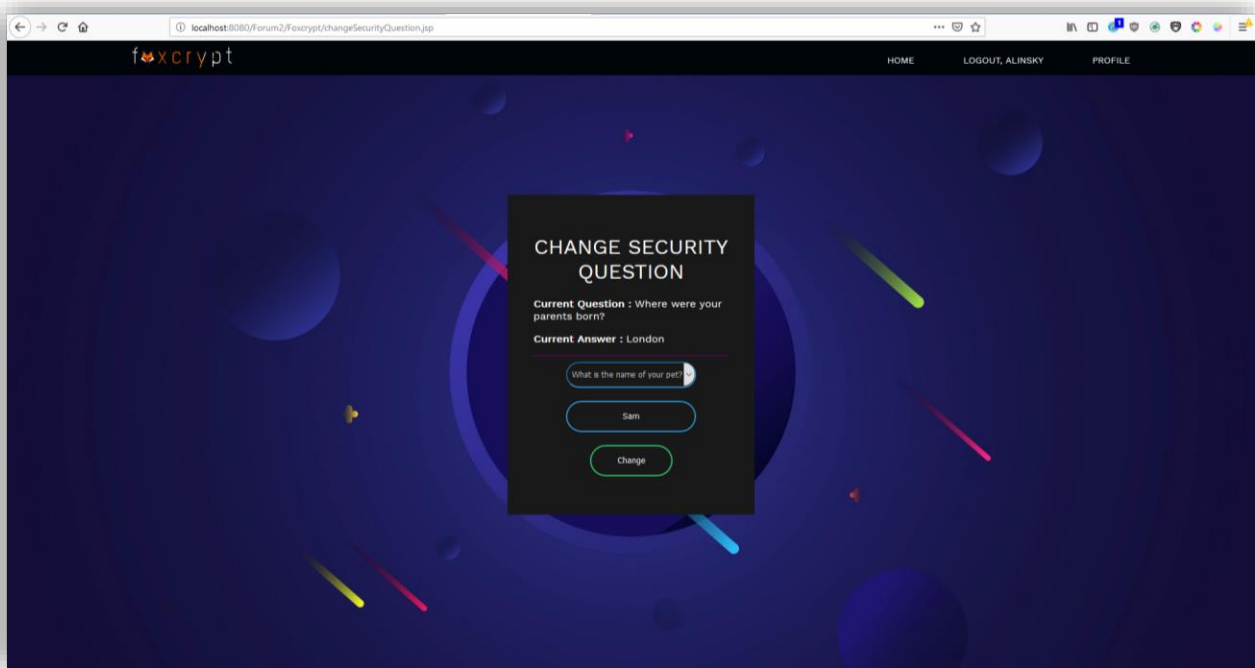
This page allows user to change their email.



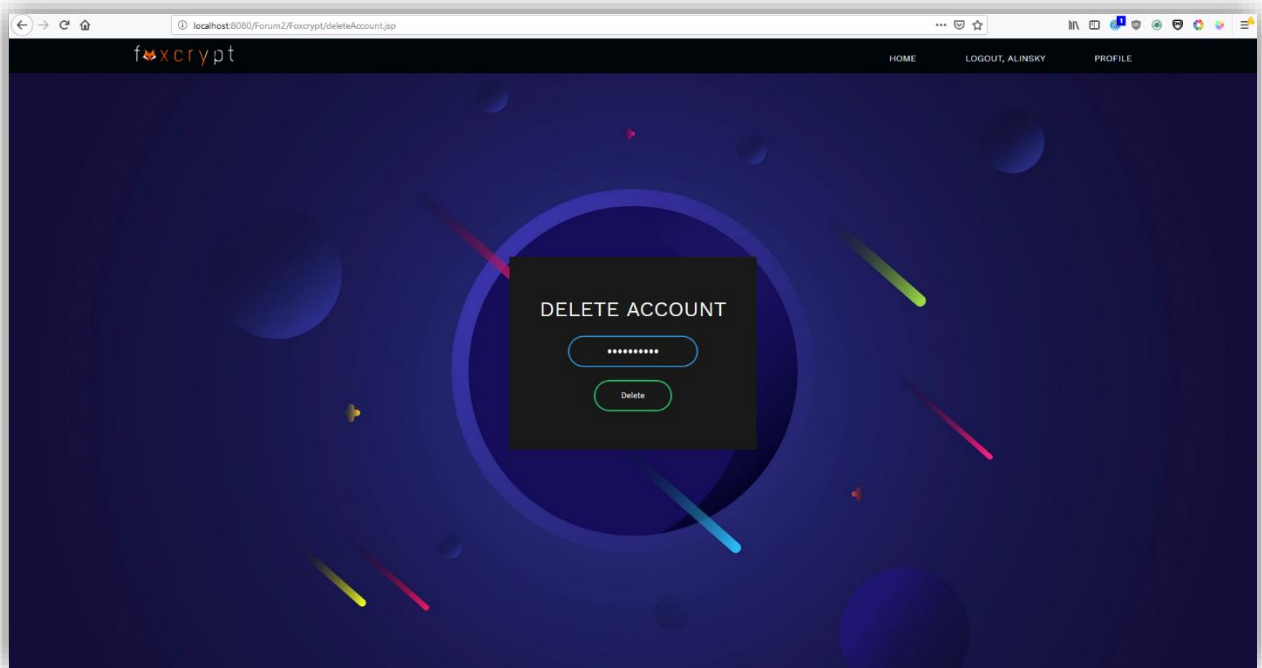
This page allows user to change their password.



