FINAL SUBMISSION REPORT ON PROJECT

ATM MANAGEMENT SYSTEM

SUBMITTED BY: -

NAME	REGISTRION NO:	ROLL NO:
AMRIT RAJ	12110951	46
MUKUL GOND	12109391	24
RAJVEER SINGH	12109319	25

ACKNOWLEDGEMENT

We would like to express our heartfelt appreciation and gratitude to all those who have contributed to the successful completion of our Java project. Without their support, dedication, and expertise, this project would not have been possible. We would like to acknowledge the following individuals and organizations for their invaluable contributions:

Our project supervisor, Mrs. Amarinder Kaur Mam for their guidance, mentorship, and continuous support throughout the project. Their insights, feedback, and encouragement were instrumental in shaping the project and ensuring its success.

The online Java community, for their vast resources, tutorials, and forums that provided us with valuable insights and solutions during the development process. Their support was invaluable in resolving technical issues and improving our project.

We would also like to extend our gratitude to anyone else who has supported us in any way, directly or indirectly, during this Java project. Thank you for your contribution and commitment to our project's success.

INTRODUCTION

The ATM (Automated Teller Machine) Management System is a Java-based software application that simulates the functionality of a real-world ATM. It provides various banking services such as account balance inquiry, cash withdrawal, cash deposit, and fund transfer to authorized bank account holders. The system is designed to provide a secure and convenient way for customers to manage their bank accounts remotely using an ATM machine.

The Java-based ATM Management System is developed using object-oriented programming concepts, making it modular, scalable, and easily maintainable. The system employs Java's core features such as classes, objects, inheritance, polymorphism, and encapsulation to model the different components of an ATM and its functionalities.

The system is equipped with robust security measures, including user authentication and authorization, to ensure that only authorized users can access and perform banking operations. It also maintains a transaction log to keep track of all the transactions that occur in the system, providing an audit trail for security and accountability purposes.

The ATM Management System project is a practical application of Java programming skills and can serve as a foundation for further enhancements, such as integrating with a real bank's backend system, adding additional features like bill payments, and extending the system to support multiple languages or currencies.

The ATM Management System is a Java-based software application that emulates the functionality of a real-world ATM, providing secure and convenient banking services to authorized users. It is developed using OOP concepts, incorporates robust security measures, and offers opportunities for future enhancements.

Key Features of the ATM Management System:

1. User Authentication: The system requires users to enter their valid credentials, such as a unique user ID and PIN, to authenticate themselves before accessing their accounts.
2. Users can perform various account management tasks, including checking their account balance, depositing cash into their accounts, and transferring funds between accounts.
3. Transaction Logging: The system keeps a record of all transactions, including withdrawal, deposit, and transfer activities, for auditing and tracking purposes.
4. Error Handling: The project incorporates robust error handling mechanisms to handle exceptions, input validation, and provide appropriate feedback to users in case of errors.
5. Security Measures: The ATM Management System includes security measures, such as encryption and secure communication protocols, to ensure the confidentiality and integrity of user data and transactions.
6. User-Friendly Interface: The system provides an intuitive and easy-to-use interface for customers to interact with, making it simple and convenient to perform transactions.

7. Modularity and Extensibility: The project is designed with a modular approach, making it easy to extend or modify functionalities in the future, such as adding new features or integrating with other systems.

Overall, the ATM Management System is a comprehensive Java project that simulates the functionalities of an ATM, providing users with a secure and efficient way to manage their bank accounts and perform transactions.

MODULES: -

1. ATMSystem Module: -
   File Name: “ATMSystem.java”
   This module contains the main function i.e., the execution of the program starts from here. This calls the ATMSystem module.

2. Login Module: -
   This helps the user to enter the login credentials such as ID and password by taking them to the login page. If the credentials are correct, then only the system takes the user to the next module.

3. checkBalance Module: -
   This module helps the user to check the availability of the current balance amount in their ATM account.

4. Withdraw module: -

   It helps the user to withdraw the money from the ATM by asking the amount they want to withdraw and if the withdrawing amount is greater than the balance amount it will print “Insufficient Funds”.

5. Deposit Module: -

   This module helps the user to deposit funds in their ATM account after this the money will be added to the balance funds.

