# Source Code for CA3

Hugo Hewitt and Cameron Wragge

April 2021

## Contents

# 1 SocialMedia.java

```
1  package socialmedia;
2
3  import java.io.*;
4  import java.util.ArrayList;
5
6  import socialmedia.AccountIDNotRecognisedException;
7  import socialmedia.IllegalHandleException;
8  import socialmedia.InvalidHandleException;
9  import socialmedia.SocialMediaPlatform;
10 import socialmedia.HandleNotRecognisedException;
11 import socialmedia.PostIDNotRecognisedException;
12
13 /**
14  * A class that will implement the methods of the social
        media platform and mini social media platform
        interfaces.
15  * This provides the majority of the functionallity
16  */
17 public class SocialMedia implements SocialMediaPlatform {
18
19     //creating the array lists needed to manage accounts
20     ArrayList<Account> accounts = new ArrayList<Account>()
           ;
21     ArrayList<Integer> ids = new ArrayList<Integer>();
22
23     //creating the array lists to manage the posts
24     ArrayList<Post> posts = new ArrayList<Post>();
25     ArrayList<Integer> postIds = new ArrayList<Integer>();
26
27
28      @Override
29     public int createAccount(String handle) throws
           IllegalHandleException, InvalidHandleException {
30         // Method that creates an account object
31
32         //validating the inputs from the user
33           if (handle.length() > 0){
34               boolean unique = true;
35               int i=0;
36               //looping over the accounts to make sure the
                     handle is unique
37               for(Account loopAccount : accounts){
```

```
38                  if(loopAccount.getHandle().equals(handle)
                        ){
39                      unique = false;
40                  }
41                  i++;
42              }
43              //creating a new account if the handle is
                    unique
44              if (unique){
45                  ids.add(ids.size());
46                  Account newAccount = new Account(ids.get(
                        ids.size()-1), handle);
47                  accounts.add(newAccount);
48                  //returning the account id
49                  return ids.get(ids.size()-1);
50              } else {
51                  //throwing the exceptions for invalid
                        handles
52                  throw new IllegalHandleException("handle
                        already in use");
53              }
54          } else {
55              throw new InvalidHandleException("invalid
                    handle length");
56          }
57      }
58
59      @Override
60      public int createAccount(String handle, String
            description) throws IllegalHandleException,
            InvalidHandleException {
61          // Method that creates an account object
62
63          //validating the inputs from the user
64          if (handle.length() > 1){
65                boolean unique = true;
66                int i=0;
67                //looping over the accounts to make sure the
                      handle is unique
68                for(Account loopAccount : accounts){
69                    if(loopAccount.getHandle().equals(handle)
                          ){
70                        unique = false;
71                    }
72                    i++;
73                }
```

```
74              //creating a new account if the handle is
                    unique
75              if (unique){
76                  ids.add(ids.size());
77                  Account newAccount = new Account(ids.get(
                        ids.size()-1), handle, description);
78                  accounts.add(newAccount);
79                  //returning the account id
80                  return ids.get(ids.size()-1);
81              } else {
82                  //throwing the exceptions for invalid
                        handles
83                  throw new IllegalHandleException("handle
                        already in use");
84              }
85          } else {
86              throw new InvalidHandleException("invalid
                    handle length");
87          }
88      }
89
90      @Override
91       public void removeAccount(int id) throws
             AccountIDNotRecognisedException {
92          //a function to remove accounts by their id
93          boolean exists = false;
94          int j = 0;
95          String loopHandle = "";
96          //looping through the accounts until the id and
                handle for that account are found and making
                sure they exists
97          while (j < accounts.size()) {
98              if (accounts.get(j).getid() == id) {
99                  exists = true;
100                 loopHandle = accounts.get(j).getHandle();
101             }
102             j++;
103         }
104         //if the acocunt exists, removing the posts from
                that account
105         if (exists){
106             //looping through the posts to find the posts
                    from that acount
107             for (Post loopPost : posts) {
108                 if (loopPost.getHandle().equals(
                        loopHandle)) {
```

```
109                     //using the delete post function for
                            every post, which needs a try
                            catch
110                     try{
111                         deletePost(loopPost.getId());
112                     } catch (PostIDNotRecognisedException
                            e) {
113                         System.out.println("
                                PostIDNotRecognisedException
                                thrown");
114                     }
