

JAVA PROJECT ON ATM APPLICATION BY USING OOPs CONCEPT

PROJECT STATEMENT:

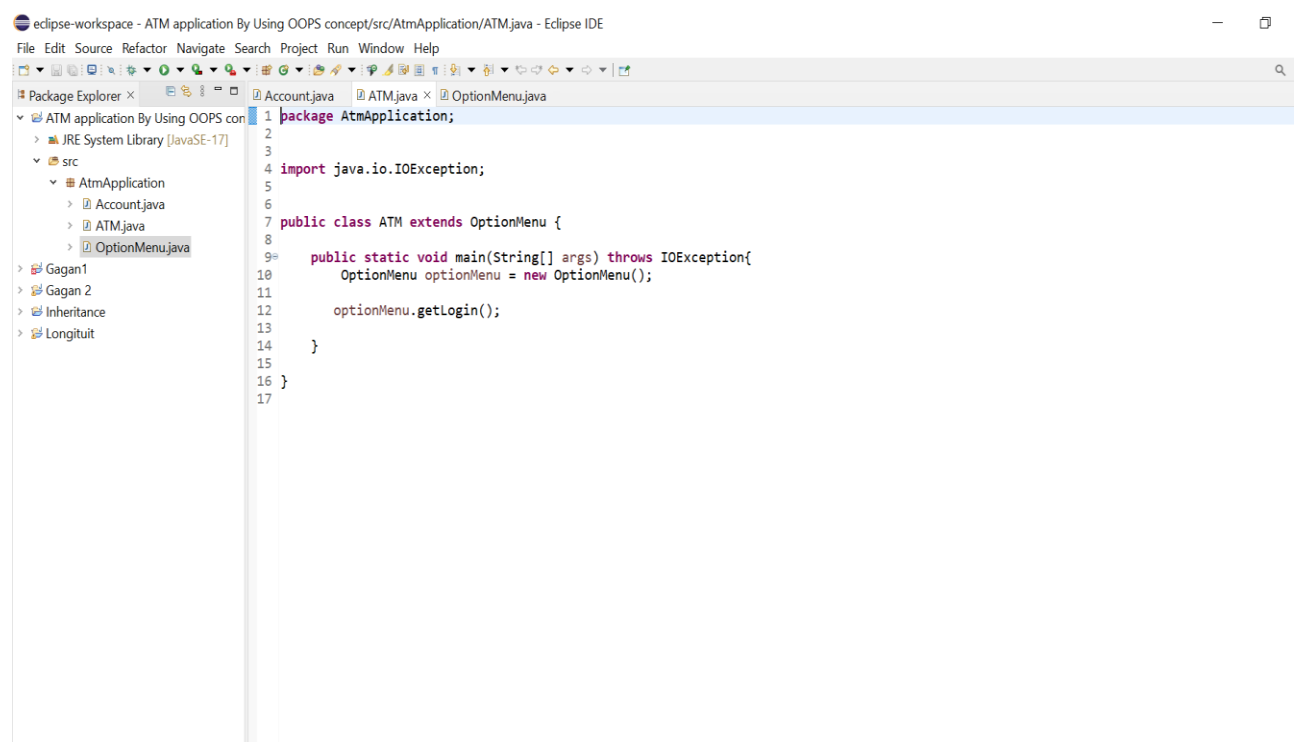
This project is to make an automated teller machine (ATM) with customer number and customer pin number with that customer are able to withdraw, deposit and view their account balance.

Basically, I created three main classes:

- 1) **ATM Class**
- 2) **OptionsMenu Class**
- 3) **Account Class**

1) **ATM Class:**

It is mainly an entry point class which basically initializes OptionMenu class and calls the get login method. It is a first method in my ATM application program in which ATM extends OptionMenu class by using EXTENDS keyword.



```
1 package AtmApplication;
2
3
4 import java.io.IOException;
5
6
7 public class ATM extends OptionMenu {
8
9     public static void main(String[] args) throws IOException{
10         OptionMenu optionMenu = new OptionMenu();
11
12         optionMenu.getLogin();
13
14     }
15
16 }
17
```

2) **OptionsMenu Class:**

It will give the understanding of what kind of functionality iam using,
Here mainly iam using three main class functions:

- 1) Java.io.IOException
- 2) Java.text.DecimalFormat
- 3) Java.util.HashMap
- 4) Java.util.Scanner

- IOException class is used to check exception that occurs during the compilation time of the java program.
- HashMap class is used to provide functionality of the hash table data structure and it is used to store key/value pairs.
- We use Decimal Format class in order to denote what kind of format of money we have.
- We use HashMap for storing couple of records of account number to pin mapping.

