Teacher Information, Course Description and Lecture-wise Plan

of

Advanced Web Technologies (CSC337)

under

Pilot Project:
Uniform Teaching, Assessment and Quality Assurance Across All The Campuses

by

Faculty of Information Science and Technology
COMSATS University Islamabad

A: Teacher Information

Course Name: Advanced Web Technologies	Credit Hours: 2+1
Course Code: CSC337	Pre-requisite: CSC336 - Web Technologies
Teacher Name: Aamir Pare	Teacher E-mail: aamir_shabbir@comsats.edu.pk
Program: BSCS, BSSE, BSAI	Semester: Fall 2026
Department: Computer Science	Campus: Islamabad

B: Course Contents

This course provides hands-on learning of current web technologies and production-grade full-stack engineering. Students cover enterprise architecture, REST APIs, security, authentication, OAuth 2.0, OWASP Top 10, and communication approaches including REST, GraphQL, JSON-RPC, and WebSockets. The course emphasizes modern frontend development with Next.js, including routing, rendering, server-side capabilities, performance, caching, scalability, database optimization, testing, and production deployment. Students learn to design, build, secure, optimize, and deploy modern web applications.

Recommended Books:

1. Web Development with Node and Express, Ethan Brown, O'Reilly.
2. Node.js Design Patterns, Mario Casciaro & Luciano Mammino.
3. Designing Data-Intensive Applications, Martin Kleppmann.
4. Real-World Next.js, Michele Riva.
5. MongoDB: The Definitive Guide, Shannon Bradshaw, Eoin Brazil & Kristina Chodorow.
6. Official Next.js Documentation (nextjs.org/learn).

C: Lecture-wise Plan

Week #	Lecture #	Topic (Chapter/Topics Covered)	Practice/ Evaluation	Remarks
1	1 (Date)	***Course Overview and Enterprise Web Application Architecture:*** • *Course introduction, CLOs and assessment plan; how AWT builds on Web Technologies* • *Web application architectures: monolithic, modular monolith, microservices and serverless* • *Layered architecture: presentation, application/service, business/domain, data access and infrastructure* • *Scalability and reliability fundamentals: scaling, stateless applications, load balancing, single points of failure*	*Exercises as given in the books*	*Casciaro & Mammino, Chapter No. 1*
	2 (Date)	***Technology Selection and Cross-Cutting Concerns:*** • *Microservices fundamentals: service boundaries, benefits, challenges and inter-service communication* • *Architecture and technology selection: when to choose what.* • *Trade-offs: complexity, cost, performance and maintainability* • *Cross-cutting concerns: security, performance (latency, throughput, caching), communication and deployment* • *Multi-tenancy SaaS Architecture*	*Exercises as given in the books*	*Casciaro & Mammino, Chapter No. 1*
2	3 (Date)	***Full-Stack Application Architecture - Business, Infrastructure and Application Layers*** • *What is software architecture; the layers of a full-stack application: presentation, business, infrastructure, application and data* • *Business layer: domain understanding, requirements, business rules, use cases, user stories; functional vs. non-functional requirements* • *Infrastructure layer: servers, VMs, containers, orchestration, cloud service models (IaaS/PaaS/SaaS), load balancing and CDN* • *Application layer: API gateway, service layer, cross-cutting concerns, stateful vs. stateless design, fault tolerance and graceful degradation*	*Exercises as given in the books* + *Assignment 1*	*Kleppman n, Chapter No. 1 & 2*
	4 (Date)	***Data Layer and Microservices Architecture:*** • *Data layer design: data modelling, SQL vs. NoSQL, polyglot persistence,*	***Quiz # 1***	*Kleppman n, Chapter*

