

INDUSTRIAL TRAINING REPORT

ON

JAVA PROGRAMMING



**MADAN MOHAN MALAVIYA UNIVERSITY OF
TECHNOLOGY, GORAKHPUR**

BATCH 2023 - 25

MCA 2nd YEAR

SUBMITTED TO:

MS. PRANJAL MAURYA

SUBMITTED BY:

AHMAD FARAZ ANSARI

(2023073006)

**DEPARTMENT OF
INFORMATION TECHNOLOGY
& COMPUTER APPLICATION**

DECLARATION

I, **Ahmad Faraz Ansari**, a student enrolled in the **Department of Information Technology and Computer Application (ITCA) at Madan Mohan Malaviya University of Technology, Gorakhpur (UP)** with roll number **2023073006**, hereby solemnly and sincerely declare that the comprehensive Industrial Training report on "JAVA PROGRAMMING" has been diligently prepared and submitted as a part of the academic requirements for completion of my training program.

This document encapsulates the culmination of my efforts, learnings, and practical experiences gained during the Industrial Training period. The duration of the training, spanning over one month, provided me with a profound opportunity to delve into the intricate realms of Java Programming within the domain of information technology.

I earnestly assert that the contents of this report are the result of my independent research, observations, and active participation in the industrial setting where I had the privilege of gaining invaluable insights into the practical applications of Java Programming.

I would like to express my sincere gratitude to **CODSOFT** for providing me with the opportunity to undertake this industrial training, enabling me to bridge the gap between theoretical knowledge and practical implementation.

Ahmad Faraz Ansari

(2023073006)

MCA 2023-25

Date: 27-12-2024

CERTIFICATE

C.ID: ac33b77



CERTIFICATE

OF COMPLETION
PROUDLY PRESENTED TO

Ahmad Faraz Ansari

has successfully completed 4 weeks of a virtual internship program in

Java Programming

with wonderful remarks at **CODSOFT** from 05/08/2024 to 05/09/2024.

We were truly amazed by his/her showcased skills and invaluable contributions to the tasks and projects throughout the internship.



A handwritten signature in black ink, appearing to read "Raza Ansari".

Founder



MSME
MICRO, SMALL & MEDIUM ENTERPRISES
सूक्ष्म, लघु एवं मध्यम उद्यम

contact@codsoft.in

www.codsoft.in

Date: 08/09/2024

ACKNOWLEDGEMENT

I express my heartfelt gratitude to CODSOFT for providing me with an invaluable professional training experience in Java Programming during my one-month internship.

I am immensely thankful to the entire CodSoft team for sharing their expertise and knowledge, which greatly enhanced my understanding and skills. Their constant support, advice, and motivation were instrumental in the successful completion of this project.

I am especially grateful to my subject matter expert at CodSoft for their insightful guidance and encouragement throughout this journey. Their suggestions and feedback were crucial in refining my project work.

I also extend my sincere thanks to the Computer Labs staff and the library staff for granting me access to the necessary resources, which played a significant role in completing my project.

Lastly, I convey my deep gratitude to my college teachers, whose unwavering support and encouragement inspired me to strive for excellence in preparing this report.

Ahmad Faraz Ansari

(2023073006)

INDEX

Serial No.	Topic	Page No.
A.	Declaration	I
B.	Certificate	II
C.	Acknowledgement	III
1.	Introduction	1
	1.1.Definition	1
	1.2.About Java Programming training	1
	1.3.Objective	1
2.	Technology Description	2
	2.1. Software Requirements	2
	2.2. Hardware Requirements	2
3	Project Description	3
	3.1. Project Name	3
	3.2. Objective	3
	3.3. Modules	3
4	Project Screenshot	4
5	Future Scope	6
6	Conclusion	7

1. INTRODUCTION

1.1 Definition:

- To design and implement a simulated Automated Teller Machine (ATM) interface that replicates essential banking operations such as balance inquiry, deposits, withdrawals, transaction history management, and PIN security, while providing a secure, user-friendly experience for the end user.
- The project focuses on building an ATM system that is simple, interactive, and capable of handling essential banking functionalities.
- The system ensures basic data security through PIN validation and provides a user interface for seamless navigation.
- It emphasizes the practical application of object-oriented programming principles such as encapsulation, modularity, and inheritance in Java.

