

Banking App Program using JDBC Screenshots

1. Create Accounts & View all Accounts.

No Data in Accounts Table

The screenshot shows a database IDE with a 'training' schema. The 'account' table is selected in the 'Tables' folder. The SQL editor shows the following queries:

```
1 • use training;
2
3 • CREATE TABLE account
4 (
5   Id int NOT NULL AUTO_INCREMENT,
6   Name varchar(255),
7   Type varchar(255),
8   Balance double,
9   IsActive boolean ,
10   PRIMARY KEY (Id)
11 );
12
13
14 • select * from account;
```

The 'Result Grid' shows the following columns: Id, Name, Type, Balance, IsActive. The table is currently empty.

The screenshot shows a Java IDE with the 'BankingApp.java' file open. The code is as follows:

```
1 package com.indium.bankingapp;
2
3 import java.util.Collection;
4 import java.util.InputMismatchException;
5 import java.util.Iterator;
6 import java.util.Scanner;
7 import java.util.concurrent.Callable;
8 import java.util.concurrent.ExecutorService;
9 import java.util.concurrent.Executors;
10 import java.util.concurrent.Future;
11 import com.indium.bankingapp.model.Account;
12 import com.indium.bankingapp.service.AccountServiceImpl;
13
14 public class BankingApp {
15     private static Scanner in;
16     private static AccountServiceImpl accountService;
17
18     public static void main(String[] args) {
19         in = new Scanner(System.in);
20         accountService = new AccountServiceImpl();
21         ExecutorService executor = Executors.newCachedThreadPool();
22         System.out.print("Welcome to Banking Application!");
23         while (true) {
24             System.out.println("\n");
25             System.out.println("1. Add Accounts");
26             System.out.println("2. View All Accounts");
27             System.out.println("3. View Account");
28             System.out.println("4. Update Account");
29             System.out.println("5. Delete Account");
30             System.out.println("6. Print Statistics");
31             System.out.println("7. Import");
32             System.out.println("8. Export");
33             System.out.println("9. Exit");
34             System.out.print("Enter the option: ");
35
36             // Get option from user
37             int option = 0;
```

