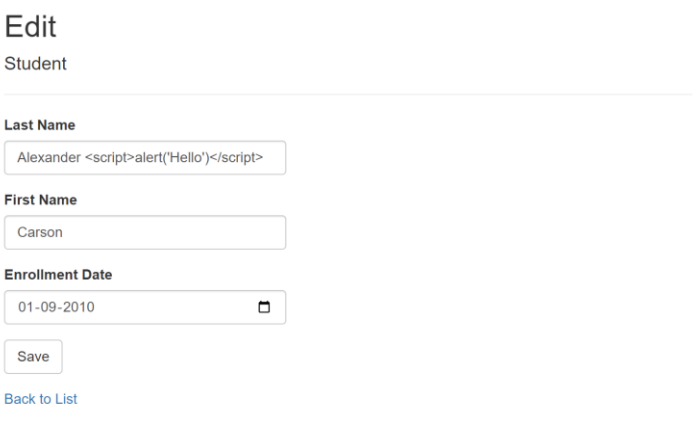
CST1 - Practicum les 4 – CSRF

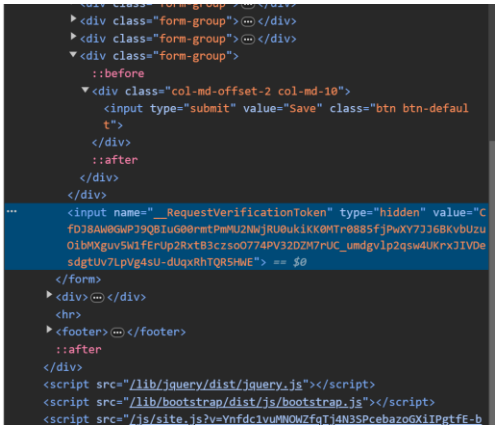
Opdracht:

Onderzoek het sample project van deze week.

Anti forgery token

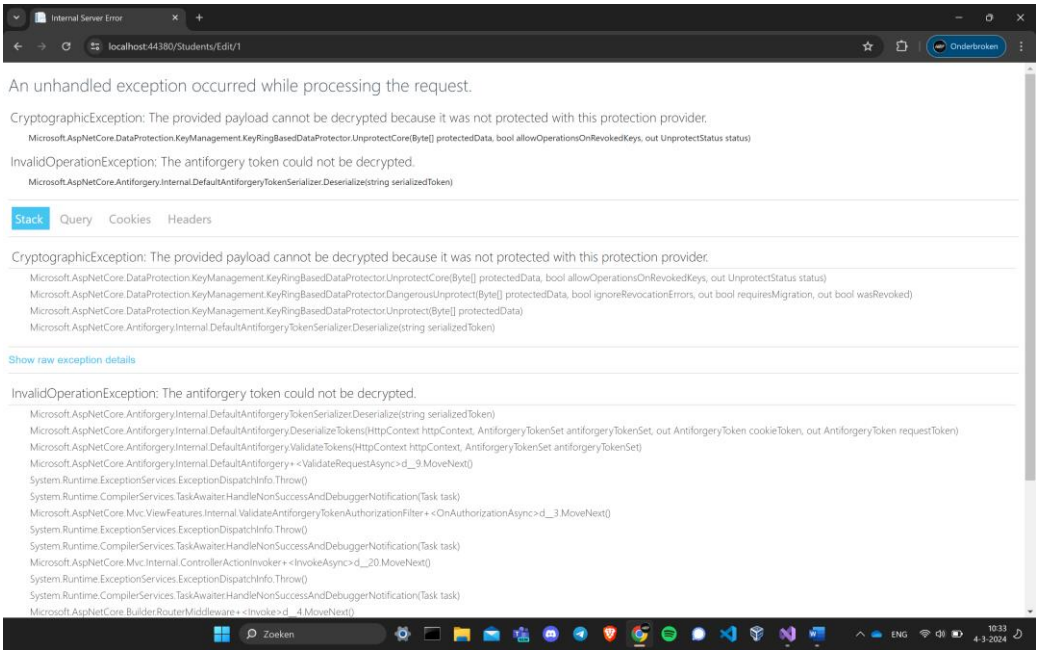
a) *Ga naar wijzigen student. Laat zien dat er een anti forgery token staat in het formulier.*





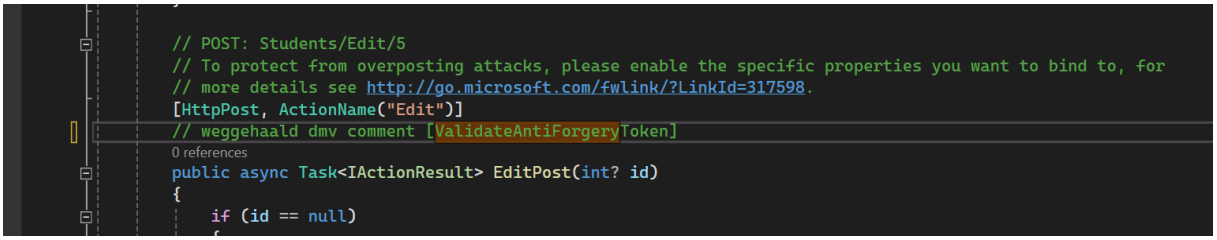
b) Waarom is deze token niet kwetsbaar voor een brute force aanval?
De token wordt willekeurig gegenereerd. Dit is onvoorspelbaar voor de aanvaller. Vervolgens is deze token maar voor een bepaalde duur geldig. Ook is deze token alleen voor de huidige gebruikerssessie geldig.

c) *Wat is het effect als je de token aanpast (via F12)? Hoe antwoord de server?*
Ik heb de token aangepast. Zodra je dan de wijzigingen van een student probeert op te slaan krijg je een foutmelding te zien. Je krijgt de melding “The antiforgery token could not be decrypted.” Vanuit de server. Dit komt omdat de juiste token niet meer aanwezig is.

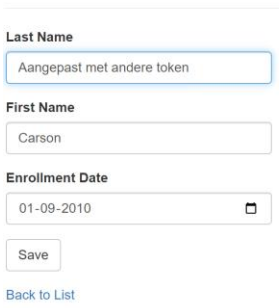


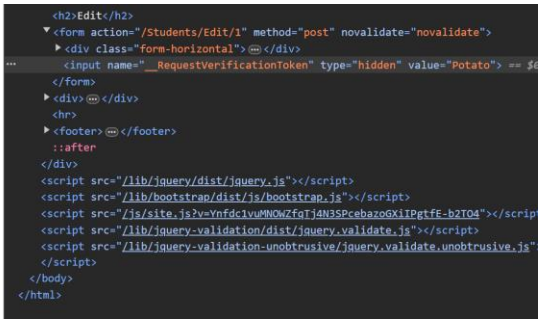
d) Wat is het effect als je de token weghaalt? Hoe antwoord de server?
Zodra je de token weghaalt krijg je een 400 error. Dit komt doordat mijn request dat naar de web server wordt gestuurd, niet correct kan worden verwerkt. Dit is logisch aangezien de website een token verwacht die vervolgens niet mee wordt gegeven.

e) Haal de annotatie weg [validateantiforgery] bij de action methode. Wat gebeurt er nu als je de token aanpast?
Ik heb de annotatie weg gehaald.



Vervolgens heb ik de token aangepast en deze aanpassing van de student opgeslagen.





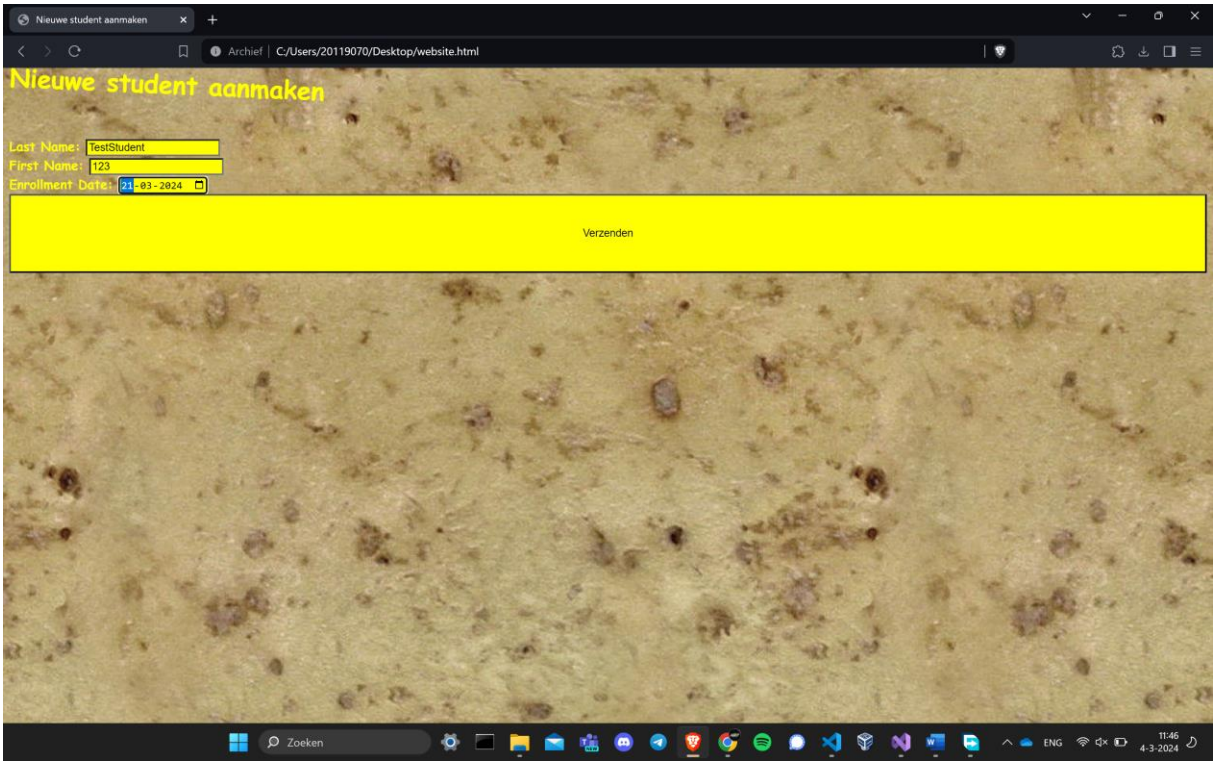
Het lukte vervolgens om deze aanpassingen van de student op te slaan.

