**EXPERIMENT-1**

**ATM System**

**1. Introduction**

**1.1 Purpose**

The purpose of this document is to define the needs for a simple Automated Teller Machine (ATM) system, which allows users to understand basic banking transactions. This project aims to enhance programming skills by implementing functionality of C or C++.

**1.2 Scope**

The ATM system operates in a console-based environment. Users will interact through a menu to perform basic transactions, such as:

1. Entering a valid PIN to access the bank account.
2. Viewing their account balance.
3. Depositing money into their account.
4. Withdrawing money from their account.
5. Exiting the system.

The system is designed to focus on the implementation of core ATM functionalities.

**1.3 Definitions and Abbreviations**

* **ATM**: Automated Teller Machine, a system that allows users to perform banking transactions.
* **PIN**: Personal Identification Number, used for user authentication in ATM systems.
* **Console-based**: An application that runs in a terminal or command-line interface.

**2. Specific Requirements**

**2.1 Functional Requirements**

1. **Menu Display**:
   * The system shall display a menu with the following options:

1. Enter a valid pin

2. Check Balance

3. Deposit Money

4. Withdraw Money

5. Exit

1. **Balance Inquiry**:
   * The system shall display the user’s current balance.
2. **Cash Deposit**:
   * The system shall prompt the user to enter the deposit amount.
   * The amount entered shall be added to the balance.
3. **Cash Withdrawal**:
   * The system shall prompt the user to enter the withdrawal amount.
   * If the entered amount exceeds the balance, the system shall display an error message.
   * If the balance is sufficient, the entered amount shall be deducted from the balance.
4. **Exit**:

The system shall end the program when the user selects the "Exit" option.

**2.2 Non-Functional Requirements**

1. **Performance**:
   * The system shall process the transactions and inputs of the user within 1 second.
2. **Usability**:
   * The system shall display clear instructions and error messages for invalid inputs.
3. **Reliability**:
   * The system shall handle invalid inputs by re-prompting the user without crashing.
4. **Portability**:
   * The program shall run on any C or C++ compiler including GCC and Turbo C++.

**3. Design Constraints**

* The system will use only basic in-memory data handling.
* If a database is integrated in the future, the required setup will include simple SQL commands.