First Method I used is Here mainly we use getLogin() method where we had couple of entries to map, Where I entered customer number with customer pin number and I done same for 2 more entries(It can be done for infinity numbers).

If we call the getLogin() method it will insert the customer number and customer pin number.

If we give wrong customer number it would give wrong customer number or Invalid which I was written in catch block

Second Method I used is getaccount type(). In this where the program asks the user what type of account you want to access.

There are mainly two type of account I initialized,

- 1) **Checking Account**
- 2) **Saving Account**
- 3) **Exit**

Here I used the Switch condition to perform some operations where if I press 1 it goes to Checking account, similarly if I press 2 it goes to Saving Account and if I press 3 it Exits from the ATM application.

Third Method and Fourth Method is getChecking method and getSaving method

In this it performs three different Account Functions

- 1) **View Balance**
- 2) **Withdraw Funds**
- 3) **Deposit Funds**
- 4) **Exit**

Here I used the Switch condition to perform some operations where if I press 1 it checks account balance, if I press 2 it debits money from account, similarly if I press 3 it credits money to account and if I press 4 it Exits from the ATM account Functions (This method is same for both Account Types).

eclipse-workspace - ATM application By Using OOPS concept/src/AtmApplication/OptionMenu.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer x

ATM application By Using OOPS concept

- JRE System Library [JavaSE-17]
- src
 - AtmApplication
 - Account.java
 - ATM.java
 - OptionMenu.java
 - Gagan1
 - Gagan 2
 - Inheritance
 - Longituit

```
1 package AtmApplication;
2
3 import java.io.IOException;
4
5
6
7
8 public class OptionMenu extends Account {
9     Scanner menuInput = new Scanner(System.in);
10    DecimalFormat moneyFormat = new DecimalFormat("'$'###,##0.00");
11
12    HashMap<Integer, Integer> data = new HashMap<Integer, Integer>();
13
14    public void getLogin() throws IOException {
15        int x = 1;
16
17        do {
18            try {
19
20                data.put(123456789, 1234);
21                data.put(987654321, 4321);
22
23                System.out.println("Welcome to the ATM Project!");
24
25                System.out.println("Enter your Customer Number: ");
26                setCustomerNumber(menuInput.nextInt());
27
28                System.out.print("Enter your PIN Number: ");
29                setPinNumber(menuInput.nextInt());
30            } catch (Exception e) {
31                System.out.println("\n" + "Invalid Character(s). Only Numbers." + "\n");
32                x = 2;
33            }
34            int cn = getCustomerNumber();
35            int pn = getPinNumber();
36            if (data.containsKey(cn) && data.get(cn) == pn) {
37                getAccountType();
38            } else
39                System.out.println("\n" + "Wrong Customer Number or Pin Number" + "\n");
40        } while (x == 1);
41    }
42}
```

eclipse-workspace - ATM application By Using OOPS concept/src/AtmApplication/OptionMenu.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer x

ATM application By Using OOPS concept

- JRE System Library [JavaSE-17]
- src
 - AtmApplication
 - Account.java
 - ATM.java
 - OptionMenu.java
 - Gagan1
 - Gagan 2
 - Inheritance
 - Longituit

```
40
41 }
42
43 public void getAccountType() {
44     System.out.println("Select the Account you want to Access: ");
45     System.out.println(" Type 1 - Checking Account");
46     System.out.println(" Type 2 - Saving Account");
47     System.out.println(" Type 3 - Exit");
48
49
50     int selection = menuInput.nextInt();
51
52     switch (selection) {
53     case 1:
54         getChecking();
55         break;
56
57     case 2:
58         getsaving();
59         break;
60
61     case 3:
62         System.out.println("Thank you for using this ATM, bye. \n");
63         break;
64
65     default:
66         System.out.println("\n" + "Invalid Choice." + "\n");
67         getAccountType();
68     }
69 }
70
71
72
73 public void getChecking() {
74     System.out.println("Checking Account: ");
75     System.out.println(" Type 1 - View Balance");
76 }
```

eclipse-workspace - ATM application By Using OOPS concept/src/AtmApplication/OptionMenu.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer x

