In this problem I will use multi-processing techniques. First the teller file will be built and will receive a parameter to define its role (1,2,3,4). Next a customer file will be built. The customer will select its option and will get a role number from the shared memory variables counters (payment_ticket_counter and charging_ ticket _counter). The generated number for their tickets will be odd for payment purpose and even for charging purpose. We will use message queue to organize the order of severing customers. The customer will send its process id to the teller in message queue type equal to its ticket number.

The payments tellers see a shared memory variable counter which carry the ticket number of customer who should be served for payment purpose. the charging teller will see a shared memory variable counter which carry the ticket number of customer which should be served to make payment and the two counters will be used for the receiving message queue for a specific type.

The payment teller will receive message queue with type of odd numbers and the charging teller will receive message queue with even type of even numbers.

If teller was not busy and received the message, he will call for the specific customer using signal.

The teller and customer will communicate using message queue at type equal to the ticket number.