PROTOCOL RACER

HTTP/2 and HTTP/3 Performance Tutorial

Sejal Khedekar - skhedek2
Ritu Malav - rmalav
Rachana Angara - rangara1
Sahil Rastogi - Srasto12

Learning Outcomes

At the end of this tutorial, the student will be able to…

- Explain why HTTP/1.1 is limited for modern web apps.
- Describe core features of HTTP/2 and HTTP/3 (QUIC).
- Compare HTTP/2 vs HTTP/3 on latency, throughput, and loss.
- Deploy a simple Spring Boot + Vite/React app using REST APIs.
- Run and interpret protocol experiments in Protocol Racer (metrics, waterfall).
- Decide and justify when to use HTTP/3 instead of HTTP/2.

01

History, Problem, Benefits, Challenges

Problem & Motivation

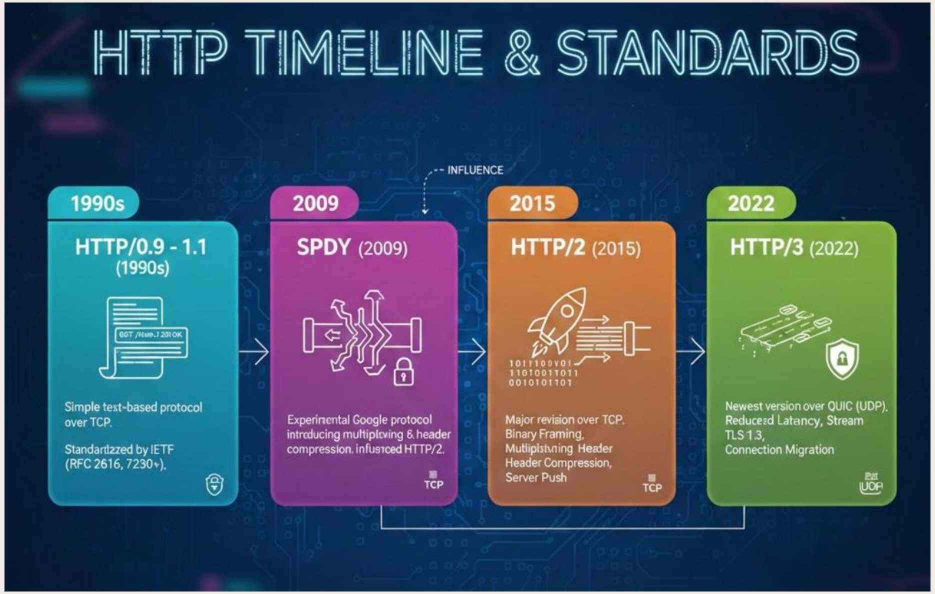
What Problem Are We Solving?

- Classic web stack evolved around HTTP/1.1:
  - One request per TCP connection (or limited pipelining).
  - Head-of-line blocking when one slow response stalls others.
  - Extra round-trips for TLS and connection setup.
- Modern apps (SPAs, media-heavy sites) need:
  - Dozens hundreds of resources per page.
  - Low latency on mobile & lossy networks.
- Our tutorial demonstrates how HTTP/2 and HTTP/3 address these issues in practice.

History & Standards

HTTP Timeline & Standards

- HTTP/0.9: 1.1 (1990s)
    - Simple text-based protocol over TCP.
    - Standardized by IETF (e.g., HTTP/1.1 → RFC 2616, later RFC 7230+).
- SPDY (2009)
    - Experimental Google protocol introducing multiplexing & header compression.
    - Influenced the HTTP/2 design.


HTTP TIMELINE & STANDARDS
INFLUENCE
1990s
HTTP/0.9 - 1.1 (1990s)
Simple text-based protocol over TCP.
Standardized by IETF (RFC 2616, 7230+).
2009
SPDY (2009)
Experimental Google protocol introducing multiplexing & header compression. Infusrced HTTP/2.
TCP
2015
HTTP/2 (2015)
Major revision over TCP. Binary Framing, Mutiplstuning Header Header Compression, Server Push
TCP
2022
HTTP/3 (2022)
Newest version over QUIC (UDP). Reduced Latancy, Stream TLS 1.3, Connection Migration
UDP

