# Session 12 HTTP Request/Response in Wireshark

## 1. What is HTTP?

HyperText Transfer Protocol (HTTP) enables communication between client and server.
Operates on TCP port 80 (unencrypted).
Uses a request–response model.

2. HTTP Request (Client → Server)

Request Line → e.g., `GET /index.html HTTP/1.1`
Headers → Host, User-Agent, Accept-Language, etc.
Body → Present in methods like POST (form data, JSON).
Common Methods:

- GET → Retrieve data
- POST → Send data
- PUT → Update resource
- DELETE → Remove resource
- HEAD → Request headers only

3. HTTP Response (Server → Client)

Status Line → e.g., `HTTP/1.1 200 OK`
Headers → Content-Type, Content-Length, Server, etc.
Body → Actual content (HTML, JSON, image, etc.)
Common Status Codes:

- 200 OK → Success
- 301/302 → Redirect
- 400 → Bad Request
- 401 Unauthorized / 403 Forbidden → Access denied
- 404 Not Found → Resource not found
- 500 Internal Server Error → Server problem

4. Analyzing HTTP in Wireshark

Apply filter: `http` or `tcp.port == 80`.
Use Follow → HTTP Stream to view full conversation.

- Observe:
- Requests (GET/POST lines, headers, body).
- Responses (status code, headers, content).
- Timing between request and response.

5. Practical Uses

Troubleshooting: Identify why sites don’t load or fail.
Performance: Detects delays and slow server responses.
Debugging: Verify data in web requests/responses.
Security/SOC: Detects leaks, suspicious requests, or unencrypted credentials.
Forensics: Reconstruct user activity, downloads, or attacks.

6. Security Note

HTTP traffic is plaintext and easily captured.
Modern websites use HTTPS (port 443) for encryption.

Summary:
HTTP in Wireshark lets us inspect how web clients and servers communicate. By analyzing requests and responses, we can troubleshoot, debug, and investigate security events effectively.