|  |
| --- |
|  |
| Eksamensopgave |
| Application Programming - S2 |

|  |  |
| --- | --- |
| Formalia: |  |
| Afdeling | AspIT Sønderjylland |
| Underviser | Jens Clausen |
| Censor | Dima Victor Omkvist |
| Forløb | S2-1-M1-2022 |
| Opgaven udleveres | Tirsdag den 08. marts 2022 kl. 08:45 |
| Opgaven afleveres | Onsdag den 09. marts 2022 kl. 15:00 |
| Der arbejdes følgende dage i tidsrummet 08:45 – 15:00 | Tirsdag og onsdag |

Indholdsfortegnelse

[1 Opgave MeatGross 3](#_Toc96788030)

[1.1 Problemformulering 3](#_Toc96788031)

[1.2 Kravspecifikation 3](#_Toc96788032)

[1.2.1 Generelt 3](#_Toc96788033)

[1.2.2 Brugergrænseflade – Hovedformen 4](#_Toc96788034)

[1.2.3 Brugergrænseflade – Oprettelse og vedligehold af kunder 5](#_Toc96788035)

[1.2.4 Brugergrænseflade – Vedligeholdelse af kødpriser og antal kg. på lager 6](#_Toc96788036)

[1.3 UML-Class diagram for MeatGross 7](#_Toc96788037)

[1.4 ER-diagram for Databasen MeatGross 8](#_Toc96788038)

[2 Bedømmelseskriterier og evaluering 9](#_Toc96788039)

[2.1 Eksamen 9](#_Toc96788040)

[2.2 Karakterskala 10](#_Toc96788041)

# Opgave MeatGross

## Problemformulering

Til firmaet MeatGross skal der udvikles en applikation, der skal hjælpe sælgerne med at holde styr på lagerbeholdning, salg og priser på kødet.

Selve applikationen er delvist påbegyndt af et andet softwareudviklingsfirma, som desværre er gået konkurs, hvilket har medført, at vi har fået opgaven med at færdiggøre projektet.

Det tidligere firma har udviklet brugergrænsefladen til håndtering af forespørgsler på lagerbeholdning, pris og angivelse af et salg til en given kunde.

Der mangler at blive udviklet en brugergrænseflade til håndtering af oprettelse og vedligeholdelse af kundekartoteket og en brugergrænseflade til opdatering af priser og lagerbeholdning af kød.

Der er udarbejdet et UML Class-Diagram over, hvordan applikationen skal opbygges med angivelse af relationer og nedarvning imellem de enkelte projektelementer og klasser i programmet.

Der skal i udviklingsprojektet implementeres et testprojekt, hvor der skal udføres unit-test på udvalgte elementer af koden.

## Kravspecifikation

### Generelt

Brugerne af systemet skal via en database kunne se beholdningen af hakket kød på lageret.

Hakkekødet kan være af følgende typer:

Gris – Kalv – Okse – Kylling – Kalkun – Hest

Når brugeren forespørger på en given type kød, skal programmet vise den aktuelle kilopris på kødet. Når bruger indtaster, hvor mange kilo kunden ønsker, skal der kontrolleres for, om lagerbeholdningen er stor nok, hvis lagerbeholdningen ikke er stor nok, skal det indtastede antal kg. rettes til, hvad der ligger på lageret. Samtidig med at brugeren indtaster, hvor mange kilo kunden vil købe, skal systemet automatisk udregne prisen, så snart der sker en indtastning.

Hvis salget bliver gennemført, skal lagerbeholdningen nedskrives, og ordren skal gemmes på databasen med alle relevante data.

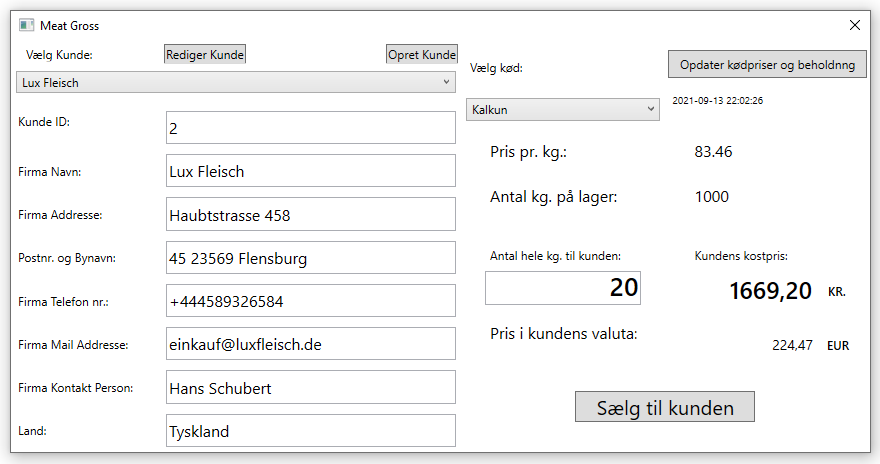
På databasen skal der oprettes tabeller til lagerstyring og ordrehåndtering (der er udarbejdet et ER-diagram over de nødvendige tabeller og deres relationer).

