.NET Interview Topics - Complete Checklist

Core .NET Framework & CLR Concepts

✅ Fundamentals

- [ ] **Value Types vs Reference Types** ⭐ (Already covered)
- [ ] **Common Language Runtime (CLR)**
- [ ] **Just-In-Time (JIT) Compilation**
- [ ] **Application Domains (AppDomains)**
- [ ] **Assembly and Namespace**
- [ ] **.NET Framework vs .NET Core vs .NET 5+**

✅ Memory Management

- [ ] **Garbage Collection (GC)**
  - Generation-based GC
  - GC algorithms (Mark & Sweep, Generational)
  - Finalization vs IDisposable
  - Memory leaks prevention
- [ ] **Stack vs Heap Memory**
- [ ] **Memory Profiling and Optimization**

✅ Type System

- [ ] **Boxing and Unboxing**
- [ ] **Nullable Types**
- [ ] **Generic Types and Constraints**
- [ ] **Covariance and Contravariance**
- [ ] **Reflection and Metadata**

Object-Oriented Programming

✅ Core OOP Concepts

- [ ] **Encapsulation, Inheritance, Polymorphism**
- [ ] **Abstract Classes vs Interfaces**
- [ ] **Method Overloading vs Overriding**
- [ ] **Virtual, Override, New keywords**
- [ ] **Static vs Instance members**

✅ Advanced OOP

- [ ] **SOLID Principles**
  - Single Responsibility Principle (SRP)
  - Open/Closed Principle (OCP)
  - Liskov Substitution Principle (LSP)
  - Interface Segregation Principle (ISP)
  - Dependency Inversion Principle (DIP)
- [ ] **Design Patterns**
  - Creational: Singleton, Factory, Builder
  - Structural: Adapter, Decorator, Facade
  - Behavioral: Observer, Strategy, Command

C# Language Features

✅ Core Language Features

- [ ] **Properties vs Fields**
- [ ] **Indexers**
- [ ] **Operator Overloading**
- [ ] **Extension Methods**
- [ ] **Partial Classes and Methods**
- [ ] **Anonymous Types and Methods**

✅ Advanced Language Features

- [ ] **LINQ (Language Integrated Query)**
  - Query syntax vs Method syntax
  - Deferred execution
  - IEnumerable vs IQueryable
- [ ] **Lambda Expressions and Delegates**
- [ ] **Events and Event Handling**
- [ ] **Generics and Constraints**
- [ ] **Nullable Reference Types (C# 8+)**

✅ Modern C# Features

- [ ] **Pattern Matching (C# 7+)**
- [ ] **Local Functions**

- [ ] **Tuples and Deconstruction**
- [ ] **Record Types (C# 9+)**
- [ ] **Init-only Properties**
- [ ] **Top-level Programs**
- [ ] **Global Using Statements**

Asynchronous Programming

✅ Async/Await

- [ ] **Task and Task<T>**
- [ ] **async/await keywords**
- [ ] **ConfigureAwait(false)**
- [ ] **Deadlock scenarios and prevention**
- [ ] **Exception handling in async methods**

✅ Threading and Concurrency

- [ ] **Thread vs Task**
- [ ] **ThreadPool**
- [ ] **Synchronization primitives**
  - lock statement
  - Monitor
  - Mutex, Semaphore
  - ReaderWriterLock
- [ ] **Concurrent Collections**
- [ ] **Parallel LINQ (PLINQ)**
- [ ] **Task Parallel Library (TPL)**

Collections and Data Structures

✅ Built-in Collections

- [ ] **Array vs List<T> vs LinkedList<T>**
- [ ] **Dictionary<K,V> vs Hashtable**
- [ ] **HashSet<T> and SortedSet<T>**
- [ ] **Queue<T> and Stack<T>**
- [ ] **IEnumerable vs ICollection vs IList**

✅ Performance Considerations

- [ ] Big O notation for collections
- [ ] When to use which collection
- [ ] Memory overhead of collections
- [ ] Concurrent collections for multithreading

Exception Handling

✅ Exception Management

- [ ] try-catch-finally blocks
- [ ] Exception hierarchy
- [ ] Custom exceptions
- [ ] Exception handling best practices
- [ ] Global exception handling
- [ ] Structured exception handling

✅ Advanced Exception Concepts

- [ ] Inner exceptions
- [ ] Exception filters (when)
- [ ] Exception handling in async methods
- [ ] Performance impact of exceptions

Data Access and Entity Framework

✅ ADO.NET

- [ ] Connection, Command, DataReader
- [ ] DataSet vs DataReader
- [ ] Connection pooling
- [ ] SQL injection prevention
- [ ] Transactions

