Mappe-prosjekt IDATx1001 - H2022 - Del 2

Fra Del 1 - Systembeskrivelse

Det skal utvikles en programvare som skal brukes av et varehus – Smarthus As. Programvaren skal brukes til å håndtere varelageret til Smarthus AS (så kalt "Warehouse management system (WMS) " på engelsk).

Smarthus AS leverer hovedsakelig varer til bygg-industrien, så typiske varer er laminatgulv, dører og vinduer, lister og annet trevirke.

Løsningen skal til slutt bestå av et tekstbasert brukergrensesnitt og et register som lagrer informasjon.

Løsningen skal utvikles igjennom 3 iterasjoner; Del 1, 2 og 3. Denne oppgaveteksten beskriver del 2 av oppgaven og bygger videre på del 1!

I tillegg til selve programmet, skal du også skrive en rapport. Denne startet du på i del 1 av prosjektet og du jobber videre med den i del 2 (se detaljer under).

Den endelige løsningen skal leveres i Mappen for emnet, for vurdering sammen med rapporten i slutten av semesteret.

Tilbakemelding/samtale på del 2 - "innlevering"

Du skal ikke levere inn noe for hver av delene i prosjektet (heller ikke del 2), men du vil bli tilbudt mulighet til å få muntlig tilbakemelding på arbeidet ditt så langt. Denne tilbakemeldingen gis på lab samme uke som ny del publiseres, og gis av faglærer eller læringsassistent.

Smart Hus AS - Del 2:

Basert på tilbakemedlingene du fikk på del 1: Gjør nødvendige forbedringer på entitetsklassen, og utvid applikasjonen iht oppgavebeskrivelsen under.

Oppgavebeskrivelse

I **del 2** av mappen skal du implementere et **vareregister** i løsningen din. Et vareregister holder på en eller flere varer (objekter av entitetsklassen du implementerte i del 1).

I tillegg skal du implementere et enkelt **brukergrensesnitt**.

Vareregister

Lag et vareregister som implementerer nødvendig funksjonalitet for å kunne brukes av brukergrensesnittet. Du velger selv hvilke klasse(r) fra biblioteket (JDK) du bruker for å holde på alle varene. Dokumenter i rapporten hvilket valg du falt på og hvorfor.

Vareregisteret bør ha en metode for å fylle registeret med en samling default varer for å kunne teste funksjonaliteten til registeret.

Brukergrensesnitt

Brukergrensesnittet skal tilby følgende funksjonalitet til brukeren. Brukeren er en person som arbeider på lageret.:

1. Skrive ut all varer på lageret
2. Søke etter en gitt vare basert på Varenummer og/eller Beskrivelse
3. Legge en ny vare til registeret. Her skal all informasjon fra 1-10 felter (gitt over) innhentes fra bruker input
4. Øke varebeholdningen til eksisterende vare. M.a.o. du har en vare med et gitt antall på lager (f.eks. 10 stk laminatgulv). Du mottar så en ny forsyning av laminatgulv som så skal registreres inn på lageret (f.eks. 20 stk).
5. Ta ut varer fra varebeholdningen (eksisterende vare). M.a.o. du har en vare med et gitt antall på lageret (f.eks. 20 stk laminatgulv). Du tar så ut 5 stk fra lageret.
6. Slette en vare fra varelageret (fordi den for eksempel er utgått eller ikke i produksjon lenger). M.a.o. du skal ikke lenger ha varen "Laminatgulv" i

   butikken din lenger. NB! Ikke det samme som å sette antall varer til 0
7. Endre rabatt, pris og/eller varebeskrivelse for en vare

Du velger selv hvordan du vil presentere disse valgene for brukeren (meny, kommando el.l.).

Hvilke læringsmål vi ønsker å teste i del 2

I del 2 har vi fokus på:

- robuste design prinsipper som *modularisering*, *coupling*, *cohesion*, *responsibility driven design* osv.
- bruk av tabeller og lister (ArrayList/HashMap/HashSet i tillegg til løkker og iterator)
- fornuftige metoder med gode navn og riktig/god dokumentasjon (bruk CheckStyle;-))
- brukervennlighet/god brukerinteraksjon

NB! Selv om dette er en "kravspek" betyr ikke det at du MÅ følge den til punkt og prikke. Det er selvsagt rom for å gjøre personlige tilpassinger. Pass på da at du dokumenterer i rapporten de endringene/tilleggene du har valgt å innføre og begrunn dine valg. Husk da at ved vurdering er robusthet, god design og godt testet løsning mye viktigere enn "fancy" funksjonalitet

Krav til del 2

Krav til Programmet/koden

Følgende krav gjelder (fortsatt) til denne delen av oppgaven:

- Koden skal følge en bestemt ***kodestil*** (enten Google eller "BlueJ"-stilen gitt i regelfilen "IDATx1001" for Chekstyle)
- Kodestilen **skal** verifiseres med **CheckStyle**-plugin (for BlueJ, IntelliJ osv) og vise ingen regelbrudd ved levering.
- **Klassen** og alle **metoder**, **variabler** (felt, parametre, lokale variabler) **skal** ha gode, beskrivende navn som tydelig gjenspeiler hvilken tjeneste en metode tilbyr, eller hvilken verdi variablene representerer/holder på.
- Alle navn på klasser, metoder og variabler **skal** være på **engelsk**.

Du velger selv hvilken IDE (utviklingsverktøy) du vil bruke på prosjektet (BlueJ, IntelliJ, VisualStudio Code etc).

Krav til rapport

Rapporten fra del 1 videreføres med tilbakemeldingene du fikk fra LA/Faglærer:

Rapporten skal for del 2 utvides med:

- En oppdatert terminologi-liste, som inneholder begreper relatert til oppgaven, både på norsk og engelsk
- Under hovedavsnittet "design/implementasjon".
    - Beskriv med egne ord hvilke valg du har gjort når du implementerte klassen som skal representere et varelager/vareregister:
        - Hvilke klasse fra biblioteket (JDK) valgte du for å holde på varene, og hvorfor?
        - Hva har du gjort for å oppnå **løs kobling (loose coupling**) i designet av register-klassen din?
        - Reflekter/diskuter hvilke tiltak du har innført for å sikre at klassen er implementert som en **robust klasse** (en robust klasse er en klasse der det ikke er mulig å angi ugyldige verdier til feltene i klassen).
    - Brukergrensesnitt/brukerinteraksjon:
        - Hvordan valgte du å løse **brukergrensesnittet** til applikasjonen din og **hvorfor**?
        - Hvordan har du valgt å håndtere det at brukeren skriver inn feil verdier/informasjon? Og hvorfor valgte du å gjøre det slik?
    - Hvilke øvrige vurderinger har du gjort for å ende på implementert løsning ?

Lykke til ☺