115                 }
116             }
117         }
118
119         int i=0;
120         while (i < accounts.size()){
121             //removing that account from the array list
122             if(accounts.get(i).getid() == id){
123                 exists = true;
124                 accounts.remove(i);
125             }
126             i++;
127         }
128         //throwing the relevant exceptions
129         if (!exists){
130             throw new AccountIDNotRecognisedException("
                    account id not recognised");
131         }
132     }
133
134     @Override
135     public void removeAccount(String handle) throws
            HandleNotRecognisedException {
136         //a function to remove accounts by their handle
137
138         //looping through the posts and removing the
                posts that match the inputed account handle
139         for (Post loopPost : posts) {
140             if (loopPost.getHandle().equals(handle)) {
141                 try{
142                     deletePost(loopPost.getId());
143                 } catch (PostIDNotRecognisedException e)
                        {
144                     System.out.println("
                            PostIDNotRecognisedException
```

```
                        thrown");
145                 }
146             }
147         }
148 
149         boolean exists = false;
150         int i=0;
151         //looping through the accounts and removing the
                account with that handle
152         while (i < accounts.size()){
153             if(accounts.get(i).getHandle().equals(handle)
                    ){
154                 exists = true;
155                 accounts.remove(i);
156             }
157             i++;
158         }
159         //if that handle doesn't exist throw an exception
160         if (!exists){
161             throw new HandleNotRecognisedException("
                    handle not recognised");
162         }
163     }
164 
165     @Override
166     public void changeAccountHandle(String oldHandle,
            String newHandle) throws
            HandleNotRecognisedException,
            IllegalHandleException, InvalidHandleException {
167         //a method that takes an old account handle and
                replaces it with a new one
168         boolean exists = false;
169         //checking the new handle is correct
170         if (newHandle.length() > 1){
171             boolean unique = true;
172             //looping through the accounts to make sure
                    that the handle is unique
173             for(Account loopAccount : accounts){
174                 if(loopAccount.getHandle().equals(
                        newHandle)){
175                     unique = false;
176                 }
177             }
178             if (unique){
179                 //looping through the current accounts to
                        make sure that the handle exists
```

```
180                 for (Account loopAccount : accounts) {
181                     if (loopAccount.getHandle().equals(
                            oldHandle)) {
182                         exists = true;
183                         loopAccount.setHandle(newHandle);
184                     }
185                 }
186             //throwing the relevent exceptions if
                    statements aren't satisfied
187                 if (!exists){
188                     throw new
                            HandleNotRecognisedException("
                            handle not recognised");
189                 }
190             } else {
191                 //######throw an exception here######
192                 throw new IllegalHandleException("handle
                        already in use");
193             }
194         } else {
195             throw new InvalidHandleException("invalid
                    handle length");
196         }
197     }
198
199
200     @Override
201     public void updateAccountDescription(String handle,
            String description) throws
            HandleNotRecognisedException {
202         //a method that takes an account handle and
                description to update the account
203         boolean exists = false;
204         //looping thwough the accounts to make sure the
                account exists and then updating the description
205         for (Account loopAccount : accounts) {
206             if (loopAccount.getHandle().equals(handle)) {
207                     exists = true;
208                 loopAccount.setDescription(description);
209             }
210         }
211         //if the account doesn't exists throw an exception
212         if (!exists){
213                 throw new HandleNotRecognisedException("
                        handle not recognised");
214         }
```

```
215     }
216
217     @Override
218     public String showAccount(String handle) throws
            HandleNotRecognisedException {
219         //a method to show an account
220         boolean exists = false;
221        String printedAccount = "";
222        //looping through the accounts to  make sure the
            account exists
223        for (Account loopAccount : accounts) {
224            if (loopAccount.getHandle().equals(handle)) {
225                exists = true;
226                //then creating a string from the account
                       attributes
227                printedAccount += "ID: " + Integer.toString(
                       loopAccount.getid()) + "\n" + "Handle: " +
                        loopAccount.getHandle() + "\n" + "
                       Description: " + loopAccount.getDesc() + "
                       \n" + "Post count: " + loopAccount.
