Serverside web develoment individueel  
Opdracht beschrijving

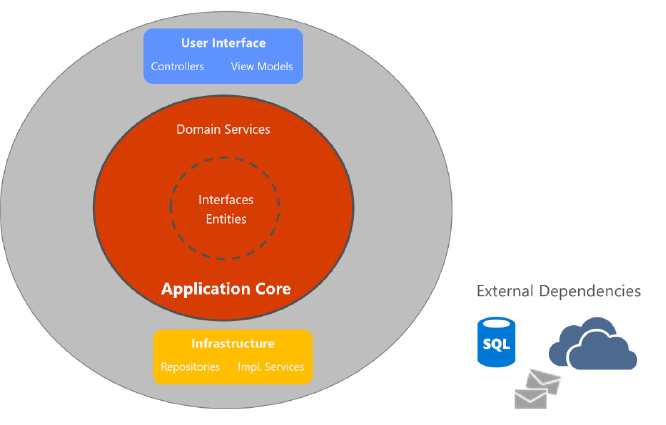
****

# Versiehistorie

|  |  |  |
| --- | --- | --- |
| *Datum* | *Versie* | *Wijziging* |
| 15-09-2019 | 1.0 | Initiële versie |
| 08-10-2019 | 1.1 | Na feedback studenten:   * Identity database toegevoegd in conceptuele opzet * Aan de toelichting op de conceptuele opzet informatie toegevoegd over de identity database |

# Inleiding

Aan het begin van periode IVT2.1 heb je kennisgemaakt met het ontwikkelen van een web-applicatie met behulp van het ASP.net MVC Core framework, en heb je ook een aantal vaardigheden geleerd op het gebied van UX-design. In Softwareaontwerp & -architectuur 2 komt aantal applicatie-architecturen aan bod, zoals de clean (onion) architecture (Smith, 2017). In deze architectuurstijl staat de *application core*, het domeinmodel met bijbehorende services, centraal, zoals Figuur 1 aangeeft. Ook het ontwikkelen van een RESTful web API komt daar aan bod. De opdracht bij het onderdeel ServerSide Web Development Individueel (EIIN-SSWPI) integreert de genoemde vakken.



In EIIN-SSWPI ontwikkel je een aantal web applicaties en maakt daarbij gebruik van een software-ontwikkelstraat met automatische tests en deployment.

# Doel

Het doel van deze opdracht is het ontwikkelen van een web-applicatie in ASP.net MVC Core waarbij je een UX-design maakt, een goede applicatie-architectuur toepast en waarbij je een ontwikkelstraat gebruikt, die automatisch de unit tests uitvoert en deployt. Tijdens het afrondend assessment demonstreer je je uitwerking en licht je gemaakte keuzes toe.

# Casus

EasyMeal is een maaltijdenservice waar je voor een hele week een warme maaltijd kunt bestellen, en heeft als motto: “Uw keuzevrijheid staat bij ons voorop”.

Wie zijn de actoren?

Er zijn koks in dienst. Hun naam is bekend in het systeem, net als hun e-mailadres en telefoonnummer (niet voor klachten uiteraard, maar vooral voor complimenten). Als klant heb je daarnaast nog een klantnummer, adresgegevens, mogelijke dieetbeperking(en). Omdat je een gratis maaltijd krijgt op je verjaardag, registreer je zeker ook je geboortedatum.

Wat zijn de user stories?

US\_01 Als kok wil ik het aanbod van maaltijden kunnen beheren, zodat ik klanten elke week een gevarieerd aanbod kan bieden waarbij rekening wordt gehouden met dieetbeperkingen.

Toelichting:  
Een maaltijd bestaat uit een voor-, hoofd- en nagerecht. Per dag van de week is er een andere maaltijd, waarbij elke dag een ander hoofdgerecht gekozen kan worden. Voor- en nagerechten hoeven niet per dag te variëren. Omdat we te maken hebben met klanten met dieetbeperkingen, is het belangrijk daar rekening mee te houden. Concreet gaat het om:

* Zoutloos
* Diabetes
* Glutenallergie

Het kan natuurlijk niet zo zijn dan een klant met een bepaalde dieetbeperking geen geschikte maaltijd kan kiezen in een week. Het is de verantwoordelijkheid van de kok om ervoor te zorgen dat alle klanten met een dieetbeperking in ieder geval één complete maaltijd per dag kunnen ‘kiezen’. Het systeem moet hier wel waarschuwingen voor geven als dit niet het geval is.