1.2 About Java Programming training:

- The **Java Programming Internship** by CodSoft was a month-long program aimed at building a strong foundation in Java programming and its real-world applications. During this internship, I learned essential concepts of object-oriented programming, data structures, and algorithmic problem-solving through hands-on coding sessions and practical assignments. The training emphasized industry-relevant skills by providing exposure to real-world scenarios and challenges.
- As part of the internship, I successfully completed a project titled "**ATM Interface in Java**", which involved implementing functionalities like account balance inquiry, deposits, withdrawals, transaction history, and PIN security. This experience not only strengthened my understanding of Java fundamentals but also enhanced my ability to design modular and efficient code, preparing me for professional roles in software development.

1.3 Objective:

- To gain a strong foundation in core Java concepts, including object-oriented programming, data structures, and algorithms.
- To develop practical problem-solving skills through hands-on coding sessions and real-world project implementation.
- To design and implement modular, efficient, and scalable Java-based applications.
- To enhance understanding of industry-relevant practices by working on projects like the **ATM Interface** with essential banking functionalities.
- To build technical confidence and prepare for professional roles in software development by applying learned concepts effectively.

2. TECHNOLOGY DESCRIPTION

2.1 SOFTWARE REQUIREMENTS:

2.1.1 IntelliJ idea Ultimate

IntelliJ IDEA Ultimate is a powerful, feature-rich Integrated Development Environment (IDE) designed for professional developers, available for Windows, macOS, and Linux. It offers extensive support for various programming languages, including Java, Kotlin, Python, and JavaScript, along with advanced features like intelligent code completion, robust debugging, and seamless integration with build tools like Maven and Gradle. The IDE also supports frameworks such as Spring, Hibernate, and Angular, and provides tools for version control, database management, and code analysis. With its vast ecosystem of plugins and customizability, IntelliJ IDEA Ultimate streamlines the development process and enhances productivity for complex projects.

2.1.2 GitHub:

GitHub is a widely used platform for version control and collaborative software development, but it also serves as an excellent option for hosting websites. Leveraging GitHub for website hosting comes with several advantages, making it a popular choice for individuals and organizations alike.

GitHub is built on the Git version control system, allowing developers to track changes, collaborate seamlessly, and manage different versions of their code. This version control functionality ensures that you can easily roll back to previous versions of your website if needed.

2.2 HARDWARE REQUIREMENTS:

- 4 GB RAM
- 10 GB HDD or SSD Space
- i3 6TH Gen or AMD Ryzen 3000 series Processor or later

3. PROJECT DESCRIPTION

3.1 PROJECT NAME:

ATM Interface

3.2 OBJECTIVE:

To design and implement a functional ATM Interface in Java, simulating core banking operations while applying object-oriented principles for a user-friendly and secure system.

3.3 MODULES:

1. User Authentication Module:

- Handles user login through PIN validation.
- Allow users to securely access their account.

2. Balance Inquiry Module:

- Displays the current balance of the user's account.
- Deposit Module:
- Enables users to deposit money into their account.
- Updates the account balance accordingly.

3. Withdrawal Module:

- Allow users to withdraw money from their account.
- Checks for sufficient balance and processes the transaction.

4. Change PIN Module:

- Provides functionality for users to change their PIN securely.
- Ensures PIN meets specified validation criteria.