The console output shows the following messages:

```
BankingApp (8) [Java Application] E:\eclipse-jee-2022-09-R-win32-x86_64\eclipse\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64.jre\bin\java.exe
Welcome to Banking Application!
1. Add Accounts
2. View All Accounts
3. View Account
4. Update Account
5. Delete Account
6. Print Statistics
7. Import
8. Export
9. Exit
Enter the option: 1
Enter Account Name: Stella
Enter Account Type: Savings
Enter Account Balance: 100000
Is Account Active(Enter true or false): true
Connection created successfully. com.mysql.cj.jdbc.ConnectionImpl@4c163e3
Account has been added successfully!
1. Add Accounts
2. View All Accounts
3. View Account
4. Update Account
5. Delete Account
6. Print Statistics
7. Import
8. Export
9. Exit
Enter the option: 1
Enter Account Name: Steffy
Enter Account Type: Savings
Enter Account Balance: 300000
Is Account Active(Enter true or false): true
Account has been added successfully!
```

```
1 package com.indium.bankingapp;
2
3 import java.util.Collection;
4 import java.util.InputMismatchException;
5 import java.util.Iterator;
6 import java.util.Scanner;
7 import java.util.concurrent.Callable;
8 import java.util.concurrent.ExecutorService;
9 import java.util.concurrent.Executors;
10 import java.util.concurrent.Future;
11 import com.indium.bankingapp.model.Account;
12 import com.indium.bankingapp.service.AccountServiceImpl;
13
14 public class BankingApp {
15     private static Scanner in;
16     private static AccountServiceImpl accountService;
17
18     public static void main(String[] args) {
19         in = new Scanner(System.in);
20         accountService = new AccountServiceImpl();
21         ExecutorService executor = Executors.newCachedThreadPool();
22         System.out.print("Welcome to Banking Application!");
23         while (true) {
24             System.out.println("\n");
25             System.out.println("1. Add Accounts");
26             System.out.println("2. View All Accounts");
27             System.out.println("3. View Account");
28             System.out.println("4. Update Account");
29             System.out.println("5. Delete Account");
30             System.out.println("6. Print Statistics");
31             System.out.println("7. Import");
32             System.out.println("8. Export");
33             System.out.println("9. Exit");
34             System.out.print("Enter the option: ");
35
36             // Get option from user
37             int option = 0;
38             try {
39                 option = Integer.parseInt(in.next());
40             } catch (NumberFormatException e) {
41                 System.out.println("Invalid option. Please enter valid option.");
42                 continue;
43             }
44             try {
45                 switch (option) {
46                     case 1:
47                         // Add Accounts
48                         System.out.print("Enter Account Name: ");
49                         String name = in.next();
50                         System.out.print("Enter Account Type: ");
51                         String type = in.next();
52                         System.out.print("Enter Account Balance: ");
53                         double balance = in.nextDouble();
54                         boolean isActive = true;
55                         Account account = new Account(name, type, balance, isActive);
56                         accountService.addAccount(account);
57                         System.out.println("Account has been added successfully!");
58                     case 2:
59                         // View All Accounts
60                         Collection<Account> accounts = accountService.getAllAccounts();
61                         if (accounts.isEmpty()) {
62                             System.out.println("No accounts found.");
63                         } else {
64                             System.out.println("All Accounts:");
65                             for (Account account : accounts) {
66                                 System.out.println(account);
67                             }
68                         }
69                     case 3:
70                         // View Account
71                         System.out.print("Enter Account ID: ");
72                         int id = in.nextInt();
73                         Account account = accountService.getAccountById(id);
74                         if (account != null) {
75                             System.out.println(account);
76                         } else {
77                             System.out.println("Account not found.");
78                         }
79                     case 4:
80                         // Update Account
81                         System.out.print("Enter Account ID: ");
82                         int id = in.nextInt();
83                         Account account = accountService.getAccountById(id);
84                         if (account != null) {
85                             System.out.print("Enter new Account Name: ");
86                             String newName = in.next();
87                             System.out.print("Enter new Account Type: ");
88                             String newType = in.next();
89                             System.out.print("Enter new Account Balance: ");
90                             double newBalance = in.nextDouble();
91                             boolean newIsActive = true;
92                             Account newAccount = new Account(newName, newType, newBalance, newIsActive);
93                             accountService.updateAccount(id, newAccount);
94                             System.out.println("Account has been updated successfully!");
95                         } else {
96                             System.out.println("Account not found.");
97                         }
98                     case 5:
99                         // Delete Account
100                        System.out.print("Enter Account ID: ");
101                        int id = in.nextInt();
102                        Account account = accountService.getAccountById(id);
103                        if (account != null) {
104                            accountService.deleteAccount(id);
105                            System.out.println("Account has been deleted successfully!");
106                        } else {
107                            System.out.println("Account not found.");
108                        }
109                    case 6:
110                        // Print Statistics
111                        System.out.println("Print Statistics");
112                    case 7:
113                        // Import
114                        System.out.println("Import");
115                    case 8:
116                        // Export
117                        System.out.println("Export");
118                    case 9:
119                        // Exit
120                        System.out.println("Exit");
121                        break;
122                }
123            }
124        }
125    }
126}
```

Console

1. Add Accounts
2. View All Accounts
3. View Account
4. Update Account
5. Delete Account
6. Print Statistics
7. Import
8. Export
9. Exit
Enter the option: 1
Enter Account Name: Hadi
Enter Account Type: Current
Enter Account Balance: 400000
Is Account Active(Enter true or false): true
Account has been added successfully!

