RDT 2.2 Simulation

- Niranjan Gopal
(IMT2022543)

Working

The code simulates a reliable data transfer (RDT) protocol using a sender and receiver communicating over a simulated channel. The sender sends data through the channel, and the receiver receives the data and sends acknowledgements (ACKs) back to the sender. The sender retransmits packets if it does not receive the expected ACK.

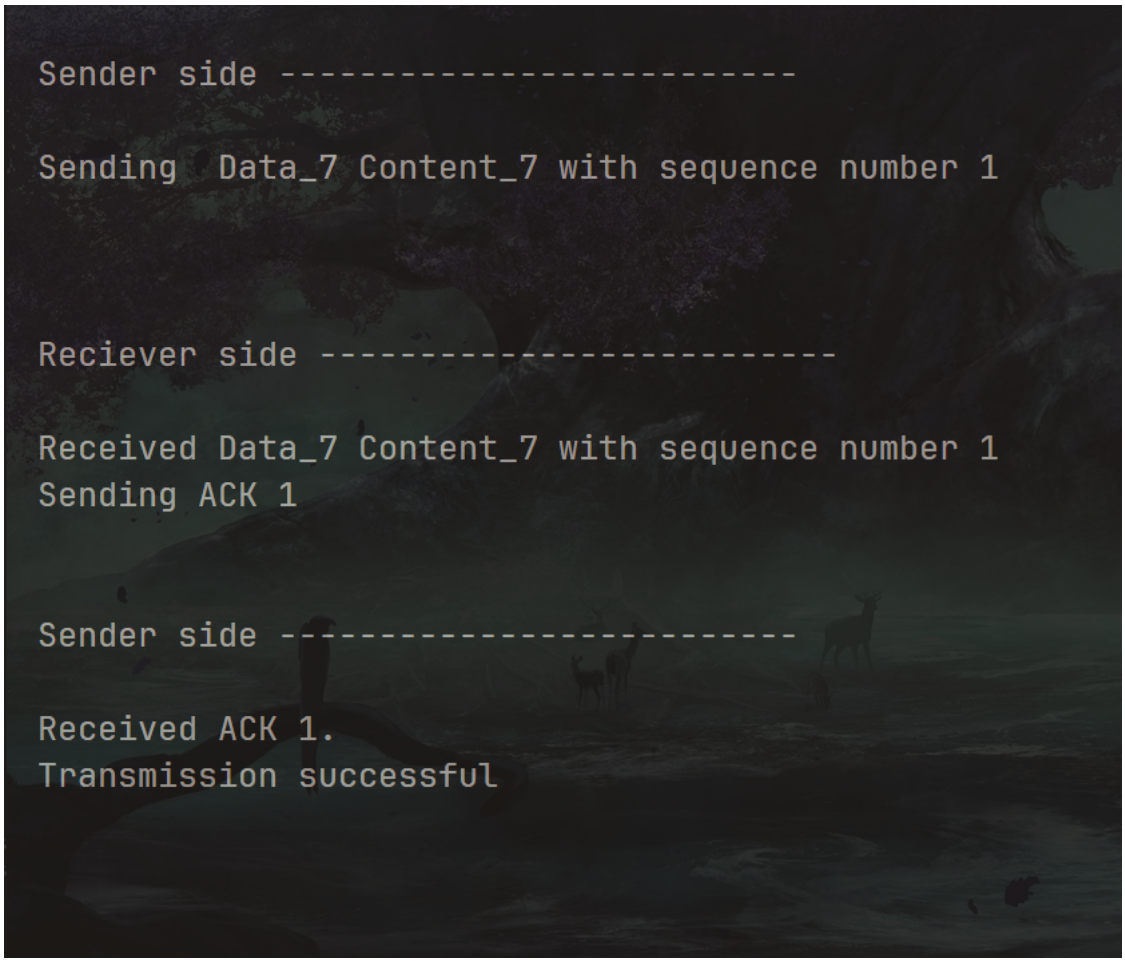
The SimulatedChannel class represents a simulated channel used by both the sender and receiver. It has methods for sending and receiving packets. The channel introduces packet loss based on a specified loss rate.

The RDTSender class represents the sender. It has a rdt_send method that sends data through the channel. The sender constructs packets with a sequence number, sends them through the channel, and waits for ACKs. If the expected ACK is not received or a different ACK is received, the sender retransmits the packet until the correct ACK is received.

The RDTReceiver class represents the receiver. It has a rdt_receive method that receives packets through the channel. The receiver checks the received sequence number against the expected sequence number. If they match, the receiver considers the data successfully received and sends the expected ACK. Otherwise, the data is discarded, and the receiver sends an ACK for the previous sequence number.