                       getNoPosts() + "\n" + "Endorse count: " +
                       loopAccount.getNoEndorsements() + "\n";
228            }
229        }
230          //if the account exists return the created string
                and if not throw an exception
231        if (exists){
232            return printedAccount;
233          } else {
234            throw new HandleNotRecognisedException("
                   handle not recognised");
235          }
236     }
237
238      @Override
239      public int createPost(String handle, String message)
            throws HandleNotRecognisedException,
            InvalidPostException {
240          //a method to create a post
241          boolean exists = false;
242          Account temp = null;
243          //looping through the accounts to make sure the
                account making the post exists
244          for (Account loopAccount : accounts) {
245              if (loopAccount.getHandle().equals(handle)) {
246                  exists = true;
```

```
247                 temp = loopAccount;
248             }
249         }
250         //if that account does exist, the character
                length of the post is checked to make sure it
                is within the limit
251         if (exists){
252             if (message.length() < 100 && message.length
                    () > 0){
253                 postIds.add(postIds.size());
254                 //new post is made if conditions are met
255                 Post newPost = new Post(postIds.size()-1,
                        message, handle);
256                 posts.add(newPost);
257                 temp.addPost();
258                 return postIds.get(postIds.size()-1);
259             } else {
260                 //if that post does not meet the required
                        conditions an exception is thrown
261                 throw new InvalidPostException("message
                        length is incorrect");
262             }
263         } else {
264             //if an account with the handle does not
                    exist an exception is thrown
265             throw new HandleNotRecognisedException("
                    handle not recognised");
266         }
267     }
268
269     @Override
270     public int endorsePost(String handle, int id) throws
            HandleNotRecognisedException,
            PostIDNotRecognisedException,
            NotActionablePostException {
271         //a method to endorse a post
272         boolean created = false;
273         boolean exists = false;
274         Account temp = null;
275         //looping through the accounts to check if an
                account with that handle exists
276         for (Account loopAccount : accounts) {
277             if (loopAccount.getHandle().equals(handle)) {
278                 exists = true;
279             }
280         }
```

```
281         //if that account does exist, the existence of
                the post the user is attempting to endorse is
                checked
282         if (exists){
283             int i = 0;
284             while (i < posts.size()){
285                 if (posts.get(i).getId() == id) {
286                     created = true;
287                     postIds.add(postIds.size());
288                     //if those conditions are met, a new
                            post is created with the correct
                            formatting for an endorsement
289                     String message = "EP@" + posts.get(i)
                            .getHandle() + ":" + posts.get(i).
                            getMessage();
290                     Post newPost = new Post(postIds.size
                            ()-1, message, handle);
291                     //total number of endorsements a post
                             has received is incremented by 1
292                     posts.get(i).addEndorsement();
293                     posts.add(newPost);
294                     //add endorsement to post handle
295                     for (Account loopAccount : accounts)
                            {
296                         if (loopAccount.getHandle().
                                equals(posts.get(i).getHandle
                                ())) {
297                             //total number of
                                    endorsements the account
                                    with that handle has
                                    received is incremented by
                                     1
298                             loopAccount.addEndorsement();
299                         }
300                     }
301                 }
302                 i++;
303             }
304             if (created) {
305                 return postIds.get(postIds.size()-1);
306             } else {
307                 //if the post id does not exist an
                        exception is thrown
308                 throw new PostIDNotRecognisedException("
                        post id does not exist");
309             }
```

```
310             } else {
311                 //if an account with that handle does not
                        exist an exception is thrown
312                 throw new HandleNotRecognisedException("
                        handle not recognised");
313             }
314         }
315
316         @Override
317         public int commentPost(String handle, int id, String
                message) throws HandleNotRecognisedException,
                PostIDNotRecognisedException,
                NotActionablePostException, InvalidPostException {
318             //a method to post a comment to a post
319             boolean created = false;
320             boolean exists = false;
321             Account temp = null;
322             //looping to check if an account with that handle
                    exists
323             for (Account loopAccount : accounts) {
324                 if (loopAccount.getHandle().equals(handle)) {
325                     exists = true;
326                     temp = loopAccount;
327                 }
328             }
329             /*if that account does exist, the existence of
                    the post the user is trying to comment on is
                    checked by its id,
330             as well as checking if the post the user is
                    trying to comment on is not an endorsement and
                    that the comment is
331             withing the character limit for a post
332                 */
333             if (exists){
334                 int i = 0;
335                 while (i < posts.size()){
336                     if (posts.get(i).getId() == id && !posts.
                            get(i).getMessage().substring(0,2).
                            equals("EP@") && message.length() <
                            100 && message.length() > 0) {
337                         created = true;
338                         postIds.add(postIds.size());
339                         //if those conditions are met, a new
                                post is created that is marked as
                                a comment
```

```
340                     Post newComment = new Post(postIds.
                            size() - 1, message, handle, "c");
341                     posts.add(newComment);
342                     posts.get(i).addCommentId(postIds.
                            size() - 1);
343                     posts.get(i).addComment();
344                     temp.addPost();
345                 }
346                 i++;
347             }
348         } else {
349             //if an account with that handle does not
                    exist an exception is thrown
350             throw new HandleNotRecognisedException("
                    handle not recognised");
351         }
352
353         if (created) {
354             return postIds.size()-1;
355         } else if (message.length() < 100 && message.
