Campus Nyköping

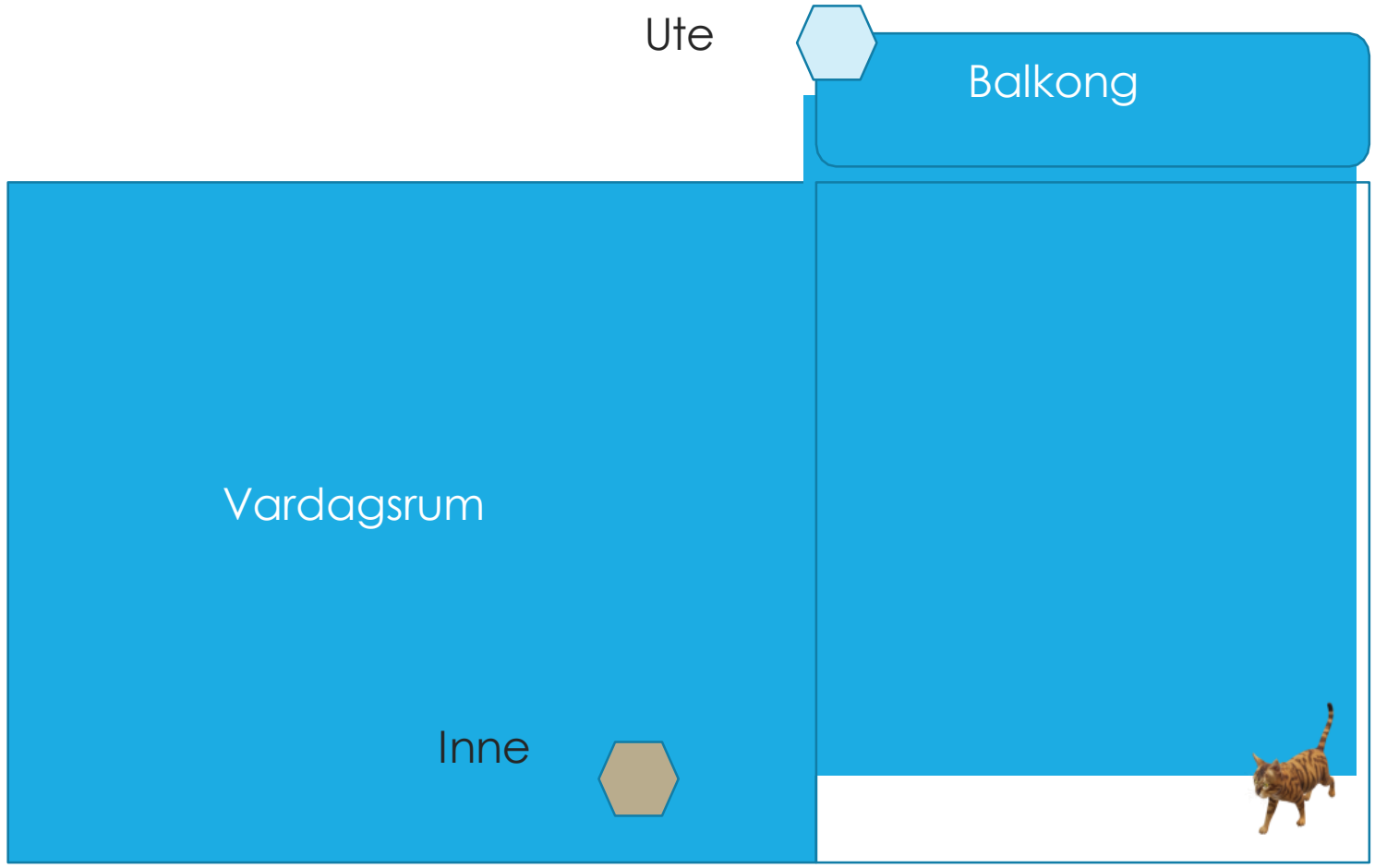
Projekt Väderdata

Projektarbete inom kursen **Arkitektur av applikationer i C#**

Projektarbete – Väderdata

- Projektet är en applikation som, utifrån befintlig temperatur- och luftfuktighetsdata kan söka, sortera och dra slutsatser.
- I det här projektet jobbar vi **i par**.
- Datafil i textformat, kommaseparerat.
- Datafilen är autentisk, och har datafel, och luckor.
- Exempel på data: https://docs.google.com/spreadsheets/d/1r0JNrqXmbFJF28CaF3nqTc9MwnKUpuXyv2hoImWwjoc
- Hämta annars från https://drive.google.com/file/d/1CcRiPfBaNC_sBWqNKJLDOHXS-4Al8xUy

Planritning



Applikationen

Er applikation skall

- Automatiskt skapa databasen om den inte redan finns
  - Låt gärna er grupps namn ingå som en del av databasnamnet!
- Vid behov läsa in datafilen TempFuktData.csv och fylla på databasen
- Använda ett ändamålsenligt användargränssnitt, så som
  - Ett enkelt konsolgränssnitt som visar att alla kraven är uppfyllda – det behövs inte ens någon input från användaren
  - Ett webbgränsnitt skapat med MVC
  - Ett webbgränssnitt med någon annan teknik (som vi inte har gått igenom ännu), till exempel Blazor
  - Windows Forms eller WPF eller en mobilapp eller något annat

