# Hashing

Dag 3 : Hashing.

## Indhold:

## Faglige mål:

Dagens øvelse dækker følgende målpinde:

1. *Eleven kan udvikle serverside webapplikationer, der kan levere HTML-kode til browseren, samt Web API eller webservices, som kan udveksle data med en client-application, f.eks. en browser eller en mobil App.*
2. *Eleven kan redegøre for forskellige arkitekturer for web Applikationer og web API (web Services), med fordele og ulemper.*
3. ***Eleven kan opbygge og konfigurere en web Application og web API (web service) vha. et framework.***
4. *Eleven kan benytte validering af brugerinput i en web Applikation.*
5. *Eleven kan implementere passende ViewModels eller DTO klasser.*
6. *Eleven kan anvende Unit Test og mocking af objekter.*
7. *Eleven kan konfigurere routing i en applikation.*
8. ***Eleven kan udvide en applikation med en database, evt. med et ORM-framework.***
9. *Eleven kan programmere services til brug for en applikation, f.eks. data- og logging-services. r*
10. *Eleven kan benytte en hensigtsmæssig strategi for Exception handling.*
11. ***Eleven kan implementere sikkerhed og brugeradministration i en web applikation.***
12. *Eleven kan udrulle (deploy) en applikation, både On-Premises og Cloud baseret.*
13. *Eleven kan udføre Parallel Programmering.*
14. **Eleven kan redegøre for fordele/ulemper ved forskellige teknikker inden for Cryptography.**
15. **Eleven kan anvende Hashing, Symmetric og Asymmetric Encryption**

# Øvelse:

## Baggrund:

(se afsnit ”**Hasing, definition”** på side 4)

## Krav:

1. Forneden er 4 hashing algorytmer. I en backend klasse (.cs fil) implementer hashing metoder for ALLE det viste hashing algorytmer.
   - SHA2
   - HMAC
   - PBKDF2
   - BCRYPT

- Opdater din web app, så applikation implementer hashing mod brugerens cpr.nr.
  - Det oprettet back-end .cs fra punkt 1 foroven må kun tilgås via dependency injection i din web app. Er metoderne ikke kaldt ved brug af dependency injection betragtes det som en væsenlig mangel.
  - Opdater din web app, så den hasher brugerens cpr.nr. i CprNr side, når cpr.nr. gemmes i ToDoDb databasen. Du skal bruge **mindst én** af de 4 implementeret hashing metoder til hashing. Mindst én af de hashing metoder du bruger til hashing skal implementer: salt, iterationer, angivelse af en hashing algorytme til kryptering samt ”enhance entropi”. Er dette ikke opfyldt, betragtes det som en væsenlig mangel.
  - Det er ikke ligegyldigt hvilket hashing algorytme du anvender. En af de 4 metoder bør helst IKKE anvendes til login valideringer, hverken for en password, eller en cpr.nr. Være derfor opmærksom på (undersøge), hvilket formål de hashing algorytmer har inden du vælger hvilket hashing metode du vil bruge. Vælger du forkert, betragtes det som en væsenlig mangel.

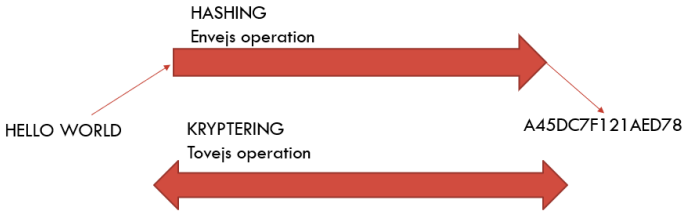
## Hvordan du bliver bedømmet

- Du skal kunne køre/fremvis din applikation fejlfrit på din browser med https://, men nu skal du kunne vise, at bruger CprNr er krypteret med hashing i den database. Brug punkterne under krav som check list, at din kode opfylder de stillede krav.

# Hashing, difinition

## HASHING, DEFINITION

Hashing bruges til at krypter sensitiv data, så data ikke er tilgængelige i simpel læsbar tekst, og bruges der hvor der ikke er behov for "reverse" kryptering.




## HASHING ALGORITME OG METODER

**Hash-algoritme**: en specifik, detaljeret, trin-for-trin procedure til at generere en hash-værdi fra inputdata.
**Hash-metode**: et programmerings metode udvikler kaldes når et stykke tekst skal hashes. Metoderne indeholder skjult hashing-algoritmer til generering af hash-værdierne.

- Følgende er hash-metoder og deres tilhørende hash-algoritmer:

| Algorithm | Metode |
|---|---|
| **M**essage **D**igest algorithm (**Obsolete!**) | *MD5* (**Obsolete!**) |
| **S**ecure **H**ashing **A**lgorithm | SHA-1 (**Deprecated!**), SHA-2, SHA-3 |
| **M**essage **A**uthentication **C**ode algorithm | **H**ash-based **MAC** **K**eccak **MAC** |
| **P**assword-**B**ased **K**ey **D**erivation **F**unction **2** | PBKDF2 |
| **B**lowfish **C**ipher algorithm vs. | BCRYPT |

- I nogle hashing metoder kan man specificer specifikt hashing algorytme at anvende (man kan derved dynamisk ændre hashing algoryme og forvirre hacker).
- I nogle hashing metoder kan man tilføj ekstra trin til komplicering af hashing så som itirationer, og "salt".
- Hashing metoderne er kodet til at være langsomme
  - *When it is said that a hashing function is coded to be slow, it usually refers to the intentional design of the hashing algorithm to require a significant amount of computational time to process each input.*
  - Bruteforce bliver meget svære at gennemføre jo langsomme en hashing metode er, men altid muligt.

## KMAC VIRKER IKKE PÅ ALLE PC

- **Important Note:** KMAC availability depends on your system:
  - Linux requires OpenSSL 3.0 or later.
  - Windows requires Windows 11 Build 26016 or later.

## Hashing metode eksempel