Eindopdracht PROG6

1718

Hotel Tamagotchi

"De eindopdracht voor PROG6 is een assessment waar je in duo's een hotel applicatie voor kleine monsters, Tamagotchi's."

Inhoud

1. Introductie

Dit document beschrijft de eindopdracht voor PROG6. Deze eindopdracht dient in duo's gemaakt te gaan worden. Vooraf dient bij je docent bekend te zijn in welke duo's er gewerkt wordt.

In de tentamenweek zal er een assessment plaats vinden. Tijdens dit assessment ga je als **duo** je eindopdracht aan de docent demonstreren en verdedigen. De docent beoordeelt de eindopdracht en hier uit volgt een cijfer.

Je gaat een webapplicatie maken voor een **hotel**. In dit hotel zitten geen normale mensen maar kleine monsters genaamd Tamagotchi's. De webapplicatie dient ontwikkeld te worden in ASP.NET MVC C# en Visual Studio (zie "Technische eisen").

Wat was een Tamagotchi ook al weer?
https://nl.wikipedia.org/wiki/Tamagotchi

---

2. Functionele eisen

De set van functionele eisen zijn opgedeeld in de eisen omtrent Tamagotchi's, hotelkamers en overnachtingen.

2.1 Tamagotchi

Er moet een Tamagotchi CRUD beschikbaar zijn om Tamagotchi's te beren.

1. Het moet mogelijk zijn een lijst van Tamagotchi's te bekijken
    - Maak hier een duidelijke onderscheiding tussen dode en levende Tamagotchi
2. Het moet mogelijk zijn om een nieuwe Tamagotchi aan te maken
3. Het moet mogelijk zijn een Tamagotchi te verwijderen

Een Tamagotchi heeft de volgende eigenschappen

- Een Naam – een String
    - Maximale lengte van tien
- Leeftijd – een getal – Beginwaarde = 0
    - Aantal nachten dat de Tamagotchi oud is
- Centjes - een getal – Beginwaarde = 100
    - Altijd een positief getal of nul (>= 0)
- Level – een getal – Beginwaarde = 0
- Gezondheid – Een getal – Beginwaarde = 100
    - Waarde van 0 t/m 100
- Verveling – een getal – Beginwaarde = 0
    - Waarde van 0 t/m 100
- Levend of dood – een boolean

2.2 Hotelkamers

Het hotel is niet zo maar een hotel. Er zijn speciale kamers waar onze Tamagotchi kunnen overnachten. Een verblijf in een speciale kamer heeft invloed op de eigenschappen van de Tamagotchi.

1. Met behulp van de webapplicatie moet het mogelijk zijn voor gebruikers (het hotelpersoneel) om hotelkamers in te voeren.
2. Voor gebruikers moet het ook mogelijk zijn ingevoerde hotelkamers te wijzigen.
3. Voor gebruikers moet het ook mogelijk zijn hotelkamers te verwijderen.
4. Het hotel heeft **minimaal 4** hotelkamers ter beschikking

Een hotelkamer heeft de volgende eigenschappen:

- Een kamer heeft een grootte (aantal bedden) met de waarde 2, 3 of 5.
- Er zijn vier verschillende type hotelkamers
  - De rustkamer
  - De vechtkamer
  - De gamekamer
  - De werkkamer

2.3 Overnachten

Elke dag is het mogelijk voor een Tamagotchi om een kamer te boeken voor een nacht. De gebruiker van de applicatie kan dan via een knop aangeven dat de overnachting kan beginnen. Hierna begint de volgende dag en begint het hele riedeltje weer opnieuw.

2.3.1 Boekingen

Het moet mogelijk zijn voor de gebruiker van de applicatie om een kamer te reserveren door een boeking aan te maken. Dit kan alleen voor kamers waar nog geen boekingen voor zijn. Hierbij zijn de volgende stappen belangrijk.

1. Selecteer een kamer die nog niet bezet is (nog geen boeking)
2. Geef aan met hoeveel Tamagotchi's je wilt overnachten
3. Kies met welke Tamagotchi je wil overnachten
4. Toon een overzicht van de kamer en de Tamagotchi's
5. De gebruiker kan de boeking nu bevestigen
6. De boeking moet pas worden doorgevoerd als de gebruiker bevestigd heeft.

2.3.2 De nacht

Als de gebruiker klaar is met het aanmaken van alle boekingen kan hij aangeven dat de overnachting kan beginnen. Maak één grote knop 'Begin nacht'. Na het indrukken van de knop worden de eigenschappen van alle **levende** Tamagotchi's aangepast. Eerst gebeuren de standaard mutaties en daarna de mutaties die afhankelijk zijn van de kamer waarin de Tamagotchi overnacht. Hieronder volgen eerst de standaard mutaties en daarna de mutaties per kamer.