1. Add Accounts
2. View All Accounts
3. View Account
4. Update Account
5. Delete Account
6. Print Statistics
7. Import
8. Export
9. Exit
Enter the option: 1
Enter Account Name: Eliza
Enter Account Type: Current
Enter Account Balance: 600000
Is Account Active(Enter true or false): true
Account has been added successfully!

1. Add Accounts
2. View All Accounts
3. View Account
4. Update Account
5. Delete Account
6. Print Statistics
7. Import
8. Export
9. Exit
Enter the option: 2

ID	Name	Type	Balance	Is Active
1	Stella	Savings	100000.0	true
2	Steffy	Savings	300000.0	true
3	Hadi	Current	400000.0	true
4	Eliza	Current	600000.0	true

```
1 package com.indium.bankingapp;
2
3 import java.util.Collection;
4 import java.util.InputMismatchException;
5 import java.util.Iterator;
6 import java.util.Scanner;
7 import java.util.concurrent.Callable;
8 import java.util.concurrent.ExecutorService;
9 import java.util.concurrent.Executors;
10 import java.util.concurrent.Future;
11 import com.indium.bankingapp.model.Account;
12 import com.indium.bankingapp.service.AccountServiceImpl;
13
14 public class BankingApp {
15     private static Scanner in;
16     private static AccountServiceImpl accountService;
17
18     public static void main(String[] args) {
19         in = new Scanner(System.in);
20         accountService = new AccountServiceImpl();
21         ExecutorService executor = Executors.newCachedThreadPool();
22         System.out.print("Welcome to Banking Application!");
23         while (true) {
24             System.out.println("\n");
25             System.out.println("1. Add Accounts");
26             System.out.println("2. View All Accounts");
27             System.out.println("3. View Account");
28             System.out.println("4. Update Account");
29             System.out.println("5. Delete Account");
30             System.out.println("6. Print Statistics");
31             System.out.println("7. Import");
32             System.out.println("8. Export");
33             System.out.println("9. Exit");
34             System.out.print("Enter the option: ");
35
36             // Get option from user
37             int option = 0;
38             try {
39                 option = Integer.parseInt(in.next());
40             } catch (NumberFormatException e) {
41                 System.out.println("Invalid option. Please enter valid option.");
42                 continue;
43             }
44             try {
45                 switch (option) {
46                     case 1:
47                         // Add Accounts
48                         System.out.print("Enter Account Name: ");
49                         String name = in.next();
50                         System.out.print("Enter Account Type: ");
51                         String type = in.next();
52                         System.out.print("Enter Account Balance: ");
53                         double balance = in.nextDouble();
54                         boolean isActive = true;
55                         Account account = new Account(name, type, balance, isActive);
56                         accountService.addAccount(account);
57                         System.out.println("Account has been added successfully!");
58                     case 2:
59                         // View All Accounts
60                         Collection<Account> accounts = accountService.getAllAccounts();
61                         if (accounts.isEmpty()) {
62                             System.out.println("No accounts found.");
63                         } else {
64                             System.out.println("All Accounts:");
65                             for (Account account : accounts) {
66                                 System.out.println(account);
67                             }
68                         }
69                     case 3:
70                         // View Account
71                         System.out.print("Enter Account ID: ");
72                         int id = in.nextInt();
73                         Account account = accountService.getAccountById(id);
74                         if (account != null) {
75                             System.out.println(account);
76                         } else {
77                             System.out.println("Account not found.");
78                         }
79                     case 4:
80                         // Update Account
81                         System.out.print("Enter Account ID: ");
82                         int id = in.nextInt();
83                         Account account = accountService.getAccountById(id);
84                         if (account != null) {
85                             System.out.print("Enter new Account Name: ");
86                             String newName = in.next();
87                             System.out.print("Enter new Account Type: ");
88                             String newType = in.next();
89                             System.out.print("Enter new Account Balance: ");
90                             double newBalance = in.nextDouble();
91                             boolean newIsActive = true;
92                             Account newAccount = new Account(newName, newType, newBalance, newIsActive);
93                             accountService.updateAccount(id, newAccount);
94                             System.out.println("Account has been updated successfully!");
95                         } else {
96                             System.out.println("Account not found.");
97                         }
98                     case 5:
99                         // Delete Account
100                        System.out.print("Enter Account ID: ");
101                        int id = in.nextInt();
102                        Account account = accountService.getAccountById(id);
103                        if (account != null) {
104                            accountService.deleteAccount(id);
105                            System.out.println("Account has been deleted successfully!");
106                        } else {
107                            System.out.println("Account not found.");
108                        }
109                    case 6:
110                        // Print Statistics
111                        System.out.println("Print Statistics");
112                    case 7:
113                        // Import
114                        System.out.println("Import");
115                    case 8:
116                        // Export
117                        System.out.println("Export");
118                    case 9:
119                        // Exit
120                        System.out.println("Exit");
121                        break;
122                }
123            }
124        }
125    }
126}
```

