Code Explanation & Developer Handbook

1. Repository Structure

```
network-topology-simulator/
├── src/
│   ├── __init__.py
│   ├── config_parser.py
│   ├── topology_builder.py
│   ├── validator.py
│   ├── optimizer.py
│   ├── simulator.py
│   ├── ipc.py
│   └── main.py
├── tests/
│   ├── test_parser.py
│   ├── test_validator.py
│   ├── test_simulator.py
│   └── sample_configs/
│       ├── R1.dump
│       ├── R2.dump
│       ├── SW1.dump
│       └── SW2.dump
├── outputs/
└── requirements.txt
```

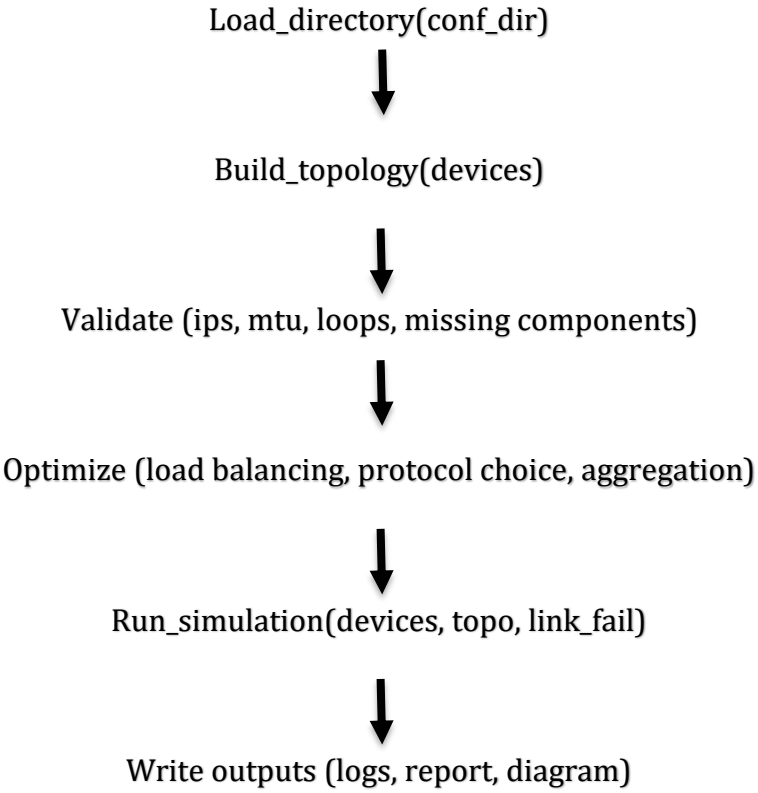
2. Data Model (Dataclasses)

- **Core entities (in config_parser.py):**
  - Interface: name, ip, vlan, mtu, bw_mbps, neighbor
  - Endpoint: name, app_type, peak_mbps, avg_mbps
  - Device: hostname, role, interfaces[], endpoints[], gateway, ospf_enabled

3. Module-by-Module Walkthrough

- **config_parser.py:** parses configs into Device objects
- **topology_builder.py:** constructs Topology with Links
- **validator.py:** detects duplicate IPs, MTU mismatches, loops, missing components
- **optimizer.py:** recommends load balancing, protocol choice, node aggregation
- **ipc.py:** simple FIFO-style in-memory IPC
- **simulator.py:** multithreaded simulation, ARP/OSPF discovery, link failure injection
- **main.py:** orchestrates the workflow and writes output

4. Execution Flow



5. CLI & Configuration

- **Run with sample configs:**
  - python -m src.main --conf tests/sample_configs --out outputs
- **Inject a link failure:**
  - python -m src.main --link-fail R1-R2
- **Config grammar example:**
  - interface Gi0/1 ip 10.0.1.1/24 vlan 20 mtu 1500 bw_mbps 50 neighbor SW1
  - endpoint app type web peak_mbps 60 avg_mbps 20

6. Algorithms & Complexity Notes

- **Topology construction:** O(E)
- **Duplicate IP detection:** O(N) per VLAN
- **Loop detection:** O(V+E)
- **MTU mismatch scan:** O(E)
- **Load recommendations:** O(E)

7. Extensibility Guide

- Extend parser for Cisco/JunOS syntax
- Replace IPC with TCP/FIFO
- Implement packet-level protocol simulation
- Add traffic engineering models
- Use NetworkX/Graphviz for visualization

8. Testing Strategy

- Unit tests with pytest for parser, validator, simulator
- Golden configs in tests/sample_configs validate outputs
- Potential for fuzz testing the parser

9. Common Pitfalls & Debugging Tips

- Missing neighbor configs → 'Missing device configuration' warnings
- Empty topology.png → install matplotlib
- No MTU mismatch detected → configs must differ in MTU
- Thread issues → keep simulation bounded, add timeouts