Malicious Website

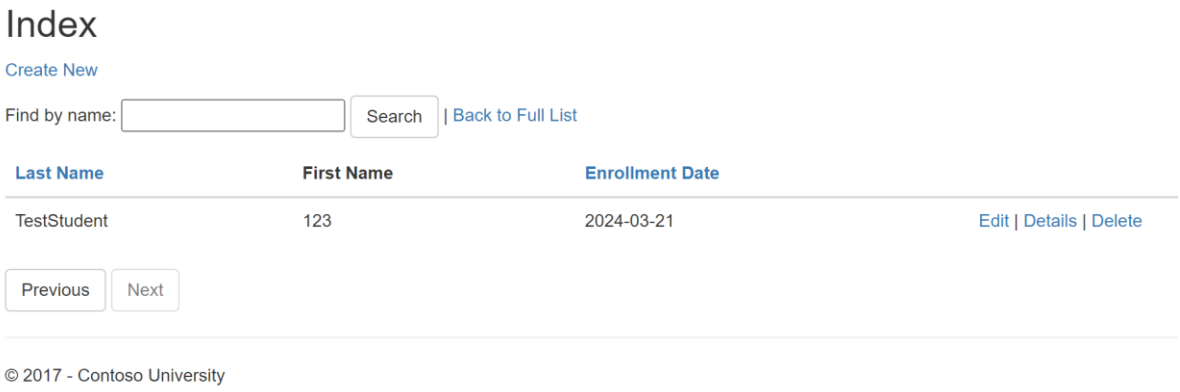
a) *Maak een html pagina met een formulier om een nieuwe student aan te maken. Je kan "spieken" in de Contoso University applicatie welke formulier velden er worden verwacht.*

```
C: > Users > 20119070 > Desktop > website.html > html > body > form
<!DOCTYPE html>
<html lang="nl">
<head>
  <meta charset="UTF-8">
  <title>Nieuwe student aanmaken</title>
  <style>
    html {
        color: yellow;
        font-family: "Comic Sans MS";
        font-weight: bold;
        background-image: url("https://blogger.googleusercontent.com/img/b/R29vZ2xl/AVvXsEhX-PxEiZsjea_WK62BqoSmvwHYyZU2lJXuwFjBUDb0i20vULrA_7P8fr22aWCLgo5blZ9dQUYaQeX
    }
  </style>
</head>
<body>
  <h1 style="transform: rotate(2.5deg);">Nieuwe student aanmaken</h1>
  <form action="https://localhost:44380/Students/Create" method="post">
    <label for="LastName">Last Name:</label>
    <input type="text" name="LastName" id="LastName" required style="background-color: yellow;">
    <br>
    <label for="First Name">First Name:</label>
    <input type="text" name="FirstMidName" id="FirstMidName" style="background-color: yellow;">
    <br>
    <label for="Enrollment Date">Enrollment Date:</label>
    <input type="date" name="EnrollmentDate" id="EnrollmentDate" required style="background-color: yellow;">
    <br>
    <input type="submit" value="Verzenden" style="width: 100%; height: 100px; background-color: yellow;">
  </form>
</body>
</html>
```

b) *Open deze pagina in de browser via het file protocol. Kan je via dit formulier nu een nieuwe student toevoegen in de Contoso University database?*
Ik heb de informatie van de nieuwe student ingevuld en erna op verzenden gedrukt.



Deze student was vervolgens terug te vinden op de website. De student is succesvol toegevoegd aan de database.

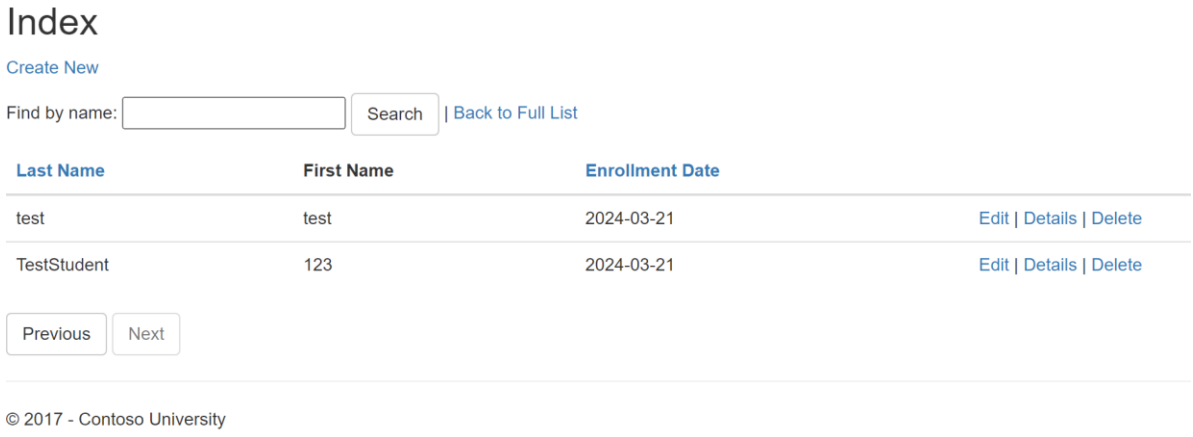


c) *Maak nu een verborgen formulier waarbij de eindgebruiker op een onschuldige knop klikt. Geef de knop een naam die uitlokt om er op te klikken*

Eerst heb ik een nieuwe form aangemaakt. Bij de input kan er door gebruik te maken van "value=..." een standaard waarde worden meegegeven. Nu hoeft er niks worden ingevuld en is de knop voldoende om de student aan te maken. Verder heb ik de labels weggehaald aangezien deze hier niet nodig zijn.

```
<form action="https://localhost:44380/Students/Create" method="post">
  <input type="text" name="LastName" value="test" id="LastName" required style="background-color: yellow;">
  <br>
  <input type="text" value="test" name="FirstMidName" id="FirstMidName" style="background-color: yellow;">
  <br>
  <input type="date" name="EnrollmentDate" value="2024-03-21" id="EnrollmentDate" required style="background-color: yellow;">
  <input type="submit" value="Klik op mij (ik ben veilig)" style="width: 100%; height: 100px; background-color: yellow;">
</form>
```

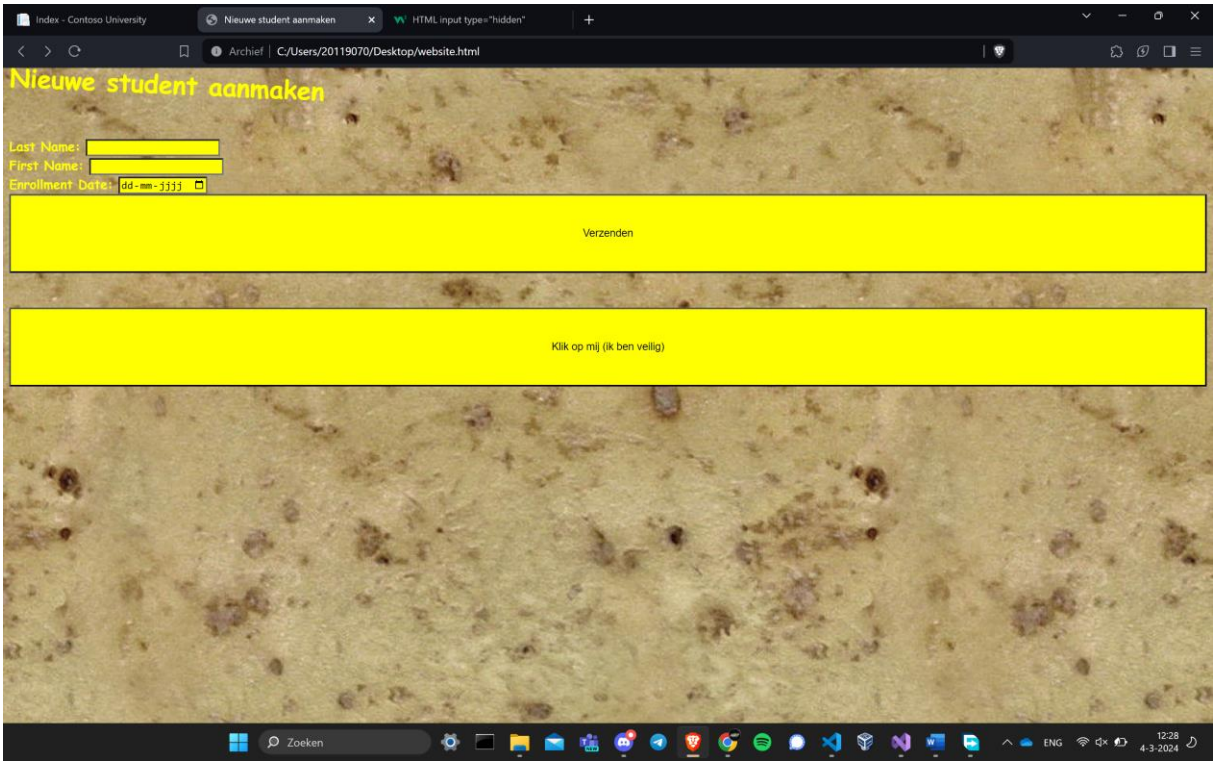
Dit wilde ik testen voordat ik de velden zou verbergen. Toen ik op de knop drukte werd er een student aangemaakt. Dit werkt dus. Nu nog de fields verbergen.



