```java
package com.company;
import java.security.NoSuchAlgorithmException;
import java.sql.ResultSet;
import java.util.Locale;
import java.util.Scanner;
import java.sql.SQLException;
import java.util.concurrent.atomic.AtomicBoolean;
import java.util.concurrent.atomic.AtomicInteger;

public class Main {
    public static void main(String[] args){
        AccountManager accountManager = new AccountManager();
        Scanner keyboard = new Scanner(System.in);

        while(true) {
            Thread loginAndSignUpThread = new Thread(() -> { try { loginAndSignUp(accountManager); } catch (NoSuchAlgorithmException | SQLException | InterruptedException exception) { exception.printStackTrace(); } });
            AtomicBoolean getOutOfLoginScreen = new AtomicBoolean(false);
            boolean isOtherThreadsCreated = false;

            Thread informSchedulesThread = null;
            Thread notifyGranularSchedules = null;
            Thread decrementStreaksForOverdueTasksManager;

            try {
                loginAndSignUpThread.start();
                loginAndSignUpThread.join();

                while (true) {
                    //let user choose what to do before displaying upcoming schedules

                    System.out.println("""

                            Menus:
```

```
34                                  \t0. Show/Print
   schedules (based on their priority)\t1. Show/Print
   schedules\t2. Show/Print Streaks status\t3. Show/
   Print upcoming schedules\t4. Add a schedule\t5.
   Update/Edit a schedule
35                                  \t6. Complete/Delete a
   schedule\t7. Logout\t8. Change a username\t9.
   Change a password\t10. Delete an account\t(Type "
   exit" for Exit)
36                                  Choice:\t""");
37
38                          int userId = accountManager.
   getUserId(accountManager.getUsername());
39                          TimeDateCheck timeDateCheck =
   new TimeDateCheck(userId);
40                          if(!isOtherThreadsCreated) {
41                              //execute thread for dozens
    of minutes interval schedule notification
42                              informSchedulesThread = new
    Thread(() -> {
43                                  while (true) {
44                                      timeDateCheck.
   informSchedules();
45
46                                      final int
   APPROX_THIRTY_EIGHT_MINUTES = 2300000;
47                                      try {
48                                          Thread.sleep(
   APPROX_THIRTY_EIGHT_MINUTES);
49                                      } catch (
   InterruptedException exception) {
50                                          exception.
   printStackTrace();
51                                      }
52                                  }
53                              });
54                              informSchedulesThread.start
   (); //no .join; concurrent operation
55
56
57                              //execute thread for minute
```

```
57 -to-minute interval schedule notification
58                          notifyGranularSchedules =
   new Thread(() -> {
59                              while (true) {
60                                  try {
61                                      timeDateCheck.
   notifyGranularSchedules();
62                                      timeDateCheck.
   printLocalTime();
63                                      accountManager.
   printStreakAndStreakLevel(userId);
64
65                                      final int
   A_MINUTE_AND_A_HALF = 90000;
66                                      Thread.sleep(
   A_MINUTE_AND_A_HALF);
67
68                                  } catch (
   InterruptedException | SQLException exception) {
69                                      exception.
   printStackTrace();
70                                  }
71                              }
72                          });
73                          notifyGranularSchedules.
   start(); //no .join; concurrent operation
74
75
   decrementStreaksForOverdueTasksManager = new Thread
   (() -> {
76                              while(true){
77                                  try {
78                                      timeDateCheck.
   periodicallyGiveBonusStreaks();
79                                      timeDateCheck.
   updateDailyStreak(userId);
80                                      timeDateCheck.
   decrementStreaksForOverdueTasks(userId);
81
82                                      int ONE_HOUR =
   3600000;
```

```
 83
 84                                      Thread.sleep(
    ONE_HOUR);
 85                                  } catch (
    SQLException | InterruptedException exception) {
 86                                      exception.
    printStackTrace();
 87                                  }
 88                              }
 89                          });
 90
    decrementStreaksForOverdueTasksManager.start();
 91
 92                          isOtherThreadsCreated =
    true;
 93                      }
 94
 95
 96                      String optionChoice = keyboard
    .nextLine().replaceAll("\\s", "").toLowerCase(
    Locale.ROOT);
 97                      switch (optionChoice) {
 98                          case "show/printschedules(
    basedontheirpriority)", "printpriorityschedule", "
    printprioityschedules", "printpriority", "
    showpriorityschedule", "showprioityschedules", "0
    " -> {
 99                              Thread
    printPriorityTasksThread = new Thread(()-> {
100                                 try {
101
    SchedulePriority.printScheduleBasedOnPriority(
    userId);
102                                 } catch (
    SQLException exception) {
103                                     exception.
