KLE Society's

KLE Technological University



**Structured Enquiry Report**

**On**

**ATM MANAGEMENT SYSTEM**

**Object Oriented Programming (20ECSC204)  
Object Oriented Programming Lab (20ECSP203)**

Submitted by

| **Name** | **Roll no** | **SRN** |
| --- | --- | --- |
| Chandni Kumari | 11 | 02FE22BCS026 |
| Prashant Uppar | 21 | 02FE22BCS069 |
| Wilfred Borges | 69 | 02FE22BCS177 |
| Yukta Sannaki | 71 | 02FE22BCS179 |
| **Team Number:**  A11 | | |

Faculty In-charge:

**Prof. Vaishali Y. Parab**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Academic year 2022-23

| **1.** | **Introduction** | |
| --- | --- | --- |
|  | 1.1 | Overview of the problem statement  Our task is to create a simple program in C++ that acts like a bank ATM. Users can do typical ATM activities like checking their balance, depositing and withdrawing money, and managing their accounts. We’ll need to set up classes to handle all these tasks, including user authentication, keeping track of transactions, and managing different types of accounts. The goal is to make it easy for users to interact with the ATM while ensuring their transactions are secure and accurate. |
|  | 1.2 | Features of Application  1. User Authentication:  The system requires users to enter a Personal Identification Number (PIN) to access their accounts. This authentication step ensures that only authorized individuals can perform transactions on the account. By verifying the PIN against stored account data, the system prevents unauthorized access and enhances security, protecting users' financial information from potential misuse or fraud.  2. Account Management:  Users can manage different types of accounts, specifically Savings and Checking accounts. Savings accounts are designed with a minimum balance requirement to encourage saving, while Checking accounts often have features suitable for frequent transactions, such as higher daily withdrawal limits. This distinction allows the system to cater to varied financial needs and preferences, providing flexibility and tailored banking experiences for users.  3. Balance Inquiry:  Users can check their current account balance to stay informed about their available funds. This feature is crucial for financial planning and budgeting, allowing users to make informed decisions about their spending and savings. The system retrieves and displays the balance information quickly, providing a convenient way for users to monitor their financial status.  4. Deposit and Withdrawal:  Users can deposit money into their accounts to increase their balance or withdraw money for their spending needs. Depositing is straightforward, allowing users to add funds to their account securely. Withdrawals are processed with checks in place to ensure the requested amount is within the available balance, preventing overdrafts and associated fees. This dual functionality supports the fundamental banking operations that users rely on for managing their money.  5. Transaction History:  The system records all transactions, including deposits and withdrawals, creating a comprehensive transaction history for each account. Users can view their transaction history to track their financial activities over time. This feature aids in personal financial management by providing a detailed log of all account movements, helping users identify spending patterns, monitor account activity, and verify the accuracy of transactions.  6. Error Handling:  The system includes mechanisms to handle common errors and exceptional situations gracefully. For example, it prompts users to re-enter their PIN if it is incorrect and prevents withdrawals that exceed the available balance, issuing appropriate error messages. These safeguards enhance the user experience by providing clear feedback and preventing actions that could lead to negative outcomes, such as insufficient funds or invalid transactions.  7. Security Measures:  Beyond PIN authentication, the system incorporates other security measures to protect user data and transactions. These measures might include encrypted storage of sensitive information, secure communication protocols, and session management to prevent unauthorized access. By prioritizing security, the system ensures that users' financial information remains confidential and protected against potential threats.  8. Simple User Interface:  The system is designed with a straightforward user interface, making it accessible even to users without technical expertise. Clear menus and prompts guide users through various banking operations, minimizing confusion and errors. The simplicity of the interface ensures that users can quickly and easily perform transactions, enhancing overall user satisfaction and efficiency.  9. Modularity and Reusability:  The codebase is structured with modularity and reusability in mind, meaning that different components of the system are designed as independent, interchangeable modules. This design approach facilitates easy maintenance and updates, allowing developers to enhance or expand the system without disrupting existing functionality. Modularity also promotes code reuse, making it more efficient to implement similar features in other projects or contexts. |
| **2.** | **Design**  **2.1** Class Diagrams | |
|  |  |  |
|  |  |  |
| **4** | **Implementation** | |
|  | 4.1 | Results |