Databasen bakom det hela

- Databasen skall skapas av Entity Framework
- Ni skall alltså arbeta enligt metoden **Code First**
- Det är inget krav att använda SQL Server
- Datamodellen behöver inte normaliseras – det räcker med en enda tabell

Följande information skall kunna visas

- Utomhus:
  - Medeltemperatur för valt datum (sökmöjlighet)
  - Sortering av varmaste till kallaste dagen enligt medeltemperatur per dag
  - Sortering av torraste till fuktigaste dagen enligt medelluftfuktighet per dag
  - Sortering av minst till störst risk för mögel
  - Datum för meteorologisk Höst
  - Datum för meteorologisk Vinter (OBS! Vintern 2016 var mild)

Följande information skall kunna visas

- Inomhus:
  - Medeltemperatur för valt datum (sökmöjlighet)
  - Sortering av varmaste till kallaste dagen enligt medeltemperatur per dag
  - Sortering av torraste till fuktigaste dagen enligt medelluftfuktighet per dag
  - Sortering av minst till störst risk för mögel

Frågeställningar som ska lösas

- Val av datatyper i DB för den här typen av data.
- Inläsning av textfil till databasen
- Algoritmer som räknar fram aggregerad data
- Meteorologiska regler för Höst, Vinter och Mögelindex
- Mögelrisk, hitta formel.

Andra krav

- Källkoden dokumenterad i löpande kod, med särskild stor vikt vid användandet av algoritmer. Förklara dina val av algoritmer och datastrukturer.
- Jämför gärna de resultat du får i din applikation med andra grupper, för att dela med er av hur ni tänkt.
- Samtliga krav och specifikationer kan ändras under arbetets gång.
- Dela upp systemet i (minst) tre beståndsdelar:
    1. Core, där all logik bor
    2. UI, för användargränssnitt
    3. DataAccess för dataåtkomst

Krav för VG

- För att få VG ska programmet även kunna göra bedömningar kring följande frågor:
  - Hur länge är balkongdörren öppen per dag, och sortera på detta.
    - Antagandet är att om balkongdörren öppnas så sjunker innertemperaturen lite.
    - Yttertemperaturen höjs också lite grann, eftersom termometern sitter nära balkongdörren.
  - Sortering på då inne- och utetemperaturerna skiljt sig mest och minst.

Ytterligare krav för VG

- På den här nivån förutsätts att ni kan:
  - Bli klara i tid, dvs lämna in före deadline (en minut innan räcker…)
  - Leverera en lösning som fungerar på första försöket
    - Om granskaren behöver göra några handgrepp för att få igång er lösning skall det beskrivas. Helst i en medföljande **Readme.md**
  - **Ni får alltså bara en chans till VG**
    - Om det finns buggar vid redovisningen gör inget så länge de är borta när ni lämnar in – vilket kan göras senare samma dag.

- Dessutom krävs en inlämning av en personlig reflektion över uppgiften
  - Det räcker alltså inte med en lösning i världsklass

Redovisning och inlämning

- Varje par redovisar sin lösning och demonstrerar hur det fungerar för klassen
- Bygg redovisningen som en kombinerad Sprint Review och Retrospective
- Redovisning på onsdag **2022-02-23** med start 09:00
- Inlämning senast: **söndag 2021-02-27 23:59**
- Inlämning görs i Moodle, som vanligt
- Den som står först i respektive par lämnar in för båda
  - Om ni lämnar in i form av en GitHub-länk, kom ihåg att lämna in en liten textfil med länken. Annars syns det inte att ni har lämnat in något.
  - ***Var och en*** som vill ha chans på VG skall lämna in en personlig reflektion över uppgiften

Länkar till hjälp

- http://www.penthon.com/vanliga-fragor/faq/vad-innebar-mogelindex/
- https://www.smhi.se/kunskapsbanken/host-1.1257
- https://www.smhi.se/kunskapsbanken/vinter-1.22843
- Jämföra temperatur: https://rl.se/vadret/period.php
- Beräkna mögelrisk:
  - https://www.byggahus.se/forum/threads/formel-foer-riskkurva.311612/
    - Särskilt inlägg 8 och 9
  - https://pastebin.com/VXyATTWw

Några råd

- Projektet gäller programmering mot databaser, *inte* korrekt beräkning av mögeldata. Så länge ni använder en beräkning som ger olika resultat beroende på temperatur och fuktighet så spelar detaljerna ingen roll.
  - Metoden som hänvisas i länkarna duger gott. Översättning från Python till C# får då bli en liten extra övning.
  - Hittar ni någon bättre metod så använd den.
- Var observanta på hur ni tolkar temperaturdata. De är angivna i datafilen som "10.3", alltså med punkt som decimalavgränsare. Det kan ge upphov till problem. (Det har med Culture i C# att göra)
- Låt LINQ göra det tunga beräkningsarbetet!

Gruppindelning

Till sist

Glöm inte att det här är kul!