Console

1. Add Accounts
2. View All Accounts
3. View Account
4. Update Account
5. Delete Account
6. Print Statistics
7. Import
8. Export
9. Exit
Enter the option: 2

ID	Name	Type	Balance	Is Active
1	Stella	Savings	100000.0	true
2	Steffy	Savings	300000.0	true
3	Hadi	Current	400000.0	true
4	Eliza	Current	600000.0	true

After Creation of accounts

Navigator

SCHEMAS

Filter objects

sys

training

Tables

account

employee

Columns

Indexes

Foreign Keys

Triggers

Views

Stored Procedures

Functions

Query 1

```
1 use training;
2
3 CREATE TABLE account
4 (
5     Id int NOT NULL AUTO_INCREMENT,
6     Name varchar(255),
7     Type varchar(255),
8     Balance double,
9     IsActive boolean,
10     PRIMARY KEY (Id)
11 );
12
13
14 select * from account;
```

Result Grid

	ID	Name	Type	Balance	IsActive
▶	1	Stella	Savings	100000	1
	2	Steffy	Savings	300000	1
	3	Hadi	Current	400000	1
	4	Eliza	Current	600000	1
•	NULL	NULL	NULL	NULL	NULL

account 7

Apply

Revert

2. View Account, Update Accounts & View all accounts.

The screenshot shows an IDE with a Java file named `BankingApp.java` and its console output. The code defines a `BankingApp` class with a `main` method that uses a `Scanner` to interact with the user. The console shows the application running, displaying a menu of options (1-9) and the user's interactions. The user has selected option 3 to view an account, then option 4 to update an account, and finally option 2 to view all accounts. The console output shows the state of the accounts after each operation.

```
1 package com.indium.bankingapp;
2
3 import java.util.Collection;
4 import java.util.InputMismatchException;
5 import java.util.Iterator;
6 import java.util.Scanner;
7 import java.util.concurrent.Callable;
8 import java.util.concurrent.ExecutorService;
9 import java.util.concurrent.Executors;
10 import java.util.concurrent.Future;
11 import com.indium.bankingapp.model.Account;
12 import com.indium.bankingapp.service.AccountServiceImpl;
13
14 public class BankingApp {
15     private static Scanner in;
16     private static AccountServiceImpl accountService;
17
18     public static void main(String[] args) {
19         in = new Scanner(System.in);
20         accountService = new AccountServiceImpl();
21         ExecutorService executor = Executors.newCachedThreadPool();
22         System.out.print("Welcome to Banking Application!");
23         while (true) {
24             System.out.println("\n");
25             System.out.println("1. Add Accounts");
26             System.out.println("2. View All Accounts");
27             System.out.println("3. View Account");
28             System.out.println("4. Update Account");
29             System.out.println("5. Delete Account");
30             System.out.println("6. Print Statistics");
31             System.out.println("7. Import");
32             System.out.println("8. Export");
33             System.out.println("9. Exit");
34             System.out.print("Enter the option: ");
35
36             // Get option from user
37             int option = 0;
38             try {
39                 option = Integer.parseInt(in.next());
40             } catch (NumberFormatException e) {
41                 System.out.println("Invalid option. Please enter valid");
42                 continue;
43             }
44             try {
45                 switch (option) {
46                     case 1:
47                         addAccount();
48
```