                length() > 0) {
356             //if the post is invalid an exception is
                    thrown
357             throw new InvalidPostException("invalid post"
                    );
358         } else if (!postIds.contains(id)) {
359             //if the post id does not exist an exception
                    is thrown
360             throw new PostIDNotRecognisedException("post
                    id does not exist");
361         } else {
362             for (Post loopPost : posts) {
363                 if (loopPost.getHandle().equals(handle)
                        && loopPost.getMessage().substring
                        (0,2).equals("EP@")) {
364                     /*if the post the user is attempting
                            to comment on is not actionable (e
                            .g. it is an endorsement)
365                     an exception is thrown
366                      */
367                     throw new NotActionablePostException(
                            "Not actionable post");
368                 }
369             }
370             return -1;
371         }
```

```
372
373     }
374
375     @Override
376     public void deletePost(int id) throws
            PostIDNotRecognisedException {
377         boolean exists = false;
378         int i=0;
379         //looping to check if a post with the id entered
                by the user exists
380         while (i < posts.size()){
381             if(posts.get(i).getId() == id){
382                 exists = true;
383                 for (Account loopAccount : accounts){
384                     if (loopAccount.getHandle().equals(
                            posts.get(i).getHandle())){
385                         /*the number of endorsements that
                                post had is removed from the
                                total number of endorsements
                                the
386                         account has
387                             */
388                         loopAccount.removeEndorsements(
                                posts.get(i).getNoEndorsements
                                ());
389                     }
390                 }
391                 //the handle above the post is set to
                        empty and the post message is set to a
                         generic message
392                 posts.get(i).setHandle("");
393                 posts.get(i).setMessage("The original
                        content was removed from the system
                        and is no longer available.");
394                 //the endorsements are removed from that
                        post
395                 posts.get(i).removeEndorsements();
396             }
397             i++;
398         }
399         if (exists) {
400             //looping to remove endorsement posts tied to
                    the post that was deleted
401             for (Post loopPost : posts) {
402                 if (loopPost.getId() == id) {
```

```
403                         String message = "EP@" + loopPost.
                                getHandle() + ":" + loopPost.
                                getMessage();
404                         int j = 0;
405                         while (j < posts.size()) {
406                             if (posts.get(j).getMessage().
                                    equals(message)) {
407                                 posts.remove(j);
408                             }
409                             j++;
410                         }
411                     }
412                 }
413             } else {
414                 //if the deleted post with the id the user
                        entered does not exist an exception is
                        thrown
415                 throw new PostIDNotRecognisedException("post
                        id does not exist");
416             }
417         }
418
419         @Override
420         public String showIndividualPost(int id) throws
                PostIDNotRecognisedException {
421             String individualPost = "";
422             boolean exists = false;
423             //looping to check if post with the id entered by
                    the user exists
424             for (Post loopPost : posts) {
425                 if (loopPost.getId() == id) {
426                     exists = true;
427                     /*String is set to contain all the
                            relevant information including the id
                            of the post, handle that made
428                         the post, number of endorsements on the
                            post, number of comments on the post
                            and the post message
429                         */
430                         individualPost = "ID: " + Integer.
                            toString(loopPost.getId()) + "\n" + "
                            Account: " + loopPost.getHandle() + "\
                            n" + "No. endorsements: " + Integer.
                            toString(loopPost.getNoEndorsements())
                             + " | No. comments: " + Integer.
                            toString(loopPost.getNoComments()) + "
```

```
                    \n" + loopPost.getMessage() + "\n";
431             }
432         }
433         if (exists) {
434           return individualPost;
435         } else {
436             //if the post with the id the user entered
                    does not exist an exception is thrown
437             throw new PostIDNotRecognisedException("post
                    id does not exist");
438         }
439     }
440
441     @Override
442     public StringBuilder showPostChildrenDetails(int id)
            throws PostIDNotRecognisedException,
            NotActionablePostException {
443         //a method that will show the post and then all
                the comments under the post
444         //it will also include the number of endorsments
                on that post
445
446         boolean exists = false;
447         //using a string builder to make the process
                faster
448         StringBuilder postTree = new StringBuilder();
449
450         //looping through the posts to find the post with
                the id in the parameter
451         for (Post loopPost : posts) {
452             if (loopPost.getId() == id) {
453                 exists = true;
454                 //if that id is true, add to the string
                        builder the 'show individual post' of
                        that post id and some formatting
455                 if (loopPost.getNoComments() > 0){
456                     postTree.append(showIndividualPost(id
                            ) + "|" + "\n" + "| > ");
457                 } else {
458                     postTree.append(showIndividualPost(id
                            ));
459                 }
460             }
461         }
462         if (exists) {
463             int i = 0;
```

```
464             //looping through the posts again to get the
                    'children' (or comments) on the post id
                    that was inputed in the parameter
465             while (i < posts.size()) {
466                 if (posts.get(i).getId() == id) {
467                     //getting all the comments on that
                            post,a nd looping through each one
468                     for (int commentId : posts.get(i).
                            getCommentIds()) {
469                         //using a temp to create the
                                indentation needed in the spec
470                         StringBuilder temp =
                                showPostChildrenDetails(
                                commentId);
471
472                         int j = 0;
473                         //looping through each character
                                of the temp and if it is a
                                newline adding some spaces for
                                 indentation
474                         while (j < temp.length()-1){
475                             //.insert(pos, c);
476                             if (temp.substring(j,j+1).