Det færdige produkt skal afspejle det medfølgende ER-Diagram og UML-Classdiagram.

### Brugergrænseflade – Hovedformen

Brugergrænsefladen må ikke ændres.

Hvis udvikleren finder det nødvendigt, må der ændres i navngivningen af elementerne på formen.



### Brugergrænseflade – Oprettelse og vedligehold af kunder

Der skal i videst muligt omfang benyttes samme indstillinger til UserControlCustomerEdit vedrørende oprettelse og vedligehold af kundedata som dem der er benyttet til UserControlCustomer.

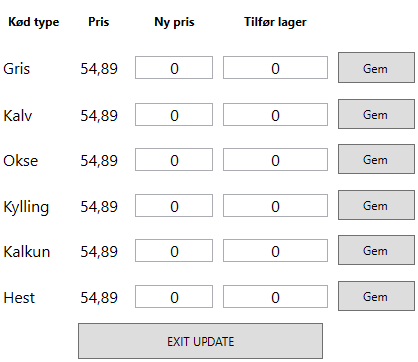
Når formen til redigering af kundedata er åbnet, må man ikke kunne tilgå andre UserControls.

Når man er færdig med redigering eller oprettelse, skal alle relevante data automatisk opdateres på relevante UserControls.



### Brugergrænseflade – Vedligeholdelse af kødpriser og antal kg. på lager

UserControlen skal i videst muligt omfang oprettes som vis i neden stående eksempel:



Når UserControlen til redigering af priser og lagerbeholdning er åbnet, må man ikke kunne tilgå andre UserControls.

Når man klikker på knappen ud for de enkelte typer kød, er det kun de værdier der står til venstre for knappen der skal overføres til databasen.

Ny pris: Er altid i DKK og skal erstatte den tidligere pris.

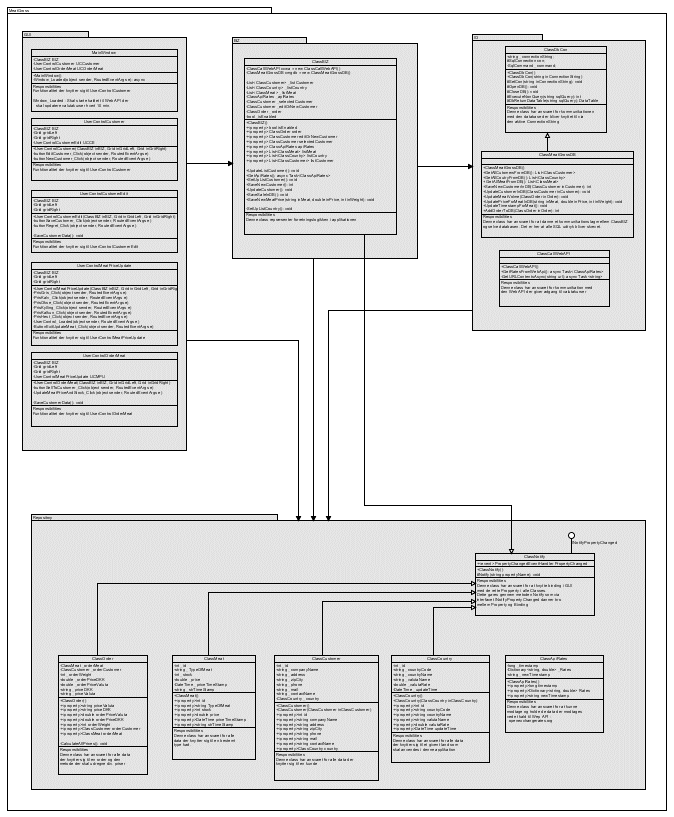
Tilfør lager: Er altid i hele kg. og skal tilføjes den vægtangivelse der er i databasen i forvejen.

Hvis værdien i *Ny pris* eller *Tilfør lager* er 0, skal der ikke opdateres i databasen.

## UML-Class diagram for MeatGross

Dette UML-Class Diagram skal ses som vejledende i forhold til, hvordan løsningen skal laves. Dog skal du overholde den overordnede struktur i dit projekt. Det betyder, at dit projekt skal afspejle en overholdelse af 3-lags arkitekturen med GUI – BIZ – OI og et REPOSITORY.