De kok moet de planning voor een bepaalde week dus uiterlijk zondagavond 23:59 de week van tevoren in het systeem invoeren. Zodra de klanten van dit menu kunnen kiezen, mogen de betreffende maaltijden niet meer aangepast worden.

Elke gerecht heeft vanzelfsprekend een naam, prijs en kunnen er dieetbeperkingen van toepassing zijn. Daarnaast is er een korte beschrijving en plaatje.

Bij het samenstellen van het maaltijdaanbod kunnen sommige gerechten die eerder op het menu stonden hergebruikt worden: om te voorkomen dat een kok elke week de gegevens van bijvoorbeeld tomatensoep, friet met biefstuk, spaghetti aglio e olio of vanille vla weer opnieuw in moet voeren, is het mogelijk gerechten te kunnen importeren in een nieuwe week.

US\_02 Als klant wil ik gebruik kunnen maken van een account met mijn e-mailadres als gebruikersnaam

Toelichting:

Een klant kan zich aanmelden waarbij zijn e-mailadres als gebruikersnaam dient. De gebruiker dient verder een wachtwoord op te geven en zijn naam, adres en geboortedatum. Ook kruist de gebruiker zijn dieetbeperkingen aan. Met uitzondering van het e-mailadres, zijn de gegevens nog te wijzigen.

US\_03 Als klant wil ik maaltijden kunnen bestellen voor de komende week, zodat ik elke dag waarop ik dat wil een compleet verzorgde maaltijd kan eten waarbij rekening wordt gehouden met mijn dieetbeperkingen.

Toelichting:

Een maaltijd bestaat uit 3 gangen, maar je hoeft ze niet alle te gebruiken. Het minimaal aantal gangen is 2 – je kunt óf het voorgerecht óf het nagerecht weglaten. Je bestelt overigens voor een hele week; een bestelling voor een losse dag kan helaas niet. Een week begint op maandag en eindigt op zondag. Natuurlijk kan het wel eens gebeuren dat je een dag wilt slaan. Het minimum aantal te bestellen maaltijden is voor 4 werkdagen en het weekend is optioneel. Voor een maaltijd kunnen grote eters kiezen voor de grote maaltijd (prijs +20%), en mensen die wat minder eten voor een kleine portie (prijs -20%). Standaard is de ‘medium size’.

Bij het bestellen moet de cliënt aan de bovenkant van het scherm het totale verschuldigde bedrag voor die week zien en een indicatie voor welke dagen hij al besteld heeft. Muteren van een reeds gemaakte menukeuze moet ook mogelijk zijn, zolang de sluitingstijd niet verstreken is. De sluitingstijd is op donderdagavond 23.59 uur, zodat de kok op vrijdag de ingrediënten voor de nieuwe week nog kan bestellen.

US\_04 Als klant wil ik per maand een factuur kunnen maken, zodat ik inzicht krijg in de gemaakte kosten