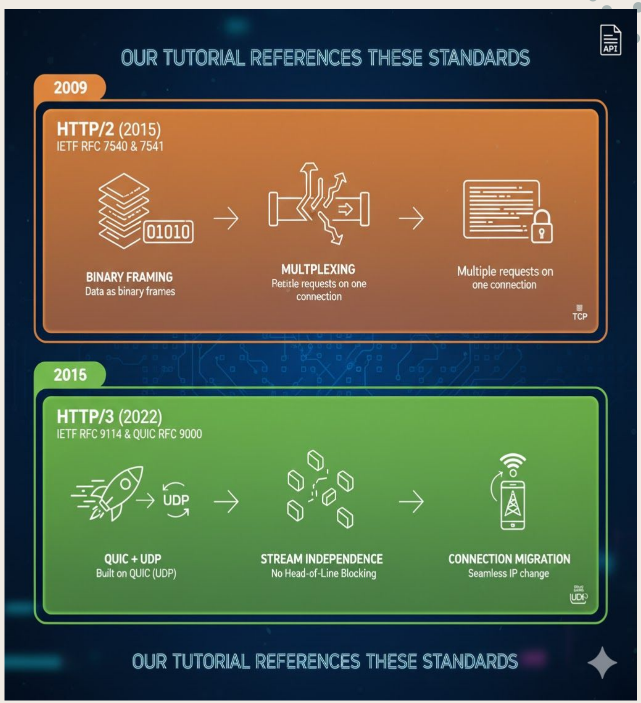
HTTP/2 and HTTP/3 Standards

HTTP/2 (2015)

- IETF RFC 7540 (binary framing, multiplexing, stream prioritization).
- RFC 7541 for HPACK header compression.

HTTP/3 (2022)

- IETF RFC 9114: HTTP mapped over QUIC instead of TCP.
- QUIC transport defined in RFC 9000 (plus related RFCs).
- Goals: reduce latency, fix head-of-line blocking at the transport layer, support connection migration.



Our tutorial references these standards in the api-design.md and discussion slides.

02

Step by Step Learning Activities

Learning Path Overview

- Module 1: Concept warm-up: Why HTTP/2 & HTTP/3?
- Module 2: Environment setup: Run backend + frontend.
- Module 3: Guided experiments: Vary protocol & network conditions.
- Module 4: Analysis: Interpret charts, connect to SER421 concepts.
- Module 5: Reflection: Pros/cons and future of HTTP/3.