- Verhoog het level van de Tamagotchi's met 1
- Als verveling groter dan of gelijk aan 70 → Gezondheid -20
- Als gezondheid gelijk aan 0 → Tamagotchi is **dood**

2.3.3 De kamers

Tijdens de nacht zal afhankelijk van het kamertype waarin de Tamagotchi overnacht het volgende gebeuren:

De rustkamer
Een rustgevende maar saaie kamer:

- Kosten: 10 centjes
- Effect: + 20 Gezondheid
- Effect: + 10 Verveling

De gamekamer
De kamer voor de Tamagotchi die zich vervelen:

- Kosten: 20 centjes
- Verveling = 0

De werkkamer
De werkkamer is saai maar levert centjes op:

- Effect: +10-60 centjes (random)
- Effect: +20 Verveling

De vechtkamer
In deze speciale kamer vechten de Tamagotchi voor geld en eer! Als er meer dan één Tamagotchi in de kamer is, wint er één random Tamagotchi het 'gevecht'. Dit is de winnaar en de overige Tamagotchi zijn allemaal verliezers.

- Kosten: Gratis (nul centjes)
- Effect-winnaar: +20 centjes per verliezer
- Effect-winnaar: +1 Level
- Effect-verliezer: -20 centjes
- Effect-verliezer: -30 Gezondheid

Geen kamer
Buiten slapen is maar niks voor een Tamagotchi.

- Effect: -20 Gezondheid
- Effect: +20 Verveling

Verzin zelf een kamertype
Ontwikkel zelf een kamertype met leuke effecten op de overnachtende Tamagotchi!

- Kosten: ???
- Effect: ???

3. Technische eisen

De volgende basis technische eisen gelden voor de eindopdracht:

3.1 Basiseisen

- De data omtrent Tamagotchi's, hotelkamers en boekingen moet geregistreerd worden in een lokale Microsoft SQL database.
- De webapplicatie moet gemaakt worden in ASP.NET MVC.
- Voor data access moet er gebruik gemaakt worden van de Entity Framework.

3.2 Unit testing

- Unit Testing van business & controller logic
- Test minimaal de spelregels van de kamers en de validatieregels van de kamers, Tamagotchi's en boekingen.

3.3 Design Patterns

- Implementatie van een Repository om je Database Context te verbergen.
- Gebruik maken van Dependency Injection (DI) om het mogelijk te maken een Repository in een controller te injecteren voor het unit testen.
    - **Tip**: Gebruik 2 constructors per controller.

---

4. Extra's

- Gebruik een DI container voor het aanmaken van controllers en Repositories.
- Maak gebruik van een WCF service voor het uiteindelijk doorvoeren van de mutaties die plaatsvinden in de nacht.
- Voeg autorisatie en authenticatie toe met 2 verschillende rollen:
    - Rol hotelmedewerker
        - Mag kamers beheren
    - Rol eindgebruiker
        - Mag Tamagotchi's beheren die hij zelf heeft aangemaakt
        - Mag boekingen beheren

5. Checklist

De onderstaande lijst zal gebruikt worden tijdens het nakijken van je opdracht. Als je alle knock-out criteria hebt behaald krijg je al het cijfer 4. De rest van de punten ontvang je bij het behalen van de overige criteria. Je moet een voldoende halen met de basis criteria voor je in aanmerking komt voor de bonus criteria.

#	Beschrijving
Knock-out criteria - Tot de 4	
K.O.	Het moet mogelijk zijn om Kamers te beheren
	Het moet mogelijk zijn om Tamagotchi te beheren
	Het moet mogelijk zijn boekingen aan te maken voor 1 Tamagotchi
	Er zijn werkende unit test aanwezig met een minimale code coverage van 25%
	Het moet mogelijk zijn om 1 nacht op de juiste manier te laten verlopen
Basis criteria - Tot de 7	
Functioneel	Als gebruiker wil ik geen foute input kunnen geven of validatie meldingen zien als ik dat wel doe
	Als gebruiker wil ik een overzichtelijk scherm met al mijn levende Tamagotchi
	Als gebruiker wil ik dat de nacht op de juiste manier verloopt
	Als gebruiker wil ik dat boekingen op de juiste manier verlopen
Unit Testing	Als ontwikkelaar wil ik een goede code coverage door unit testing
	Als ontwikkelaar wil ik goede en slimme unit test
patterns	Als architect wil ik de database ontsluiten via Repository
	Als architect wil ik controllers ontkoppelen via depency injection
	Als architect wil ik ook andere klasse ontkoppelen via depency injection
MVC	Als architect wil ik dat er op de juiste momenten gebruik is gemaakt van ViewModels
	Als security manager wil ik validatie op alle formulieren via 'Validatable objects'
	Als architect wil ik dat de verantwoordelijkheden netjes zijn verdeeld over verschillende controllers
Zilver criteria - Tot de 8	
Extra: Zilver	Als security manager wil ik autorisatie en authenticatie met 2 verschillende rollen
Goud criteria - Tot de 10	
Extra: Goud	Als architect wil ik dat het aanmaken van controllers verloopt via een DI container
	Als architect wil ik dat het muteren van Tamagotchi's in de nacht wordt geregeld via een WCF service.