Got it 👍
Below is a **clean, ready-to-use slide content** for **ALL sections except 7**, written in **simple academic language**, aligned with your **NetClassroom TCP project**, and suitable for a **5-minute presentation**.

You can directly copy–paste these into PPT slides.

---

# 1. PROBLEM DOMAIN & MOTIVATIONS

## Problem Domain

- Online classrooms require **real-time communication**, **assignment management**, and **secure interaction** between teachers and students.
- Many existing systems rely on heavy web frameworks and external services.
- Understanding **low-level networking concepts** is often hidden in high-level tools.

## Motivation

- To design and implement a **custom classroom system** using **pure TCP socket programming**.
- To **apply networking theory concepts practically** instead of relying on pre-built platforms.
- To build a system that demonstrates **client–server communication**, **data transfer**, and **session handling** clearly.

---

# 2. OBJECTIVES / AIMS

- To design a **TCP-based classroom management system**.
- To support **real-time chat** between participants.
- To enable **teacher–student interactions** including:

    - Announcements
    - Private messaging
    - Assignment creation
    - Group assignments
- To demonstrate **reliable data transfer** and **connection handling**.
- To build both **terminal-based** and **JavaFX GUI-based clients**.

---

# 3. PROJECT FEATURES

## Core Features

- User login with **role-based access** (Teacher / Student)
- Public classroom chat
- Private messaging:
    - Student ↔ Student
    - Teacher ↔ Student
- Assignment management:
    - SOLO assignments
    - GROUP assignments
- Group creation for group assignments
- Assignment submission (file upload)
- Submission status tracking for teachers

## Interface

- Terminal-based client (initial version)
- JavaFX GUI client (enhanced usability)

---

# 4. BLOCK DIAGRAM / WORK FLOW DIAGRAM

## High-Level Workflow

1. Client connects to server using **TCP socket**
2. User joins classroom with role (student/teacher)
3. Server maintains active sessions
4. Client sends requests using **custom HTTP-like protocol**
5. Server processes request and responds accordingly
6. Messages and files are transferred reliably over TCP
7. Server broadcasts or routes messages based on type

## Components

- Client Application (Java / JavaFX)
- TCP Server
- Session Manager
- Chat Room Manager
- Assignment & Group Managers
- File Storage System

*(You can draw this as a simple block diagram in PPT)*

---

# 5. TOOLS & TECHNOLOGIES

### Programming Language

- Java

### Networking

- TCP Sockets (`java.net.Socket`, `ServerSocket`)

### UI

- JavaFX (for GUI client)

### Development Tools

- IntelliJ IDEA / VS Code
- Command Line (javac, java)

### Others

- Custom HTTP-style request/response parsing
- File I/O for assignment submissions

---

# 6. APPLIED NETWORKING CONCEPTS

### 1. Socket Programming

- Used **TCP sockets** for client–server communication.
- Each client maintains a persistent socket connection to the server.

### 2. Client–Server Architecture

- Centralized server handles:
    - Multiple clients
    - Request routing
    - Message broadcasting
- Clients act as request initiators.

## 3. Reliable Data Transfer

- TCP ensures:
    - Ordered delivery
    - No data loss
    - Error-free transmission
- Used for:
    - Chat messages
    - Private messages
    - File uploads

## 4. Flow Control

- TCP flow control automatically manages data rate.
- Server processes requests sequentially per client.
- Prevents buffer overflow during large file transfers.

## 5. Congestion Control

- TCP congestion control adjusts sending rate based on network conditions.

- Large file uploads rely on TCP's congestion window mechanism.

### 6. Session Management

- Server tracks active users using session objects.
- Ensures only authenticated users can send messages or submit assignments.

### 7. Multiplexing

- Server handles multiple clients using **multi-threading**.
- Each client connection runs on a separate handler thread.

---

# 8. LIMITATIONS AND FUTURE PLAN

## Limitations

- No encryption (data is sent in plain text)
- Basic UI design (not web-based)
- No persistent database (data stored in memory/files)
- Limited scalability for very large classrooms

## Future Enhancements

- Add **encryption (TLS/SSL)** for secure communication
- Implement **database storage** for users and submissions
- Improve UI with better styling and notifications
- Add attendance tracking
- Implement reconnection and fault tolerance

- Add role-based analytics for teachers