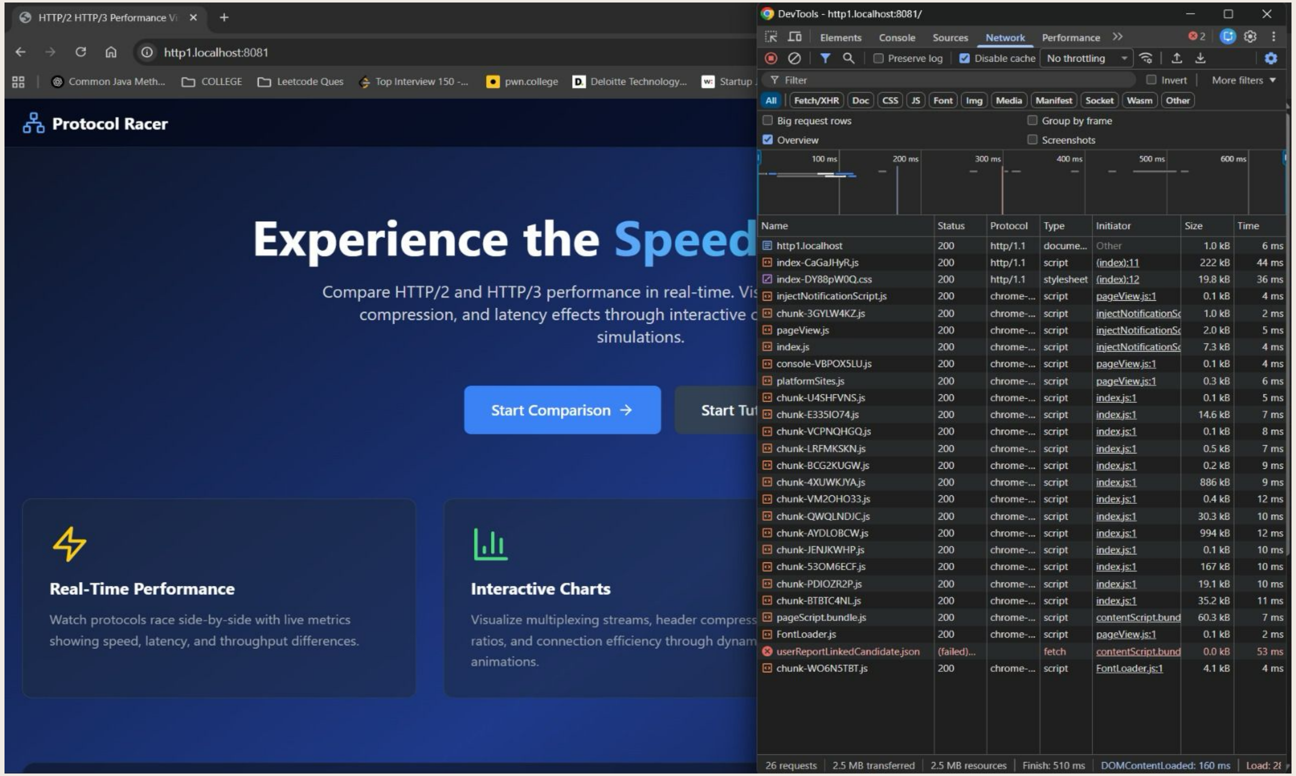
Module 1: Concept Warm-up

Activities:

- Short mini-lecture (or reading) covering:
    - HTTP/1.1 limitations, multiplexing, head-of-line blocking.
    - High-level features of HTTP/2 & HTTP/3.
- Quick think-pair-share:
    - “Where would you expect HTTP/3 to shine? High-loss mobile? Data center? Why?”

Connect HTTP evolution to SER421 topics: web architectures, protocols, latency, head-of-line blocking.

HTTP1



Experience the Speed

Compare HTTP/2 and HTTP/3 performance in real-time. Vis
compression, and latency effects through interactive c
simulations.

Real-Time Performance

Watch protocols race side-by-side with live metrics showing speed, latency, and throughput differences.

Interactive Charts

Visualize multiplexing streams, header compress
ratios, and connection efficiency through dynam
animations.

Name	Status	Protocol	Type	Initiator	Size	Time
http1.localhost	200	http/1.1	docume...	Other	1.0 kB	6 ms
index-CaGaJHyR.js	200	http/1.1	script	(index):11	222 kB	44 ms
index-DY88pW0Q.css	200	http/1.1	stylesheet	(index):12	19.8 kB	36 ms
injectNotificationScript.js	200	chrome-...	script	pageView.js:1	0.1 kB	4 ms
chunk-3GYLW4KZ.js	200	chrome-...	script	injectNotificationSc	1.0 kB	2 ms
pageView.js	200	chrome-...	script	injectNotificationSc	2.0 kB	5 ms
index.js	200	chrome-...	script	injectNotificationSc	7.3 kB	4 ms
console-VBPOX5LU.js	200	chrome-...	script	pageView.js:1	0.1 kB	4 ms
platformSites.js	200	chrome-...	script	pageView.js:1	0.3 kB	6 ms
chunk-U4SHFVNS.js	200	chrome-...	script	index.js:1	0.1 kB	5 ms
chunk-E335IO74.js	200	chrome-...	script	index.js:1	14.6 kB	7 ms
chunk-VCPNQHGQ.js	200	chrome-...	script	index.js:1	0.1 kB	8 ms
chunk-LRFMKSKN.js	200	chrome-...	script	index.js:1	0.5 kB	7 ms
chunk-BCG2KUGW.js	200	chrome-...	script	index.js:1	0.2 kB	9 ms
chunk-4XUWKJYA.js	200	chrome-...	script	index.js:1	886 kB	9 ms
chunk-VM2OHO33.js	200	chrome-...	script	index.js:1	0.4 kB	12 ms
chunk-QWQLNDJC.js	200	chrome-...	script	index.js:1	30.3 kB	10 ms
chunk-AYDLOBCW.js	200	chrome-...	script	index.js:1	994 kB	12 ms
chunk-JENJKWHP.js	200	chrome-...	script	index.js:1	0.1 kB	10 ms
chunk-53OM6ECF.js	200	chrome-...	script	index.js:1	167 kB	10 ms
chunk-PDIOZR2P.js	200	chrome-...	script	index.js:1	19.1 kB	10 ms
chunk-BTBTC4NL.js	200	chrome-...	script	index.js:1	35.2 kB	11 ms
pageScript.bundle.js	200	chrome-...	script	contentScript.bund	60.3 kB	7 ms
FontLoader.js	200	chrome-...	script	pageView.js:1	0.1 kB	2 ms
userReportLinkedCandidate.json	(failed)...		fetch	contentScript.bund	0.0 kB	53 ms
chunk-WO6N5TBT.js	200	chrome-...	script	FontLoader.js:1	4.1 kB	4 ms