- ATM application By Using OOPS con
 - JRE System Library [JavaSE-17]
 - src
 - AtmApplication
 - Account.java
 - ATM.java
 - OptionMenu.java
 - Gagan1
 - Gagan 2
 - Inheritance
 - Longituit

```
73 public void getChecking() {
74     System.out.println("Checking Account: ");
75     System.out.println(" Type 1 - View Balance");
76     System.out.println(" Type 2 - Withdraw Funds");
77     System.out.println(" Type 3 - Deposit Funds");
78     System.out.println(" Type 4 - Exit");
79     System.out.println("Choice: ");
80
81
82     int selection = menuInput.nextInt();
83
84     switch (selection) {
85     case 1:
86         System.out.println("Checking Account Balance: " + moneyFormat.format(getCheckingBalance()));
87         getAccountType();
88         break;
89
90     case 2:
91         getCheckingWithdrawInput();
92         getAccountType();
93         break;
94
95     case 3:
96         getCheckingDepositInput();
97         getAccountType();
98         break;
99
100    case 4:
101        System.out.println("Thank you for using this ATM, bye.");
102        break;
103
104    default:
105        System.out.println("\n" + "Invalid choice." + "\n");
106        getChecking();
107    }
108 }
109
```

eclipse-workspace - ATM application By Using OOPS concept/src/AtmApplication/OptionMenu.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer x

- ATM application By Using OOPS con
 - JRE System Library [JavaSE-17]
 - src
 - AtmApplication
 - Account.java
 - ATM.java
 - OptionMenu.java
 - Gagan1
 - Gagan 2
 - Inheritance
 - Longituit

```
113
114 public void getsaving() {
115     System.out.println("Saving Account: ");
116     System.out.println("Type 1 - View Balance");
117     System.out.println("Type 2 - Withdraw Funds");
118     System.out.println("Type 3 - Deposit Funds");
119     System.out.println("Type 4 - Exit");
120     System.out.print("Choice: ");
121
122
123     int selection = menuInput.nextInt();
124
125     switch (selection) {
126     case 1:
127         System.out.println("Saving Account Balance: " + moneyFormat.format(getSavingBalance()));
128         getAccountType();
129         break;
130
131     case 2:
132         getSavingWithdrawInput();
133         getAccountType();
134         break;
135
136     case 3:
137         getSavingDepositInput();
138         getAccountType();
139         break;
140
141     case 4:
142         System.out.println("Thank you for using this ATM, bye.");
143         break;
144
145     default:
146         System.out.println("\n" + "Invalid choice." + "\n");
147         getsaving();
148    }
149 }
```

```
110 System.out.println("Type 3 - Deposit runus ");
119 System.out.println("Type 4 - Exit");
120 System.out.print("Choice: ");
121
122 int selection = menuInput.nextInt();
123
124 switch (selection) {
125     case 1:
126         System.out.println("Saving Account Balance: " + moneyFormat.format(getSavingBalance()));
127         getAccountType();
128         break;
129
130     case 2:
131         getSavingWithdrawInput();
132         getAccountType();
133         break;
134
135     case 3:
136         getSavingDepositInput();
137         getAccountType();
138         break;
139
140     case 4:
141         System.out.println("Thank you for using this ATM, bye.");
142         break;
143
144     default:
145         System.out.println("\n" + "Invalid choice." + "\n");
146         getsaving();
147
148 }
149
150 }
151
152 }
153
154 }
```

3) Account Class:

The entire transaction calculation part of a program happens in the account class and Here I defined private member variables like Customer number, Pin number, checking balance and Saving balance.

Then mainly I used Getters and Setters method for what I specified in private member variables.

```
1 package AtmApplication;
2
3
4
5 import java.text.DecimalFormat;
6
7
8
9 public class Account {
10
11     private int CustomerNumber;
12     private int pinNumber;
13     private double checkingBalance = 0;
14     private double savingBalance = 0;
15     Scanner input = new Scanner(System.in);
16     DecimalFormat moneyFormat = new DecimalFormat("$###,##0.00");
17
18     public int setCustomerNumber(int CustomerNumber) {
19         this.CustomerNumber = CustomerNumber;
20         return CustomerNumber;
21     }
22
23     public int getCustomerNumber() {
24         return CustomerNumber;
25     }
26
27     public int setPinNumber(int pinNumber) {
28         this.pinNumber = pinNumber;
29         return pinNumber;
30     }
31
32     public int getPinNumber() {
33         return pinNumber;
34     }
35
36     public double getCheckingBalance() {
37         return checkingBalance;
38     }
39 }
```

eclipse-workspace - ATM application By Using OOPS concept/src/AtmApplication/Account.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer x