		replication/partitioning, indexing and the CAP theorem • *Monolithic vs. modular-monolith vs. microservices vs. serverless architectures selection* • *Microservices principles: service decomposition, bounded contexts and service boundaries* • *Inter-service communication: REST, gRPC, message queues; service discovery, API gateway, database-per-service* • *Benefits, challenges and anti-patterns of microservices; when a monolith is the better choice; architecture/technology selection trade-offs*		*No. 1 & 2*
3	5 (Date)	***Professional API Design Practices:*** • *REST architectural constraints: client-server, statelessness, cacheability, layered system, uniform interface* • *Resource modelling and URI naming conventions; nouns vs. verbs, nesting of resources; HTTP methods, idempotency and safe methods* • *Status codes and consistent error responses; pagination, filtering, sorting and partial responses* • *API versioning strategies and HATEOAS; API documentation with OpenAPI/Swagger and API-Gateway basics* • *Breaking vs. non-breaking changes* • *Deprecation policy, sunset headers and consumer-driven contracts*	*Exercises as given in the books*	*Ethan Brown Chapter No 15 -18*
	6 (Date)	***REST and GraphQL*** • *REST recap: resource-oriented request/response; over-fetching and under-fetching* • *GraphQL schema, types and the Schema Definition Language* • *Queries, mutations, subscriptions and resolvers* • *When GraphQL makes sense; caching and query-complexity concerns*	*Exercises as given in the books*	*GraphQL official document ation*
4	7 (Date)	***JSON-RPC and WebSockets*** • *JSON-RPC: procedure-oriented communication, message format, batching and notifications* • *WebSocket protocol and handshake; persistent bidirectional communication* • *Real-time communication with Socket.IO: server/client setup, events, rooms, namespaces, broadcasting; scaling with the Redis adapter* • *Comparison with Server-Sent Events and long*	*Exercises as given in the books*	*JSON-RPC 2.0 specificati on; Socket.IO document ation*

		polling; real-time use cases (chat, notifications, dashboards)		
	8	***Selecting a Suitable Communication Approach*** • *Requirement-driven technology-selection criteria* • *CRUD/resource API → REST; flexible client queries → GraphQL* • *Procedure-oriented API → JSON-RPC; real-time bidirectional → WebSockets* • *Non-functional factors (caching, tooling, security, expertise); decision exercise on a case study*	*Exercises as given in the books*	*Kleppman, Chapter No. 4*
5	9 (Date)	***Application Security: Authentication and Authorization*** • *Authentication vs. authorization; stateful sessions vs. stateless tokens* • *Session-based authentication and server-side session stores* • *Token-based authentication and JWT: structure, signing, expiry and refresh tokens; secure token storage (httpOnly cookies vs. localStorage)* • *Role-Based Access Control (RBAC); overview of access-control models (DAC, MAC, RBAC, ABAC)* • *Password hashing (bcrypt) and salting, password policies, multi-factor authentication overview*	*Exercises as given in the books*	*Ethan Brown Chapter No 13*
	10 (Date)	***OAuth 2.0 and OpenID Connect*** • *OAuth 2.0 as an authorization framework: roles and terminology* • *Authorization Code flow with PKCE; overview of other grant types* • *Access tokens, refresh tokens, scopes and user consent* • *OpenID Connect and the ID token; social login and third-party identity providers* • *Passport.js strategies; common OAuth mistakes and secure implementation*	*Exercises as given in the books*	*OAuth 2.0 / OIDC specifications*
6	11 (Date)	***Web-Specific Security*** • *Cross-Site Scripting (stored, reflected, DOM-based) and output encoding* • *Cross-Site Request Forgery, anti-CSRF tokens and SameSite cookies* • *CORS: preflight requests, headers and safe configuration*	*Exercises as given in the books*	*OWASP Cheat Sheet Series*