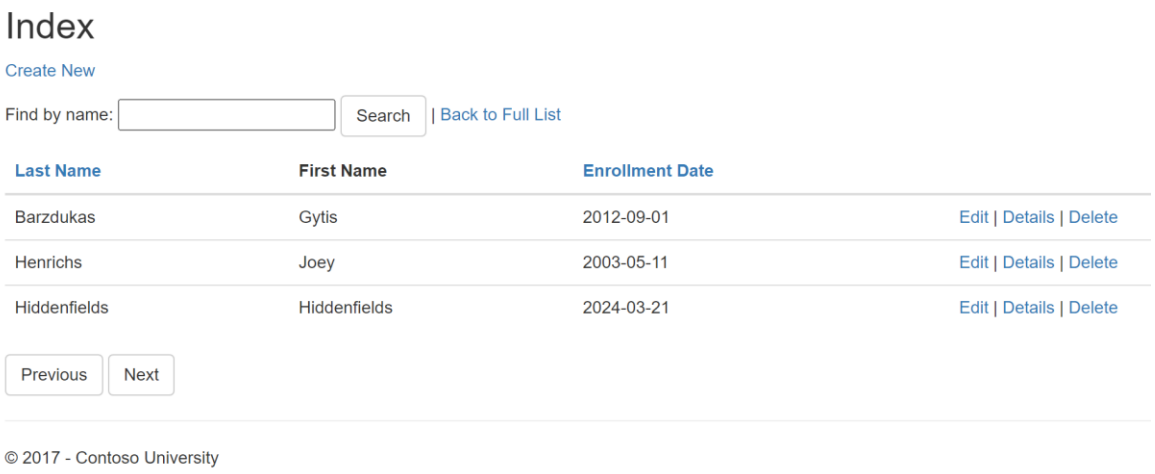
Ik heb de input type aangepast naar hidden. Dit zou voldoende moeten zijn om ze te verbergen.

```
<form action="https://localhost:44380/Students/Create" method="post">
  <input type="hidden" name="LastName" value="Hiddenfields" id="LastName" required style="background-color: yellow;">
  <br>
  <input type="hidden" value="Hiddenfields" name="FirstMidName" id="FirstMidName" style="background-color: yellow;">
  <br>
  <input type="hidden" name="EnrollmentDate" value="2024-03-21" id="EnrollmentDate" required style="background-color: yellow;">
  <input type="submit" value="Klik op mij (ik ben veilig)" style="width: 100%; height: 100px; background-color: yellow;">
</form>
```

Vervolgens waren deze velden op de pagina verdwenen. Er was alleen nog maar de knop over van deze form. Ik heb er op geklikt om het vervolgens te testen.

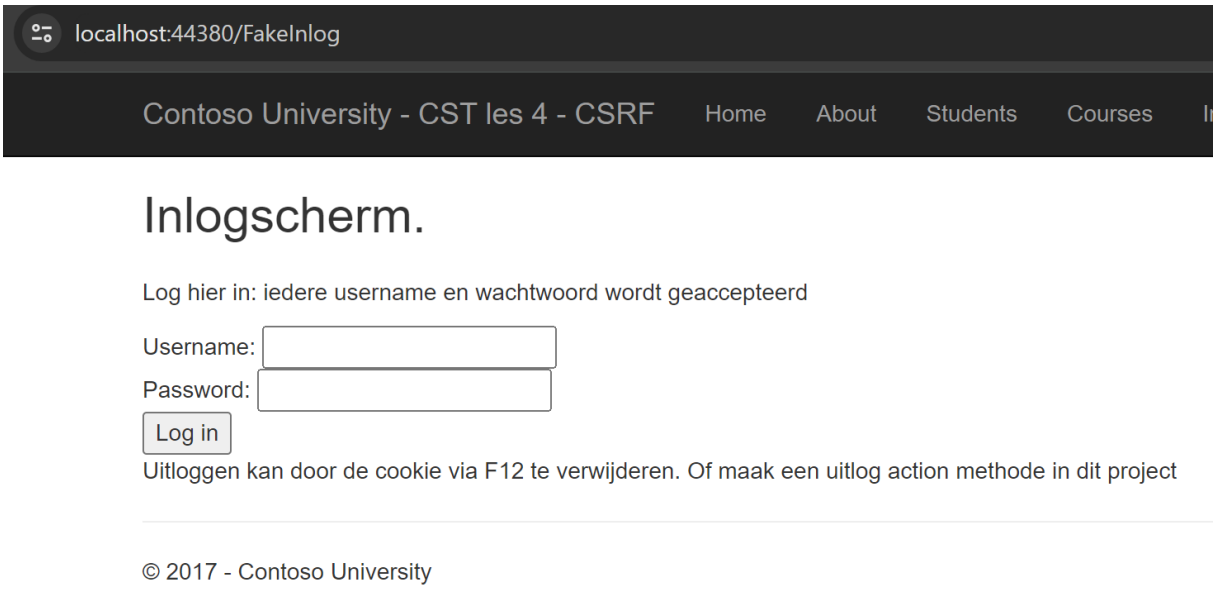


Zodra ik op de knop drukte was de student aangemaakt. De student met de naam “hiddenfields” is succesvol toegevoegd.



Cookies en SameSite:

a) *Ga naar de about pagina. Wanneer kan je deze pagina zien?*
Je kunt deze pagina zien zodra je bent ingelogd of de cookie hebt.

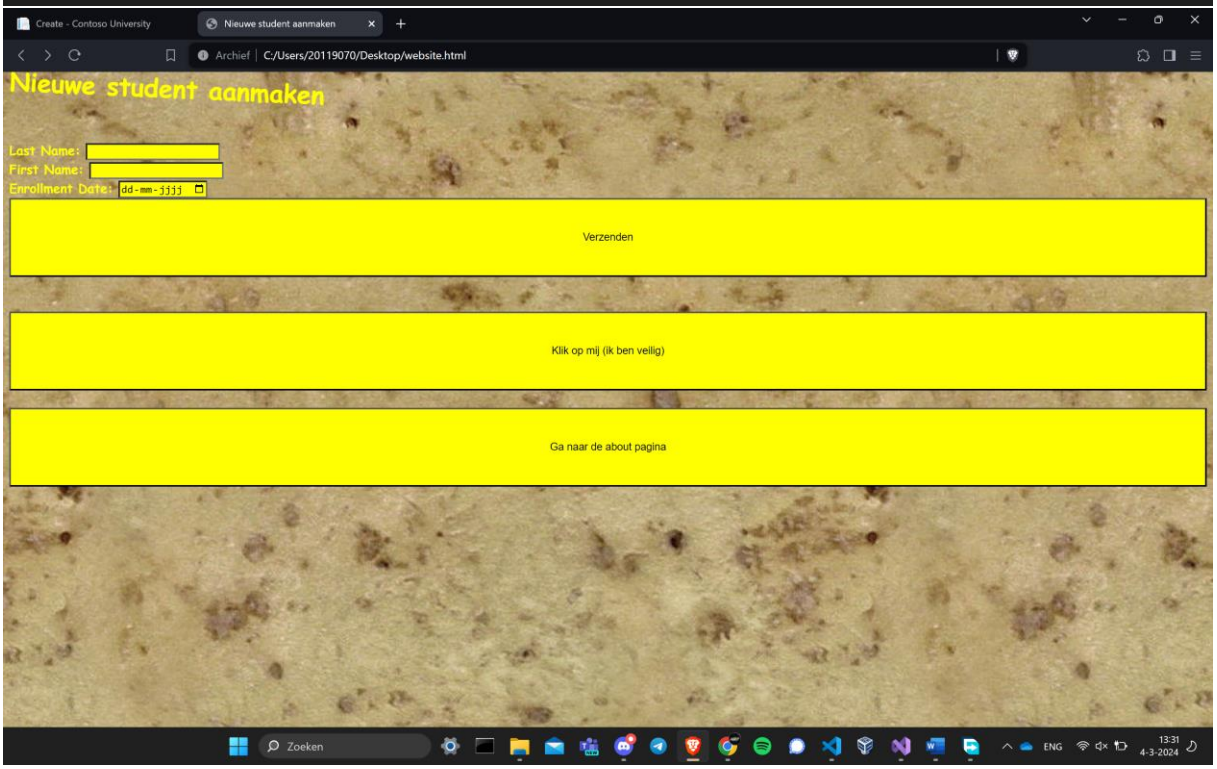


b) *Welke code in de action method is toegevoegd?*
De volgende code is toegevoegd in de methode. Hierin wordt een cookie aangevraagd bij het inloggen. Zodra er al een cookie aanwezig is wordt deze vertoond op het scherm.