- ATM application By Using OOPS con
 - JRE System Library [JavaSE-17]
 - src
 - AtmApplication
 - Account.java
 - ATM.java
 - OptionMenu.java
 - Gagan1
 - Gagan 2
 - Inheritance
 - Longituit

Account.java x ATM.java OptionMenu.java

```
40 public double getSavingsBalance() {
41     return savingBalance;
42 }
43 public double calcCheckingWithdraw(double amount) {
44     checkingBalance = (checkingBalance - amount);
45     return checkingBalance;
46 }
47 public double calcSavingsWithdraw(double amount) {
48     savingBalance = (savingBalance - amount);
49     return savingBalance;
50 }
51 public double calcCheckingDeposit(double amount) {
52     checkingBalance = (checkingBalance + amount);
53     return checkingBalance;
54 }
55 public double calcSavingsDeposit(double amount) {
56     savingBalance = (savingBalance + amount);
57     return savingBalance;
58 }
59 public void getCheckingWithdrawInput() {
60     System.out.println("Checking Account Balance: " + moneyFormat.format(checkingBalance));
61     System.out.println("Amount you want to withdraw from Checking Account: ");
62     double amount = input.nextDouble();
63
64     if ((checkingBalance - amount) >= 0) {
65         calcCheckingWithdraw(amount);
66         System.out.println("New Checking Account balance: " + moneyFormat.format(checkingBalance));
67     } else {
68         System.out.println("Balance cannot be negative." + "\n");
69     }
70 }
71
72
73 public void getSavingsWithdrawInput() {
74     System.out.println("Checking Account Balance: " + moneyFormat.format(savingBalance));
75     System.out.println("Amount you want to withdraw from saving Account: ");
76     double amount = input.nextDouble();
```

eclipse-workspace - ATM application By Using OOPS concept/src/AtmApplication/Account.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer x

- ATM application By Using OOPS con
 - JRE System Library [JavaSE-17]
 - src
 - AtmApplication
 - Account.java
 - ATM.java
 - OptionMenu.java
 - Gagan1
 - Gagan 2
 - Inheritance
 - Longituit

Account.java x ATM.java OptionMenu.java

```
68     System.out.println("Balance cannot be negative." + "\n");
69 }
70 }
71
72
73 public void getSavingsWithdrawInput() {
74     System.out.println("Checking Account Balance: " + moneyFormat.format(savingBalance));
75     System.out.println("Amount you want to withdraw from saving Account: ");
76     double amount = input.nextDouble();
77
78     if ((savingBalance - amount) >= 0) {
79         calcCheckingDeposit(amount);
80         System.out.println("New Saving Account balance: " + moneyFormat.format(savingBalance));
81     } else {
82         System.out.println("Balance cannot be negative." + "\n");
83     }
84 }
85 public void getCheckingDepositInput() {
86     System.out.println("Checking Account Balance: " + moneyFormat.format(checkingBalance));
87     System.out.println("Amount you want to Deposit from Checking Account: ");
88     double amount = input.nextDouble();
89
90     if ((checkingBalance + amount) >= 0) {
91         calcCheckingDeposit(amount);
92         System.out.println("New Checking Account balance: " + moneyFormat.format(checkingBalance));
93     } else {
94         System.out.println("Balance cannot be negative." + "\n");
95     }
96 }
97 public void getSavingsDepositInput() {
98     System.out.println("Checking Account Balance: " + moneyFormat.format(checkingBalance));
99     System.out.println("Amount you want to Deposit from Checking Account: ");
100     double amount = input.nextDouble();
101
102     if ((checkingBalance + amount) >= 0) {
103         calcCheckingDeposit(amount);
```

```

75      System.out.println("Checking Account balance: " + moneyFormat.format(savingBalance));
76      System.out.println("Amount you want to withdraw from saving Account: ");
77      double amount = input.nextDouble();
78      if ((savingBalance - amount) >= 0) {
79          calcCheckingDeposit(amount);
80          System.out.println("New Saving Account balance: " + moneyFormat.format(savingBalance));
81      } else {
82          System.out.println("Balance cannot be negative." + "\n");
83      }
84  }
85  public void getCheckingDepositInput() {
86      System.out.println("Checking Account Balance: " + moneyFormat.format(checkingBalance));
87      System.out.println("Amount you want to Deposit from Checking Account: ");
88      double amount = input.nextDouble();
89      if ((checkingBalance + amount) >= 0) {
90          calcCheckingDeposit(amount);
91          System.out.println("New Checking Account balance: " + moneyFormat.format(checkingBalance));
92      } else {
93          System.out.println("Balance cannot be negative." + "\n");
94      }
95  }
96  }
97  public void getSavingDepositInput() {
98      System.out.println("Checking Account Balance: " + moneyFormat.format(checkingBalance));
99      System.out.println("Amount you want to Deposit from Checking Account: ");
100     double amount = input.nextDouble();
101     if ((checkingBalance + amount) >= 0) {
102         calcCheckingDeposit(amount);
103         System.out.println("New Checking Account balance: " + moneyFormat.format(checkingBalance));
104     } else {
105         System.out.println("Balance cannot be negative." + "\n");
106     }
107 }
108 }
109 }
110 }