26 requests | 2.5 MB transferred | 2.5 MB resources | Finish: 510 ms | DOMContentLoaded: 160 ms | Load: 2

Module 2: Environment Setup

Activities (mirrors your README):

- Clone repository, build & run:
  - Backend (Spring Boot): mvn clean package + mvn spring-boot:run.
  - Frontend (Vite/React): npm install + npm run dev.
- Verify:
  - GET /api/health returns { "status": "ok" }.
  - Playground page loads and passes health check.

Connect the project setup to SER421 topics: layered apps, REST APIs, build tools (Maven, npm).

Module 3: Guided Protocol Experiments

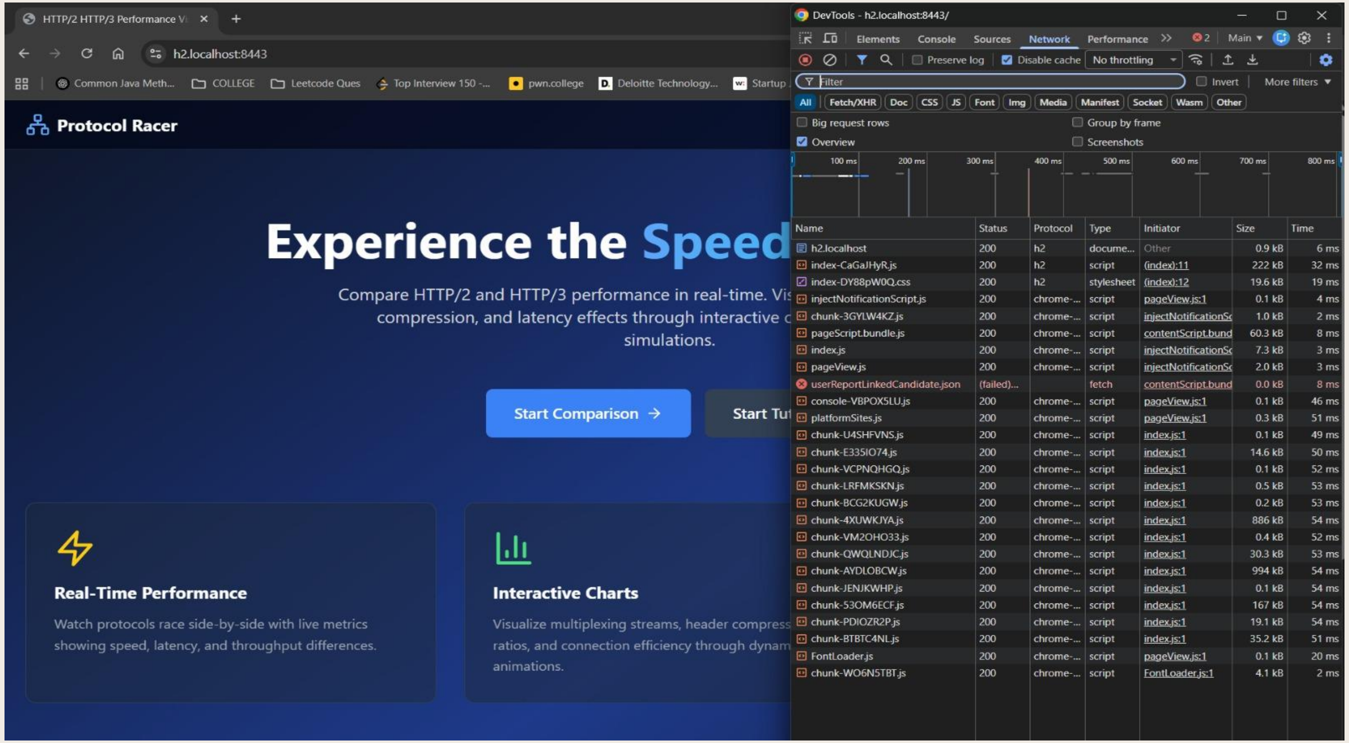
Activities:

- In the Playground:
  - Select protocol: HTTP/2 or HTTP/3.
  - Choose network conditions: 5G, WiFi, Slow 3G, etc.
  - Click Start Simulation and observe:
    - Latency card
    - Multiplexing streams
    - Throughput
    - Waterfall chart (resource fetch timing)
- Students complete a results table, e.g.