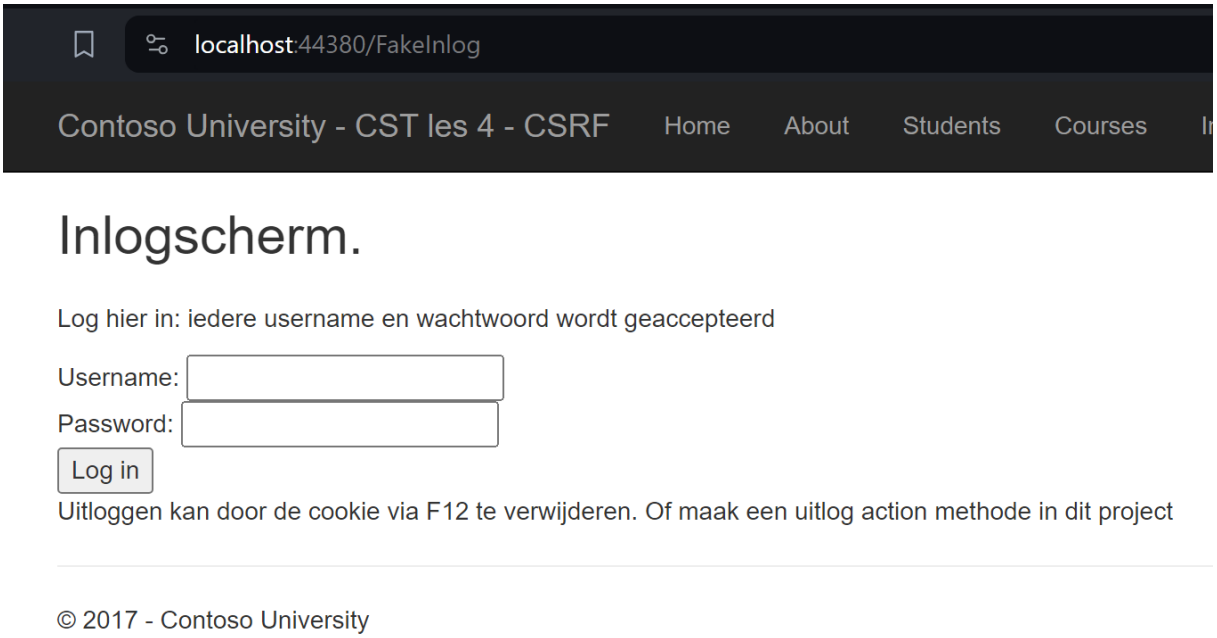
```
// GET: /<controller>/
0 references
public IActionResult Index()
{
    string naam = Request.Cookies.Where(c => c.Key == "FakeAuthId").FirstOrDefault().Value;
    if (naam!=null) { ViewData["Message"] = "U bent al ingelogd. De FakeAuthID cookie is " + naam; }
    return View();
}
```

c) *Maak een html pagina met een link naar de about pagina. Open deze in de browser via het file protocol. Dit is goed genoeg. Wanneer zie je about pagina en wanneer niet.*

```
<form action="https://localhost:44380/Home/About" method="get">
  <br>
  <input type="submit" value="Ga naar de about pagina" style="width: 100%; height: 100px; background-color: yellow;">
</form>
```

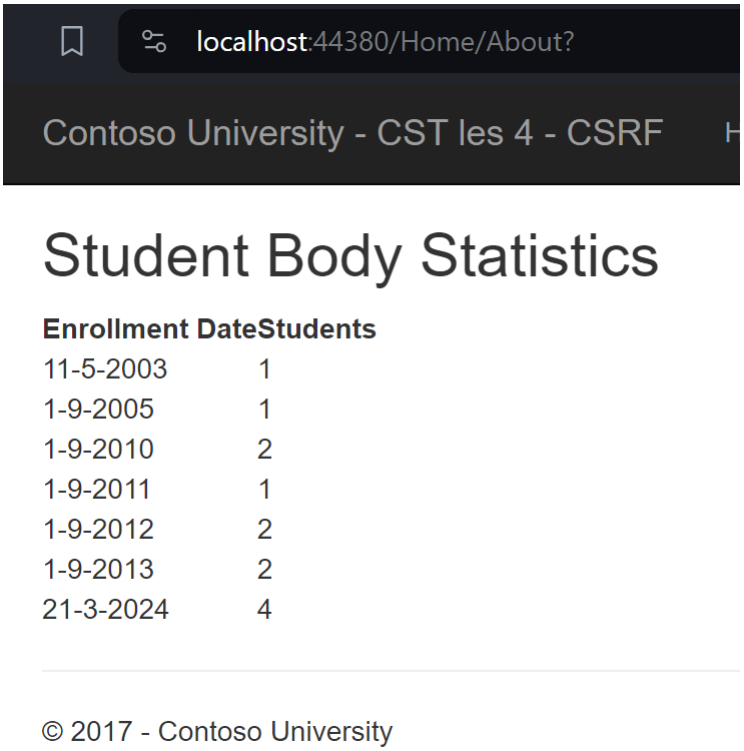


Zonder een aanwezig cookie door een vorige login:
Je krijgt inlogscherm te zien.



Om nu met een cookie op deze pagina te komen heb ik eerst een inlogpoging op de normale pagina gedaan.

Na een eerdere inlogpoging:
Je krijgt dan de inhoud van de about pagina te zien.

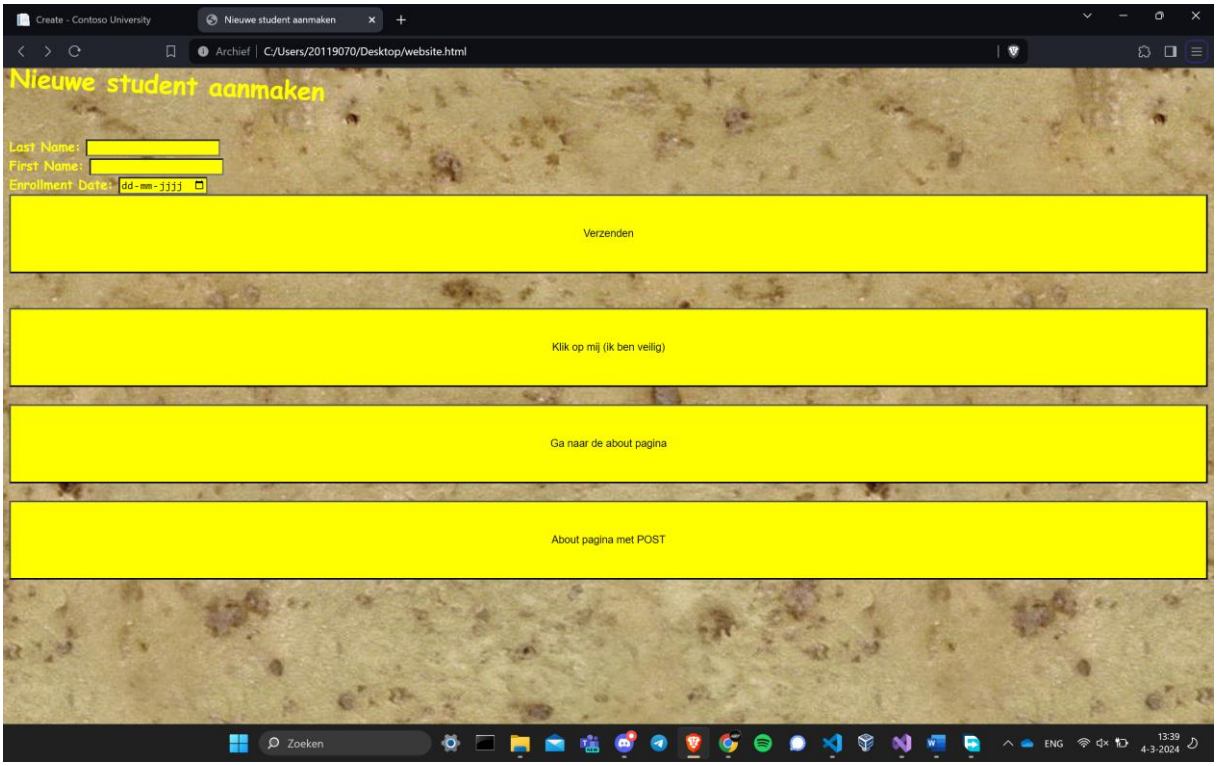


d) *Leg uit hoe dit komt.*
De redirect werkt niet zodra er geen (geldige) cookie is opgeslagen in de browser. Dit komt omdat er geen cookie aanwezig is waarvan gebruik gemaakt kan worden om in te loggen op de pagina. Dit resulteert in een prompt om alsnog in te loggen
Zodra er een cookie is opgeslagen in de browser die geldig is, lukt het je om met een redirect op de pagina te komen. Dit komt doordat we de cookie van de pagina gebruiken om in te loggen. Je krijgt de pagina te zien.