    printStackTrace();
104                                 }
105                             });
106
    printPriorityTasksThread.start();
```

```
107
    printPriorityTasksThread.join();
108                         }
109                         case "printschedule", "
    printschedules", "print", "showschedule", "
    showschedules", "show", "1" -> {
110                             Thread
    printTasksThread =  new Thread (()-> { try {
    accountManager.printTasks(userId, accountManager.
    getUsername()); } catch (SQLException |
    InterruptedException exception){ exception.
    printStackTrace(); } });
111                             printTasksThread.start
    ();
112                             printTasksThread.join
    ();
113                         }
114
115                         case "showstreak", "
    showstreakstatus", " showstreaks", "
    showstreaksstatus", "printstreak", "
    printstreakstatus", "printstreaks", "
    printstreaksstatus", "2" -> accountManager.
    printStreakAndStreakLevel(userId);
116
117                         case "
    printupcomingschedules", "showupcomingschedules",
    "printupcomings", "showupcomings", "printupcoming"
    , "showupcoming", "3" -> {
118                             timeDateCheck.
    informSchedules();
119                             timeDateCheck.
    notifyGranularSchedules();
120                             timeDateCheck.
    printLocalTime();
121                         }
122
123                         case "addschedule", "
    addaschedule", "addtheschedule", "4" -> {
124                             Thread
    addScheduleThread = new Thread(()-> { try {
```

```
124 accountManager.addManualSchedule(userId); } catch
     (SQLException exception) { exception.
    printStackTrace(); } });
125                                 addScheduleThread.
    start();
126                                 addScheduleThread.join
    ();
127                             }
128
129                             case "editschedule", "
    editaschedule", "updateschedule", "updateaschedule
    ", "edit/updateschedule", "edit/updateaschedule",
    "editorupdateschedule", "editorupdateaschedule", "
    update/editschedule", "update/editaschedule", "
    updateoreditschedule", "updateoreditaschedule","5"
     -> {
130                                 Thread
    editScheduleThread = new Thread(()-> { try {
    accountManager.editSchedule(userId); } catch (
    SQLException | InterruptedException exception) {
    exception.printStackTrace(); } });
131                                 editScheduleThread.
    start();
132                                 editScheduleThread.
    join();
133                             }
134
135                             case "completeschedule", "
    deleteschedule", "completeaschedule", "
    deleteaschedule", "complete/deleteschedule", "
    complete/deleteaschedule", "
    completeordeleteschedule", "
    completeordeleteaschedule", "6" -> {
136                                 Thread
    editScheduleThread = new Thread(()-> { try {
    accountManager.completeOrRemoveSchedule(userId
    ); } catch (SQLException | InterruptedException
    exception) { exception.printStackTrace(); } });
137                                 editScheduleThread.
    start();
138                                 editScheduleThread.
```

```
138 join();
139                         }
140
141                         case "logout", "7" -> {
142                             getOutOfLoginScreen.
    set(accountManager.logout()); //too costly too
    make a new thread
143                             informSchedulesThread.
    stop();
144
    notifyGranularSchedules.stop();
145                         }
146
147                         case "changeusername", "
    changeausername", "8" -> {
148                             //variables in lambda
    expression must be either final or effectively
    final
149                             Thread
    finalInformSchedulesThread = informSchedulesThread
    ;
150                             Thread
    finalNotifyGranularSchedules =
    notifyGranularSchedules;
151                             Thread
    changeUsernameThread = new Thread(()-> {
152                                 try {
    accountManager.changeUsername(userId); } catch (
    NoSuchAlgorithmException | SQLException exception
    ) { exception.printStackTrace(); }
153
    getOutOfLoginScreen.set(accountManager.logout());
154
    finalInformSchedulesThread.stop();
155
    finalNotifyGranularSchedules.stop();
156                             });
157                             changeUsernameThread.
    start();
158                             changeUsernameThread.
    join();
```

```
159                             }
160
161                             case "changepassword", "
    changeapassword", "9" -> {
162                                 Thread
    finalInformSchedulesThread1 =
    informSchedulesThread;
163                                 Thread
    finalNotifyGranularSchedules1 =
    notifyGranularSchedules;
164                                 Thread
    changeUsernameThread = new Thread(()-> {
165                                     try {
    accountManager.changePassword(userId); } catch (
    NoSuchAlgorithmException | SQLException exception
    ) { exception.printStackTrace(); }
166
    getOutOfLoginScreen.set(accountManager.logout());
167
    finalInformSchedulesThread1.stop();
168
    finalNotifyGranularSchedules1.stop();
169                                 });
170                                 changeUsernameThread.
