Økonomi

# Titelblad

## Vejleder

Søren Præstegaard - SPR

## Fag

Programmering B

## Klasse og skole

3.D - Odense Tekniske Gymnasium

## Antal sider

X A4-sider

X sider a 2400 tegn

## Dato for udlevering

11-03-2020

## Dato for aflevering

07-05-2020

# Indledning

Økonomi er et vigtigt begreb i dagligdagen som alle mennesker skal holde styr på, det kan til tider være svært at have et overblik hvor mange penge man har haft på forskellige datoer. Jeg arbejder derfor med at lave et program som har en grafisk brugerflade, som ved hjælp af en database kan hjælpe brugeren med at holde lidt styr på brugerens økonomi. I denne synopsis vil jeg dokumentere min arbejdsgang med udviklingen af programmet. Her vil jeg komme ind på de forskellige metoder fra programmeringsfaget som jeg har brugt til planlægningen og udførelsen af programmet.

# Planlægning af programmet

## Krav

For at kunne lave et program, så kræver det at man sætter sig nogle krav for hvad der skal være med i programmet. Kravene skal også være med til at sætte nogle rammer for hvordan programmet skal fungere når man bruger det som bruger. Dette gør det nemmere for programmøren at programmerer programmet. Jeg har derfor lavet X krav til programmet som skal sikre at programmet fungere som planlagt, samtidigt med at programmet har de funktionaliteter som der vil være passende.

### Krav 1 - GUI

Programmet skal have en GUI. På programmets GUI skal brugeren blandt andet kunne indtaste hvor ofte brugeren får løn, hvor meget brugeren får i løn, hvor mange penge brugeren har brugt/fået siden sit sidste besøg. Brugeren skal til det sidste have mulighed for at kunne lave sine egne kategorier såsom ”Brændstof”, som kan opdele brugerens økonomi i kategorier som brugeren har lavet. De forskellige informationer skal være vist på GUIen og brugerens kontosaldo skal vises med en graf.

Programmets GUI skal starte med at bede brugeren om at logge ind, eller lave en konto. Derefter vil brugeren kunne komme ind og ændre sin økonomi.

### Krav 2 - Administrering af økonomi

Programmet skal kunne administrere brugernes økonomi igennem udregninger. Programmet skal kunne administrere økonomien, ved at udregne hvor mange penge der er tilbage på kontoen, hvis brugeren har brugt penge osv. Programmet skal også kunne holde styr på løn fra brugerens arbejde, hvis brugeren har tilføjet sit arbejde.

### Krav 3 - Visning af økonomi

Programmet skal via programmet GUI vise brugerens nuværende kontosaldo. Programmet skal også via programmet GUI kunne vise forløbet for brugerens kontosaldo for de datoer der er sket ændringer i brugerens kontosaldo gennem en graf. Visningen af disse økonomiske genstande skal kunne give brugeren et hurtigt overblik over sin kontosaldo igennem dagene.

## brugerhistorier

Jeg har til mit program valgt at lave 3 brugerhistorier. Brugerhistorierne vil jeg bruge til at beskrive nogle mere detaljerede krav og en beskrivelse af hvordan brugeren skal interagere med programmet. Jeg har her valgt at beskrive de 2 brugerhistorier som står for hoveddelen af mit program. I Bilag 1 har jeg den sidste brugerhistorie, som beskriver hvordan brugeren tilføjer og fjerner sit job. I afsnittet ”Brugerhistorie x - Something” vil jeg beskrive hvordan jeg har udført de forskellige brugerhistorier.

### Brugerhistorie 1 - Login system

Denne brugerhistorie kan kun startes når programmet bliver åbnet af brugeren.

* Brugeren åbner programmet
* Programmet registrerer at brugeren ikke er logget ind og viser skærmen for login
* Hvis brugeren har en konto
  + Brugeren indtaster sit brugernavn og adgangskode i de tilhørende felter
  + Brugeren trykker på ”login”
  + Programmet logger brugeren ind og viser hovedsiden for programmet
* Hvis brugeren ikke har en konto
  + Brugeren trykker på knappen ”Don’t have an account? Sign up here”
  + Programmet viser ”Sign up” skærmen
  + Brugeren indtaster sit brugernavn, fornavn, efternavn, adgangskode, bekræftelse af adgangskode og e-mail ind i de tilhørende felter
  + Brugeren trykker på ”Create account”
  + Programmet tjekker om brugernavnet er ledigt
  + Hvis brugernavnet ikke er ledigt
    - Programmet laver en fejl, hvor der bliver beskrevet hvad fejlen er. Teksten ”username” bliver vist med rødt. Brugeren kan ændre brugernavn og prøve igen
  + Hvis brugernavnet er ledigt
    - Programmet tjekker om felterne ”password” og ”confirm password” er ens
    - Hvis felterne er ens
      * Programmet opretter brugerens konto og viser login siden
    - Hvis felterne ikke er ens
      * Programmet laver en fejl, hvor der bliver beskrevet hvad fejlen er, og felterne der er forkerte, bliver vist med rødt. Brugeren kan rette felterne og prøve igen

### Brugerhistorie 2 - Hvor mange penge har brugeren brugt eller fået

Denne brugerhistorie kan kun startes når brugeren er logget ind på programmet, og befinder sig på programmets hovedside.