Trial	Protocol	Network	Avg latency	Streams	Notes

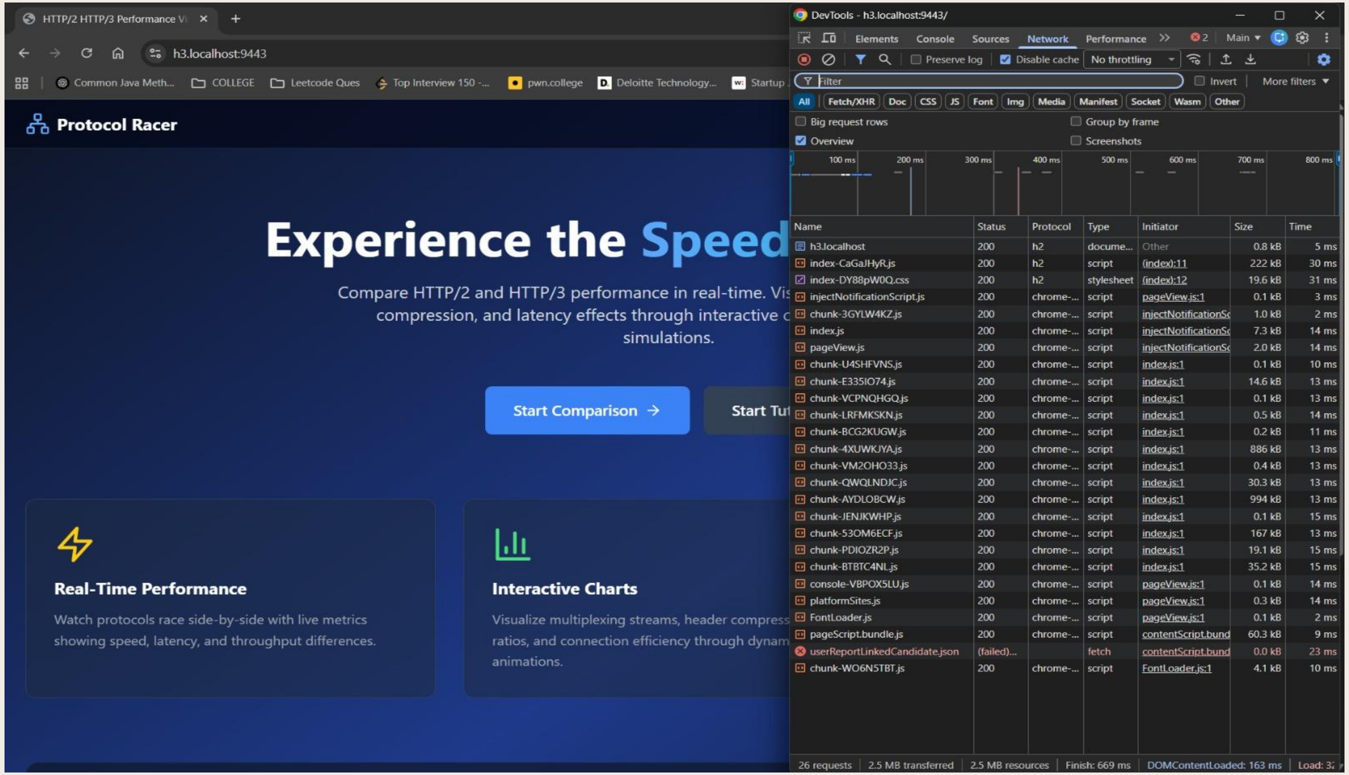
Connect the Playground runs to SER421 topics: latency, throughput, reliability, network conditions.

