|  |  |
| --- | --- |
| D:\SYNTRA\FRONTENDEV\MODULE-LERENPROGRAMMEREN\Algemeen\Syntra_Templates\SyntraWest-logo.jpg | **Datum:** 15/06/2020  **Opleiding**: C# Programmeur  **Lesmodule**: Programmeren 2  **Schriftelijk EXAMEN C# Programmeren 2** |
| **EindTotaal**: / 50 | **Naam**: |

# **DEEL 1 : Schriftelijk**

# **(gedeelte op papier en gedeelte met laptop en internet)**

**Beantwoord de opgaven in C#. Maak een Folder Ex\_Programmeren2\_FamilieNaam\_Voornaam (Maak binnen deze folder voor elke Programmeer opgave een aparte solution). Maak je projecten in .Net Core 3.1**

**Om het resultaat in te dienen, comprimeer de volledige solutions folders en upload dit via het leerplatform (moodle) vóór het verstrijken van de deadline.**

# Kennistest

**Beantwoord op papier de volgende vragen:**

1. Leg in je eigen woorden uit MVC pattern. Waarom zou je dit gebruiken?
2. Wat is een extension method, wanneer zou je dit nodig hebben. Zet de volgende code om naar een extension method die hetzelfde doet:

public int AantalKeerVoorkomendeString(List<string> lijst, string zoekterm)

{

return

(from word in lijst

where word.ToLower() == zoekterm.ToLower()

select word).Count();

}

1. Wat is een Generic class? Waarvoor dient dit? Wanneer zou je dit gebruiken? Zet de volgende code om naar de Generieke versie (haakjes niet belangrijk):

public class Factory

{

public Cat CreateCat()

{

return new Cat();

}

public Dog CreateDog()

{

return new Dog();

}

}

1. Wat is Dependency Injection? Op welke manier hebben we dit toegepast in tijdens de lessen?

# OPGAVE programmeren in C#: MVC Web App ‘FamilieNaam\_VoorNaam\_Kadoo’ (deel 1: papier – deel 2: met laptop en internet)

1. **Maak op papier** een schema van een **conceptueel database model** (diagram) (Entiteiten, relaties tussen de entiteiten (1 to 1, 1 to many of many to many) en de attributen bij elke entiteit).( Af te geven **vóór je start met programmeren**)
2. **Schrijf op je eigen laptop, nadat je het conceptueel database ontwerp op papier hebt afgegeven**:

**Een ASP.NET Core 3.1** **MVC Web app in Visual Studio 2019 met naam '** **FamilieNaam\_VoorNaam\_KADOO'**

**Gebruik een Sql Server database, Entity Framework Core 3.1 als ORM en Razor Views voor de CRUD pagina’s in je project.**

**Gebruik voor authenticatie en authorisatie ASP.NET Core Identity**

Je mag kiezen of je EF Core **database first of code-first methode** (of een combinatie van deze) **gebruikt**.

**Opgave:**

Maak een **MVC** Web App die gadgets(kattebelletje, Travel-scheerapparaat,..) kan beheren voor de firma ‘**Kadoo’**.

Deze **producten, evenals hun categorieën en subcategorieën** kunnen via deze web app worden beheerd (lijsten raadplegen, details bekijken , toevoegen, wijzigen, verwijderen).

In **elke categorie zijn er 1 of meer subcategorieën** bv

**Categorie ‘Huisdieren’**:

**Subcategorieën**: honden, katten, knaagdieren,…

**Categorie ‘Reizigers’**:

**Subcategorieën:** Travel-ToiletArtikelen, Travel-electro, …

**Categorie ‘Kinderen’**:

**Subcategorieën:** peuters, kleuters, basisscholieren, tieners,…

In **elke subcategorie zijn er 1 of meer producten** (bv **Categorie** Huisdieren/ **subcategorie** Katten: **Producten**: kattebelletje, fopmuis,…

Een **product** kan **maar tot 1 subcategorie behoren** bv product Travel-scheerapparaat behoort enkel tot subcategorie Travel-electro

Voorzie per **product minstens deze velden**:

* Unieke Id
* Naam
* Beschrijving
* AantalInVoorraad

Voor een **Categorie** voorzie **minstens** deze velden:

* Unieke Id
* Naam (uniek)
* Beschrijving

Voor een **SubCategorie** voorzie **minstens** deze velden:

* Unieke Id
* Naam

**Nadat je het conceptueel database ontwerp op papier hebt afgegeven, mag je je eigen laptop nemen**:

1. Maak een **Sql Server** **database ‘FamilieNaam\_VoorNaam\_KadooDb’** om de Categorieën, SubCategorieën en Producten bij te houden.
2. **Maak Models, Controllers en Razor Views (MVC):**

* Maak de nodige **Model** classes aan
* Maak een **CategorieController** met CRUD functionaliteit (Lijst van categoriën, Creatie van een nieuwe categorie, Details van een Categorie, Wijzigen van een bestaande categorie en Verwijderen van een Categorie) en voorzie bijhorende Razor views Index, Create, Details, Edit en Delete.
* Maak een **SubCategorieController** met CRUD functionaliteit en bijhorende Razor views
* Maak een **ProductController** met CRUD functionaliteit en bijhorende Razor views

1. Voeg **Authenticatie** toe:

* Gebruik **ASP.NET Core Identity** en maak de nodige wijzigingen en toevoegingen in je app
* Voeg een **AccountController** toe met functionaliteit met bijhorende razor views om een nieuwe gebruiker te **registreren**, in te **loggen en uit te loggen** van een gebruiker

3. Voeg **Authorisatie** toe:

* De **Index** (lijst) en **Details** van Categorieën, Subcategorieën en Producten, zijn **voor iedereen toegankelijk en zichtbaar.**
* Het **toevoegen, wijzigen en verwijderen van categorieën, subcategorieën en producten** mag **enkel door ingelogde gebruikers** worden uitgevoerd. Zet de authorisatie op het niveau van de Controllers en action methoden
* Wat zou je nog moeten veranderen/toevoegen aan de Web App om enkel de gebruikers met rol “Admin” toe te laten om producten, categorieën en subcategorieën te verwijderen? (Code niet vereist, mag tekstueel antwoord zijn)