* Hvis brugeren vil tilføje/fjerne penge
  + Brugeren vælger en kategori i dropdown-menuen ved ”Select category for obtained/used money”
  + Brugeren finder inputfeltet ved enten ”Money obtained” / ”Money used”, og indtaster hvor mange penge brugeren har fået / brugt
  + Brugeren trykker på knappen ”Add money obtained” eller ”Add money used”
  + Programmet tjekker om inputfeltet kun består af tal
  + Hvis feltet ikke kun består af tal
    - Programmet laver en fejl og skriver på skærmen ”Please make sure you only used numbers!”, og teksten ved inputfeltet bliver vist med rødt. Brugeren kan rette og prøve igen.
  + Hvis feltet kun består af tal
    - Programmet tjekker om brugeren har valgt en category
    - Hvis brugeren ikke har valgt en category
      * Programmet laver en fejl og skriver på skærmen ”Please select a category!”, og teksten ved dropdown-menuen bliver vist med rødt. Brugeren kan rette og prøve igen.
    - Hvis brugeren har valgt en category
      * Programmet modtager tallet fra inputfeltet og kategorien og indsætter i databasen for brugeren.
      * Programmet opdaterer programmets GUI og grafen for brugerens saldo ved den nuværende dato er opdateret og viser den nye kontosaldo.

## iterationer

# Udførelse af programmet

## Oprettelse af databasen

## Oprettelse af klasserne til behandling af data

## Brugerhistorie x - Something

Beskrivelse af hvordan brugerhistorien er blevet udført.

# Test af programmet

## Test 1

Pisse god ide og noget han gerne vil bruge.

Mangler nogle infoting til hvordan de forskellige ting fungerer.

Grafen viser intet ved den første gang man tilføjer penge på den første dag.

## Test 2

## Test 3

# Beskrivelse af arbejdsprocessen

## Trello

Jeg har i mit projekt haft brug for at have et overblik over de delopgaver som jeg skal have arbejdet med. Jeg har derfor brugt hjemmesiden trello.com til at hjælpe mig med at holde et overblik over mit projekt, hvilket har sikret mig en let tilgang til at holde styr på hvor meget der mangler osv. Man kan på de forskellige elementer angive hvornår de skal være færdige, så man ender med at bruge tiden fornuftigt.

## Tidsplan

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Moduler  (12 moduler  i alt) | 2 | 4 | 2 | 1 | 3 |
| Arbejdsopgaver | Krav specifikation til programmet | Program opsætning | Database opsætning | Test af programmet | Synopsis skrivning |

# Litteraturliste

# Bilag

## Bilag 1

### Brugerhistorie 3 - Add job / Remove job

Denne brugerhistorie kan kun startes når brugeren er logget ind på programmet, og befinder sig på programmets hovedside.

* Hvis brugeren vil tilføje et job
  + Programmet tjekker om brugeren allerede har et job
  + Hvis brugeren ikke har et job
    - Brugeren skriver navnet på brugerens job i inputfeltet ved ”Add job name”
    - Brugeren skriver lønnen som brugeren får i inputfeltet ved ”Add job salary”
    - Brugeren skriver dagene der er mellem hver gang brugeren får løn i inputfeltet ved ”Days between payments”
    - Brugeren trykker på knappen ”Add job”
    - Programmet tjekker om alle 3 felter er udfyldte
      * Hvis der er én eller flere felter der ikke er udfyldte
        + Programmet laver en fejl og skriver en passende fejltekst alt efter hvilke felter der mangler. Teksten ved inputfeltet bliver markeret med rødt, og brugeren kan rette og prøve igen.
      * Hvis alle felter er udfyldte
        + Programmet tjekker om der kun er tal i inputfelterne ved ”Add job salary” og ”Days between payments” (”Days between payments må kun være heltal”).
        + Hvis felterne ikke kun består af tal

Programmet laver en fejl og skriver ”Please make sure you only used numbers!”, og teksten ved inputfeltet bliver markeret med rødt. Brugeren kan rette oplysningerne og prøve igen.

* + - * + Hvis felterne kun består af tal

Programmet modtager de forskellige informationer fra inputfelterne og tilføjer dem til databasen.

Programmet opdaterer programmets GUI og viser nu oplysningerne for brugerens job.

Programmet udregner hvornår brugeren får løn næste gang og viser datoen for næste løn på skærmen.

* + Hvis brugeren har et job
    - Programmet laver en fejl og skriver ”You already have a job. Remove your job before adding a new job.”
    - Brugeren kan fjerne sit job og prøve igen.
* Hvis brugeren vil fjerne sit job
  + Brugeren trykker på knappen ”Remove job”.
  + Programmet fjerner brugerens job oplysninger fra databasen op opdaterer programmets GUI så informationer om brugerens job ikke vises længere.