Dotnet Core MVC + Microservices Interview Cheat Sheet

1. What is the difference between .NET Core MVC and Web API?

- MVC is used for building web apps with views (Razor), while Web API is for RESTful services.
- MVC returns Views or JSON; Web API returns pure JSON or XML responses.
- MVC: [Route("Home/Index")]; Web API: [HttpGet("api/products/{id}")]

2. What are Microservices and how do you implement them in .NET Core?

- Microservices: Independent services for specific features/modules.
- Implement using ASP.NET Core Web APIs, separate projects for each service.
- Use tools like Ocelot for API Gateway, RabbitMQ for communication.
- Each service should have its own DB.

3. Explain Dependency Injection in .NET Core

- DI is a design pattern to inject service dependencies at runtime.
- Register services in Program.cs/Startup.cs:

```
services.AddScoped<IMyService, MyService>();
```

- Constructor injection is most common.

4. What is JWT and how is it used in authentication?

- JWT = JSON Web Token: a secure token passed between client and server.
- Contains claims like userId, role.
- Used in Authorization header as 'Bearer <token>'.
- Validate using middleware in Startup.cs.

5. What are Filters in MVC?

- Filters are used for cross-cutting concerns.
- Types: Authorization, Action, Result, Exception.
- Example: [Authorize], custom logging filter via IActionFilter.

6. How do you handle exceptions globally in .NET Core?

- Use a custom middleware or built-in `UseExceptionHandler`.
- Can log error and return a custom error response.

- Example:

```
app.UseExceptionHandler("/Home/Error");
```

7. What is the role of API Gateway in Microservices?

- Central entry point for all services.
- Handles routing, authentication, rate limiting.
- Tool: Ocelot (ASP.NET Core-compatible).

8. What is the difference between ViewData, ViewBag, TempData?

- ViewData: Dictionary, per-request.
- ViewBag: Dynamic, per-request.
- TempData: Persists across one redirect, uses session behind the scenes.

9. How do you implement communication between Microservices?

- Synchronous: REST API calls via HttpClient or Refit.
- Asynchronous: Message brokers like RabbitMQ, Kafka.
- gRPC: High-performance RPC for internal services.

10. What are some best practices for Microservices?

- Keep services small and focused.
- Handle failures (retry, circuit breaker via Polly).
- Use health checks and logging.
- Secure APIs with OAuth/JWT.
- Maintain separate database per service.

11. Explain Code-First approach in EF Core

- Define models as C# classes.
- Run `Add-Migration` to create migrations.
- `Update-Database` to apply changes.
- Fluent API or Data Annotations to configure model.

Dotnet Core MVC + Microservices Interview Cheat Sheet

12. Explain how Redux works in a React application

- Redux has: Store (holds state), Actions (describe events), Reducers (update state).
- Component dispatches an action.
- Reducer processes it and updates state.
- Components subscribe to store for state updates.