**.NET**

Валидация модели

Worker Service

Source Generators (Generatoare de surse)

Dependency Injection

1. Ce reprezinta „Service”?
2. Ce este Dependency Injection? Care este scopul lui?
3. Mecanism care face obiectele usor cuplabile(слабосвязанными )
4. Care sunt metodele de injectare a unui service?
5. prin constructor
6. prin parametru catre o metoda
7. prin proprietate
8. Ce reprezinta IServiceCollection?
9. Reprezinta o colectie de servicii care vor fi inregistrate in DI Container. Transmitind astfel ciclul de viata a serviciului si parametrii sai.
10. Cum se manifesta Adaugarea in IServiceCollection? Pentru ce este necesar?
11. Pentru lucrarea cu timpul de viata al unui serviciu precum si cu dependentele acestor servicii.
12. Cum se adauga serviciile in IServiceCollection?
13. Prin metodele AddTransient, AddScoped sau AddSingletone
14. Unde Tservice reprezinta interfata sau clasa abstracta care va fi folosita pentru a inregistra dependenta iar Timplementation reprezinta clasa care va realiza aceasta dependenta.
15. Ce este un Middleware?
16. Componenta care se afla intre un web server si aplicatie,
17. Este implementat ca un lant de componente care proceseaza cererile HTTP de intrare si raspunsurile de iesire
18. Fiecare Middleware are rolul sau, una pentru logare, alta pentru prinderea exceptiilor etc...
19. Ce reprezinta Delegatul „RequestDelegate ‚next’”?
20. Legatura catre urmatorul component Middleware din lantul de componente. Lucreaza precum o lista inlantuita.
21. Cum sunt creati middleware? Exemple: Ce reprezinta .Use, .UseMiddleware?

Metodele Run si Run(code)

* Metoda Run este o modalitate pentru a rula aplicatia din asp.net core. Este folosita pentru a procesa cererea si a genera un raspuns pe baza parametrilor specificati.

**ASP.NET core**

Маршрутизация (Sistem de rutare) Definirea adreselor catre resurse

1. Utilizarea metodei .Map, .UseRouting, UseEndpoints-???? (Expired)
2. .Map este folosit pentru definirea rutelor mai simple
3. .UseEndpoints este folosit pentru definirea punctelor finale si rute mai complexe care proceseaza cereri intr-o aplicatie cu diferite ramuri din conducta (конвейера обработки запросов)
4. Versiunea pentru UseEndpoints deja s-a finalizat ( <https://metanit.com/sharp/aspnet6/3.1.php> )
5. Adaugarea unei restrictii pentru uitilizarea .Map (RouteOptions)

Статические файлы (Fisiere statice)

1. UseStaticFiles (Utilizeaza fisierele statice din catalogul indicat), HTML, css, js
2. UseDefaultFiles (Utilizeaza fisierele default din catalogul indicat), index.html, default.html, index.htm...
3. UseFileServer (Combina cele doua metode indicate mai sus si permite transmiterea codului de raspuns in cazul in care fisierul nu este gasit)

Конфигурация (Configurare) (Un set de optiuni utilizate pentru a personaliza comportamentul aplicatiei)

* Adaugarea dinamica a fisierelor Config / Json (   
  folosirea optiunilor  
  optional: true,   
  reloadOnChange: true)

1. Interfata IConfiguration (Ofera metode de a accesa valorile de configurare prin intermediul cheilor)
2. Interfata IConfigurationBuilder (Un builder pentru a crea o instanta a obiectului de configurare IConfiguration)
3. Interfata IConfigurationRoot (Folosit pentru a accesa valorile din alte surse si/sau dintr-un fisier, Fara obiectul builder creat nu va putea functiona)
4. Interfata IConfigurationSection (Reprezinta sectiunea din interfata Iconfiguration sau sectiunea de unde sa luam valorile datelor)
5. Adaugarea unui ConfigurationBuilder la WebApplicationBuilder (Ajuta la construirea si configurarea unei aplicatii web).
6. Exemple.