                                    equals("\n")){
477                                 temp.insert(j+1, "    ");
478                             }
479                             j++;
480                         }
481                         //appending this to the tree
482                         postTree.append(temp);
483                     }
484                 }
485                 i++;
486             }
487             return postTree;
488         //throwing and exception if the id is not found
489         } else {
490             throw new PostIDNotRecognisedException("post
                    id does not exist");
491         }
492     }
493
494     @Override
495     public int getNumberOfAccounts() {
496         //a method to return the number of accounts in the
                platform
```

```
497         return accounts.size();
498     }
499 
500     @Override
501     public int getTotalOriginalPosts() {
502         //a method to return the number of posts that are
                origional and not comments or endorsments
503         int total = 0;
504         //looping through the posts and adding to the
               total origioal posts if, not a comment and not
                an endorsement
505         for (Post loopPost : posts) {
506             if (!loopPost.getMessage().substring(0,3).
                    equals("EP@") && !loopPost.getIsComment()
                    && !loopPost.getHandle().equals("")) {
507                 total++;
508             }
509         }
510         return total;
511     }
512 
513     @Override
514     public int getTotalEndorsmentPosts() {
515         //a method that will return the total number of
               posts that are endorsements
516         int total = 0;
517         //looping through the posts and if the posts is an
              endorsement adding to the total endorsment posts
518         for (Post loopPost : posts) {
519             if (loopPost.getMessage().substring(0,3).
                    equals("EP@")) {
520                 total++;
521             }
522         }
523         return total;
524     }
525 
526     @Override
527     public int getTotalCommentPosts() {
528         //a method that will return the number of total
               comments on the platform
529         int total = 0;
530         //looping through the posts and if it is a comment
              adding to the total comment posts
531         for (Post loopPost : posts) {
532             if (loopPost.getIsComment()) {
```

```
533                 total++;
534             }
535         }
536         return total;
537     }
538 
539     @Override
540     public int getMostEndorsedPost() {
541         //a method that returns the post that has the
                most number of endorsements
542         int id = 0;
543         int max = -1;
544         //a max functions that loops through the posts to
                find the one with the largest number of
                endorsments
545         for (Post loopPost : posts) {
546             if (loopPost.getNoEndorsements() > max) {
547                 max = loopPost.getNoEndorsements();
548                 id = loopPost.getId();
549             }
550         }
551         return id;
552     }
553 
554     @Override
555     public int getMostEndorsedAccount() {
556         //a method that will return the account id with the
                most endorsments
557         int id = 0;
558         int max = -1;
559         //a max function that loops through the accounts to
                find the one with the largest number of
                endorsments
560         for (Account loopAccount : accounts) {
561             if (loopAccount.getNoEndorsements() > max) {
562                 max = loopAccount.getNoEndorsements();
563                 id = loopAccount.getid();
564             }
565         }
566         return id;
567     }
568 
569     @Override
570     public void erasePlatform() {
571         //a method to remove all the accounts and posts
                from the platform
```

```
572
573             accounts.clear();
574             ids.clear();
575             posts.clear();
576             postIds.clear();
577         }
578
579         @Override
580         public void savePlatform(String filename) throws
                IOException {
581             //a method to save the platform by writing to a ser
                    file
582             try{
583
584                 FileOutputStream writeData = new
                        FileOutputStream(filename + ".ser");
585                 ObjectOutputStream writeStream = new
                        ObjectOutputStream(writeData);
586                 writeStream.writeObject(accounts);
587                 writeStream.writeObject(ids);
588                 writeStream.writeObject(posts);
589                 writeStream.writeObject(postIds);
590
591                 writeStream.flush();
592                 writeStream.close();
593
594             }catch (IOException e) {
595                 e.printStackTrace();
596             }
597         }
598
599
600         @Override
601         public void loadPlatform(String filename) throws
                IOException, ClassNotFoundException {
602             //a method to load a platfomr froma ser file that
                    was saved
603             try{
604                 FileInputStream readData = new
                        FileInputStream(filename + ".ser");
605                 ObjectInputStream readStream = new
                        ObjectInputStream(readData);
606
607                 accounts = (ArrayList<Account>) readStream.
                        readObject();
```

```
608                 ids = (ArrayList<Integer>) readStream.
                        readObject();
609                 posts = (ArrayList<Post>) readStream.
                        readObject();
610                 postIds = (ArrayList<Integer>) readStream.