Output

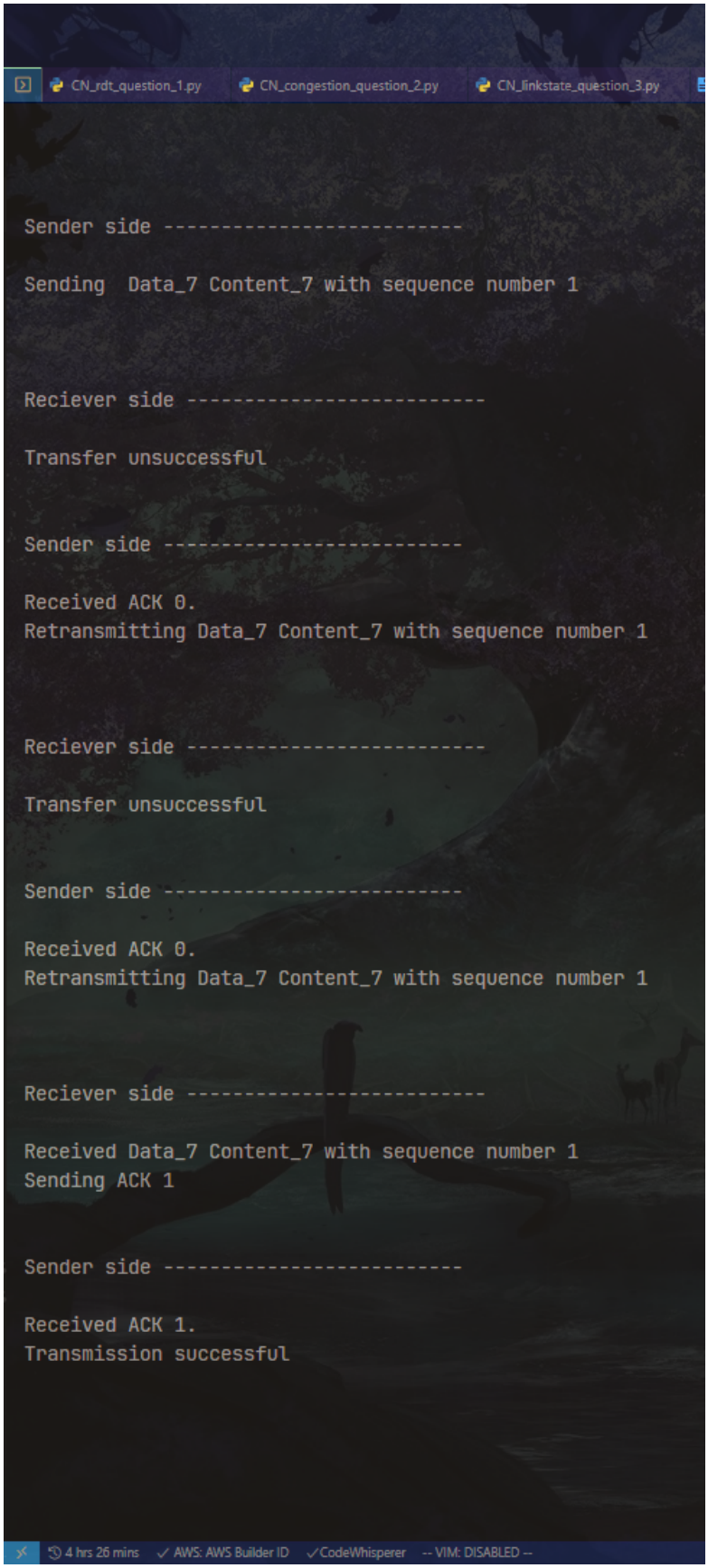
Below is the favourable case when the seen loss rate < predicted loss rate and therefore no packet loss happens and retransmission is not needed.



NOTE :

RDT 2.2 in itself doesn’t account for loss of ACK etc therefore the implementation also doesn't account for the situation where the ACK losses are happens.

And in the event of a packet loss the implementation is so that the receiver sends ACK of the previously received sequence number. This is demonstrated in the output below :-


CN_rdt_question_1.py
CN_congestion_question_2.py
CN_linkstate_question_3.py
Sender side --------------------------
Sending Data_7 Content_7 with sequence number 1
Reciever side --------------------------
Transfer unsuccessful
Sender side --------------------------
Received ACK 0.
Retransmitting Data_7 Content_7 with sequence number 1
Reciever side --------------------------
Transfer unsuccessful
Sender side --------------------------
Received ACK 0.
Retransmitting Data_7 Content_7 with sequence number 1
Reciever side --------------------------
Received Data_7 Content_7 with sequence number 1
Sending ACK 1
Sender side --------------------------
Received ACK 1.
Transmission successful
4 hrs 26 mins
AWS: AWS Builder ID
CodeWhisperer
-- VIM: DISABLED --