e) *De about pagina is ook bereikbaar via Http Post. Maak nu een leeg formulier met submit button op je eigen html pagina die submit naar about pagina. Kan je nu de about pagina zien?*

Ik heb de code van de vorige form gebruikt, de methode aangepast en de tekst.

```
<form action="https://localhost:44380/Home/About" method="post">
  <br>
  <input type="submit" value="About pagina met POST" style="width: 100%; height: 100px; background-color: yellow;">
</form>
```

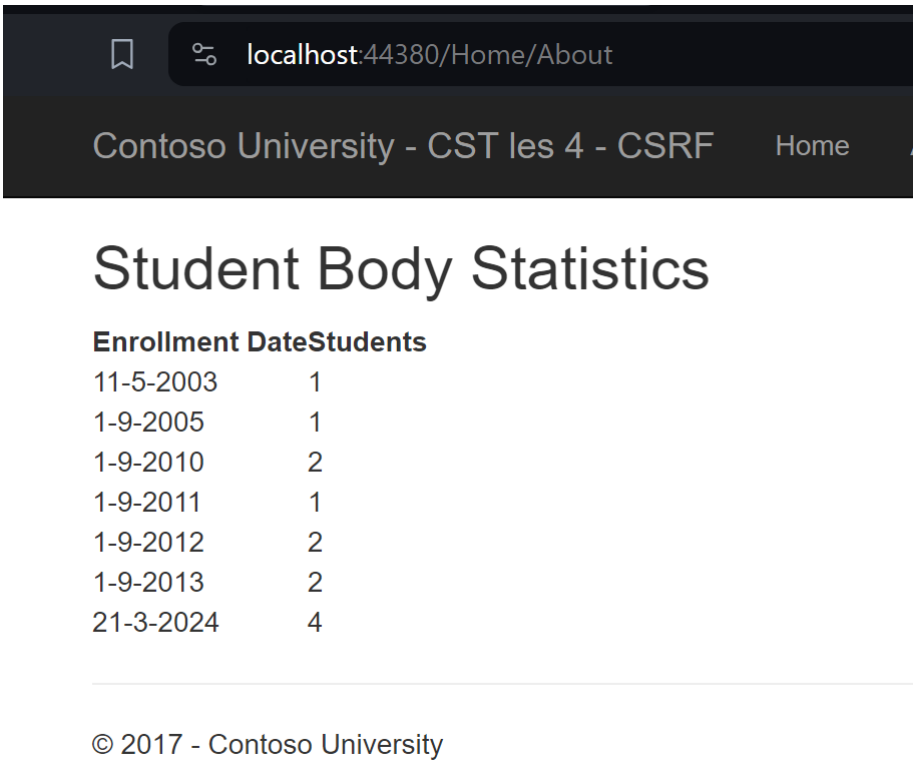


Zodra je erop klikt, krijg je het volgende te zien:

Dit komt doordat de cookie niet aanwezig is.

Vervolgens heb ik gekeken wat er zou gebeuren als ik eerst zou inloggen en dus zo een cookie aanmaken, gevolgd door de redirect met een post request. Ik kreeg het volgende te zien:

Ik kon de about pagina zien.

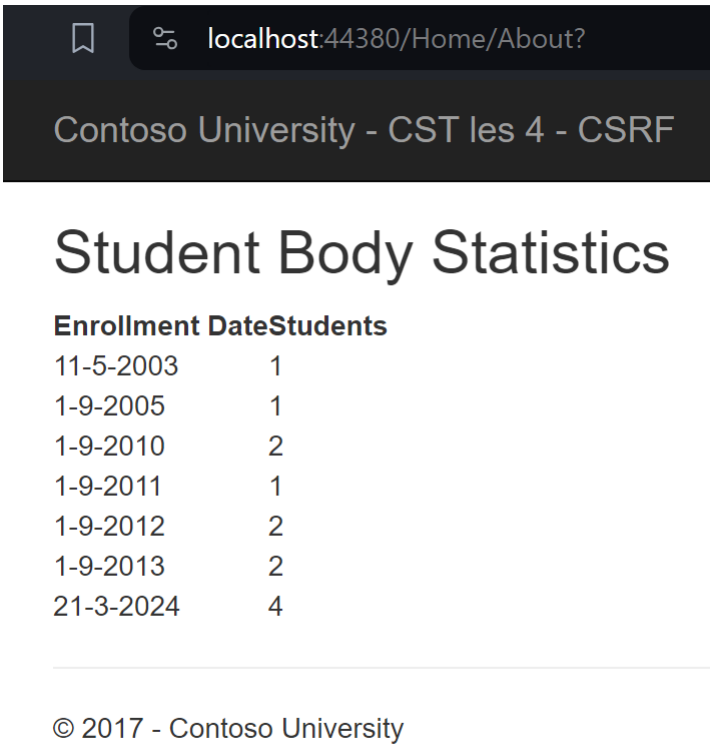


f) *In de fakeinlogcontroller kan je de SameSite Atrribute. Zet deze op Strict of Lax. Test effect als je nu about pagina probeert te openen vanuit je eigen html pagina.*
De code was eerst uitgecomment. Zodra je de code weer op actief zet, staat de SameSite Attribute op Strict.

Ik ben ik begonnen met het testen van Lax.

```
cookieOptions.Path += "; SameSite=Lax"; //workaround for version 1.1
```

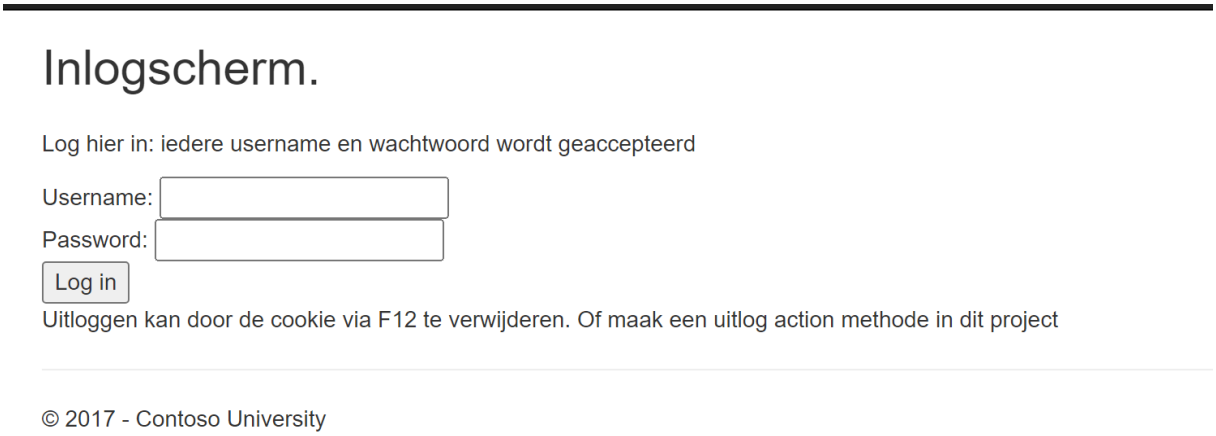
Zodra je inlogt op de pagina, wordt er een cookie aangemaakt. Ik heb dan 2 minuten om gebruik te maken van deze cookie met mijn redirect. De redirect laat mij dan ook de pagina zien zolang de cookie geldig is.