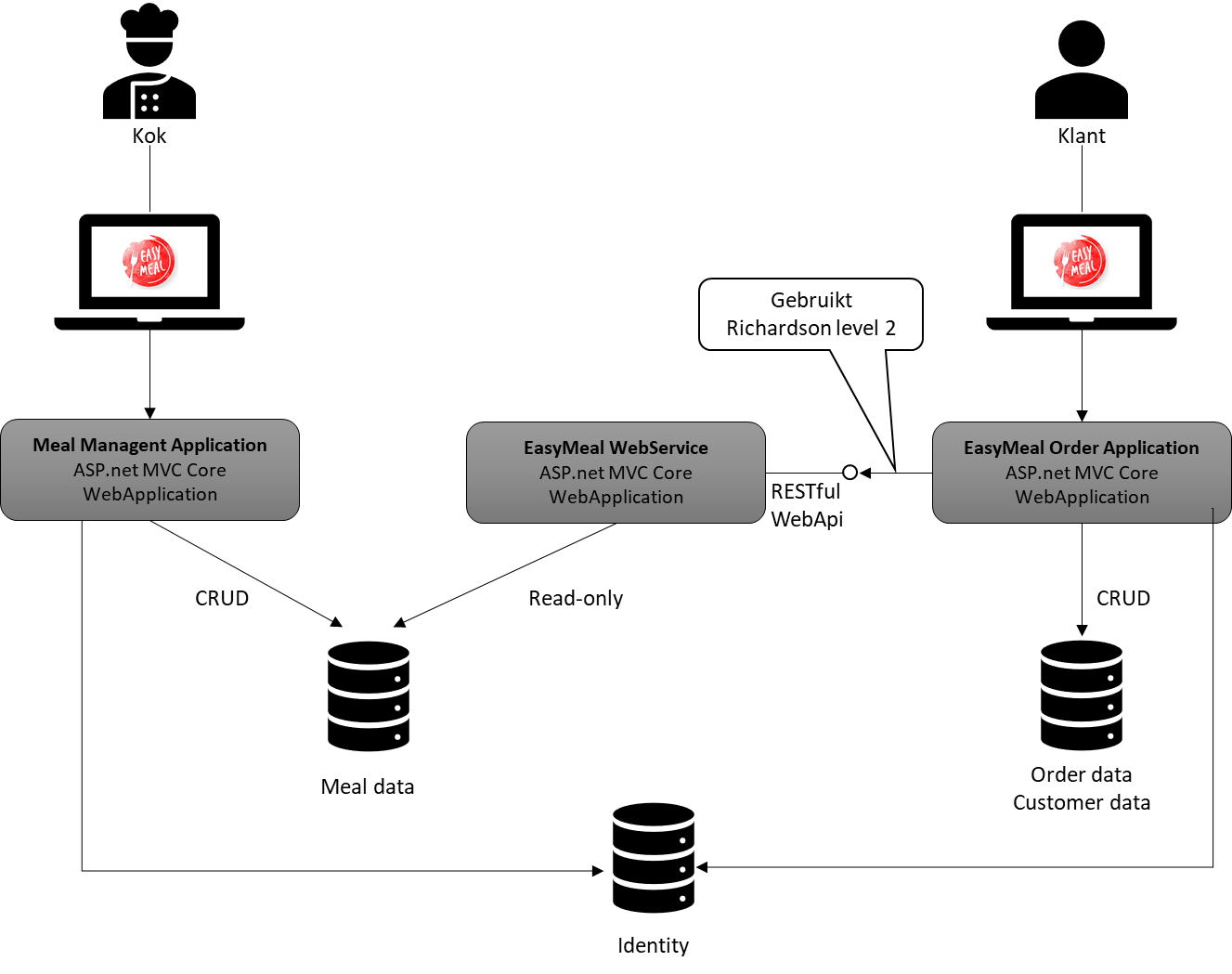
Toelichting:

Bij het maken van een bestelling, krijgt de klant een indicatie van de kosten te zien. Een factuur kan echter tot een ander totaalbedrag leiden, in het voordeel van de klant. Indien een klant een maaltijd had besteld voor zijn verjaardag, dan krijgt hij/zij deze gratis. Verder geldt dat bij 15 of meer bestelde maaltijden in die maand, de klant 10% korting krijgt. De mogelijk gratis maaltijd op de verjaardag, telt hiervoor niet mee. Kortingen zijn expliciet op in factuuroverzicht terug te vinden.

# Opdracht

Hieronder volgt de volledige opdrachtbeschrijving. Voor studenten die deze opdracht maken als herkansing uit voorgaande jaren, kan het zijn dat maar een deel van toepassing is. In het hoofdstuk Oplevering vind je wat van toepassing is voor jouw situatie.