ROLES AND RESPONSIBILITIES

MUKUL GOND	By using the concept of java, will be implementing the module 1 and module 3. Also, will be collecting information and writing the report.
RAJVEER SINGH	By using the concept of java, will be implementing the module 2 and module 4. Also, testing the functionality of this program and designing and writing the report.
AMRIT RAJ	By using the concept of java, will be implementing the module 1 and module 5. Also, collecting information and creating content for report making.

GANTT CHART

PROJECT PHASES	WEEK 1	WEEK 2	WEEK 3	WEEK 4	WEEK5
PLANNING					
DESIGN					
CODING					
TESTING					
FINAL TOUCH					

Week1-

Planning and Research on the topic allocated.

Week2-

Data collection and designing the project.

Week3-

Writing the code for our project.

Week4-

Testing is done to check the functionality of code.

Week5-

Report writing and submission is done.

SCREENSHOTS OF THE CODE

```
import java.util.Scanner;
public class ATMSystem {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.print("Enter your username: ");
String username = scanner.nextLine();
System.out.print("Enter your password: ");
String password = scanner.nextLine();
User user = new User(username, password);
ATM atm = new ATM();
if (atm.login(user)) {
System.out.println("Welcome " + user.getUsername());
boolean logout = false;
while (!logout) {
System.out.println("1. Check balance");
System.out.println("2. Withdraw");
System.out.println("3. Deposit");
System.out.println("4. Logout");
int choice = scanner.nextInt();
switch (choice) {
case 1:
System.out.println("Balance: " + atm.checkBalance());
break;
case 2:
System.out.print("Enter amount to withdraw: ");
int withdrawAmount = scanner.nextInt();
System.out.println("New balance: " + atm.withdraw(withdrawAmount));
break;
case 3:
System.out.print("Enter amount to deposit: ");
int depositAmount = scanner.nextInt();
System.out.println("New balance: " + atm.deposit(depositAmount));
break;
case 4:
logout = true;
System.out.println("Logout successful");
break;
default:
System.out.println("Invalid choice");
break;
```

```
}
}
} else {
System.out.println("Invalid username or password");
}
}
}
class ATM {
private int balance;
public ATM() {
balance = 10000;
}
public boolean login(User user) {
return user.getUsername().equals("admin") && user.getPassword().equals("1234");
}
public int checkBalance() {
return balance;
}
public int withdraw(int amount) {
if (balance >= amount) {
balance -= amount;
return balance;
} else {
System.out.println("Insufficient funds");
return balance;
}
}
public int deposit(int amount) {
balance += amount;
return balance;
}
}
class User {
private String username;
private String password;
public User(String username, String password) {
this.username = username;
this.password = password;
}
public String getUsername() {
return username;
```

```
}
public String getPassword() {
return password;
}
}
```

OUTPUT OF THE CODE: -

```
PS C:\Users\Rajveer\Documents\ATM managment> javac ATMSystem.java
PS C:\Users\Rajveer\Documents\ATM managment> java ATMSystem
Enter your username: admin
Enter your password: 1234
Welcome admin
1. Check balance
2. Withdraw
3. Deposit
4. Logout
1
Balance: 10000
1. Check balance
2. Withdraw
3. Deposit
4. Logout
2
Enter amount to withdraw: 2000
New balance: 8000
1. Check balance
2. Withdraw
3. Deposit
4. Logout
3
Enter amount to deposit: 7000
New balance: 15000
1. Check balance
2. Withdraw
3. Deposit
4. Logout
4
Logout successful
```

CONCLUSION

In conclusion, the ATM Management System is a robust Java project that emulates the operations of an automated teller machine (ATM) and provides users with a user-friendly interface to perform various banking transactions. The project incorporates essential features such as user authentication, account management, transaction logging, error handling, security measures, and a user-friendly interface. It follows object-oriented programming principles, utilizes Java as the primary programming language, and is designed with modularity and extensibility in mind.

The ATM Management System is a valuable software solution for banks and financial institutions, as it provides a secure and efficient way for customers to manage their accounts and perform transactions. It can be easily extended or modified to accommodate additional features or integrate with other systems as per specific requirements. The project also demonstrates good coding practices, including error handling, input validation, and proper logging for auditing and tracking purposes.

Overall, the ATM Management System is a well-designed and functional Java project that serves as a solid foundation for building a real-world ATM system. It showcases the power of Java programming in creating reliable, secure, and user-friendly software applications for the financial industry.