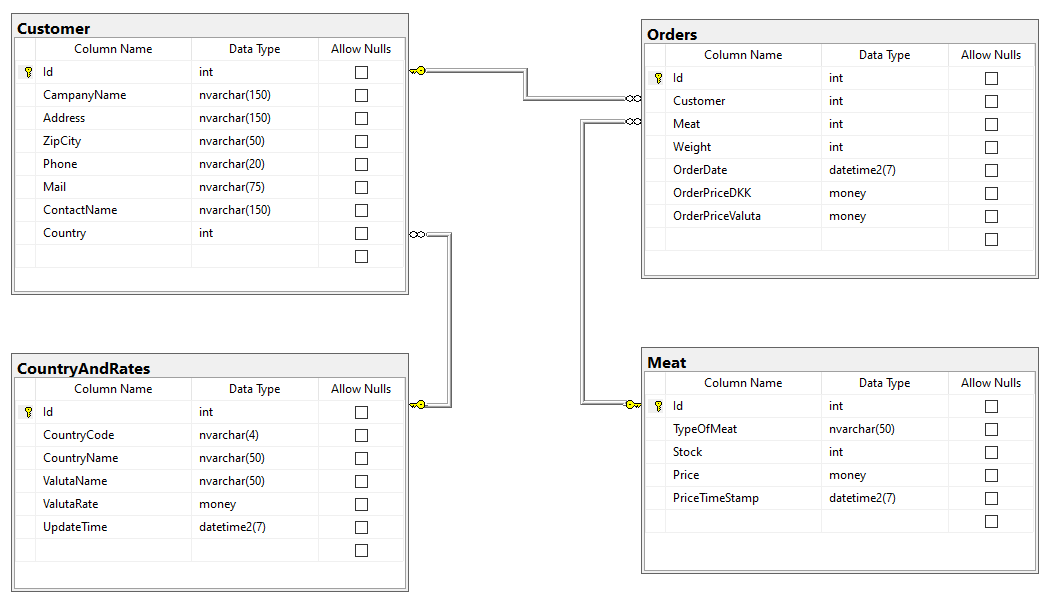
Du skal ligeledes lægge vægt på, at din løsning lever op til reglerne omkring OOP.



Dette diagram ligger som en .uxf fil sammen med din eksamensopgave på OneDrive. (MeatGross.uxf som kan åbnes med [UMLet programmet](https://www.umlet.com/download/umlet_14_3/umlet-standalone-14.3.0.zip))

## ER-diagram for Databasen MeatGross

Databasen skal modelleres som vist i nedenstående UML diagram.



CountryAndRates tabellen skal kun indeholde data vedrørende følgende lande:

* Danmark
* Sverige
* Norge
* Finland
* Island
* Tyskland
* United Kingdom (England)

Dette diagram ligger som en .png fil sammen med din eksamensopgave på OneDrive. (ER\_MeatGross\_DB.png)

# Bedømmelseskriterier og evaluering

Der vil i bedømmelsen blive lagt vægt på følgende:

* Fuldstændighed – hvor meget er lavet færdigt?
* Har du hjulpet andre, eller har modtaget hjælp fra andre. Der ses positivt på begge situationer
* Struktur – anvendelse af relevante programmeringsstrukturer og teknikker, såsom anvendelse af 3-lags-model, OOP, variabler, properties og metoder
* Robusthed – giver programmet relevante fejlmeddelelser?
* Relevant dokumentation i koden.

## Eksamen

Det fremgår af modulbeskrivelsen for *Application Programming - S2* at:

Evalueringen foretages som løsning af en applikationsopgave med en tilhørende database. Opgaven løses over 2 skoledage, á 6 daglige lektioner. Opgaven giver dig mulighed for at arbejde med:

* Centrale programmeringselementer
* Databaseprogrammering
* Kobling af brugergrænseflade til resten af systemet.
* Client / Server
* Anvendelse af OOP.

Eksaminationen gennemføres uden foregående forberedelse som en præsentation af den fremkommende løsning. Eksaminationen gennemføres over 30 minutter, inkl. tid til bedømmelse. Opgaven bedømmes efter 7-trins karakterskalaen, og karakteren meddeles eleven.

## Karakterskala

|  |  |  |
| --- | --- | --- |
| Karakter | Betegnelse | Beskrivelse |
| 12 | Den fremragende præstation | Karakteren 12 gives for den fremragende præstation, der demonstrerer udtømmende opfyldelse af fagets mål, med ingen eller få uvæsentlige mangler. |
| 10 | Den fortrinlige præstation | Karakteren 10 gives for den fortrinlige præstation, der demonstrerer omfattende opfyldelse af fagets mål, med nogle mindre væsentlige mangler. |
| 7 | Den gode præstation | Karakteren 7 gives for den gode præstation, der demonstrerer opfyldelse af fagets mål, med en del mangler. |
| 4 | Den jævne præstation | Karakteren 4 gives for den jævne præstation, der demonstrerer en mindre grad af opfyldelse af fagets mål, med adskillige væsentlige mangler. |
| 02 | Den tilstrækkelige præstation | Karakteren 02 gives for den tilstrækkelige præstation, der demonstrerer den minimalt acceptable grad af opfyldelse af fagets mål. |
| 00 | Den utilstrækkelige præstation | Karakteren 00 gives for den utilstrækkelige præstation, der ikke demonstrerer en acceptabel grad af opfyldelse af fagets mål. |
| -3 | Den ringe præstation | Karakteren -3 gives for den helt uacceptable præstation. |

Tabel 1: Karakterskala for Application Programming - S2.