

ASP Development Professional

Prüfungscode IFW-ASP-300-01 · Fassung 1.0 · Stand 18.09.2026

ASP.NET Core für anspruchsvolle Anwendungen: Dependency Injection und Konfiguration, Middleware, EF Core, Sicherheit sowie Asynchronität, Caching und Betrieb.

Die Prüfung im Überblick

Bearbeitungszeit 2 Stunden	Fragen 70
Bestehensgrenze 70 %	Fragetypen Einfach- und Mehrfachauswahl
Prüfungssprache Deutsch	Zertifikat gültig 36 Monate
Durchführung Online, zeitlich überwacht	Preis 99,00 €

Themenblöcke

Block	Thema	Gewichtung	Fragen
1	Dependency Injection und Konfiguration	20 %	14
2	Middleware und Endpunkte	20 %	14
3	Datenzugriff mit EF Core	20 %	14
4	Authentifizierung, Autorisierung und HTTPS	20 %	14
5	Asynchronität, Caching und Betrieb	20 %	14
Gesamt		100 %	70

Die Gewichtung gibt an, welchen Anteil ein Block an den Prüfungsfragen hat. Die Fragen werden beim Start zufällig und anteilig je Block aus dem Fragenpool gezogen.

Dependency Injection und Konfiguration

Services mit der passenden Lebensdauer registrieren und Einstellungen zur Laufzeit übernehmen.

- 1.1** Lebensdauern wählen
Transient, Scoped und Singleton; Captive Dependencies vermeiden
- 1.2** Scoped Services außerhalb eines Requests nutzen
IServiceScopeFactory in einem BackgroundService
- 1.3** Konfiguration zur Laufzeit neu laden
IOptions, IOptionsSnapshot, IOptionsMonitor
- 1.4** HttpClient richtig verwenden
IHttpClientFactory statt new HttpClient() je Aufruf, Socket Exhaustion

Middleware und Endpunkte

Die Request-Pipeline in der richtigen Reihenfolge aufbauen und gezielt verzweigen.

- 2.1** Eigene Middleware schreiben
app.Use, await next(), Änderungen nach dem Start der Antwort
- 2.2** Die Reihenfolge der Pipeline festlegen
UseStaticFiles, UseAuthentication, UseAuthorization
- 2.3** Die Pipeline verzweigen
Map, MapWhen, UseWhen
- 2.4** Minimal APIs mit Filtern erweitern
AddEndpointFilter, Validierung vor dem Handler

Datenzugriff mit EF Core

Abfragen formulieren, die korrekt sind und auch bei vielen Daten schnell bleiben.

- 3.1** Change Tracking verstehen
Identity Resolution, AsNoTracking()
- 3.2** Ladestrategien wählen
Lazy Loading und N+1-Abfragen, Include, kartesische Explosion, AsSplitQuery()
- 3.3** Die Ausführung von LINQ-Abfragen beurteilen
verzögerte Ausführung, Auswertung in der Datenbank oder im Client
- 3.4** Konflikte bei gleichzeitigen Änderungen behandeln
Concurrency Tokens mit [Timestamp], DbUpdateConcurrencyException

Authentifizierung, Autorisierung und HTTPS

Wer darf was – und wie die Anwendung hinter einem Proxy sicher bleibt.

- 4.1** Authentifizierung und Autorisierung unterscheiden
Schemes, Claims, [Authorize]
- 4.2** Richtlinien definieren
Policies mit RequireRole und RequireClaim, mehrere Anforderungen kombinieren
- 4.3** Hinter einem Reverse Proxy betreiben
TLS-Terminierung, UseForwardedHeaders, X-Forwarded-Proto

Asynchronität, Caching und Betrieb

Anwendungen, die unter Last reaktionsfähig bleiben und sich überwachen lassen.

- 5.1** Asynchron programmieren
async/await ohne Blockieren, ConfigureAwait in ASP.NET Core
- 5.2** Abbrüche weitergeben
CancellationToken vom Request bis zur Datenbankabfrage
- 5.3** Über mehrere Instanzen cachen
IMemoryCache gegenüber IDistributedCache, Invalidierung
- 5.4** Health Checks einrichten
AddHealthChecks, MapHealthChecks, Liveness und Readiness

Institut für Webentwicklung · unabhängiges Lernangebot ohne Verbindung zu den genannten Technologieanbietern. Maßgeblich ist die Fassung auf der Zertifizierungsseite.