Implementation:

* There will be 5 processes/nodes and each node has to get sync with each other process. When one process gets updated all other processes should get the updated information. For this we will implement sockets and assign port numbers for each process. Each process should get connected to all other process and get access to the shared memory. To avoid race condition, we will use mutexes and semaphores. The shared memory will be a database containing information of Current balance, Deposits and Withdraw information. For creating a database we will use SQL query creating a table containing the information of Current balance, Deposits and Withdraw information.
* When one process withdraws money all other process gets updated information through sockets by doing continuous polling.
* We will start with ATM1 we will deposit the money and increase the current balance. We will propose the new balance of one of the process and if everybody agrees the two phase commit is done and all other processes gets updated with new deposit. Each process will have one socket and all the communication will be done through sockets. Same way for withdrawal and balance enquiry
* Error Handling:
  + When one process done a transaction and gets crashed, all other process will be updated and the coordinator is shifted to other process and the two phase commit is implemented the same way
  + When the process gets crashed before updating then all other process should not get updated and restart the entire implementation with other process as a coordinator