**Virtuele Dierentuin - Designdocument**



**Esad Hamza Caliskan 1206800**

**Shailo Douglas S1157233**

**Inleiding**

Voor deze eindopdracht wordt een virtuele dierentuin ontwikkeld. Dit project maakt gebruik van de programmeertaal C# in combinatie met ASP.NET Core MVC om een webapplicatie te bouwen die dieren, verblijven en categorieën beheert. Het doel van de applicatie is om gebruikers in staat te stellen een dierentuin te organiseren en te beheren. Daarnaast moeten acties zoals zonsopgang, zonsondergang en etenstijd worden gesimuleerd.  
De opdracht is bedoeld om aan te tonen dat de basisprincipes van programmeren in C# beheerst worden, evenals het toepassen van frameworks en tools zoals Entity Framework Core en Razor Views.

**Doel van de Applicatie**

De virtuele dierentuin biedt een intuïtieve manier om dieren, verblijven en categorieën te beheren. Gebruikers kunnen nieuwe dieren en verblijven toevoegen, gegevens bewerken of verwijderen, en specifieke acties uitvoeren. Denk hierbij aan het controleren welke dieren wakker worden bij zonsopgang, wat dieren eten tijdens etenstijd of het controleren of een verblijf voldoet aan de vereisten voor de dieren die erin geplaatst zijn.  
Daarnaast bevat de applicatie een zoekfunctionaliteit waarmee gebruikers snel gegevens kunnen vinden. De applicatie is zowel functioneel als schaalbaar opgezet, zodat toekomstige uitbreidingen eenvoudig mogelijk zijn.

**Functionaliteiten**

**Dierenbeheer**

In de applicatie is het mogelijk om dieren toe te voegen, te bewerken en te verwijderen. Elk dier heeft een aantal eigenschappen, zoals de naam, soort, grootte, voedingsklasse en activiteitspatroon. Deze eigenschappen worden opgeslagen in de database en zijn via de webinterface zichtbaar.  
Bij het toevoegen van een dier kan een categorie worden gekozen, bijvoorbeeld "roofdieren" of "planteneters". Daarnaast biedt de applicatie een zoekfunctie waarmee dieren snel kunnen worden gefilterd op naam, soort of categorie.

Voorbeeld van eigenschappen van een dier:

* **Naam:** De naam van het dier (bijvoorbeeld "Tijger").
* **Soort:** De diersoort (bijvoorbeeld "Panthera tigris").
* **Grootte:** Hoe groot het dier is (bijvoorbeeld "Large").
* **Voedingsklasse:** Wat het dier eet (bijvoorbeeld "Carnivoor").
* **Activiteitspatroon:** Of het dier actief is overdag, 's nachts of altijd.

**Verblijvenbeheer**

Verblijven zijn een belangrijk onderdeel van de dierentuin. In de applicatie kunnen gebruikers verblijven toevoegen, bewerken of verwijderen. Elk verblijf heeft specifieke eigenschappen, zoals de naam, het klimaat (bijvoorbeeld "Tropisch"), het type habitat (bijvoorbeeld "Bos" of "Water") en het beveiligingsniveau.  
Een unieke functionaliteit van verblijvenbeheer is het uitvoeren van acties zoals zonsopgang, zonsondergang en etenstijd:

* **Zonsopgang:** Toont welke dieren in het verblijf wakker worden of gaan slapen.
* **Zonsondergang:** Toont welke dieren in het verblijf actief worden of gaan slapen.
* **Etenstijd:** Laat zien wat elk dier eet.

Deze acties helpen de gebruiker om inzicht te krijgen in het gedrag van de dieren in de verblijven.

**Categorieënbeheer**

Om de dieren beter te organiseren, kunnen ze worden ingedeeld in categorieën. Een categorie kan bijvoorbeeld "Herbivoren" of "Carnivoren" zijn. Categorieënbeheer biedt de mogelijkheid om categorieën toe te voegen, te bewerken of te verwijderen. Daarnaast kunnen dieren eenvoudig worden gekoppeld aan een categorie.  
In de webapplicatie is een filterfunctie beschikbaar waarmee de gebruiker dieren kan bekijken die tot een specifieke categorie behoren.

**Check Constraints**

Een belangrijke functionaliteit is de controle op regels binnen de dierentuin. Deze **Check Constraints**-functionaliteit controleert:

1. Of een verblijf groot genoeg is voor alle dieren die erin geplaatst zijn.
2. Of dieren met conflicterende dieetvoorkeuren niet samen in één verblijf zitten (bijvoorbeeld roofdieren samen met herbivoren).

Deze controles worden automatisch uitgevoerd en helpen de gebruiker om fouten in de indeling van de dierentuin te voorkomen.

**Technische Specificaties**

De applicatie is gebouwd met ASP.NET Core MVC en maakt gebruik van de volgende technologieën:

* **C#** als programmeertaal.
* **Entity Framework Core** voor databasebeheer.
* **SQLite** als database.
* **Razor Views** voor de presentatie van gegevens.

De database wordt opgezet met behulp van migraties en bevat seed data. Dit betekent dat bij het starten van de applicatie automatisch enkele dieren, verblijven en categorieën worden toegevoegd.  
De applicatie is opgezet volgens het **MVC-designpattern**:

1. **Model:** Bevat de gegevensstructuren (zoals Dieren, Verblijven en Categorieën).
2. **View:** Zorgt voor de presentatie in de browser.
3. **Controller:** Verwerkt gebruikersacties en communiceert met het model en de view.

**Testen**

De applicatie is uitgebreid getest op functionaliteit. Voor elk onderdeel zijn handmatige tests uitgevoerd:

1. **Dierenbeheer:** Het toevoegen, bewerken, verwijderen en zoeken van dieren is succesvol getest.
2. **Verblijvenbeheer:** Het beheren van verblijven en het uitvoeren van acties zoals zonsopgang en etenstijd werken correct.
3. **Categorieënbeheer:** Het toevoegen, bewerken en koppelen van categorieën aan dieren werkt naar verwachting.
4. **Database:** De migraties en seed data zijn succesvol uitgevoerd, en gegevens worden correct opgeslagen en opgehaald.

**Conclusie**

De Virtuele Dierentuin is een volledige webapplicatie die voldoet aan alle gestelde eisen. De gebruiker kan eenvoudig dieren en verblijven beheren, acties uitvoeren en de dierentuin organiseren. Door het gebruik van ASP.NET Core MVC en Entity Framework is de applicatie schaalbaar en eenvoudig uitbreidbaar.