De functionele requirements en business rules die van toepassing zijn, heb je kunnen lezen in de vorige paragraaf. Wat de niet-functionele eisen betreft, leggen we de focus op schaalbaarheid, onderhoudbaarheid en testbaarheid (zie ook Softwareontwerp & -architectuur). Een grove, conceptuele opzet voor de applicaties vind je in Figuur 1.



Figuur 1 Conceptuele opzet EasyMeal

De MealManagement applicatie is een ASP.net MVC Core applicatie die voor gegevens over maaltijden een SQL Server database gebruikt. Omdat we in de toekomst veel klanten verwachten, is er een aparte service waar het maaltijdenaanbod uit opgehaald kan worden, de EasyMeal WebService. Deze haalt gegevens uit dezelfde database als waar de Meal Management applicatie ze wegschrijft. De service biedt een RESTful Web API. Zo kunnen wij van EasyMeal niet alleen zelf een applicatie maken voor bestellingen,

maar kunnen andere partijen hier in de toekomst ook gebruik van maken. Verder kunnen we van de EasyMeal Web Service ook gemakkelijk meerdere instanties deployen om snelle response te garanderen. Een applicatie waarmee EasyMeal de bestellingen kan afhandelen is niet opgenomen in Figuur 1 en valt daarmee buiten de scope voor deze versie.

De EasyMeal Order application is de applicatie waarmee klanten maaltijden kunnen bestellen en facturen in kunnen zien. Deze applicatie haalt informatie over maaltijden op via de EasyMeal Web Service (en niet direct uit de database met de meal gegevens). De applicatie heeft zelf ook minimaal 1 SQL Server database voor order data en klantgegevens. Natuurlijk zal een order verwijzen naar data uit de meal database. **Het is belangrijk dat gegevens over maaltijden niet gedupliceerd worden in de database van de EasyMeal Order applicatie.** Dat zou alleen maar tot veel duplicatie en inconsistenties leiden…

Voor user accountmanagement en bijbehorende authenticatie en autorisatie gebruik je het Identity framework. De Meal Management Application als de EasyMeal Order Application gebruiken daarvoor dezelfde database. Daar hoef je niet aparte databases voor te maken. Er is natuurlijk een relatie tussen de IdentityUsers en de customer data in de EasyMeal Order database: die is op basis van e-mailadres. Een koppeling van de IdentityUser en je eigen customer in het domain is lastig in combinatie met de onion architectuur.

Ontwerp

De beschrijving van EasyMeal is vrij generiek. Je geeft zelf de opdracht meer kleur door een thema te kiezen, bijvoorbeeld voor maaltijden voor ouderen, Italiaanse maaltijden, alleen vegetarisch, super gezond enzovoort, en houd daarbij expliciet rekening met betrekking tot het UX-design. Je stuurt hiermee ook welke persona je van toepassing vindt en welke maatregelen je treft om een gebruiksvriendelijke user interface te maken. Voor ouderen gelden andere keuzes dan voor jongere doelgroepen! De uitwerking van het UX-design neem je op in het portfolio dat voor dat vak is voorgeschreven.

Je maakt ook een aantal (UML-)diagrammen:

* Package- en klassendiagram voor toepassing van clean (onion) architectuur.
* Componentdiagram voor het gehele systeem.
* Deploymentdiagram voor het gehele systeem.

**De diagrammen heb je zelf met de hand gemaakt en zijn dus niet gegenereerd  
vanuit de code door bijvoorbeeld VisualStudio.**

De EasyMeal Web Service biedt zijn functionaliteit aan via een correcte RESTful Web API en er zijn 2 varianten: de ene voldoet aan Richardson level 2 en de andere is op level 3. Beide voldoen dan ook aan de volgende criteria voor RESTful architectuur constraints (zie ook studiemateriaal SO&A 2):

|  |  |
| --- | --- |
| * Client/server | * Stateless communication |
| * Resources | * Resources with multiple representations |
| * Standard operations | * Hypermedia (HATEOAS), alleen voor Richardson level 3 |

