Cloud Contraptions
TRAINING & CONSULTING

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

Real-Time Apps with React, ASP.NET Core MVC, and SignalR

Class Duration

14 hours of live training delivered over 2 days

Student Prerequisites

- Professional experience building React applications with function components and hooks
- Working experience with ASP.NET Core MVC (controllers, routing, dependency injection)
- Comfort with C# and modern TypeScript or JavaScript

Target Audience

Full-stack developers who already build with React and ASP.NET Core MVC and now need to add live, server-pushed features: dashboards, notifications, chat, collaborative editing, and progress updates. This focused two-day course assumes both halves of the stack are familiar and concentrates entirely on the real-time layer that connects them. Teams wanting to deepen the client side afterward can continue with *Advanced React*.

Description

This two-day course teaches experienced full-stack developers how to add real-time features to applications that pair a React front end with an ASP.NET Core MVC back end, using SignalR as the bidirectional transport. Students learn the SignalR architecture on .NET 10, build strongly-typed hubs, and wire them to a React client with the `@microsoft/signalr` package managed cleanly inside React's component lifecycle. The course covers the patterns that matter in production: targeting individual users and groups, securing hubs with the same authentication and authorization the MVC app already uses, and handling automatic reconnection and connection state in the UI. Students also work with server-to-client streaming, the MessagePack protocol for efficient payloads, and the connection-lifecycle events that keep client state consistent. The final topics address scale-out with the Redis backplane and the Azure SignalR Service, along with logging, diagnostics,

and load considerations for many concurrent connections. By the end, students can design and ship reliable real-time features without disrupting the existing React and MVC codebase.

Learning Outcomes

- Explain the SignalR architecture on .NET 10: hubs, transports, the WebSockets fallback chain, and where SignalR fits in an existing React and ASP.NET Core MVC application.
- Build strongly-typed SignalR hubs with hub methods, strongly-typed clients, and dependency-injected services, and host them alongside MVC controllers.
- Connect a React client with `@microsoft/signalr`, managing the connection inside the component lifecycle with hooks, refs, and clean teardown.
- Push messages to all clients, specific users, and groups, and design group membership for rooms, tenants, and per-user channels.
- Secure hubs with the MVC app's authentication and authorization, flowing identity and access tokens through the SignalR connection.
- Handle automatic reconnection, connection-state changes, and message buffering, and reflect connection status reliably in the React UI.
- Use server-to-client streaming and the MessagePack protocol for efficient, high-volume real-time payloads.
- Scale SignalR out with the Redis backplane or the Azure SignalR Service, and apply logging, diagnostics, and load planning for many concurrent connections.

Training Materials

All students receive comprehensive courseware covering all topics in the course. Courseware is distributed via GitHub in the form of documentation and extensive code samples. Students practice the topics covered through challenging hands-on lab exercises.

Software Requirements

Students will need a free, personal GitHub account to access the courseware. Students will need permission to install the .NET 10 SDK, Node.js, and Visual Studio Code (or Visual Studio 2026) on their computers, along with permission to install NuGet and NPM packages and Visual Studio Code

extensions. If students are unable to configure a local environment, a cloud-based environment can be provided.

Training Topics

Real-Time on the React and MVC Stack

- Why polling falls short: the case for server-pushed updates
- Real-time use cases: dashboards, notifications, chat, presence, collaboration
- Where SignalR fits between a React client and an ASP.NET Core MVC server
- SignalR on .NET 10: hubs, transports, and the WebSockets fallback chain

Building SignalR Hubs

- Creating hubs and defining hub methods
- Strongly-typed hubs with typed client interfaces
- Dependency injection inside hubs
- Mapping hub endpoints next to MVC routes
- Sending from outside a hub with `IHubContext`

The React SignalR Client

- The `@microsoft/signalr` client: building and starting a connection
- Managing the connection in React: hooks, refs, and effect cleanup
- Registering and removing handlers without leaks
- Invoking hub methods and awaiting results
- A reusable connection hook and context for the app

Targeting Clients

- Broadcasting to all connections
- Sending to specific users by identity
- Groups: rooms, tenants, and per-user channels
- Managing group membership across connections
- Connection IDs and the user identifier model

Security and Authorization

- Reusing the MVC app’s authentication for hubs
- Flowing access tokens through the SignalR connection
- `[Authorize]` on hubs and hub methods

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

- Mapping users to connections and groups securely
- Guarding against unauthorized invocations

Connection Lifecycle and Resilience

- Connection state and lifecycle events
- Automatic reconnection and configurable retry policies
- Reflecting connection status in the React UI
- Handling missed messages and resynchronizing state
- Graceful disconnect and cleanup

Streaming and Efficient Payloads

- Server-to-client streaming for incremental results
- Client-to-server streaming basics
- The MessagePack hub protocol for compact payloads
- Designing message contracts shared by client and server
- Throttling and batching high-frequency updates

Scaling and Operations

- Why a single server limits SignalR scale-out
- The Redis backplane for multi-server deployments
- The Azure SignalR Service as a managed option
- Logging, diagnostics, and tracing connections
- Load and capacity planning for many concurrent clients

