**Programmeerimine I osa: kodutöö - Middleware**

1. **Mis on Middleware?**

*Middleware* on tarkvara, mis toimib vahendajana kahe rakenduse või teenuse vahel, et hõlbustada nende suhtlust või ühenduvust (taotluste ja vastuste käitlemine). Iga komponent:

* Valib, kas edastada taotlus järgmisele komponendile.
* Oskab teostada tööd enne ja pärast *pipline* järgmist komponenti.

*Middleware* saab sisseehitada .NET Core raamistiku osana, lisada NuGeti pakettide kaudu või olla enda kohandatud.

Tänapäevases arenduses kasutatavad vahevara levinumad kasutusjuhud:

* Uute rakenduste arendamine
* Olemasolevate rakenduste optimeerimine
* Põhjalik integratsioon
* Rakenduste programmeerimisliidesed (API)
* Andmete voogesitus
* Äri automatiseerimine

1. **Missuguseid extensione meetodeid kasutab?**

* ***Run***- laiendusmeetod, mis lisab rakenduse päringuahelasse terminali vahevaradelegaadi. Lisatakse päringute konveieri lõppu.
* ***Map***- hargneb päringute kontuuri antud taotluste vaste põhjal. Kui taotlus algab antud teega siis haru täidetakse. Kasutatakse sissetulevate päringute alusel päringu ümbersuunamiseks erinevatele vahevara komponentidele.
* ***Use*-** lisab reas määratletud vahevara delegaadi rakenduse päringuahelasse. Saab lisada järgmise parameetri, mis kutsub järgmise päringu delegaadi ette. Täidab ülesandeid enne ja pärast järgmist delegaati, ühendades komponente.

1. **Iseloomusta Middleware käskude loogikat.**

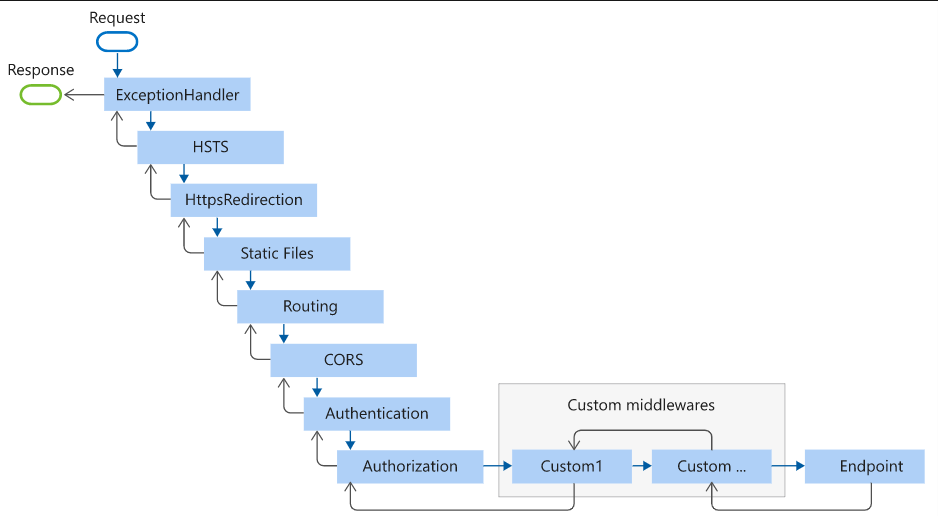
ASP.NET Core päringu konveier koosneb päringute delegaatide jadast, mida kutsutakse üksteise järel. Iga delegaat saab teha toiminguid enne ja pärast järgmist delegaati. Erandite käsitsemise delegaadid tuleks välja kutsuda varakult, et nad saaksid tabada erandeid, mis esinevad konveieri hilisemates etappides. Lihtsaim võimalik ASP.NET Core rakendus seadistab ühe päringute delegaadi, kes tegeleb kõigi päringutega. See juhtum ei hõlma tegelikku päringute jada kontuuri. Selle asemel kutsutakse vastuseks igale HTTP -päringule üks anonüümne funktsioon.