Console Output:

```
BankingApp (8) [Java Application] E:\eclipse-jee-2022-09-R-win32-x86_64\plugins\org.eclipse.justi.openjdk.k
3. View Account
4. Update Account
5. Delete Account
6. Print Statistics
7. Import
8. Export
9. Exit
Enter the option: 3
Enter the Account Id: 1
  Id      Name      Type      Balance      Is Active
  1      Stella      Savings    100000.0      true
1. Add Accounts
2. View All Accounts
3. View Account
4. Update Account
5. Delete Account
6. Print Statistics
7. Import
8. Export
9. Exit
Enter the option: 4
Enter the Account Id to be updated: 3
Enter Account Name: Juli
Enter Account Type: Savings
Enter Account Balance: 900000
Is Account Active(Enter true or false): true
Account has been updated successfully!
1. Add Accounts
2. View All Accounts
3. View Account
4. Update Account
5. Delete Account
6. Print Statistics
7. Import
8. Export
9. Exit
Enter the option: 2
  Id      Name      Type      Balance      Is Active
  1      Stella      Savings    100000.0      true
  2      Steffy      Savings    300000.0      true
  3      Juli        Savings    900000.0      true
  4      Eliza       Current    600000.0      true
```

Account table post update

The screenshot shows a database management tool interface. On the left, the 'SCHEMAS' pane shows a hierarchy of databases: `sys`, `training`, and `employee`. The `training` database is selected, and the `account` table is highlighted. The main pane shows the SQL query for creating the `account` table and selecting all records. The query is as follows:

```
1 use training;
2
3 CREATE TABLE account
4 (
5     Id int NOT NULL AUTO_INCREMENT,
6     Name varchar(255),
7     Type varchar(255),
8     Balance double,
9     IsActive boolean ,
10     PRIMARY KEY (Id)
11 );
12
13
14 select * from account;
```

The 'Result Grid' pane at the bottom shows the data returned by the query. It contains 5 rows of data, including the newly added account for 'Eliza'.

	Id	Name	Type	Balance	IsActive
▶	1	Stella	Savings	100000	1
	2	Steffy	Savings	300000	1
	3	Juli	Savings	900000	1
	4	Eliza	Current	600000	1
*	NULL	NULL	NULL	NULL	NULL

3. Delete Accounts

The screenshot shows the Eclipse IDE with two panels. The left panel displays the `BankingApp.java` file, and the right panel shows the console output.