HTTP 2



Name	Status	Protocol	Type	Initiator	Size	Time
h2.localhost	200	h2	docume...	Other	0.9 kB	6 ms
index-CaGaJHyR.js	200	h2	script	(index):11	222 kB	32 ms
index-DY88pW0Q.css	200	h2	stylesheet	(index):12	19.6 kB	19 ms
injectNotificationScript.js	200	chrome-...	script	pageView.js:1	0.1 kB	4 ms
chunk-3GYLW4KZ.js	200	chrome-...	script	injectNotificationSc	1.0 kB	2 ms
pageScript.bundle.js	200	chrome-...	script	contentScript.bund	60.3 kB	8 ms
index.js	200	chrome-...	script	injectNotificationSc	7.3 kB	3 ms
pageView.js	200	chrome-...	script	injectNotificationSc	2.0 kB	3 ms
userReportLinkedCandidate.json	(failed)...		fetch	contentScript.bund	0.0 kB	8 ms
console-VBPOX5LU.js	200	chrome-...	script	pageView.js:1	0.1 kB	46 ms
platformSites.js	200	chrome-...	script	pageView.js:1	0.3 kB	51 ms
chunk-U4SHFVNS.js	200	chrome-...	script	index.js:1	0.1 kB	49 ms
chunk-E335IO74.js	200	chrome-...	script	index.js:1	14.6 kB	50 ms
chunk-VCPNQHGQ.js	200	chrome-...	script	index.js:1	0.1 kB	52 ms
chunk-LRFMKSKN.js	200	chrome-...	script	index.js:1	0.5 kB	53 ms
chunk-BCG2KUGW.js	200	chrome-...	script	index.js:1	0.2 kB	53 ms
chunk-4XUWKJYA.js	200	chrome-...	script	index.js:1	886 kB	54 ms
chunk-VM2OHO33.js	200	chrome-...	script	index.js:1	0.4 kB	52 ms
chunk-QWQLNDJC.js	200	chrome-...	script	index.js:1	30.3 kB	53 ms
chunk-AYDLOBCW.js	200	chrome-...	script	index.js:1	994 kB	54 ms
chunk-JENJKWHP.js	200	chrome-...	script	index.js:1	0.1 kB	54 ms
chunk-53OM6ECF.js	200	chrome-...	script	index.js:1	167 kB	54 ms
chunk-PDIOZR2P.js	200	chrome-...	script	index.js:1	19.1 kB	54 ms
chunk-BTBTC4NL.js	200	chrome-...	script	index.js:1	35.2 kB	51 ms
FontLoader.js	200	chrome-...	script	pageView.js:1	0.1 kB	20 ms
chunk-WO6N5TBT.js	200	chrome-...	script	FontLoader.js:1	4.1 kB	2 ms

HTTP 3



Experience the Speed

Compare HTTP/2 and HTTP/3 performance in real-time. Vis compression, and latency effects through interactive c simulations.

Real-Time Performance

Watch protocols race side-by-side with live metrics showing speed, latency, and throughput differences.

Interactive Charts

Visualize multiplexing streams, header compress ratios, and connection efficiency through dynam animations.

Name	Status	Protocol	Type	Initiator	Size	Time
h3.localhost	200	h2	docume...	Other	0.8 kB	5 ms
index-CaGaJHyR.js	200	h2	script	(index):11	222 kB	30 ms
index-DY88pW0Q.css	200	h2	stylesheet	(index):12	19.6 kB	31 ms
injectNotificationScript.js	200	chrome-...	script	pageView.js:1	0.1 kB	3 ms
chunk-3GYLW4KZ.js	200	chrome-...	script	injectNotificationSc	1.0 kB	2 ms
index.js	200	chrome-...	script	injectNotificationSc	7.3 kB	14 ms
pageView.js	200	chrome-...	script	injectNotificationSc	2.0 kB	14 ms
chunk-U4SHFVNS.js	200	chrome-...	script	index.js:1	0.1 kB	10 ms
chunk-E335IO74.js	200	chrome-...	script	index.js:1	14.6 kB	13 ms
chunk-VCPNQHGQ.js	200	chrome-...	script	index.js:1	0.1 kB	13 ms
chunk-LRFMKSKN.js	200	chrome-...	script	index.js:1	0.5 kB	14 ms
chunk-BCG2KUGW.js	200	chrome-...	script	index.js:1	0.2 kB	11 ms
chunk-4XUWKJYA.js	200	chrome-...	script	index.js:1	886 kB	13 ms
chunk-VM2OHO33.js	200	chrome-...	script	index.js:1	0.4 kB	13 ms
chunk-QWQLNDJC.js	200	chrome-...	script	index.js:1	30.3 kB	13 ms
chunk-AYDLOBCW.js	200	chrome-...	script	index.js:1	994 kB	13 ms
chunk-JENJKWHP.js	200	chrome-...	script	index.js:1	0.1 kB	15 ms
chunk-53OM6ECF.js	200	chrome-...	script	index.js:1	167 kB	13 ms
chunk-PDIOZR2P.js	200	chrome-...	script	index.js:1	19.1 kB	15 ms
chunk-BTBTC4NL.js	200	chrome-...	script	index.js:1	35.2 kB	15 ms
console-VBPOX5LU.js	200	chrome-...	script	pageView.js:1	0.1 kB	14 ms
platformSites.js	200	chrome-...	script	pageView.js:1	0.3 kB	14 ms
FontLoader.js	200	chrome-...	script	pageView.js:1	0.1 kB	2 ms
pageScript.bundle.js	200	chrome-...	script	contentScript.bund	60.3 kB	9 ms
userReportLinkedCandidate.json	(failed)...		fetch	contentScript.bund	0.0 kB	23 ms
chunk-WO6N5TBT.js	200	chrome-...	script	FontLoader.js:1	4.1 kB	10 ms

