API & Driftsättning

Och lite MySQL

API

- Vi har tidigare byggt API consumers, dvs applikationer som konsumerar ett Rest API, eller som använder sig av data från en annan tjänst. (mocky.io och liknande)
- Vi ska även lära oss att bygga Rest API providers, dvs tjänster som låter oss hantera och servera data.

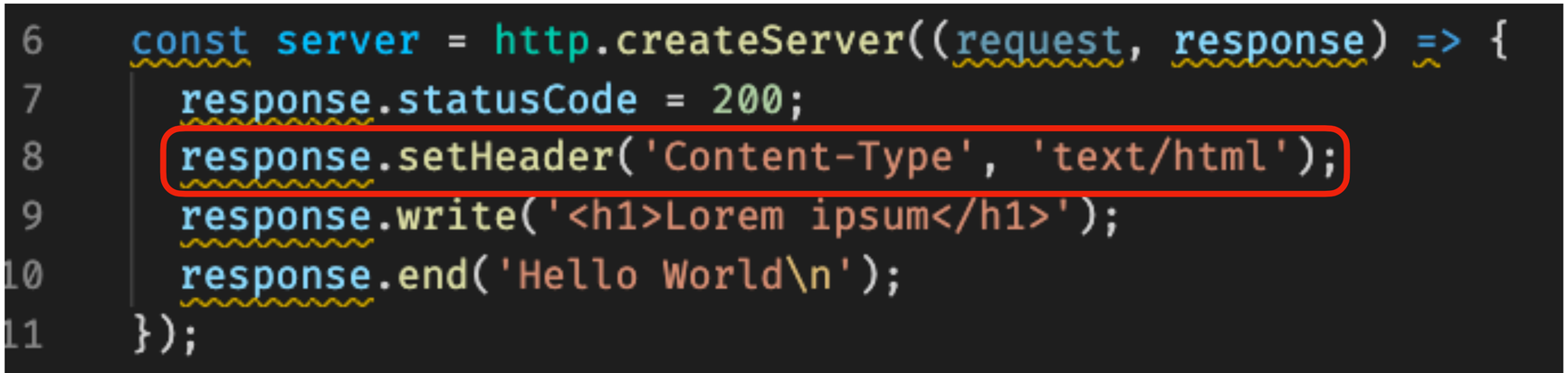
CRUD

- Http-metoder
  - `POST`: Used to submit data, typically used to create new entities or edit already existing entities
  - `GET`: Used to request data from the server, typically used to read data
  - `PUT`: Used to completely replace the resource with the submitted resource, typically used to update data
  - `PATCH`: Used to update parts of a resource
  - `DELETE`: Used to delete an entity from the server