                        readObject();
611
612                 readStream.close();
613
614             }catch (Exception e) {
615                 e.printStackTrace();
616             }
617     }
618
619 }
```

## 2 Account.java

```
1  package socialmedia;
2
3  import java.io.*;
4
5  /**
6  * A class that will hold all of the account objects
7  **/
8  public class Account implements Serializable {
9      //Attributes
10     private int id;
11
12     private String handle;
13
14     private String description;
15
16     private int noEndorsements;
17
18     private int noPosts;
19
20     //methods
21     //setters
22     public void setHandle(String h) {
23         this.handle = h;
24     }
25
26     public void setDescription(String d) {
27         this.description = d;
28     }
29
30     public void setId(int id) {
31         this.id = id;
32     }
33
34     public void addEndorsement() {
35         this.noEndorsements += 1;
36     }
37
38     public void removeEndorsements(int n) {
39         this.noEndorsements -= n;
40     }
41
42     public void addPost() {
43         this.noPosts += 1;
```

```
44      }
45
46      //getters
47      public String getHandle(){
48          return this.handle;
49      }
50
51      public int getid(){
52          return this.id;
53      }
54
55      public String getDesc() {
56          return this.description;
57      }
58
59      public int getNoEndorsements() {
60          return this.noEndorsements;
61      }
62
63      public int getNoPosts() {
64          return this.noPosts;
65      }
66
67      //Constructors
68      public Account(int id, String h) {
69          this.id = id;
70          setHandle(h);
71      }
72
73      public Account(int id, String h, String d) {
74          this.id = id;
75          setHandle(h);
76          setDescription(d);
77      }
78  }
```

## 3 Post.java

```
1  package socialmedia;
2
3  import java.util.ArrayList;
4  import java.io.*;
5
6  /**
7  * A class that will hold all the post objects
8  **/
9  public class Post implements Serializable {
10     //attributes
11     private String message;
12
13     private int postId;
14
15     private String accountHandle;
16
17     private int noEndorsements;
18
19     private int noComments;
20
21     private boolean isComment;
22
23     private ArrayList<Integer> commentIds = new ArrayList
          <Integer>();
24
25     //methods
26     public void setMessage(String m) {
27         if (m.length() < 100) {
28             this.message = m;
29         }
30     }
31
32     public void setPostId(int id) {
33         this.postId = id;
34     }
35
36     public void setHandle(String h) {
37         this.accountHandle = h;
38     }
39
40     public void setisComment(boolean c){
41         this.isComment = c;
42     }
```

```
43
44      public void addCommentId(int id) {
45          this.commentIds.add(id);
46      }
47
48      public void removeEndorsements() {
49          this.noEndorsements = 0;
50      }
51
52      public void addEndorsement() {
53          this.noEndorsements += 1;
54      }
55
56      public void addComment() {
57          this.noComments += 1;
58      }
59
60      public String getHandle() {
61          return this.accountHandle;
62      }
63
64      public String getMessage() {
65          return this.message;
66      }
67
68      public int getNoEndorsements(){
69          return this.noEndorsements;
70      }
71
72      public int getNoComments(){
73          return this.noComments;
74      }
75
76      public boolean getIsComment(){
77          return this.isComment;
78      }
79
80      public ArrayList<Integer> getCommentIds() {
81          return this.commentIds;
82      }
83
84      public int getId() {
85          return this.postId;
86      }
87
88      //Constructors
```

```
89      public Post(int id, String m, String h) {
90          setPostId(id);
91          setMessage(m);
92          setHandle(h);
93          setisComment(false);
94      }
95
96      public Post(int id_, String m_, String h_, String c)
            {
97          setPostId(id_);
98          setMessage(m_);
99          setHandle(h_);
100         setisComment(true);
101     }
102
103     public Post(String m){
104         setMessage(m);
105     }
106 }
```

## 4 SocialMediaPlatformTestApp.java

```
1  import socialmedia.AccountIDNotRecognisedException;
2  import socialmedia.SocialMedia;
3  import socialmedia.IllegalHandleException;
4  import socialmedia.InvalidHandleException;
5  import socialmedia.SocialMediaPlatform;
6  import socialmedia.HandleNotRecognisedException;
7  import socialmedia.InvalidPostException;
8  import socialmedia.PostIDNotRecognisedException;
9  import socialmedia.NotActionablePostException;
10
11 import java.io.*;
12
13 /**
14  * A short program to illustrate an app testing some
        minimal functionality of a
15  * concrete implementation of the SocialMediaPlatform
        interface -- note you will
16  * want to increase these checks, and run it on your
        SocialMedia class (not the
17  * BadSocialMedia class).