**4. Kuidas kutsutakse delegaati, kui see ei edasta päringut (request) järgmisele delegaadile?**

Kui delegaat ei edasta päringut järgmisele delegaadile, nimetatakse seda *short-circuiting the request pipeline.* Lühis on sageli soovitav, kuna see vädib tarbetut tööd.

**5. Kirjelda täieliku päringuprotsessi järjestust ASP.NET Core MVC näite puhul.**

****

*Vahevara* komponendid käivitatakse nende konveierile lisamise järjekorras ning tuleb jälgida, et need lisatakse õiges järjekorras. Vastasel juhul ei pruugi rakendus ootuspäraselt toimida. Esimene konfigureeritud *vahevara* on taotluse vastu võtnud, muudab seda (vajadusel) ja annab kontrolli järgmisele *vahevarale* üle. Lõpp-punkt käivitab filtrite jada vastava rakenduse tüübi jaoks (ressursifiltrid, mudeli sidumine, mudeli valideerimine, toimingufiltrid, erandifiltrid, tulemuste filtrid).

**6. Kuidas mõjutab middleware koodi järjestus komponeneti tööd?**

Vahevara komponentide Startup.Configure meetodisse lisamise järjekord määrab, millises järjekorras vahevara komponente päringutele kutsutakse ja vastus vastupidises järjekorras. See on turvalisuse, jõudluse ja funktsionaalsuse seisukohalt kriitilise tähtsusega .

**7. Mida teeb ?**

* *Developer Exception Page Middleware*- teatab rakenduse käitustõrkest
* *UseExceptionHandler*- püüab erandeid, mis on visatud järgmisesse middleware-i.
* *UseHsts*- lisab the Strict-Transport-Security päise.
* *UseHttpsRedirection*- suunab HTTP -päringud HTTPS -i.
* *UseStaticFile*- tagastab staatilised failid ja lühised edasiseks taotluste töötlemiseks.
* *UseCookiePolicy*- vastab rakendusele EL üldise andmekaitsemääruse (GDPR) määrustele.
* *UseRouting*- marsuudi taotlused.
* *UseAuthentication*- üritab kasutajat autentida enne, kui tal on lubatud juurdepääs turvalistele ressurssidele.
* *UseAuthorization*- annab kasutajale juurdepääsu turvalistele ressurssidele.
* *UseSession*- loob ja säilitab seansi oleku.
* *UseEndpoints koos MapRazorPages*- Razor Pagesi lõpp -punktide lisamiseks päringuahelasse.
* *UseSpaStaticFiles*- tavaliselt viimasel kohal, võimaldada kõigil teistel keskklassi esmalt vastata vastavatele päringutele. Lubada kliendipoolse marsruutimisega SPA-sid käitada kõigil marsruutidel, mida serverirakendus ei tunne.

**8 Valige välja viis Built-in Middleware ja iseloomustage neid.**

* *Diagnostics* **-** Mitu eraldi keskklassi, mis pakuvad arendajatele erandilehte, erandite käsitlemist, olekukoodi lehti ja uute rakenduste vaikeveebilehte. Enne komponente, mis tekitavad vigu. Terminal erandite jaoks või uute rakenduste vaikeveebilehtede teenindamine.
* *Health Check -*Kontrollib rakenduse ASP.NET Core tervist ja selle sõltuvusi, näiteks kontrollib andmebaasi saadavust. Terminal, kus päring vastab kontrolli lõpp- punktile.
* *Header Propagation* - Edastab HTTP päised sissetulevast päringust väljuvate HTTP kliendi päringuteni.
* *Session* - Pakub tuge kasutajaseansside haldamiseks. Enne komponente, mis nõuavad seanssi.
* *URL Rewrite* - Toetab URL -ide ümberkirjutamist ja päringute ümbersuunamist. Enne komponente, mis on vajalikud WebSocket'i päringute vastuvõtmiseks.

**Kasutatud kirjandus:**

* <https://docs.microsoft.com/en-us/aspnet/core/fundamentals/middleware/?view=aspnetcore-5.0>
* <https://www.c-sharpcorner.com/article/overview-of-middleware-in-asp-net-core/>
* <https://procodeguide.com/programming/aspnet-core-middleware/>
* <https://www.redhat.com/en/topics/middleware/what-is-middleware>