Serva data

HTML

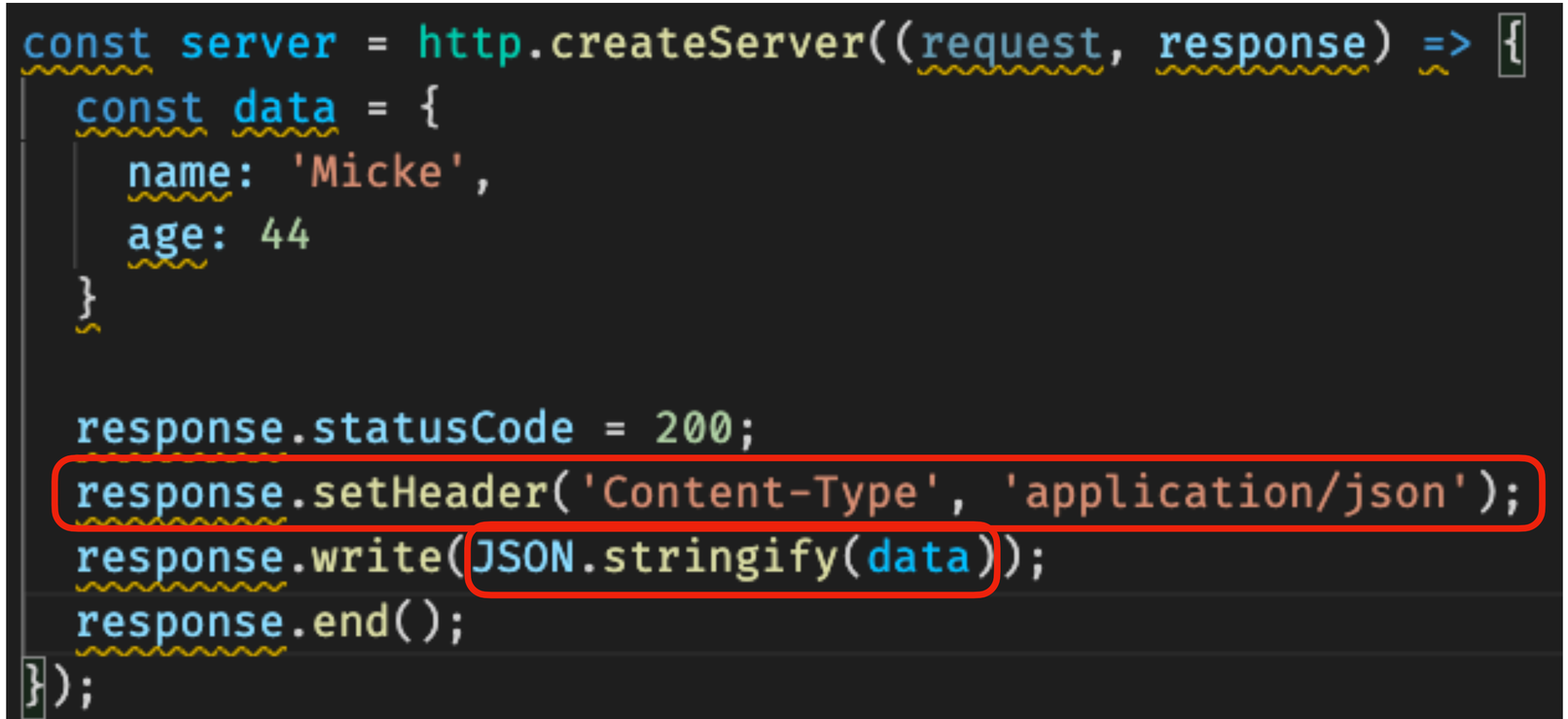
- Det vi har gjort hittills har i stor utsträckning varit att skicka tillbaka html.
- I början gjorde vi det manuellt genom att ange vilken typ av data vi skickar tillbaka till klienten.
- Detta sköts automatiskt i Express.



