Passport

Inloggning

- Det finns många sätt att hantera inloggning på.
- Det simplaste är att kanske att kontrollera ett användarnamn och lösenord mot t ex en databas för att kontrollera att användaren finns.
- Det finns massor av säkerhetsgrejer att tänka på med inloggningar. Några basic:
  - Kryptera lösenord
  - Salta

Inloggning
Kryptera lösenord

- Tänk er att man sparar inloggningsuppgifter i en databas.

```
{
  username: "micke",
  password: "hemligt"
}
```

- Om någon skulle få tillgång till databasen skulle hen kunna läsa av lösenorden. Om vi istället krypterar lösenorden skulle vi göra det svårare.

```
{
  username: "micke",
  password: "jdiwnfopakjhui38uij8u"
}
```

Inloggning
Kryptering

- Olika krypteringsmetoder är olika starka. Sha256 och blowfish är exempel på starka envägskrypteringar.
- En del krypteringar går att reversera.
  - `if (request.body.password == decrypt("jdiwnfopakjhui38uij8u"))`
  - Kallas ibland tvåvägskryptering.
  - Osäkert. Om du kan dekretera kan någon annan det också.
- Envägskrypteringar ”går” inte att dekryptera men ger samma krypterade sträng varje gång. Lite säkrare.
  - `if (encrypt(request.body.password) =="jdiwnfopakjhui38uij8u")`

Inloggning
Salt

- Korta lösenord är lättare att knäcka än långa. Säg att användaren har valt lösenordet ”abc”.
  - En bruteforce-attack kan göras genom att testa alla kombinationer av vissa tecken, t ex alla små bokstäver, alla stora bokstäver, siffror och kanske några specialtecken.
  - ```
    test_password('a')
    test_password('b')
    // …
    test_password('aa')
    test_password('ab')
    // osv…
    ```
  - Det ska inte många gissningar till för att hitta vårt lösenord.
    ```

Inloggning

Salt

- En sak vi kan göra för att ”tvinga” fram långa lösenord är att salta dem i vår applikation.

```
// request.body.password = 'abc'
const salt = 'jre9398eoke9ujmfkj'

save_password(salt + request.body.password)  // jre9398eoke9ujmfkjabc
check_password(salt + request.body.password) // jre9398eoke9ujmfkjabc
```

Inloggning
Session

- Http är stateless - håller inte reda på något mellan requests.
- För att hålla reda på att användaren är inloggad och inte behöver skicka in användarnamn och lösenord i varje request kan vi spara att användaren är inloggad i en session.

```
request.session.loggedin_user = 'micke'
```

Inloggning
JWT

- JSON Web Token
- Bygger i princip på att när man loggat in får man en hemlig sträng.
- Denna sträng har en viss giltighetstid men så länge man har en giltig sträng kan man begära att få en ny sträng.
- Gör det lite säkrare eftersom det blir en ”ständigt” pågående inloggning där det mesta sköts automatiskt.

Oauth

”Utlånad” inloggning

- Finns i olika versioner som inte är kompatibla.
- Används för att låta en tredjepart hantera inloggningen, t ex Facebook, Google, Twitter, LinkedIn.
- Du kan sätta upp att användaren får använda t ex Facebook för att logga in.
- Facebook garanterar sedan att användaren är den hen uppger sig vara och skickar med en JWT eller liknande som din applikation kan använda för att kontrollera att användaren fortfarande är korrekt.

Inloggning

Passport

- Det finns såklart olika npm-paket för att hantera inloggning. Ett av de mest populära heter passport.
- Passport har stöd för en massa inloggningssätt som de kallar strategier.
- Vi ska bygga ett inloggningssystem. Vi kommer i mångt och mycket att följa följande tutorial:
  - https://betterprogramming.pub/build-a-login-system-in-node-js-f1ba2abd19a

Eftermiddagsprojekt

Blogg

- (Ni kan även fortsätta med / skriva om receptsamlingen eller biblioteket om ni föredrar det.)
- En blogg innehåller inlägg. Man kan tänka sig olika lösningar, men säg att man visar en eller ett par ”huvudinlägg” (senaste inlägget/inläggen) och att man visar länkar till tio ytterligare inlägg. När man klickar på ett inlägg kommer man till detaljsidan för inlägget.
  - Tänkbara fält för ett inlägg: title, content, author(s), categories, tags
  - Man skulle kunna tänka sig att lägga in bilder. Än så länge vet vi inte hur man laddar upp bilder, men vi skulle kunna lägga bilder i en mapp (`/public/images/kitten.jpg`) och spara sökvägen till bilderna i en egenskap i dokumentet. (`image: '/public/images/kitten.jpg'`)
  - Funktioner man skulle kunna vilja ha: möjlighet att CRUD:a inlägg, skriva kommentarer till inlägg, kunna lista inlägg på kategori/tagg, kunna söka efter inlägg, kunna lista inlägg efter författare.
  - Användare bör kunna logga in för att kunna skapa/redigera/ta bort inlägg.