    start();
171                                 changeUsernameThread.
    join();
172                             }
173
174                             case "deleteaccount", "
    deleteanaccount", "10" -> {
175                                 Thread
    finalInformSchedulesThread2 =
    informSchedulesThread;
176                                 Thread
    finalNotifyGranularSchedules2 =
    notifyGranularSchedules;
177                                 Thread
    deleteAccountThread = new Thread(()-> {
178                                     try {
    accountManager.removeAccount(userId); } catch (
```

```
178 SQLException exception) { exception.
    printStackTrace(); }
179
    getOutOfLoginScreen.set(accountManager.logout());
180
    finalInformSchedulesThread2.stop();
181
    finalNotifyGranularSchedules2.stop();
182                             });
183                             deleteAccountThread.
    start();
184                             deleteAccountThread.
    join();
185                         }
186
187                         case "exit" -> {
188                             System.out.println("
    Bye!");
189                             System.exit(0);
190                         }
191
192                         default -> System.out.
    println("Wrong option.");
193                     }
194                     if (getOutOfLoginScreen.get
    ()) {
195                         System.out.println("Logged
     out.");
196                         break;
197                     }
198                 }
199             } catch (InterruptedException |
    SQLException exception){
200                 exception.printStackTrace();
201             }
202         }
203     }
204
205     private static void loginAndSignUp(
    AccountManager accountManager) throws
    NoSuchAlgorithmException, SQLException,
```

```
205 InterruptedException {
206         AtomicBoolean loginSucceed = new
    AtomicBoolean(false);
207         Scanner keyboard = new Scanner(System.in);
208         AtomicInteger countLoginAttempt = new
    AtomicInteger(0);
209
210         do {
211             Thread getPasswordHintThread =  new
    Thread(()-> { try { getPasswordHint(
    countLoginAttempt.get()); } catch (SQLException
    exception) { exception.printStackTrace(); } });
212             getPasswordHintThread.start();
213             getPasswordHintThread.join();
214             System.out.println("Do you want to
    Login (1) or Sign-up (2) to an account? (Type \"
    exit\" for Exit)");
215             String signUpOrLogin = keyboard.
    nextLine().toLowerCase(Locale.ROOT);
216             switch (signUpOrLogin) {
217                 case "login" , "log in", "1"-> {
218                     Thread loginThread = new
    Thread(() -> {
219                         try { loginSucceed.set(
    accountManager.login()); } catch (SQLException |
    NoSuchAlgorithmException exception) { exception.
    printStackTrace(); }
220                         countLoginAttempt.
    getAndIncrement();
221                     });
222                     loginThread.start();
223                     loginThread.join();
224                 }
225                 case "sign-up", "signup", "2" -> {
226                     Thread signUpThread = new
    Thread(()-> { try { accountManager.signUp(); }
    catch (NoSuchAlgorithmException exception) {
    exception.printStackTrace(); } });
227                     signUpThread.start();
228                     signUpThread.join();
229                 }
```

```
230                 case "exit" -> {
231                     System.out.println("Bye!");
232                     System.exit(0);
233                 }
234                 default -> System.out.println("
    Wrong option.");
235             }
236         } while (!loginSucceed.get()); //repeat
    until login succeed
237     }
238 
239     private static void getPasswordHint(int
    countLoginAttempt) throws SQLException {
240         Scanner keyboard = new Scanner(System.in);
241 
242         if(countLoginAttempt >= 6){
243             System.out.println("Shutting down the
    program for security reasons. (6 incorrect login
    attempts)");
244             System.exit(0);
245         } else if(countLoginAttempt >=3){
246             MysqlExecute mysqlExecute = new
    MysqlExecute();
247             System.out.println("In order to get
    the password hint, what is the username of the
    user that you are trying to log in?");
248             String usernameSearch = keyboard.
    nextLine();
249             ResultSet passwordHint = mysqlExecute.
    mysqlQuery("SELECT passwordHint FROM UserList
    WHERE username = \"" + usernameSearch + "\";");
250             if(passwordHint == null){
251                 throw new SQLException();
252             }
253             if(passwordHint.next()){
254                 System.out.println("Password Hint
    : \"" + passwordHint.getString(1) + "\".");
255             }else{
256                 System.out.println("Sorry, there
    is no user called \"" + usernameSearch + "\".");
257             }
```

```
258             }
259         }
260 }
261
262
```