Softwaresikkerhed

webapp med sikkerhed framework

Dagens mål

Implementer en webapp med stærk sikkerhed

- 1. Valg af webapp template
- 2. HTTPS implementation i det valgte webapp
- 3. Bruger database implementation i det valgte webapp.
- 4. Authentikering (m/u 2-factor) implementation i det valgte webapp.
- 5. Autorisation implementation i det valgte webapp.
- 6. Strong password implementation i det valgte webapp.

Vælg en webapp template

Vælg selv hvilket C#/.NET webapp template i vil bruge!

- ASP.NET Core Web App (Model-View-Controller)
- ASP.NET Core Web App (Razor Pages)
- Blazor (anbefales)
- Mere besværlige men tilladt
  - Client-side webapp med Web API - vil kræver 2 paraelle projekter (som begge skal testes)
  - Eget valgt webapp template som ikke behøver at være C#/.NET, men i skal så selv stå for undersøg og implementering af følgende på det valgte platform: Certifikat, authentikation (2-factor), authorisation, stærke password med beskyttelse via kryptering.

Øvelse

Code-along(sammen) oprettelse af det valgte webapp template

- Gennemgåelse og forstå strukturen af det oprettet webapp projekt.
- Genopret projekt, denne gang med en sikkerhed framework (Identity) tilføjet.

Certifikat

Certifikater

- Man se og få overblik over sin installeret certifikater (udgivet og/eller self-sign) med ”Administrer brugercertifikater”/”Manage user certificates” program.
  - Brug eventuelt Windows Søge felt, skriv ”cert” i søge feltet, og vælg ”Administrer brugercertifikater”/”Manage user certificates”. Eller eksekver følgende kommando i CLI, eller Windows Søge felt: certmgr.msc

Self-sign certifikater, til udvikling og test

- **Opret** selv-sign cert : dotnet dev-certs https --trust
- **Slet** selv-sign cert : dotnet dev-certs https --clean

Afgræsning, dækkes først under kyrptografi, under fag: Serverside programmering 3

Selv-sign certifikat behøves ikke at være auto generert. En auto genereret certifikat kan man ikke flytte rundt med fysisk, den er automatisk placeret og tilgås med certmgr.msc.

Det er ikke ualmindeligt, at opret certifikat manuelt, så man får en fysisk certifikat at flytte rundt med, og også mulighed for selv at specificer certifikats navn og password.

Øvelse

Code-along(sammen) oprettelse af self-sign certifikat

1. Slet eksisterende selv-sign certifikat (med CLI kommando).
2. Genstart applikation og se fejlmeddelelse.
3. Opret en selv-sign certifikat (med CLI kommando).
4. Genstart applikation og se at applikation køre igen.

Overblik over burger databasen

Øvelse

- Kode-along(sammen), vi skal gennemgå processen for **oprettelse** af databasen til bruger rolle og password kryptering.
  1. Kør applikation og register en bruger.
  2. Med en database værktøj, undersøger vi hvordan databasen og bruger data er oprettet.

Authentikering (m/u 2-factor) implementation i det valgte webapp

Øvelse

Kode-along(sammen), vi skal gennemgå processen for **oprettelse** af 2-factor authentikation og processen for **håndhævelse** af "authentikering krævet" områder i webapp.

1. Aktiver 2-factor authentikation i webapp.
2. Generering af 2-factor authentikation kode.
3. Håndhævelse af "authentikering krævet" områder i webapp, både i view og i koden.

Autorisation implementation i det valgte webapp

Øvelse

Kode-along(sammen), vi skal gennemgå processen for **oprettelse** af bruger rolle.

1. Oprettelse af bruger roller.
2. Håndhæv authorization (rolle for log-in brugeren) i webapp.

Strong password implementation i det valgte webapp

Øvelse

Følgende link fra Microsoft dokumentation, beskriver Identity framework, blandt andet for password styrke:

https://learn.microsoft.com/en-us/aspnet/core/security/authentication/identity?view=aspnetcore-8.0&tabs=visual-studio

Undersøg via linket til det beskrevet dokumentationen, hvad implementationen af password settings er, og implementer det i webappen.

Strong password eksempel:

- **RequireDigit** : Password skal indeholde mindst et tal (0-9).
- **RequireLowercase** : Password skal indeholde mindst et lille bogstav (a-z).
- **RequireNonAlphanumeric** : Password skal indeholde mindst et specialtegn (fx. !@#$%^&*()_+).
- **RequireUppercase** : Password skal indeholde mindst et stort bogstav (A-Z).
- **RequiredLength** : Password skal være mindst 6 tegn langt.
- **RequiredUniqueChars** : Password skal indeholde mindst 1 unikke tegn.