18  *
19  *
20  * @author Diogo Pacheco
21  * @version 1.0
22  */
23 public class SocialMediaPlatformTestApp {
24
25    /**
26     * TEST method. This will test all the functionality
           of our social media class we created
27     *
28     * @param args not used
29     */
30    public static void main(String[] args) {
31       System.out.println("The system compiled and started
             the execution...");
32
33         //creating the new platform
34       SocialMediaPlatform platform = new SocialMedia();
35
36         //TEST checking the platform is created correctly
37       assert (platform.getNumberOfAccounts() == 0) : "
            Innitial SocialMediaPlatform not empty as
```

```
            required.");
38          assert (platform.getTotalOriginalPosts() == 0) : "
                Innitial SocialMediaPlatform not empty as
                required.";
39          assert (platform.getTotalCommentPosts() == 0) : "
                Innitial SocialMediaPlatform not empty as
                required.";
40          assert (platform.getTotalEndorsmentPosts() == 0) :
                "Innitial SocialMediaPlatform not empty as
                required.";
41
42          Integer id=null;
43          Integer id1;
44          Integer id2;
45
46
47          //TEST to check that the creation of the accounts
                and showing the accounts works
48          try {
49                System.out.println("----------------------showIDs
                        ----------------------");
50              id = platform.createAccount("my_handle");
51              System.out.println(id);
52              id1 = platform.createAccount("my_handle2", "
                    hello");
53              System.out.println(id1);
54              id2 = platform.createAccount("my_handle3", "
                    hello");
55              System.out.println(id2);
56
57              System.out.println("----------------------
                    showAccounts----------------------");
58              System.out.println(platform.showAccount("
                    my_handle"));
59              System.out.println(platform.showAccount("
                    my_handle2"));
60              System.out.println(platform.showAccount("
                    my_handle3"));
61
62          } catch (IllegalHandleException e) {
63              System.out.println("IllegalHandleException
                    thrown");
64          } catch (InvalidHandleException e) {
65              System.out.println("InvalidHandleException
                    thrown");
66          } catch (HandleNotRecognisedException e) {
```

```
67              System.out.println("HandleNotRecognisedException
                    thrown");
68          }
69
70
71          //TEST to check that the change account handle
                works
72          try {
73                  System.out.println("--------------------
                        changeAccountHandle----------------------");
74                  platform.changeAccountHandle("my_handle", "
                        Jeff");
75                  System.out.println(platform.showAccount("Jeff
                        "));
76
77          } catch (IllegalHandleException e) {
78              System.out.println("IllegalHandleException
                    thrown");
79          } catch (InvalidHandleException e) {
80              System.out.println("InvalidHandleException
                    thrown");
81          } catch (HandleNotRecognisedException e) {
82              System.out.println("HandleNotRecognisedException
                    thrown");
83          }
84
85
86          //TEST to check that the update account description
                works
87          try {
88                  System.out.println("--------------------
                        updateAccountDescription
                        ----------------------");
89                  platform.updateAccountDescription("my_handle2
                        ", "my name is");
90                  System.out.println(platform.showAccount("
                        my_handle2"));
91
92          } catch (HandleNotRecognisedException e) {
93              System.out.println("HandleNotRecognisedException
                    thrown");
94          }
95
96
97          //TEST to check the post works
98          try {
```

```
99              System.out.println("----------------------
                    createPosts----------------------");
100             System.out.println(platform.createPost("
                    my_handle2", "this is my first post"));
101             System.out.println(platform.createPost("Jeff"
                    , "this is MY first post"));
102             System.out.println("----------------------
                    showIndividualPosts----------------------");
103             System.out.println(platform.
                    showIndividualPost(0));
104             System.out.println(platform.
                    showIndividualPost(1));
105
106         } catch (HandleNotRecognisedException e) {
107             System.out.println("HandleNotRecognisedException
                    thrown");
108         } catch (InvalidPostException e) {
109             System.out.println("InvalidPostException
                    thrown");
110         } catch (PostIDNotRecognisedException e) {
111             System.out.println("
                    PostIDNotRecognisedException thrown");
112         }
113
114         //TEST to check the endorsments and post details
                works
115
116
117
118         //TEST to check the comments and post details works
119         try {
120             System.out.println("----------------------
                    commentPost----------------------");
121             //comments on posts
122             System.out.println(platform.commentPost("Jeff
                    ", 0, "a comment"));
123             System.out.println(platform.commentPost("
                    my_handle2", 1, "another comment"));
124             System.out.println(platform.commentPost("
                    my_handle3", 0, "a aklfrav comment"));
125
126             //comments on comments
127             System.out.println(platform.commentPost("Jeff
                    ", 2, "another imb comment"));
128             System.out.println(platform.commentPost("
                    my_handle3", 4, "another kjasflj comment")
```

```
                );
129
130             System.out.println("------------------
                    showPostChildrenDetails----------------------
                    ");
131             System.out.println(platform.
                    showPostChildrenDetails(0));
132             System.out.println(platform.
                    showPostChildrenDetails(1));
133
134         } catch (HandleNotRecognisedException e) {
135            System.out.println("HandleNotRecognisedException
                   thrown");
136         } catch (InvalidPostException e) {
137             System.out.println("InvalidPostException
                    thrown");
138         } catch (PostIDNotRecognisedException e) {
139             System.out.println("
                    PostIDNotRecognisedException thrown");
140         } catch (NotActionablePostException e) {
141             System.out.println("
                    NotActionablePostException thrown");
142         }
143
144
145         //TEST
146         try {
147             System.out.println("------------------
                    endorsePost----------------------");
148             //comments on posts
149             System.out.println(platform.endorsePost("
                    my_handle3", 0));
150             System.out.println(platform.endorsePost("Jeff
                    ", 1));
151
152             System.out.println("------------------
                    showPostChildrenDetails----------------------
                    ");