Het ontwerp voor de RESTful Web API’s bestaat uit een schema met de het gedrag voor de CRUD operaties op de resources in termen van HTTPS verbs. Omdat we de interface read-only gebruiken, geldt voor C, U en D dat er vooraf gedefinieerde foutmelding geïmplementeerd zijn. Het leesgedrag wordt ondersteund voor zowel de verzameling resources als individuele. Je past de richtlijnen volgens Brian Mulloy van Apigee (Mulloy, 2012) toe op de Web API’s, daar waar het basisschema niet voldoende is. Je documenteert je Web API’s beide (level 2 + 3) met behulp van Swagger.

Code

Je programmeert je ontwerp in ASP.net MVC Core 2.2 en C#. De code is netjes: volgens coding guidelines, geen uitgecommentarieerde codeblokken, netjes uitgelijnd etc.

Je past de volgende concepten toe:

|  |
| --- |
| 1. Dependency injection d.m.v. een dependency injection container. |
| 1. Persistentie in de database d.m.v. entity framework code first en migrations. 2. Relaties tussen modellen d.m.v. navigational properties. 3. Het gebruik maken van repositories op basis van interfaces zodat we eenvoudig de manier van dataopslag kunnen wijzigen. 4. Het kunnen maken van validatieregels op de velden van de models. 5. Het gebruik van strongly typed views. 6. Het gebruik van Microsoft Identity voor authenticatie en autorisatie. 7. Het maken van een widget welke standaard op iedere pagina verschijnt. 8. Toepassen van Lambda expressies om selecties te maken op datasets. |

Testen

Voor de business rules van het domein implementeer je unit testen. Je test dus geen standaard get-ers en set-ers van klassen waarbij ‘gewoon’ waardes direct worden teruggegeven of opgeslagen. Je past ook mocking toe. Het gaat minimaal om de volgende business rules:

1. Er is voor elke dieetbeperking minimaal één volledige maaltijd te bestellen voor elke dag van de week waarvoor de bestelling geldt.
2. Het minimum aantal te bestellen maaltijden voor de werkdagen van een week is 4; daarnaast zijn de dagen van het weekend beide optioneel.
3. Bij een weekbestelling hoort een indicatie voor het verschuldigde bedrag. Dat is nog zonder eventueel te verrekenen kortingen; die kunnen pas na het verstrijken van de betreffende maand verrekend worden.
4. Bij het opstellen van de factuur voor een maand, gelden kortingen: maaltijd op verjaardag is gratis; 10% korting bij 15 of meer bestelde maaltijden en de gratis maaltijd op verjaardag telt hierin niet mee.

De RESTful Web API van de EasyMeal Web Service test je met behulp van een aantal collections in Postman of Swagger.

Overig

Je werkt volgens continuous deployment waarbij je gebruik maakt van een development pipeline (ontwikkelstraat) die automatisch een build start wanneer je wijzigingen levert (‘pusht’), daarna ook automatisch de unit testen uitvoert en deployt. De gerealiseerde user interface is consistent met het UX-design waarbij je de design principes uit UX-design 1 toepast.

Wat je op dient te leveren en wat beoordeeld wordt, hangt af van je cohort en welke vakken je nog open had staan uit een ouder examenprogramma (zie ook periodewijzer). Als je aan het vak UX-design 1 deelneemt, kijk dan in het materiaal van dat vak op BB voor meer informatie; voor de andere vakken kijk je in het hoofdstuk hieronder dat op jou van toepassing is.

# Oplevering cohort 2018, of cohort 2015 en ouder

Indien je een student bent van cohort 2015 of eerder en je hebt IVH7 IMINBOUWWEBAPP-A Bouw van een web-applicatie nog open staan, dan is het onderstaande ook op jouw situatie van toepassing.

**Welke onderdelen zijn van toepassing?**

Alle.

**Wat lever je in?**

Lever de opdracht in via de Blackboard inleverlink; daar staat ook de deadline voor inleveren vermeld.