✅ Entity Framework

- [ ] Code First vs Database First
- [ ] DbContext and DbSet
- [ ] LINQ to Entities
- [ ] Change tracking
- [ ] Lazy vs Eager loading
- [ ] Migration and seeding
- [ ] Performance optimization

✅ Dapper and Micro-ORMs

- [ ] When to use Dapper vs EF
- [ ] Raw SQL execution
- [ ] Parameter binding

Dependency Injection and IoC

✅ DI Concepts

- [ ] Dependency Injection principles
- [ ] Constructor vs Property vs Method injection
- [ ] Service lifetimes (Singleton, Transient, Scoped)
- [ ] IoC containers (built-in, Autofac, Unity)

✅ Advanced DI

- [ ] Service registration patterns
- [ ] Factory patterns with DI
- [ ] Decorator pattern with DI
- [ ] Circular dependencies

Web Development

✅ ASP.NET Core

- [ ] MVC pattern
- [ ] Dependency Injection in ASP.NET Core
- [ ] Middleware pipeline
- [ ] Routing
- [ ] Model binding and validation
- [ ] Action filters
- [ ] Authentication and Authorization

✅ Web API

- [ ] RESTful API design
- [ ] HTTP status codes
- [ ] Content negotiation
- [ ] API versioning
- [ ] CORS (Cross-Origin Resource Sharing)
- [ ] JWT tokens

✅ SignalR

- [ ] Real-time communication
- [ ] Hubs and clients
- [ ] Connection management

Testing

✅ Unit Testing

- [ ] xUnit, NUnit, MSTest frameworks
- [ ] Arrange-Act-Assert pattern
- [ ] Mocking with Moq
- [ ] Test-driven development (TDD)
- [ ] Code coverage

✅ Integration Testing

- [ ] Testing with TestHost
- [ ] Database testing strategies
- [ ] Testing async code

Performance and Optimization

✅ Performance Best Practices

- [ ] String concatenation optimization
- [ ] Collection performance
- [ ] Memory management
- [ ] CPU profiling
- [ ] Database query optimization

✅ Caching

- [ ] In-memory caching
- [ ] Distributed caching (Redis)
- [ ] Cache strategies and patterns
- [ ] Cache invalidation

Security

✅ Application Security

- [ ] Input validation

- [ ] SQL injection prevention
- [ ] XSS prevention
- [ ] CSRF protection
- [ ] Secure coding practices

✅ Authentication & Authorization

- [ ] Claims-based identity
- [ ] OAuth 2.0 and OpenID Connect
- [ ] JWT tokens
- [ ] Role-based vs Policy-based authorization

Deployment and DevOps

✅ Deployment

- [ ] IIS deployment
- [ ] Docker containerization
- [ ] Azure deployment
- [ ] Configuration management
- [ ] Environment-specific settings

✅ Monitoring and Logging

- [ ] Application Insights
- [ ] Structured logging
- [ ] Log levels and best practices
- [ ] Health checks

Advanced Topics

✅ Microservices

- [ ] Service communication patterns
- [ ] API Gateway pattern
- [ ] Service discovery
- [ ] Circuit breaker pattern
- [ ] Event-driven architecture

✅ Messaging and Queues

- [ ] Message queues (RabbitMQ, Azure Service Bus)
- [ ] Event sourcing

- [ ] **CQRS pattern**

✅ Cloud-Native Development

- [ ] **Azure services integration**
- [ ] **Serverless computing (Azure Functions)**
- [ ] **Blob storage and databases**
- [ ] **Service fabric**

Priority Levels

🔥 Must Know (High Priority)

- Value Types vs Reference Types
- SOLID Principles
- Async/Await and Threading
- Garbage Collection
- LINQ and Collections
- Exception Handling
- Dependency Injection

⭐ Should Know (Medium Priority)

- Design Patterns
- Entity Framework
- ASP.NET Core basics
- Unit Testing
- Performance optimization

💡 Nice to Know (Lower Priority)

- Advanced C# features
- Microservices patterns
- Cloud-specific implementations
- Advanced security topics

Study Strategy

1. **Start with fundamentals** - Master the "Must Know" topics first
2. **Practice coding** - Implement examples for each concept
3. **Build projects** - Apply multiple concepts in real applications
4. **Mock interviews** - Practice explaining concepts clearly
5. **Stay updated** - Follow latest .NET releases and features

Use this checklist to track your progress and ensure comprehensive coverage of .NET interview topics!