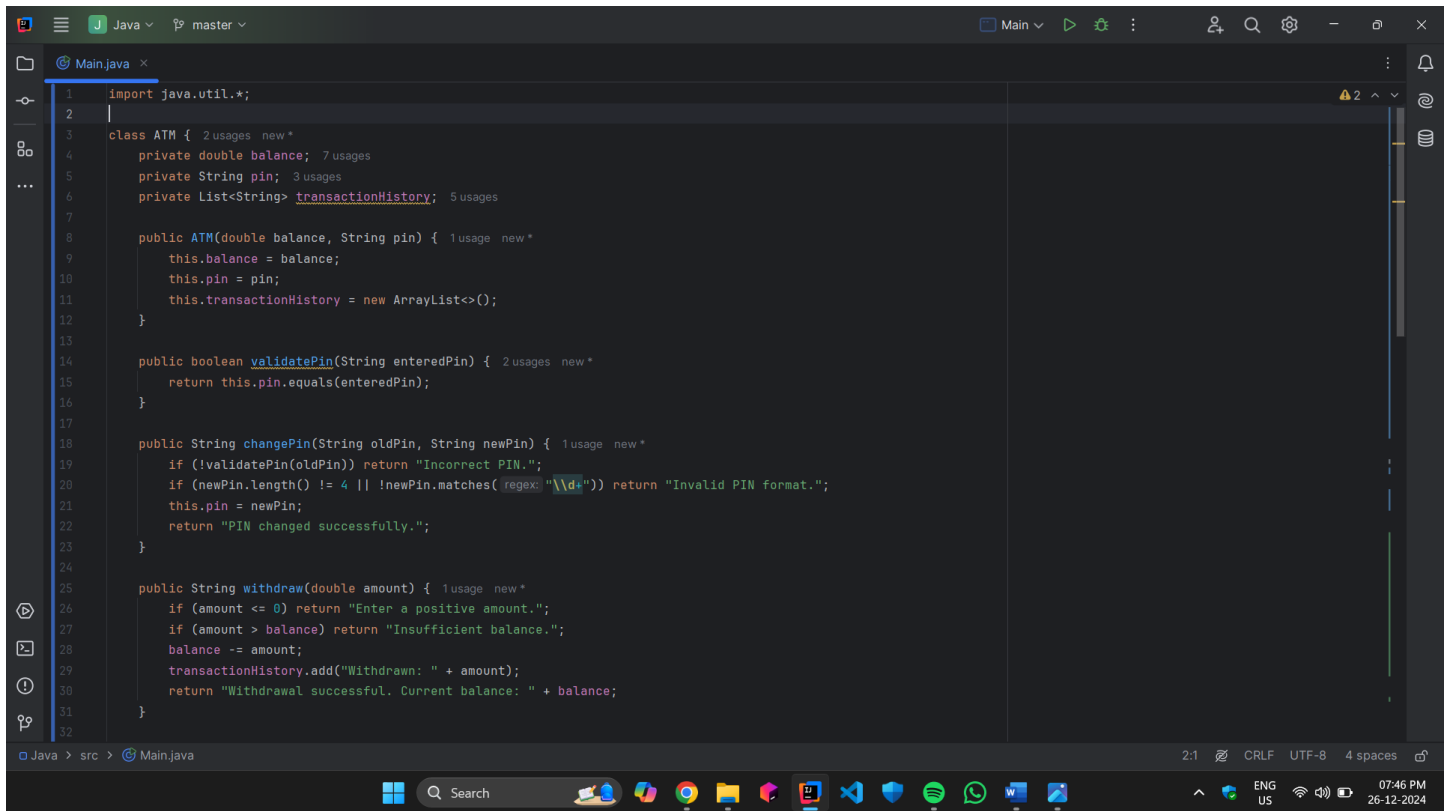
5. Transaction History Module:

- Tracks and displays the history of deposits and withdrawals.
- Keeps a record of user transactions for transparency.