Hierna ben ik gaan testen op Strict.

```
cookieOptions.Path += "; SameSite=Strict"; //workaround for version 1.1
```

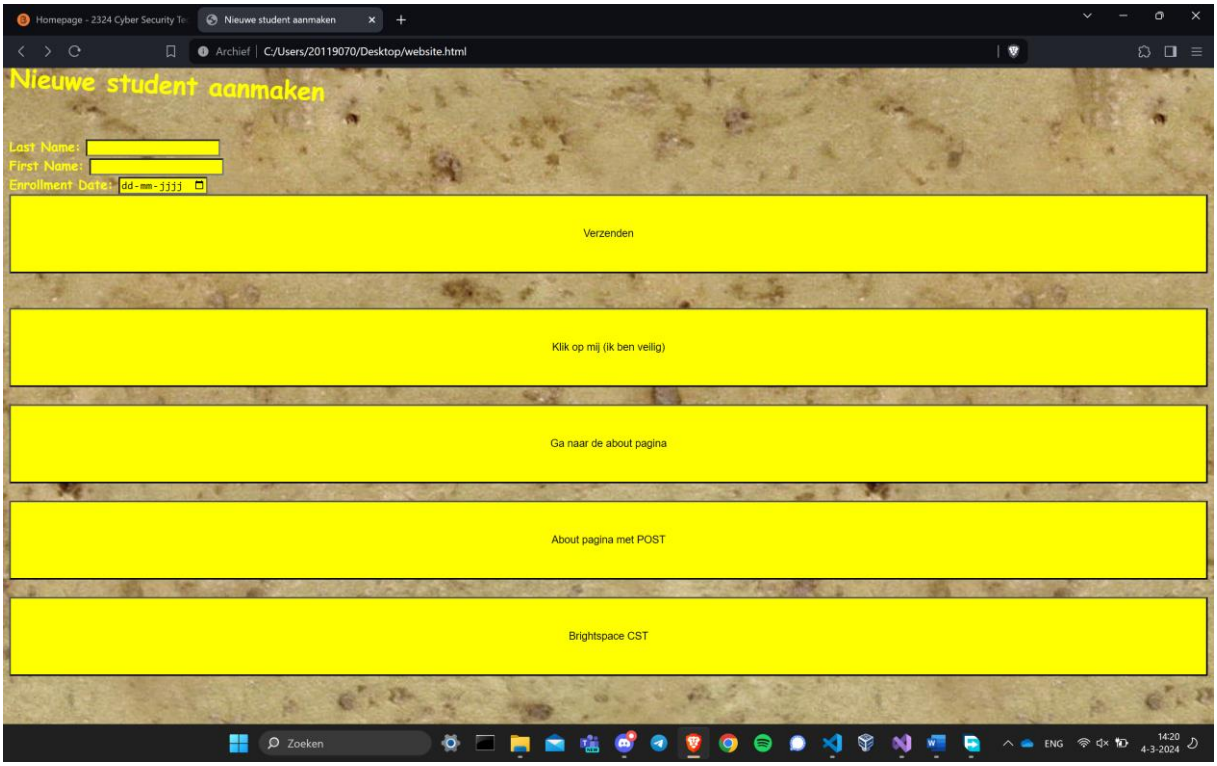
Ik ben eerst ingelogd op de pagina. Hierna heb ik de redirect gebruikt naar de about pagina. Ik krijg dat de inlog pagina te zien. Dit komt omdat met strict de cookies niet werken vanuit een redirect. Het was dan ook te voorspellen dat dit zou gebeuren.



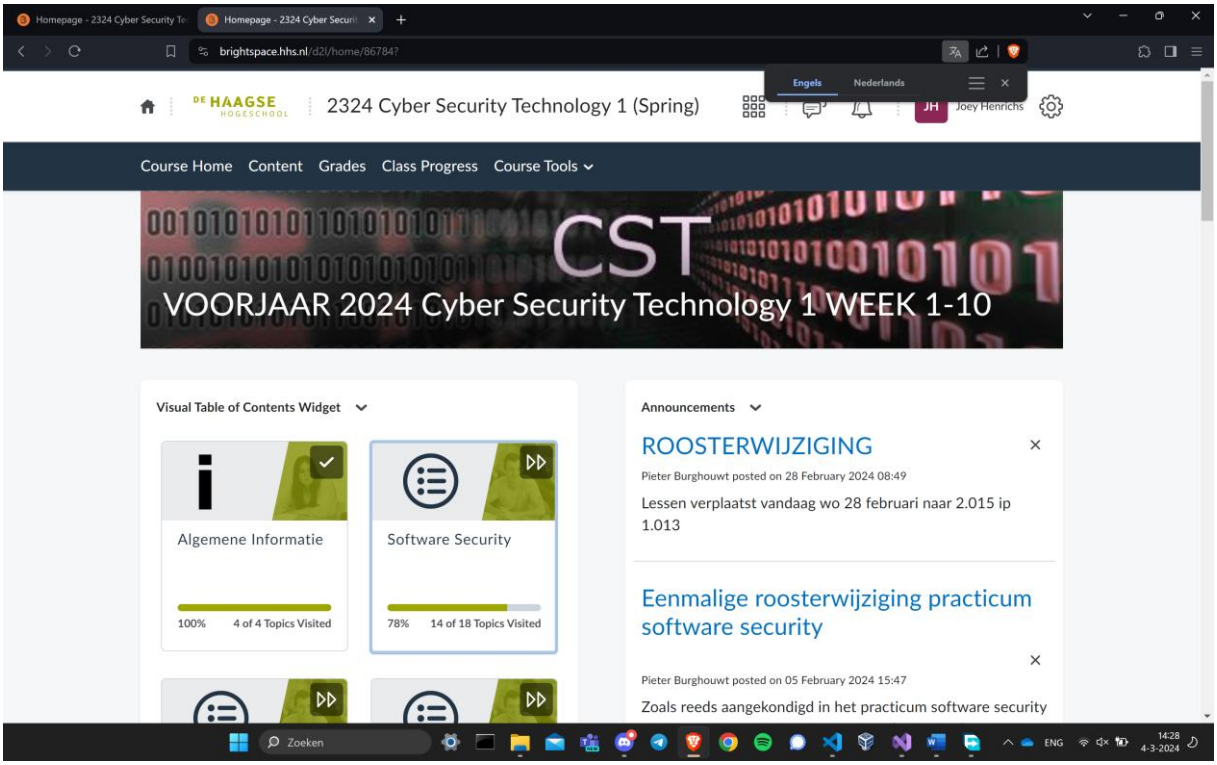
g) *Voeg een link op je eigen pagina toe naar de CST course op BrightSpace. Waarom werkt ook deze deep linking?*

Ik heb een nieuwe form aangemaakt met een knop naar de brightspace pagina.

```
<form action="https://brightspace.hhs.nl/d2l/home/86784" method="get">
  <br>
  <input type="submit" value="Brightspace CST" style="width: 100%; height: 100px; background-color: yellow;">
</form>
```



Zodra je op de knop drukt krijg je de brightspace pagina te zien. Dit komt doordat er geen SameSite Attribute aanwezig is. Deze staat dan automatisch op lax wat redirects cookies laat gebruiken. Dit houd in dat je met een redirect gebruik kan maken van de aanwezige cookie.



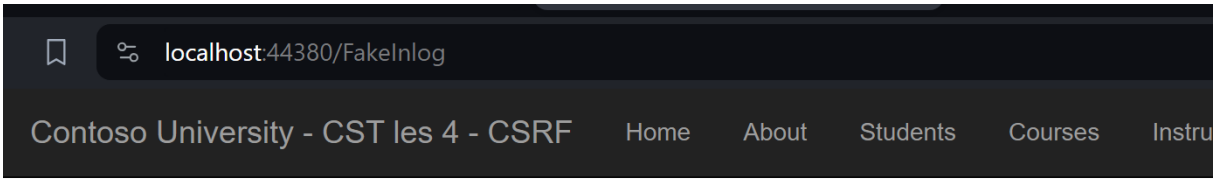
CSRF

Bij CSRF combineer je dat je een submit mag doen vanaf een andere website en dat de authenticatie cookie automatisch meegestuurd wordt

a) *Edit de html pagina die je eerder hebt gemaakt met een formulier om een nieuwe student aan te maken. Post nu echter naar de create2 action methode. In deze action methode wordt ook gecontroleerd op de "authenticatie cookie". Zet geen SameSite attribute in de fakeinlogcontroller. (Ga dus terug naar de oorspronkelijke code)*
Ik heb de mogelijkheid toegevoegd om een student aan te maken met de create2 action methode.

```
<h1>Create2</h1>
<form action="https://localhost:44380/Students/Create2" method="post">
  <label for="LastName">Last Name:</label>
  <input type="text" name="LastName" id="LastName" required style="background-color: yellow;">
  <br>
  <label for="First Name">First Name:</label>
  <input type="text" name="FirstMidName" id="FirstMidName" style="background-color: yellow;">
  <br>
  <label for="Enrollment Date">Enrollment Date:</label>
  <input type="date" name="EnrollmentDate" id="EnrollmentDate" required style="background-color: yellow;">
  <br>
  <input type="submit" value="Verzenden" style="width: 100%; height: 100px; background-color: yellow;">
  <br>
</form>
```

b) Open het formulier daarna in de browser via het file protocol. Laat zien dat je dat nu niks kan toevoegen als je niet ingelogd bent maar dit wel lukt als je al eerder als gebruiker bent ingelogd. Dit is dus toevoegen via een "malicious" website
Zodra je de post request op create 2 probeert te maken zonder ingelogd te zijn krijg je de fakeinlog pagina te zien met "u bent al ingelogd". Je kunt zo geen student aanmaken.



Inlogscherm.