Логгирование (Logare) (Procesul de inregistrare a informatiei de diagnosticare si de monitorizare a aplicatiei)

1. Ce reprezinta logarea? (Procesul de inregistrare a informatiilor de la aplicatie in fisierele de jurnal)
2. Interfata Ilogger (Este folosit pentru a scrie loguri in aplicatie)
3. Interfata IloggerFactory (Este folosit pentru a configura logarea in aplicatie)
4. Crearea unui obiect de tip Logger folosind alta metoda.?
5. Configurarea si filtrarea Logurilor
6. UseHttpLogging middleware
7. Nivelele de logare

Обработка ошибок (Exception handling) (Procesul de gestionare a exceptiilor care apar in timpul rularii unei aplicatii)

1. De ce UseExceptionHandler dar nu Try-catch? (In ASP.NET sunt 2 tipuri de exceptii, fie si exceptia DivideByZero care este generate in timpul rularii programului, dar sunt si exceptiile protocolului HTTP care ar fi 404 not found sau 500 invalid request… etc…)
2. UseDeveloperExceptionPage (Reprezinta un middleware care atunci cand in timpul programului apare o exceptie si daca programul ruleaza in mod Debug, sa fie transferat la o pagina in care va afisa exceptia in detaliat)
3. UseExceptionHandler (Reprezinta un middleware care transmite utilizatorul la o adresa indicata, acest middleware permite prelucrarea exceptiei aparute {Se foloseste rar deoarece nu-i neaparat sa-i indicam utilizatorilor simpli ce fel de eroare a aparut})
4. UseStatusCodePages (Middleware care genereaza o pagina cu un text care reprezinta exceptia requestului userului... Aceiasi 404 NotFound)

Состояние приложения. Куки. Сессии

1. HttpContext.Items cu obiectul tipului
2. IDictionary<object, object>
3. Context.Request.Cookies[string key] (cea mai simpla cale de a salva datele unui utilizator)
4. Continutul la Context ???????
5. Request – Informatii despre cererea creata de utilizator
6. Response – Informatii despre raspunsul generat pentru utilizator
7. Session – Informatii despre sesiunea creata a utilizatorului
8. User – Alte informatii despre User
9. Items – Informatii despre
10. Cookie sunt maxim de 4096 bytes si se salveaza pe calculatorul userului dar se mai pot salva pe server
11. Pentru adaugare de Cookie se foloseste metoda ...Cookies.Append(string Tkey, string TValue) Pentru stergerea unui Cookie se foloseste ...Cookies.Delete(string TKey)
12. Session reprezinta un rand de requesturi intr-un singur browser intr-un anumit interval de timp   
    AddDistributedMemoryCache() // Adaugam serviciul IDistributedMemoryCache  
    AddSession() // Adaugam serviciile sesiunilor  
    UseSession() // Middleware care permite lucrul cu Sesiunea
13. Inregistrarea sesiunii in Session Storage

Results API

1. O colectie de clase si interfete care ne permite mai simplu si mai pe inteles de transmis un raspuns Utilizatorului cum ar fi:  
   Ok(), Problem(), Text(), Json() etc...
2. Clasa Results este o clasa statica si se apeleaza metodele prin:  
   Result.Ok(), Result.Text() etc...