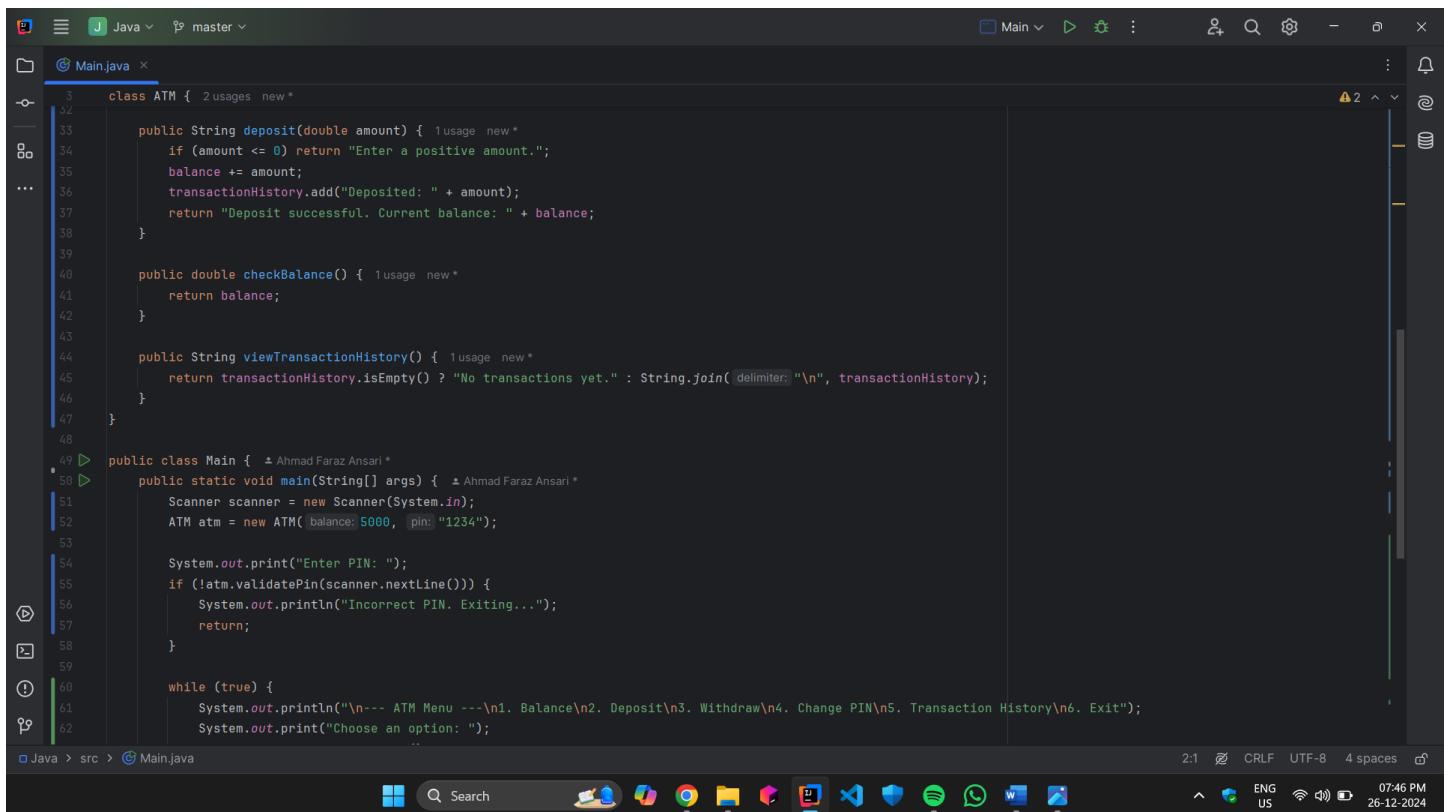
6. ATM Interface Module:

- The main interface that interacts with the user.
- Provides navigation through various options like balance check, deposit, withdrawal, etc.

4. PROJECT SCREENSHOT



```
1  import java.util.*;
2
3  class ATM { 2 usages new *
4      private double balance; 7 usages
5      private String pin; 3 usages
6      private List<String> transactionHistory; 5 usages
7
8      public ATM(double balance, String pin) { 1 usage new *
9          this.balance = balance;
10         this.pin = pin;
11         this.transactionHistory = new ArrayList<>();
12     }
13
14     public boolean validatePin(String enteredPin) { 2 usages new *
15         return this.pin.equals(enteredPin);
16     }
17
18     public String changePin(String oldPin, String newPin) { 1 usage new *
19         if (!validatePin(oldPin)) return "Incorrect PIN.";
20         if (newPin.length() != 4 || !newPin.matches(regex: "\\d+")) return "Invalid PIN format.";
21         this.pin = newPin;
22         return "PIN changed successfully.";
23     }
24
25     public String withdraw(double amount) { 1 usage new *
26         if (amount <= 0) return "Enter a positive amount.";
27         if (amount > balance) return "Insufficient balance.";
28         balance -= amount;
29         transactionHistory.add("Withdrawn: " + amount);
30         return "Withdrawal successful. Current balance: " + balance;
31     }
32 }
```



```
33     public String deposit(double amount) { 1 usage new *
34         if (amount <= 0) return "Enter a positive amount.";
35         balance += amount;
36         transactionHistory.add("Deposited: " + amount);
37         return "Deposit successful. Current balance: " + balance;
38     }
39
40     public double checkBalance() { 1 usage new *
41         return balance;
42     }
43
44     public String viewTransactionHistory() { 1 usage new *
45         return transactionHistory.isEmpty() ? "No transactions yet." : String.join(delimiter: "\n", transactionHistory);
46     }
47 }
48
49 public class Main { 1 Ahmad Faraz Ansari *
50     public static void main(String[] args) { 1 Ahmad Faraz Ansari *
51         Scanner scanner = new Scanner(System.in);
52         ATM atm = new ATM( balance: 5000, pin: "1234");
53
54         System.out.print("Enter PIN: ");
55         if (!atm.validatePin(scanner.nextLine())) {
56             System.out.println("Incorrect PIN. Exiting...");
57             return;
58         }
59
60         while (true) {
61             System.out.println("\n--- ATM Menu ---\n1. Balance\n2. Deposit\n3. Withdraw\n4. Change PIN\n5. Transaction History\n6. Exit");
62             System.out.print("Choose an option: ");
```

```
Java master Main Main.java
49 public class Main {
50     public static void main(String[] args) {
51
52         while (true) {
53             System.out.println("\n-- ATM Menu --\n1. Balance\n2. Deposit\n3. Withdraw\n4. Change PIN\n5. Transaction History\n6. Exit");
54             System.out.print("Choose an option: ");
55             int choice = scanner.nextInt();
56
57             switch (choice) {
58                 case 1: System.out.println("Balance: " + atm.checkBalance()); break;
59                 case 2: System.out.print("Deposit amount: "); System.out.println(atm.deposit(scanner.nextDouble())); break;
60                 case 3: System.out.print("Withdraw amount: "); System.out.println(atm.withdraw(scanner.nextDouble())); break;
61                 case 4:
62                     scanner.nextLine(); // Consume newline
63                     System.out.print("Old PIN: ");
64                     String oldPin = scanner.nextLine();
65                     System.out.print("New PIN: ");
66                     System.out.println(atm.changePin(oldPin, scanner.nextLine()));
67                     break;
68                 case 5: System.out.println(atm.viewTransactionHistory()); break;
69                 case 6: System.out.println("Goodbye!"); scanner.close(); return;
70                 default: System.out.println("Invalid choice.");
71             }
72         }
73     }
74 }
```

