Emilia Krogh Malmberg

Godkendelsesopgave 3

Programmering og udvikling af små systemer samt databaser

# Løsningsovervejelser samt begrundelse for valg

Nedenstående er mine overvejelser for opbygningen af de forskellige dele af opgaven, og mine overvejelser på baggrund af de metoder jeg har brugt.

## Opbygning af klasserne samt endpoints

Som indikeret i opgavebeskrivelsen skal vi ikke kode en frontend, derfor, startede jeg med at kode mine klasser, da jer ser dem som et godt udgangspunkt for resten af opgaven. Til dette bruger jeg *constructor* metoden, da vi har forskellige attributter under de forskellige klasser der skal indgå. Et eksempel er klassen for *paymentUser* som er en nedarvning af *User*:

class paymentUser extends user {

    constructor(pinNumber, cardName,cvvCode){

        this.pinNumber=pinNumber;

        this.cardName=cardName;

        this.cvvCode=cvvCode;

    }

}

Dette er gentaget med freeUser, hvorimod de andre variabler som billede og interesser, er oprette som en klasse uden undertyper. I bilag 1, ses UML diagrammet der netop afspejler denne opbygning af koden, hvor det ses at det kun er brugeren der har undertyper.

Nu hvor mine er klasser er stillet op i hver sin javascript fil kan jeg oprette mine endpoints, der er kodet i serveren (opbygningen af min server er forklaret i næste afsnit).Jeg bruger kun de to HTTP metoderne, get og delete ud af de fire, når jeg opretter mine endpoints. Jeg skriver server.get da den viser den route man skal tage for at hente data. Altså min app.get() function specificere er en callback funktion der kører når der er en http get request med en sti defineret, derefter er der defineret hvad det prævis er funktionen skal sende tilbage fra vores ”sti”, når jeg skriver res.send():

//til user

server.get('/user1', (req, res) => {

  //fortæller at den skal sende data fra mit første array tilbage

  res.send(user.hardUser[0]);

})

//til match

server.delete('/match', (req,res) => {

  res.send(hardMatch[0]);

})

Dette skyldes at jeg har kodet arrays manuelt i de filer der hedder hardUser, hardMatch og hardInterest. Denne metode, oprettelsen af arrays og sammensætningen med mine endpoints, benyttes da jeg gerne vil teste om disse endpoints rent faktisk fungere fremfor bare at kode et løst eksempel som i teorien ville kunne bruges. Dette er testet i postman, sikker på der er andre metoder til at teste om ens CRUD-endpoint virker, men jeg valgte oprettelsen af arrays manuelt for de forskellige klasser og postman, da det var den fremgangsmåde der gav mest mening for mig.

## Oprettelse af API

Efter oprettelsen af mine klasser installerede jeg Express og oprettede min API fil, der er navngivet servergotre.js. Express bruges til at oprette min API, samt så jeg kunne lave mine CRUD-endpoints. Dette var altså et step der lå imellem oprettelsen af mine klasser og endpoints.

(…)

## Login Controller og JWT

## GitHub

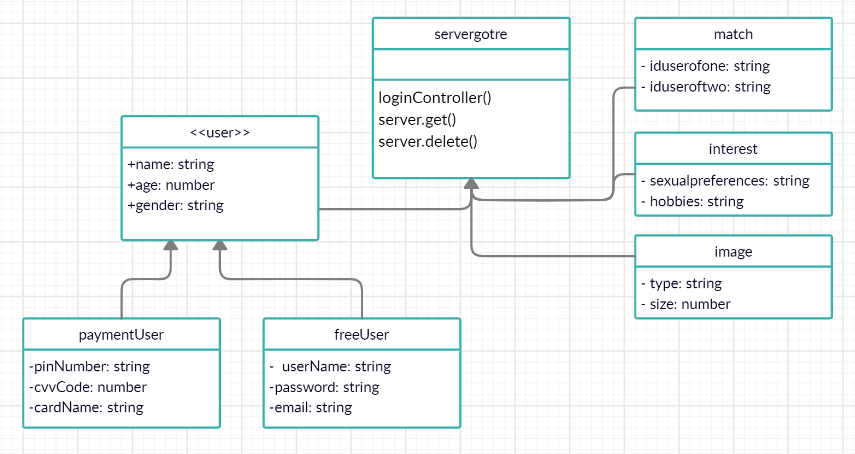
Derudover er min kode gemt i Github som krævet i opgavebeskrivelsen.

<https://github.com/emiliakmalmberg/godkendelsesopgave->

[**emiliakmalmberg/godkendelsesopgave-**](https://github.com/emiliakmalmberg/godkendelsesopgave-)

## Bilag

UML klassediagram



*\*En funktion bliver til en metode når det er inde i en klasser*

Kan ikke opdatere min server

Er min logincontroller det samme som min jwt token?

UML?

Er det okay at de er blanke ”funktionerne”?