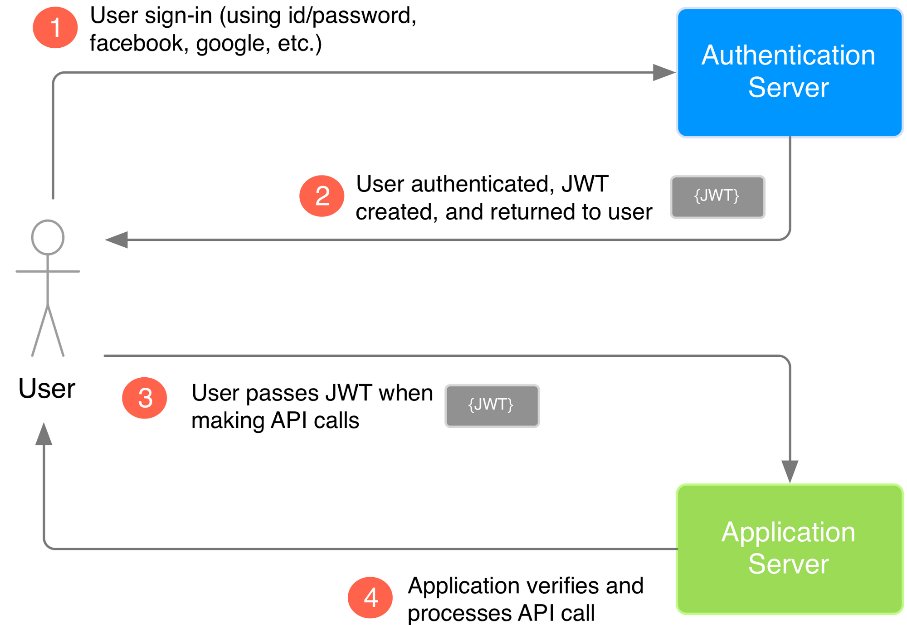
Web API

1. MapGet, MapPut, MapPost, MapDelete
2. Ce reprezinta Get, Post, Put, Delete?

Работа с базой данных и Entity Framework

1. Exemplu cu Oracle.ManagementDataAccess

Аутентификация и авторизация

1. Add/Use Authentication (Procesul prin care sistema verifica identitatea utilizatorului)
2. Add/Use Authorization (Procesul prin care sistemul decide daca acest utilizator are destule permisiuni pentru a accesa ceva)
3. Autentificarea/Autorizarea prin Cookie. Exemplu
4. Autentificarea/Autorizarea prin JWT Bearer. Exemplu   
   (Json Web Token)   
   🡪 Necesita descarcarea unui pachet aditional 🡨 Microsoft.AspNetCore.Authentication.JwtBearer
5. Header (заголовка) (JSON obiect care contine informatia despre token precum si metoda lui de creare)
6. Payload (полезной нагрузки) (JSON obiect care contine datele necesare pentru autoriazarea userului)
7. Signature (подписи или данные шифрования) (Validare a codului secret creat de catre Header si Payload)
8. Ce contine JWT?  
   Type, - reprezinta numele sau tipul informatiei spre exemplu: Name; ID; IPAddress; Role  
   Value, - reprezinta valoarea afirmatiei asociate cu claim spre exemplu, claimul de tip (rol: Admin) (nume: Tom)  
   Issuer, - reprezinta entitatea emisa de claim, poate fi un furnizor   
   Subject, - Poate fi o persoana sau o entitate la care se refera claim-ul  
   Expiration, - reprezinta Data si ora cand claimul expira si nu mai este valid  
   Proprietes – reprezinta alte informatii suplimentare care pot fi adaugate in claim spre exemplu, locatia sau contextul
9. Ce reprezinta Claim? (Afirmatie pentru indentitatea utilizatorului autentificat spre exemplu ca un dictionar fiind: „name” cu valoare „Tom” sau role cu valoarea admin )
10. ClaimsIdentity? (O identitate a unui utilizator)
11. ClaimsPrincipal(claimsIdentity object)? (Identitatea utilizatorului curent precum si claimurile legate de acest utilizator)
12. 

CORS и кросс-доменные запросы

1. Ce reprezinta Cors?
2. AddCors() // Service  
   UseCors() // middleware
3. Configurarea unui cors: (Se face in middleware (builder => {}))  
   AllowAnyOrigin – primeste requesturi de la oricare adresa  
   AllowAnyHeader - cererile cu orice antet sunt acceptate  
   AllowAnyMethod - cererile de orice tip (GET/POST) sunt acceptate  
   AllowCredentials - permis să accepte acreditări de la client (cum ar fi cookie-urile)  
     
   WithHeaders - sunt acceptate numai cererile care folosesc anumite antete  
   WithMethods - Sunt acceptate doar anumite tipuri de cereri  
   WithOrigins - sunt acceptate numai cererile de la anumite adrese  
   WithExposedHeaders - permite serverului să-și trimită anteturile către client

URL Rewriting

Мониторинг работоспособности приложения

**ASP.NET core MVC**

Контроллеры

Модели

View Component

Метаданные и валидация модели

Фильтры

Таким образом, IServiceCollection представляет описание сервисов, которые должны быть зарегистрированы в контейнере внедрения зависимостей, тогда как DI контейнер представляет реализацию контейнера, который фактически создает и управляет этими сервисами в приложении.