		• *SQL/NoSQL injection and sanitization; HTTPS/TLS, secure cookies and Content Security Policy*		
	12 (Date)	***API Security*** • *Input validation and schema-based request validation (express-validator / Joi)* • *Rate limiting, throttling and quotas; HPP, mongo-sanitize, xss-clean* • *API keys vs. tokens and mutual TLS; secrets management and key rotation* • *Authorization at the API and resource level; object-level access checks*	*Exercises as given in the books + Assignment No 2*	*OWASP API Security Top 10*
7	13 (Date)	***OWASP Top 10*** • *Purpose, structure and correct use of the OWASP Top 10* • *Broken access control, cryptographic failures and injection* • *Insecure design, security misconfiguration and vulnerable/outdated components* • *Authentication and integrity failures, logging/monitoring failures, SSRF; practical hardening checklist (Helmet, disabling x-powered-by)*	*Exercises as given in the books*	*OWASP Top 10 (2021)*
	14 (Date)	***Frontend Performance*** • *Core Web Vitals (LCP, CLS, INP) and measurement with Lighthouse* • *HTTP and browser caching of static assets* • *Asset optimisation: images, fonts, compression and minification* • *Lazy loading, code splitting and bundle analysis*	***Quiz # 2***	*web.dev performance guides*
8	15 (Date)	***Web / Server Caching*** • *Cache-Control, ETag, Last-Modified and cache validation* • *Private browser cache vs. shared caches* • *CDN and edge caching; reverse-proxy caching with Nginx/Varnish* • *Cache keys, invalidation and purging strategies*	*Exercises as given in the books*	*Kleppmann, Chapter No. 1*
	16	***Application Caching with Redis***		

		• *Introduction to Redis: in-memory data store and data structures (strings, hashes, lists, sets, sorted sets)* • *Cache-aside, read-through and write-through/write-behind patterns* • *Cache invalidation, TTL, eviction policies and stampede protection* • *Using Redis for API caching, session storage, rate limiting and leaderboards*	*Exercises as given in the books*	*Redis document ation*
9	17 (Date)	*Midterm Examination*	***Midterm (25%)***	
	18 (Date)	***Database Monitoring, Optimization and Serverless Scalability*** • *Indexing strategies and reading query plans (EXPLAIN / ANALYZE)* • *Query optimisation, the N+1 problem, connection pooling and slow-query monitoring* • *Serverless functions: execution model, cold starts, statelessness and limits* • *Horizontal vs. vertical scaling, load balancing and bottleneck identification*	*Exercises as given in the books*	*Kleppman n, Chapter No. 3-6*
10	19 (Date)	***Event-Driven Scalability with Kafka and Multi-Tenancy SaaS Products*** • *Event-driven architecture: events, commands, producers and consumers; Kafka concepts (topics, partitions, offsets, brokers, replication)* • *Delivery semantics (at-most-once / at-least-once / exactly-once), ordering and retention; integrating Kafka with Node.js (KafkaJS)* • *Software as a Service: single-tenant vs. multi-tenant architectures and business model overview* • *Tenancy models (shared DB with tenant ID, schema-per-tenant, DB-per-tenant), tenant routing, data isolation, subscription/billing basics*	*Exercises as given in the books + Assignment No 3*	*Kleppman n, Chapter No. 11 ; Redis / Kafka document ation*
	20 (Date)	***Next.js Architecture and Project Setup*** • *Recap of React fundamentals: components, props, state and hooks; limitations of client-side-only React* • *Next.js features: hybrid rendering, file-system routing, built-in optimization and full-stack capability* • *Rendering strategies overview: CSR, SSR, SSG*	***Quiz # 3***	*Riva, Chapter No. 1-2*

		and ISR - concepts and trade-offs • *Project setup (create-next-app), folder structure, configuration files; App Router vs. Pages Router*		
11	21 (Date)	***Routing and Layout Management*** • *File-based routing conventions and special files (page, layout, loading, error, not-found)* • *Nested routes, route groups, dynamic, catch-all and optional catch-all segments* • *Layouts, templates and shared UI* • *Navigation: Link component, useRouter/usePathname, programmatic navigation, redirects and rewrites*	*Exercises as given in the books*	*Riva, Chapter No. 3*
	22 (Date)	***Rendering Strategies and Server-Side Capabilities*** • *Server Components vs. Client Components; the "use client" directive and composition rules* • *Server-side data fetching, request memoization, caching and revalidation (time-based and on-demand ISR)* • *Streaming and Suspense; loading skeletons and progressive rendering* • *Route handlers / API routes, Server Actions and Next.js middleware*	*Exercises as given in the books*	*Riva, Chapter No. 4*
12	23 (Date)	***Components, Layouts and Styling*** • *Special-file behaviour; Metadata API for titles, descriptions and SEO; built-in optimizations (next/image, next/font, next/script)* • *Styling options: global CSS, CSS Modules, Tailwind CSS and CSS-in-JS* • *Component libraries (shadcn/ui, MUI) and design systems; responsive design, dark mode and accessibility* • *Building the application shell: navigation, sidebar, forms and reusable UI components*	*Exercises as given in the books*	*Riva, Chapter No. 6 & 7*
	24 (Date)	***Frontend Architecture*** • *Component organisation and folder conventions at scale* • *State categories: local UI state, server state and URL state* • *API/service layer and data-access abstraction; form handling and schema validation* • *Error boundaries, loading skeletons and code splitting*	*Exercises as given in the books*	*Riva, Chapter No. 6 & 7*
13	25 (Date)	***Data Fetching and State Management with TanStack Query:*** • *Client-side data fetching with SWR / TanStack*	*Exercises*	*TanStack*