26 requests | 2.5 MB transferred | 2.5 MB resources | Finish: 669 ms | DOMContentLoaded: 163 ms | Load: 3

Module 4: Implementation Walkthrough

Activities:

- **Walk through key code paths:**
    - **Backend:**
        - ResourceController.java – /api/resource/css, /js, /image, /api/fast.
    - **Frontend:**
        - apiClient.ts: callHealth, runFastSimulation.
        - simulationService.ts: bridges UI & backend.
        - VisualizationPage.tsx: updates metrics & waterfall.
- **Ask students to:**
    - Identify where protocol and network condition are passed.
    - Modify a parameter (e.g., number of resources) and observe impact.

Connect the code tour to SER421 topics: controllers, routing, JSON, client - server separation.

Module 5: Reflection & Check-Your-Understanding

Activities:

- Short written or discussion prompts:
    - “Under which networks did HTTP/3 outperform HTTP/2? Why?”
    - “What trade-offs do QUIC and UDP introduce for middleboxes and firewalls?”
- Quick quiz or poll:
    - MCQs on history/standards.
    - Scenario questions: “You’re designing a mobile video app: which protocol and why?”

Connect the discussion to SER421 topics: architectural trade-offs, performance vs complexity, evidence-based design.

03

Analytical Component

Our Assessment: Strengths of HTTP/3

Evidence can come from:

- Your own experiment results.
- Papers/blogs you read.

Bullets:

- **Lower latency under loss**
  - QUIC avoids TCP-level head-of-line blocking: one lost packet doesn't stall all streams.
- **Connection migration**
  - Better experience on mobile (WiFi ↔ 5G) because QUIC connection survives IP changes.
- **Modern-friendly design**
  - Tuned for encrypted, multiplexed, high-concurrency workloads (CDNs, microservices, gRPC).

Our Assessment: Challenges & Risks

Operational friction

- Some middleboxes and older networks still treat high-volume UDP as suspicious.
- More complex debugging (encryption of more layers).

Ecosystem maturity

- Servers, load balancers, and APM tools still catching up compared to HTTP/1.1/2.

Not always a win

- On low-latency, low-loss LANs, we observed only modest gains vs HTTP/2.

Future Directions:

We expect HTTP/3 to become the default for:

- Browser traffic and CDN-delivered content.
- Latency-sensitive APIs (e.g., real-time collaboration, gaming, streaming).

Interesting research/industry directions:

- QUIC-based RPC (gRPC) and microservices in data centers.
- Better observability & debugging tools for QUIC traffic.
- Interaction with serverless and edge computing platforms.

Our conclusion:

HTTP/3 is not a silver bullet, but a strong default choice for modern, mobile-heavy workloads.

Summary & Takeaways

- History and motivation for HTTP/2 and HTTP/3.
- How to run and instrument a real app that compares them.
- How protocol choice impacts latency and throughput under different network conditions.
- Key Message: “Protocol choices are architectural decisions with measurable impact on non-functional requirements.”
- Q & A

TIME FOR DEMO!