Project Title:

Reliable Data Transmission using Go-Back-N Protocol over UDP

---

1. Objective

This project implements the **Go-Back-N (GBN)** sliding window protocol using **UDP sockets** in Python to simulate reliable data transfer. Since UDP does not guarantee delivery, order, or integrity, GBN ensures:

- Reliable and ordered transmission
- Retransmission on loss/corruption
- ACK/NACK handling
- Sliding window optimization

---

2. Features Implemented

- **Custom packet structure** with:
    - Sequence number
    - Corruption flag
    - ACK flag
    - CRC32 checksum for error detection
- **Sender-side sliding window** with configurable size
- **Three simulated scenarios**:

1. Normal transmission (no loss or corruption)
2. **Packet loss** (by skipping send)
3. **Packet corruption** (modifying data before send)

- **Timeout retransmission** (resends all unACKed packets)
- **Fast retransmit** (triggered by 3 duplicate ACKs; resends only the suspected packet)

---

3. Tools & Technologies

- **Python 3**
- **UDP Sockets** (socket module)
- **CRC32 Checksum** (zlib.crc32)
- Modular design using separate Header.py for packet utilities

---

4. System Design

🔁 Sender (Client)

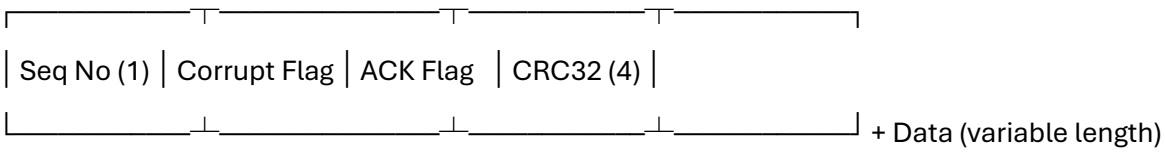
- Generates packets with headers + CRC
- Maintains a window of unacknowledged packets
- Simulates loss/corruption using sets
- Listens for ACKs, handles duplicates, and retransmits if needed

📥 Receiver (Server)

- Validates packet with CRC
- Accepts only in-order packets
- Sends cumulative ACK for last correctly received packet
- Ignores out-of-order/corrupted packets

---

5. Packet Format



---

6. Results

Scenario	Outcome
No packet loss/corruption	All packets ACKed in order
Simulated packet loss	Timeout triggers retransmission
Simulated packet corruption	CRC fails → packet ignored → retransmit
Corrupted/lost ACKs	Handled by duplicate ACKs or timeout
3 duplicate ACKs	Fast retransmit successfully triggered

---

7. Conclusion

This project demonstrates the real-world application of the Go-Back-N protocol for **reliable transmission over unreliable networks** like UDP. Key takeaways include understanding **retransmission strategies, sliding windows, timeout handling**, and the importance of **error detection** with CRC.