5. FUTURE SCOPE

The future scope of the ATM interface project includes enhancing security features by implementing multi-factor authentication, such as SMS verification or biometric recognition. It could also be integrated with a centralized bank server to enable real-time balance updates and transaction handling across multiple machines. User account management could be expanded with different roles like Admin and Customer, where Admins can manage accounts and monitor the system.

Further development could include creating a mobile application for users to perform transactions and manage their accounts remotely. Implementing AI-driven fraud detection would help identify suspicious activities by analyzing transaction patterns. The transaction history feature could be improved by offering advanced filters, downloadable reports, and more comprehensive statements. Lastly, adding services like bill payments, loan applications, and fund transfers would broaden the scope of the ATM system. These advancements would ensure the system remains secure, user-friendly, and aligned with future banking technologies.

Additionally, the ATM system can be enhanced with support for multiple languages to cater to a global audience, improving accessibility for users from diverse linguistic backgrounds. Integration with digital wallets and cryptocurrencies could expand the payment options, allowing users to access and manage their digital assets directly from the ATM interface. The system could also incorporate real-time customer support through chatbots or direct communication with bank representatives, providing immediate assistance for troubleshooting or inquiries. With continuous improvements, the ATM interface can evolve into a more versatile, user-centric solution that meets the growing demands of the digital banking era.

6. CONCLUSION

In conclusion, the ATM Interface project successfully demonstrates the core functionalities of an ATM system, such as balance inquiry, withdrawals, deposits, PIN management, and transaction history. By utilizing object-oriented programming principles in Java, the project ensures a modular and maintainable codebase. While the current system offers essential banking features, there is significant potential for future enhancements, including improved security, mobile integration, AI-driven fraud detection, and broader service offerings. As technology continues to evolve, the ATM system can be expanded to meet the demands of modern banking, providing users with a secure, efficient, and user-friendly experience.

Furthermore, the project offers valuable learning experience in real-world software development by combining theoretical concepts with practical application. The use of IntelliJ IDEA Ultimate as the development environment provided a powerful toolset for efficient coding, debugging, and project management. The experience gained in developing this ATM system has enhanced my understanding of Java programming, software architecture, and user interface design, and has prepared me for tackling more complex projects in the future.

This project also highlights the importance of user-centric design and security in developing financial applications. By focusing on features like secure PIN validation and transaction history tracking, the ATM interface aligns with industry standards for banking software. As the financial sector continues to innovate, systems like this one will play a key role in shaping the future of digital banking, ensuring both accessibility and security for users worldwide.