U bent al ingelogd. De FakeAuthID cookie is Welkom123

Log hier in: iedere username en wachtwoord wordt geaccepteerd

Username:

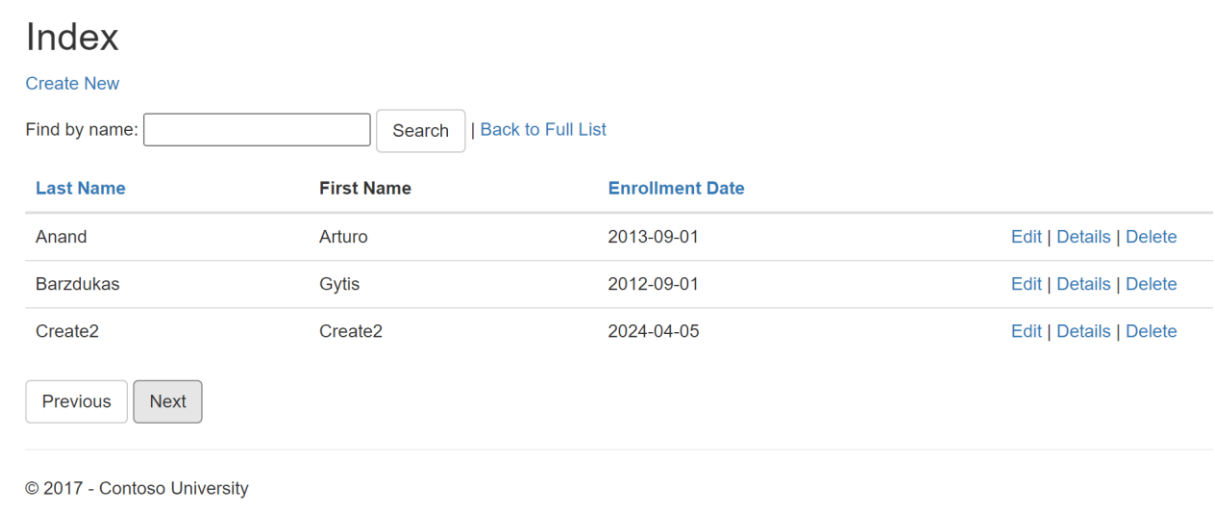
Password:

Log in

Uitloggen kan door de cookie via F12 te verwijderen. Of maak een uitlog action methode in dit project

© 2017 - Contoso University

Zodra je al ingelogd lukt het echter wel om een student aan te maken via create2.



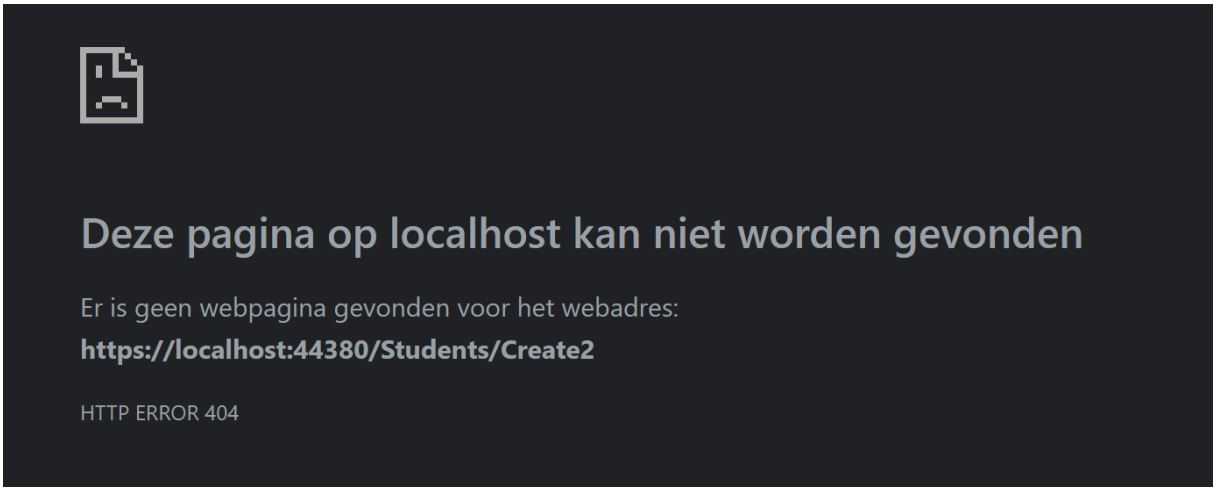
c) *Waarom werkt dit toevoegen via een andere website niet meer als bij create2 het valideren van een anti forgery token aanstaat? Test dit ook*

Antiforgery aangezet door de comments weggehaald zodat de code runt.

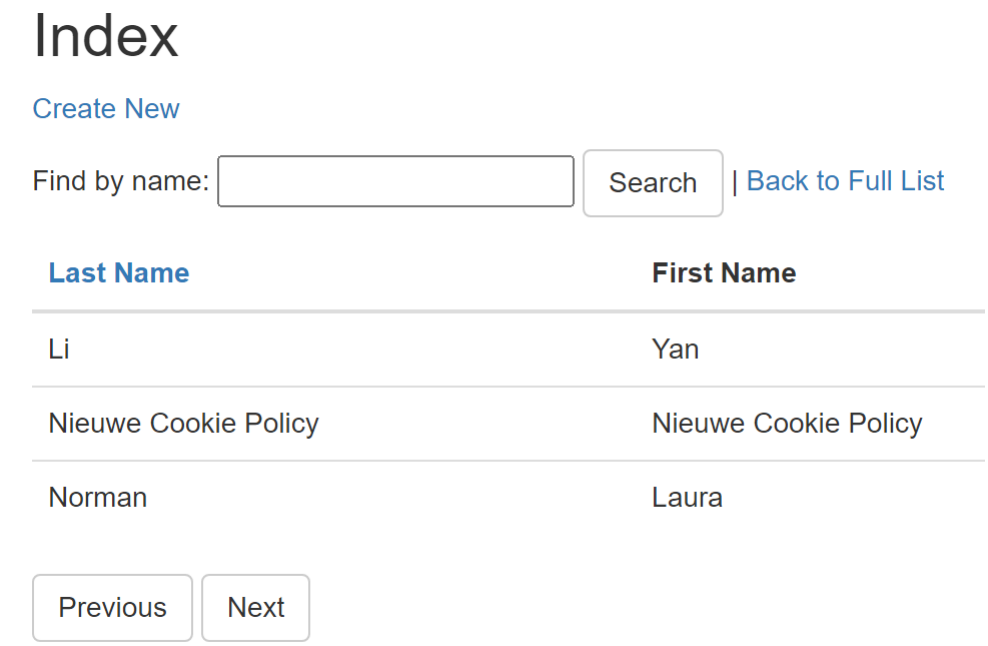
```
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public async Task<IActionResult> Create2(
    [Bind("EnrollmentDate,FirstMidName,LastName")] Student student)
{
    string naam = Request.Cookies.Where(c => c.Key == "FakeAuthId").FirstOrDefault().Value;
    if (naam == null) return Redirect("/FakeInlog");

    try
```

Je krijgt gelijk een 400 error. Dit komt omdat de token niet overheen komt met wat de server verwacht.



d) *Zet de validatie van het anti forgery token weer uit. Chrome heeft vanaf versie 80 de policy verandert voor het meesturen van cookies. In de action methode voor het inloggen werd tot nu toe het SameSite attribute niet gezet. Test of je nu wel of niet via je eigen formulier een student kan toevoegen? NB: Denk aan het 2 minuten interval van Chrome.*
Dit lukt. Met de verandering van de policy is lax nu de standaard. Dit houd in dat je 2 minuten hebt om je cookie te gebruiken met je eigen redirects. Het lukt dan om in te loggen en een student aan te maken.



e) Als je SameSite=Lax, werkt dan via je eigen “malicious” website voor het aanmaken van een student?
Nee dit werkt niet. Je krijgt het inlogscherm te zien.

f) Dezelfde vraag maar nu als je naar create3 submit?
Ik heb een nieuw form aangemaakt.

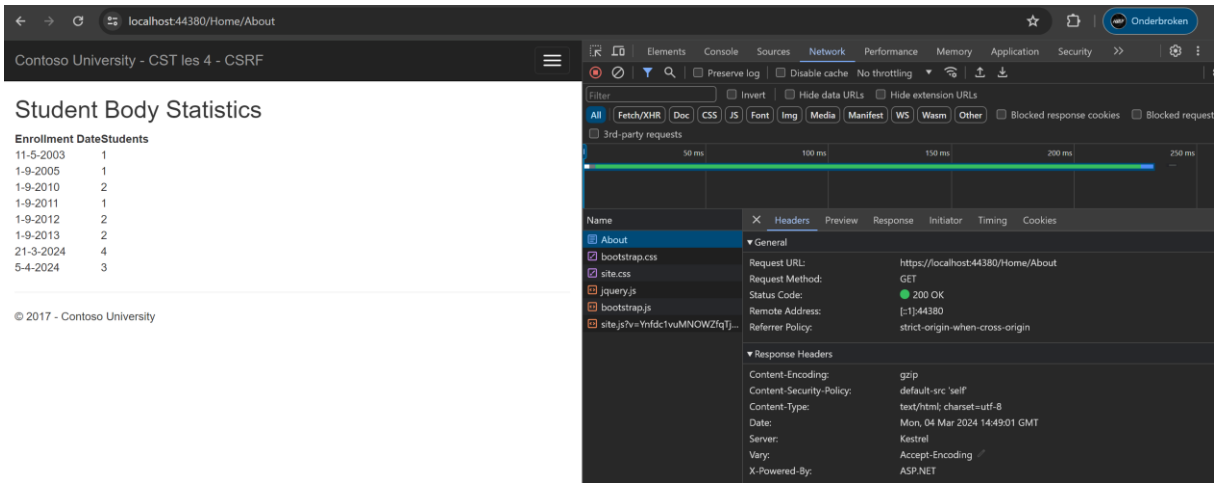
```
<h1>Create3</h1>
<form action="https://localhost:44380/Students/Create3" method="post">
  <label for="LastName">Last Name:</label>
  <input type="text" name="LastName" id="LastName" required style="background-color: yellow;">
  <br>
  <label for="First Name">First Name:</label>
  <input type="text" name="FirstMidName" id="FirstMidName" style="background-color: yellow;">
  <br>
  <label for="Enrollment Date">Enrollment Date:</label>
  <input type="date" name="EnrollmentDate" id="EnrollmentDate" required style="background-color: yellow;">
  <br>
  <input type="submit" value="Verzenden" style="width: 100%; height: 100px; background-color: yellow;">
  <br>
</form>
```

Via mijn eigen site probeerde ik een student aan te maken via create3.