BankingApp.java

```
1 package com.indium.bankingapp;
2
3 import java.util.Collection;
4 import java.util.InputMismatchException;
5 import java.util.Iterator;
6 import java.util.Scanner;
7 import java.util.concurrent.Callable;
8 import java.util.concurrent.ExecutorService;
9 import java.util.concurrent.Executors;
10 import java.util.concurrent.Future;
11 import com.indium.bankingapp.model.Account;
12 import com.indium.bankingapp.service.AccountServiceImpl;
13
14 public class BankingApp {
15     private static Scanner in;
16     private static AccountServiceImpl accountService;
17
18     public static void main(String[] args) {
19         in = new Scanner(System.in);
20         accountService = new AccountServiceImpl();
21         ExecutorService executor = Executors.newCachedThreadPool();
22         System.out.println("Welcome to Banking Application!");
23         while (true) {
24             System.out.println("\n");
25             System.out.println("1. Add Accounts");
26             System.out.println("2. View All Accounts");
27             System.out.println("3. View Account");
28             System.out.println("4. Update Account");
29             System.out.println("5. Delete Account");
30             System.out.println("6. Print Statistics");
31             System.out.println("7. Import");
32             System.out.println("8. Export");
33             System.out.println("9. Exit");
34             System.out.print("Enter the option: ");
35
36             // Get option from user
37             int option = 0;
38             try {
39                 option = Integer.parseInt(in.next());
40             } catch (NumberFormatException e) {
41                 System.out.println("Invalid option. Please enter valid option.");
42                 continue;
43             }
44             try {
45                 switch (option) {
46                     case 1:
47                         addAccount();
48                     case 2:
49                         viewAllAccounts();
50                     case 3:
51                         viewAccount();
52                     case 4:
53                         updateAccount();
54                     case 5:
55                         deleteAccount();
56                     case 6:
57                         printStatistics();
58                     case 7:
59                         importAccounts();
60                     case 8:
61                         exportAccounts();
62                     case 9:
63                         exit();
64                     default:
65                         System.out.println("Invalid option. Please enter valid option.");
66                         continue;
67                 }
68             } catch (Exception e) {
69                 System.out.println("Error: " + e.getMessage());
70             }
71         }
72     }
73
74     private void addAccount() {
75         // ...
76     }
77
78     private void viewAllAccounts() {
79         // ...
80     }
81
82     private void viewAccount() {
83         // ...
84     }
85
86     private void updateAccount() {
87         // ...
88     }
89
90     private void deleteAccount() {
91         // ...
92     }
93
94     private void printStatistics() {
95         // ...
96     }
97
98     private void importAccounts() {
99         // ...
100     }
101
102     private void exportAccounts() {
103         // ...
104     }
105
106     private void exit() {
107         // ...
108     }
109 }
```

Console Output

```
BankingApp (8) [Java Application] E:\eclipse-jee-2022-09-R-win32-x86_64\plugins\org.eclipse.justj.openjdk.h
Id      Name      Type      Balance      Is Active
1       Stella Savings  100000.0    true
2       Steffy Savings 300000.0    true
3       Juli Savings   900000.0    true
4       Eliza Current   600000.0    true

1. Add Accounts
2. View All Accounts
3. View Account
4. Update Account
5. Delete Account
6. Print Statistics
7. Import
8. Export
9. Exit
Enter the option: 5
Enter the Account Id to be deleted: 4
Account has been deleted successfully!

1. Add Accounts
2. View All Accounts
3. View Account
4. Update Account
5. Delete Account
6. Print Statistics
7. Import
8. Export
9. Exit
Enter the option: 2

Id      Name      Type      Balance      Is Active
1       Stella Savings  100000.0    true
2       Steffy Savings 300000.0    true
3       Juli Savings   900000.0    true

1. Add Accounts
2. View All Accounts
3. View Account
4. Update Account
5. Delete Account
6. Print Statistics
7. Import
8. Export
9. Exit
Enter the option: |
```

Accounts table Post Delete

The screenshot shows the SQL Developer interface. The left pane displays the database schema, and the right pane shows the SQL query and its results.

Schema

- sys
 - training
 - Tables
 - account
 - Columns
 - Indexes
 - Foreign Keys
 - Triggers
 - Views
 - Stored Procedures
 - Functions

Query 1

```
1 use trainings;
2
3 CREATE TABLE account
4 (
5     Id int NOT NULL AUTO_INCREMENT,
6     Name varchar(255),
7     Type varchar(255),
8     Balance double,
9     IsActive boolean,
10     PRIMARY KEY (Id)
11 );
12
13
14 select * from account;
```

Result Grid

Id	Name	Type	Balance	IsActive
1	Stella	Savings	100000	1
2	Steffy	Savings	300000	1
3	Juli	Savings	900000	1
4	Eliza	Current	600000	1