		Query: caching, staleness, mutations and optimistic updates • *Query keys, pagination, infinite queries, prefetching, retries and error handling* • *State-management options in Next.js (Context, Zustand, Redux Toolkit) and when to use each* • *Handling errors, empty states and performance pitfalls in data fetching*	*as given in the books + Assignment No 4*	*Query document ation*
	26 (Date)	***Authentication and Security in Next.js*** • *Authentication approaches in Next.js: session-based, JWT and third-party providers* • *NextAuth.js / Auth.js: setup, providers, callbacks, sessions and adapters* • *Protecting pages, layouts, route handlers and server actions; middleware-based route protection and role-based UI rendering* • *Secure handling of environment variables (server-only secrets vs. public variables); XSS, CSRF, CORS and secure headers*	***Quiz # 4***	*Riva, Chapter No. 4*
14	27 (Date)	***Performance Optimization and Testing in Next.js*** • *Performance metrics and Core Web Vitals (LCP, CLS, INP)* • *Code splitting, dynamic imports, lazy loading and bundle analysis; image/font/asset optimization* • *SEO in Next.js: metadata, sitemap, robots.txt, structured data and Open Graph* • *Testing the application: unit, component and end-to-end testing (Jest, React Testing Library, Playwright/Cypress)*	*Exercises as given in the books*	*Riva, Chapter No. 9*
	28 (Date)	***Deploying the Product on Vercel*** • *Build process, production build output and self-hosting vs. managed hosting* • *Introduction to Vercel: projects, Git integration, preview and production deployments* • *Configuring environment variables, secrets and build settings on Vercel* • *Serverless and edge functions/middleware; custom domains, HTTPS, CDN/caching behaviour, analytics and CI/CD rollbacks*	*Exercises as given in the books*	*Next.js document ation*
15	29 (Date)	***Web Application and API Testing*** • *Testing pyramid: unit, integration and end-to-end tests* • *Unit and integration testing tools (Jest/Vitest,*	*Exercises as given in the books*	*Ethan Brown Chapter*

		React Testing Library, Supertest) • *API testing: success paths, validation errors, authentication and edge cases; Postman collections and Swagger UI* • *End-to-end testing (Playwright/Cypress) with mocking; coverage and CI integration*		*No 5*
	30 (Date)	***Reliability and Observability: Log Levels*** • *Reliability practices: retries, circuit breakers, health checks and graceful degradation; CI/CD pipeline and deployment strategies (build, test, deploy, rollback)* • *Why logging matters: logs vs. metrics vs. traces* • *Log levels: TRACE, DEBUG, INFO, WARN, ERROR and FATAL; structured logging with correlation/request identifiers* • *Log aggregation, retention and handling of sensitive data; deciding what to log (signal vs. noise)*	*Exercises as given in the books*	*Kleppman, Chapter No. 13*
16	31 (Date)	***Application and Server Monitoring; Course Review*** • *Key metrics: latency, error rate, throughput and saturation; health checks, uptime monitoring and alerting* • *Application performance monitoring (APM) and distributed-tracing basics* • *Server monitoring (CPU, memory, disk, network); dashboards, SLIs/SLOs and incident response* • *Consolidated review of Units 1-7; semester-project demonstration and final-exam guidelines*	*Exercises as given in the books*	*Kleppman, Chapter No. 12*
	32 (Date)	• *Review*		
17	(Date)		***Final (50%)***	

Note: All Assignments/Quizes weightage is 25%