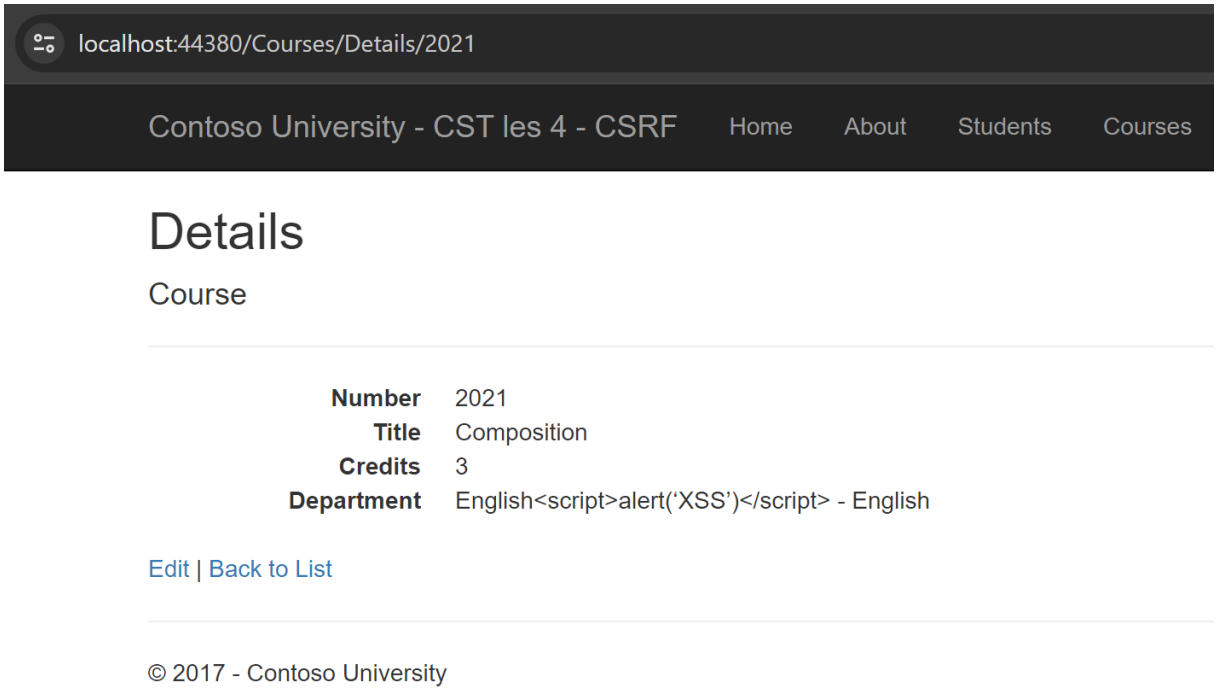
Dit gaf een 404 foutmelding.

Content Security Policy

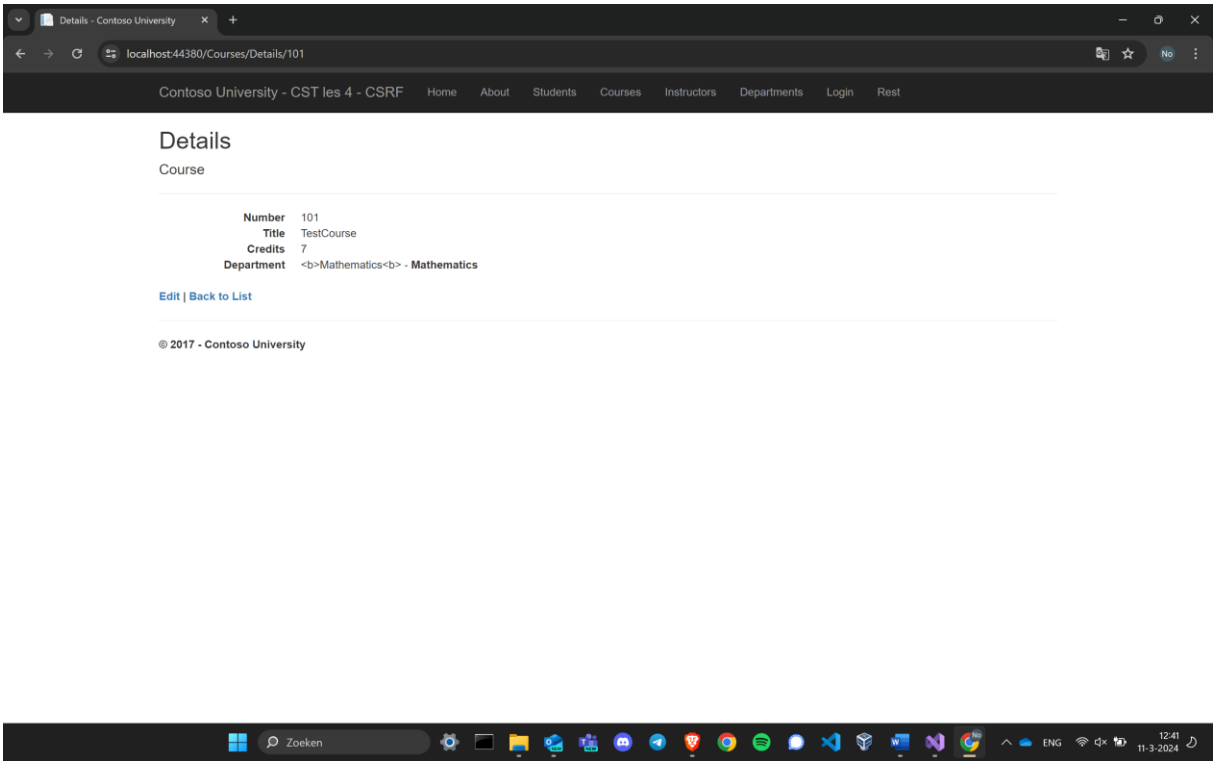
a) Hoe kan je via F12 zien dat er een Content Security Policy is ingesteld? En welke is dit?
Dit kan je via de netwerk tap zien en dan onder response headers.
Dit is “default-src 'self”



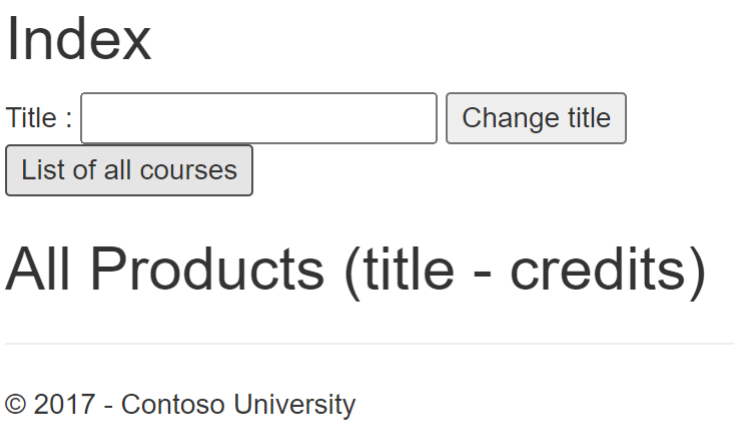
b) *Demonstreer het effect van deze policy voor de gevonden server side XSS die werkte met Html.Raw uit practicum 2 waarbij je een Department naam had gewijzigd.*
Met CSP kan je voorkomen dat inline scripts kunnen worden gerund. De XSS alert werkt dan niet.



c) *Wat gebeurt er als je in de Department naam de <b> tag gebruikt*
De department naam wordt dikgedrukt.



d) *Welk effect heeft deze Content Security Policy als je naar de menu optie Rest gaat? En waarom is dit zo?*
De knop list of all courses werkt niet aangezien inline javascript uit staat.



e) *Hoe moet een developer het probleem van vraag d oplossen? Zie een link in de sheets voor de oplossing.*
Je kunt javascript van een aparte file inladen en dan gebruik maken van listeners.
https://web.dev/articles/csp#inline-code-is-considered-harmful