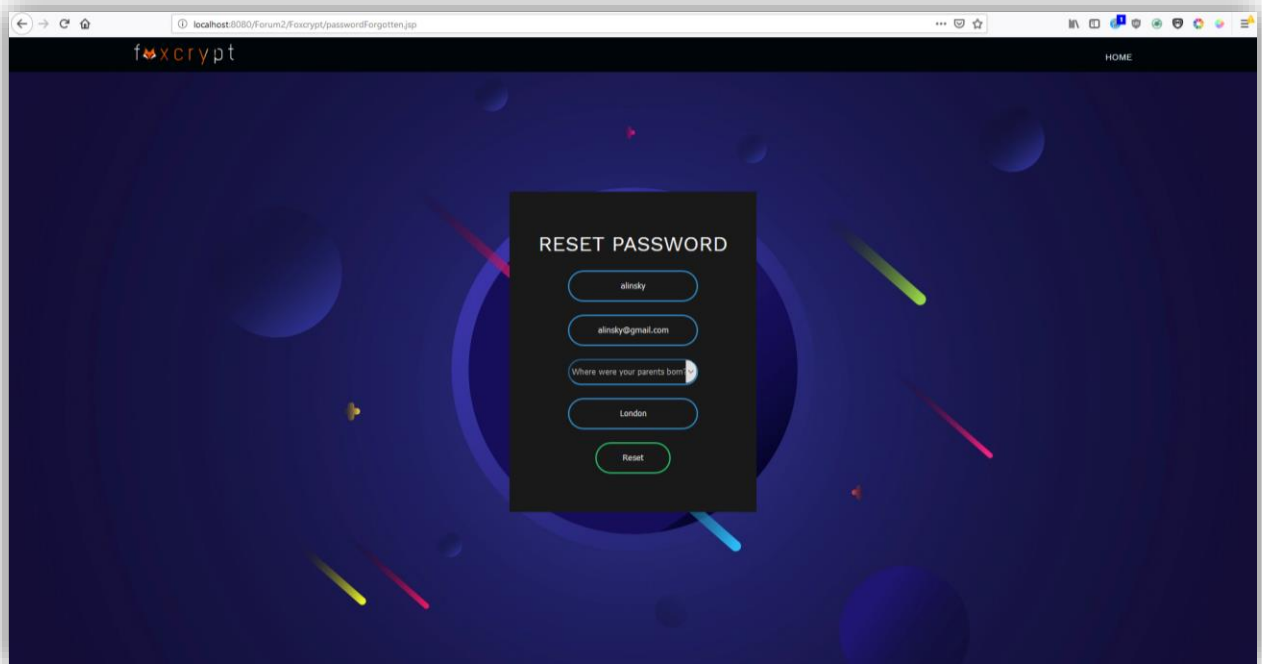
This page allows users to change their security questions.



The user also has the option to delete their profile.
After which they are redirected to the homepage.



This page allows the user to reset password if they
have forgotten. The input fields are validated first.



Bibliography & Additional Notes.

Bibliography for customizations:

- Designing login forms - <https://youtu.be/HV7DtH3J2PU?list=PLnPn0BwaMLqjrbw-YIVNT0TO7D8Qxx9Yr>
- Animated tiles - <https://youtu.be/THIQgTqT-io?list=PLnPn0BwaMLqjrbw-YIVNT0TO7D8Qxx9Yr>
- Navigation bar 1 - <https://youtu.be/gQgqJG8ld7M?list=PLnPn0BwaMLqjrbw-YIVNT0TO7D8Qxx9Yr>
- Navigation bar 2 - <https://youtu.be/FEmysQARWFU?list=PLnPn0BwaMLqjrbw-YIVNT0TO7D8Qxx9Yr>
- Tables & headers - <https://youtu.be/5sbPjvbfzk8?list=PLnPn0BwaMLqjrbw-YIVNT0TO7D8Qxx9Yr>

Important points:

- JDBC SQL Server driver is included in root of project folder.
- Java Docs are created for web service Processor class. Path: **Forum2/dist/javadoc/index.html**.
Select serviceServer when opening index file. serviceClient is generated by Net Beans.
- Log file is generated and saved on desktop.

I used GlassFish server instead of using Tomcat. The project does not really specify which server container to use. I used it because it did not run into any errors and worked flawlessly. Tomcat gave a number of problems. It does the same job as Tomcat. It also comes included with the NetBeans package.

Video tutorial is included just in case.