Serva data

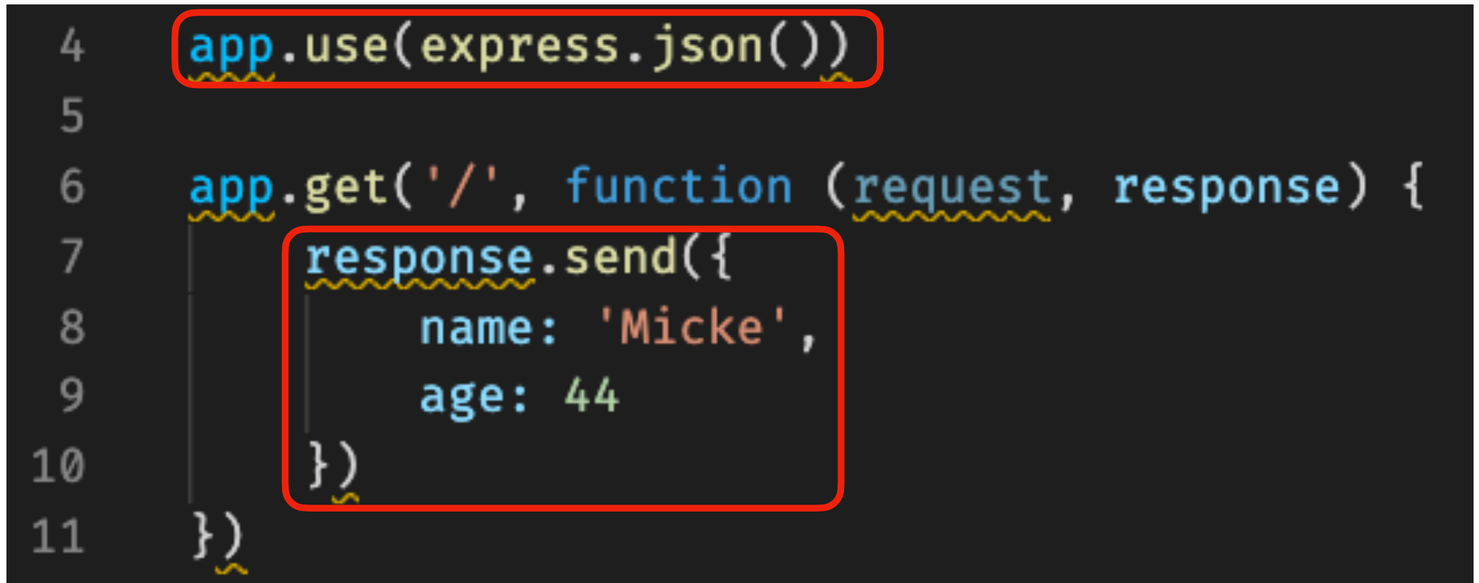
JSON

- Det skiljer inte mycket om vi istället vill skicka tillbaka JSON (eller andra typer av data).



Serva data
JSON

- Express kan ta hand om mycket av detta också åt oss.



Code Along

- Vi ska bygga ett Restful API tillsammans.
- Vårt API ska hantera böcker. Här finns data vi kan importera till vår databas: https://gist.github.com/nanotaboada/6396437
  - Om ni vill köra själva: https://stackabuse.com/building-a-rest-api-with-node-and-express/
  - Extra övning: bygg ut API:et med en endpoint som tar emot en författare och returnerar alla böcker som författaren har skrivit.

MySQL

- Vi kan även använda oss av MySQL i våra applikationer.
- En bra startpunkt är paketet mysql.
  - https://www.npmjs.com/package/mysql
- Låt oss titta på ett exempel.

```
const mysql = require('mysql');
var connection = mysql.createConnection({
    host: 'localhost',
    // port: '3306',
    socketPath: '/Users/micke/Library/Application Support/Local/run/d5uuTh7we/mysql/mysqld.sock',
    user: 'root',
    password: 'root',
    database: 'classicmodels',
    debug: false,
});
```

Deploya application

Publicera

- För att andra ska komma åt vår applikation behöver vi en dator med server-program som är tillgänglig från internet.
- Det kan vara en VPS (Virtual Private Server), ett webbhotell eller liknande.
- På ett webbhotell ser företaget till att alla servrar finns installerade, fungerar och är säkra.
  - Kan vara begränsande eftersom vi inte får installera egna program, men kan också vara lättare att slippa hantera säkerhet och sånt.
  - Ex: Heroku
- På en VPS (eller liknande) kan vi själva installera de program (webbservrar och liknande) vi vill kunna använda.
  - Ger stor frihet men kräver mer av oss i form av att sätta upp brandväggar mm.

Deploya application

Installera egen server

- Det funkar utmärkt att installera sin egen server, men det kräver en del kunskap om säkerhet, brandväggar osv.
  - En vanlig brandvägg heter `iptables`, om man vill ha en utgångspunkt.
- Om vi bortser från säkerhetskrav, så skulle vi göra samma sak som vi har gjort på våra lokala datorer; installera node, mongodb osv.
- Ett tips är att inte köra vår webbserver genom node eller nodemon direkt utan att använda ett program som kan övervaka vår applikation och starta om den om det verkar behövas.
  - https://pm2.keymetrics.io/
  - Alternativ: https://geekflare.com/nodejs-monitoring-tools/

Deploya application

Använda en tjänst

- Det finns flera tjänster som låter oss publicera vår tjänst med tillgång till MongoDB och liknande.
  - https://www.heroku.com/nodejs
    - https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/deployment
  - https://try.digitalocean.com/deploy-nodejs/
  - https://docs.microsoft.com/en-us/azure/app-service/quickstart-nodejs?pivots=platform-linux
- En del kostar, en del är gratis för åtminstone små applikationer.

Uppgift

- Bygg ett API som CRUD:ar kunder och ordrar.
- Bygg endpointen /customer/:id/order som returnerar alla ordrar för en kund.
- Bygg en endpoint som hämtar ut alla kunder i en specifik stad/ett specifikt land.
- Implementera ett frontend som hämtar en kundlista och skriver ut den i en tabell.