Single Sign-On
Logger på en sentral påloggingstjeneste og får en *token* tilbake. Kan bruke samme token om igjen for å slippe å logge inn igjen.
Autentisering i REST-tjenester

- Sesjonsbasert: enkelt, bruker servertilstand(ikke helt REST-ful), fungerer ikke over mange REST-tjenester(SSO)
- Token-basert: kan brukes på mange tjenester(SSO), token må sendes i hvert kall (auth-header), krever sentralt register
- Sertifikiat basert(signerte tokens), Rolls Royce, helt tilstandsløs(REST-ful), kan brukes på mange tjenester(SSO)

Express Middleware

- /API
- Noe usynlig du putter mellom to lag i en app.
- I Express kan vi beskytte våre endepunkter med å innstalere en funksjon som kjører for vi får tilgang til endepunktene
- Denne funksjonen kan avgjøre om vi skal få tilgang til endepunktene eller sende en feilmelding til klienten
- Vi innstalerer en middleware-funksjon ved å bruke *app.use("/thepath", enFunksjon).*
- Da vil enFunskjon kjøre for alle endepunktene som under samme path kan nås.

1 Ulike autentiseringer

Tokenbasert (usignert)

- Klient kaller REST, men brukernavn og passord
- REST generer random Token og lagrer
- Klient spør om funksjon med token i header
- Server sjekker token mot lagra og utfører funksjonen

Signaturbasert autentisert (JWT)

- Klient kaller REST, men brukernavn og passord
- Server lager en JWT med en henmmelighet og returnerer til klienten
- Klient sender JWT i headeren til funksjonen til server

- Server sjekker JWT signatur, får brukerinfo fra denne, og responderer

Server slipper å lagre tokenet på serveren. Vi sjekker ikke mot database, vi sjekker bare om signatur er gyldig

JWT (JSON Web Token)
Når vi lager JWT, lager vi en JSON. Vi fyller ut noen claims som (navn, value)par. Disse navnene er reserverte feks. iss(issuer), exp(expiration), sub(subject), aud(audience). Så signerer vi dette med et bibliotek, feks jasonwebtoken(node) ekker Java JWT som da generer en header og en signatur Vi ender opp med en kodet streng i tre deler, separert med punktum.

2 OAuth2

Åpen protokoll som gjør at vi som utvikler applikasjonen kan bruke påloggsfunksjonalititeten til en Identity provider, feks, Google, Facebook, Twitter og mer. OAuth2 bruker også signerete tokens som blir plassert i headeren. Kan bruke Passport i Node for å gjøre dette.