```

OUTPUT:

```

1 package AtmApplication;

2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

The screenshot shows the Eclipse IDE interface. The Package Explorer on the left displays the project structure: 'ATM application By Using OOPS concept' containing 'src' with 'AtmApplication', 'Account.java', 'ATM.java', and 'OptionMenu.java'. The main editor shows the code for 'package AtmApplication;'. The Console window on the right shows the application's execution: it starts with a new checking account balance of \$100,000.00, prompts for account selection (1 for Checking, 2 for Saving, 3 for Exit), then prompts for account type selection (1 for View Balance, 2 for Withdraw Funds, 3 for Deposit Funds, 4 for Exit). The user enters '1' for account selection and '1' for account type, resulting in a balance of \$100,000.00. Then, the user enters '2' for account type, resulting in a balance of \$50,000.00 after a withdrawal.

```
1 package AtmApplication;
```

```
ATM (1) [Java Application] C:\Users\HP\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.4.v20220903-1038\jre\bin\javaw.exe (18-Nov-2022, 8:27:28 pm) [pid: 3776]
100000
New Checking Account balance: $100,000.00
Select the Account you want to Access:
Type 1 - Checking Account
Type 2 - Saving Account
Type 3 - Exit
1
Checking Account:
Type 1 - View Balance
Type 2 - Withdraw Funds
Type 3 - Deposit Funds
Type 4 - Exit
Choice:
1
Checking Account Balance: $100,000.00
Select the Account you want to Access:
Type 1 - Checking Account
Type 2 - Saving Account
Type 3 - Exit
1
Checking Account:
Type 1 - View Balance
Type 2 - Withdraw Funds
Type 3 - Deposit Funds
Type 4 - Exit
Choice:
2
Checking Account Balance: $100,000.00
Amount you want to withdraw from Checking Account:
50000
New Checking Account balance: $50,000.00
Select the Account you want to Access:
Type 1 - Checking Account
```

This screenshot is identical to the one above, showing the same Eclipse IDE interface and console output for the ATM application.

```
1 package AtmApplication;
```

```
ATM (1) [Java Application] C:\Users\HP\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.4.v20220903-1038\jre\bin\javaw.exe (18-Nov-2022, 8:27:28 pm) [pid: 3776]
100000
New Checking Account balance: $100,000.00
Select the Account you want to Access:
Type 1 - Checking Account
Type 2 - Saving Account
Type 3 - Exit
1
Checking Account:
Type 1 - View Balance
Type 2 - Withdraw Funds
Type 3 - Deposit Funds
Type 4 - Exit
Choice:
1
Checking Account Balance: $100,000.00
Select the Account you want to Access:
Type 1 - Checking Account
Type 2 - Saving Account
Type 3 - Exit
1
Checking Account:
Type 1 - View Balance
Type 2 - Withdraw Funds
Type 3 - Deposit Funds
Type 4 - Exit
Choice:
2
Checking Account Balance: $100,000.00
Amount you want to withdraw from Checking Account:
50000
New Checking Account balance: $50,000.00
Select the Account you want to Access:
Type 1 - Checking Account
Type 2 - Saving Account
Type 3 - Exit
```

*** THANK YOU ***