De ingeleverde opdracht dient te bevatten:

* pdf met (UML-)diagrammen en verdere onderbouwing.
* Gehele Visual Studio solution met alle projecten, zonder obj-files, binaries en packages folders. Gebruik de functie ‘Clean Project’ uit het menu en verwijder eventueel de packages folder!
* URL van de gedeployde applicatie op Azure zodat de applicate bezocht kan worden.
* Een account (gebruikersnaam én wachtwoord) waarmee de docent kan inloggen op jouw applicatie in Azure.
* Swagger API documentation
* Postman collection

Bovenstaande producten ingepakt in 1 zipfile met de volgende naam:

* <klas> - <voornaam + achternaam> - IVT2-1-EasyMeal.zip

# oplevering cohort 2016 + 2017

**Welke onderdelen zijn van toepassing?**

Dat hangt af van welke onderdelen je nog hebt openstaan van IVT2.1 uit 2017/2018 of 2018/2019:

* **EIIN-SSWPRI Server-side web programming individueel**  
  Dit is de ‘oude ASP.net’, waarin je zowel theorietoetsen had en er een opdracht was.  
    
  Je maakt alleen de De MealManagement applicatie (het linker deel van Figuur 1), dus de applicatie die de kok gebruikt.

|  |  |
| --- | --- |
| **Requirements** | US\_01, met daarbij accounts en inlogfunctionaliteit voor koks met behulp van Identity Framework. |
| **Design** | In het ontwerp neem je alleen een klassendiagram op van het domeinmodel. Dit diagram genereer je niet vanuit je code, maar maak je zelf. |
| **Code** | Helemaal van toepassing. |
| **Testen** | Alleen unit testen op de business rules (BU\_1 t/m BU\_4). |
| **Overig** | Helemaal van toepassing. |

* **EIIN-SOFA2 Softwareontwerp & -architectuur 2**  
  Je maakt het rechter deel van Figuur 1: EasyMeal Order application en de EasyMeal Web Service (met de RESTful WebAPI). Je hoeft geen meal management applicatie te maken waarmee je maaltijden in de Meal Database beheert. Je mag daarvoor direct data in de database invoeren, of als je dat handiger vindt mag je ook een heel eenvoudige applicatie daarvoor maken, maar daar ligt niet de focus op.

|  |  |
| --- | --- |
| **Requirements** | US\_03 |
| **Design** | Helemaal van toepassing. |
| **Code** | Items 1, 2, 3, 4 en 6 zijn verplicht; de overige optioneel |
| **Testen** | Alleen de RESTful Web API van de EasyMeal Web Service met behulp van Postman. |
| **Overig** | Helemaal van toepassing. |

* **EIIN-SSWPRI Server-side web programming individueel +  
  EIIN-SOFA2 Softwareontwerp & -architectuur 2**  
    
  Alle onderdelen zijn van toepassing.

**Wat lever je in?**

Lever de opdracht in via de Blackboard inleverlink; daar staat ook de deadline voor inleveren vermeld.

De ingeleverde opdracht dient te bevatten:

* pdf met (UML-)diagrammen en verdere onderbouwing.
* Gehele Visual Studio solution met alle projecten, zonder obj-files, binaries en packages folders. Gebruik de functie ‘Clean Project’ uit het menu en verwijder eventueel de packages folder!
* URL van de gedeployde applicatie op Azure zodat de applicate bezocht kan worden.
* Indien van toepassing: een account (gebruikersnaam én wachtwoord) waarmee de docent kan inloggen op jouw applicatie in Azure.
* Indien van toepassing: Swagger API documentation en Postman collection

Bovenstaande producten ingepakt in 1 zipfile met de volgende naam:

<klas> - <voornaam + achternaam> - IVT2-1-EasyMeal-cohort1617.zip

# Bibliografie

Mulloy, B. (2012). *Web API Design.* unknown: Apigee. Opgehaald van https://pages.apigee.com/rs/apigee/images/api-design-ebook-2012-03.pdf

Smith, S. (2017). *Architecting Modern Web Applications with ASP.NET Core and Azure.* Redmond, Washington: Microsoft Corporation.