153             System.out.println(platform.
                    showPostChildrenDetails(0));
154             System.out.println(platform.
                    showPostChildrenDetails(1));
155           } catch (HandleNotRecognisedException e) {
156            System.out.println("HandleNotRecognisedException
                   thrown");
157         } catch (PostIDNotRecognisedException e) {
```

```
158             System.out.println("
                    PostIDNotRecognisedException thrown");
159         } catch (NotActionablePostException e) {
160             System.out.println("
                    NotActionablePostException thrown");
161         }
162
163
164         //TEST to check that the creation of the accounts
                and showing the accounts works
165         try {
166             System.out.println("--------------------
                    showAccounts----------------------");
167             System.out.println(platform.showAccount("Jeff"))
                    ;
168             System.out.println(platform.showAccount("
                    my_handle2"));
169             System.out.println(platform.showAccount("
                    my_handle3"));
170
171         } catch (HandleNotRecognisedException e) {
172             System.out.println("HandleNotRecognisedException
                     thrown incorrectly");
173         }
174
175
176         //TEST to check that the erase platform works
177         try{
178             platform.savePlatform("paul");
179
180         } catch (IOException e) {
181             System.out.println(e);
182         }
183
184
185         //TEST to check that the remove account by id works
                and the remove by handle works
186         try{
187             System.out.println("--------------------
                    removeAccounts----------------------");
188             platform.removeAccount(id);
189             assert (platform.getNumberOfAccounts() == 2) : "
                    number of accounts registered in the system
                    does not match";
190
191             platform.removeAccount("my_handle2");
```

```
192             assert (platform.getNumberOfAccounts() == 1) : "
                    number of accounts registered in the system
                    does not match";
193
194         } catch (HandleNotRecognisedException e) {
195             System.out.println("HandleNotRecognisedException
                     thrown incorrectly");
196         } catch (AccountIDNotRecognisedException e) {
197             System.out.println("
                    AccountIDNotRecognizedException thrown
                    incorrectly");
198         }
199
200
201         //TEST to check that the creation of the accounts
                and showing the accounts works
202         try {
203             System.out.println("----------------------
                    showAccounts----------------------");
204             System.out.println(platform.showAccount("
                    my_handle3"));
205
206             System.out.println("----------------------
                    showPostChildrenDetails----------------------");
207                 System.out.println(platform.
                        showPostChildrenDetails(0));
208                 System.out.println(platform.
                        showPostChildrenDetails(1));
209
210         } catch (HandleNotRecognisedException e) {
211             System.out.println("HandleNotRecognisedException
                     thrown");
212         } catch (PostIDNotRecognisedException e) {
213                 System.out.println("
                        PostIDNotRecognisedException thrown");
214         } catch (NotActionablePostException e) {
215                 System.out.println("
                        NotActionablePostException thrown");
216         }
217
218
219         //TEST to check that the methods that return
                information about the platform work
220         try{
221                 System.out.println("----------------------
                        getTotalOriginalPosts----------------------");
```

```
222             System.out.println(platform.
                    getTotalOriginalPosts());
223
224             System.out.println("----------------------
                    getTotalEndorsmentPosts----------------------"
                    );
225             System.out.println(platform.
                    getTotalEndorsmentPosts());
226
227             System.out.println("----------------------
                    getTotalCommentPosts----------------------");
228             System.out.println(platform.
                    getTotalCommentPosts());
229
230             System.out.println("----------------------
                    getMostEndorsedPost----------------------");
231             System.out.println("Post ID: " + platform.
                    getMostEndorsedPost());
232
233             System.out.println("----------------------
                    getMostEndorsedAccount----------------------")
                    ;
234             System.out.println("Account ID: " + platform.
                    getMostEndorsedAccount());
235
236             System.out.println();
237         //assert (platform.getNumberOfAccounts() == 1) :
                "number of accounts registered in the system
                does not match";
238
239     } catch (Exception e) {
240        System.out.println(e);
241     }
242
243
244       //TEST to check that the erase platform works
245     try{
246             platform.savePlatform("james");
247
248     } catch (IOException e) {
249        System.out.println(e);
250     }
251
252
253       //TEST to check that the erase platform works
254     try{
```

```
255                 platform.erasePlatform();
256
257         } catch (Exception e) {
258             System.out.println(e);
259         }
260
261         //TEST to check that the erase platform works
262         try{
263                 platform.loadPlatform("paul");
264
265         } catch (IOException e) {
266             System.out.println(e);
267         } catch (ClassNotFoundException e) {
268             System.out.println(e);
269         }
270     }
271
272 }
```