UserStories Part 2 – (Cards, Icons og Delete UserStory)

Denne opgave bygger videre på UserStories Part 1. I denne opgave skal der styles vha Bootstrap Cards og der introduceres brug af Icons fra font-awsome til en button for Delete UserStory.

Udgangspunkt

I UserStories Part 1 blev basis implementeret og siden så ud som følger:

Hvor alle UserStory blev vist i en simpel ”unorded list”.

Card-Layout

1. **Card Intro**
   Læs om Bootstrap Card: https://getbootstrap.com/docs/4.0/components/card/

2. **UserStory som Card**
   I Bootstrap findes bl.a. klasserne: “card”, “card-header”, “card-title”, “card-body”, “card-text”, “card-link” og “card-footer”. De kan benyttes til at style UserStory-objekterne som Cards. Prøv i første omgang at udskifte den “unorded list” med “card” ala:

```
<div class="card">
    <div class="card-eader">
       <h5 class="card-title">(@userstory.Id) @userstory.Title</h5>
    </div>
    <div class="card-body">
       <p class="card-text">@userstory.Description</p>
    </div>
</div>
```

3. **Afprøvning**
   Kør programmet og verificer at du får noget ala

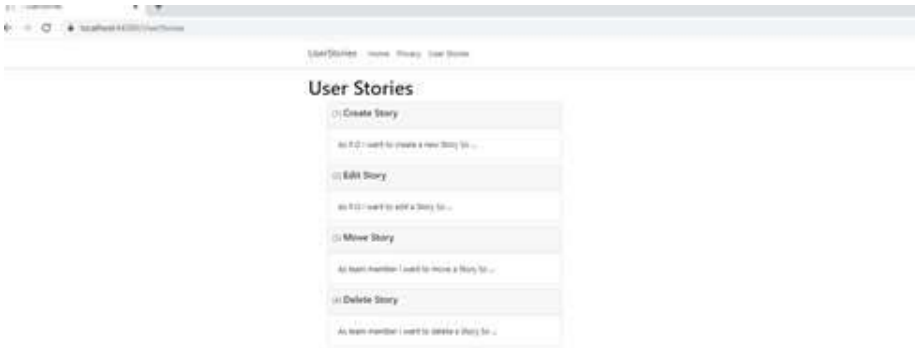
4. **Yderligere styling med Spacing (margin m og padding p)**

   Se: https://getbootstrap.com/docs/4.0/utilities/spacing/

   Benyt Bootstrap klasserne *m* og *p* (fx *mb-2* for magin bottom spacer 2) til at style dit card (prøv med forskellige værdier og se hvordan det virker) fx:

```
div class="card mb-2" style="width: 30rem;">
        <div class="card-header p-2">
```

   Bemærk *style=" width: 30rem;"* er en inline styling der sætter bredden på card´et til 30rem, hvor rem står for relativ font-size i forhold til root-element (deafault er html root-element font-size 16px) - ovenstående "styling" giver følgende:



5. **Card-footer med delete-icon (trash icon fra font-awesome)**
   Den næste udvidelse er en tilføjelse af en footer til vores card, der indeholder et Delete Icon.

   a. Første step er at tilføje et cdn link til font-awesome (stylesheet med en masse gratis ikoner). Da vi skal bruge ikoner på flere sider - tilføjes nedenstående link til layoutsiden (*_Layout.cshtml*)

```
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
```

b. Næste step er at der til card tilføjes en “card-footer” indeholdende et icon (her er benyttet class: “fa fa-trash" der er stylet af font-awesome):

```
<div class="card-footer">
        <i class="fa fa-trash"></i>
</div>
```

c. Afprøv - det skulle gerne se ud ala:

Delete UserStory

Næste udvidelse er muligheden for at slette et card (UserStory) ved at klikke på delete-ikonet. Ideen er at ikonet “wrappes” ind i et anchor-tag af typen “button” og med tag-helperne “asp-page” og “asp-route-id" :

```
<a class="btn" type="button" asp-page="DeleteUserStory" asp-route-id="@userstory.Id"><i class="fa fa-trash"></i></a>
```

Når der klikkes på den nye “button” med “trash”-ikonet, vil der sendes et “http get-request" til serveren med følgende route: “DeleteUserStory/id”. Det betyder at metoden OnGet(id) på page DeleteUserStory bliver kaldt.

6. UserStoryService

Udvid UserStoryService med en metode der kan finde en UserStory ud fra et givet id:

```
public UserStory GetUserStory(int id)
{
    foreach (UserStory userStory in userStories)
    {
        if (userStory.Id == id)
            return userStory;
    }
    return null;
}
```

Samt en metode der kan slette en UserStory:

```
public UserStory DeleteUserStory(int userstoryId)
{
        UserStory userstoryToBeDeleted = null;
        foreach (UserStory us in userStories)
        {
           if (us.Id == userstoryId)
           {
              userstoryToBeDeleted = us;
              break;
           }
        }
        if (userstoryToBeDeleted != null)
        {
           userStories.Remove(userstoryToBeDeleted);
        }
        return userstoryToBeDeleted;
 }
```

b. Gennemgå metoden og diskuter hvad den gør og hvorfor den ser ud som den gør! Kan den skrives kortere?

7. Opret en ny Razor Page (empty) – DeleteUserStory

DeleteUserStory.cshtml.cs
Når der skal slettes en Userstory skal vi først bruge en reference til det objekt der skal slettes. Derfor skal *DeleteUserStoryModel* have en property *UserStory*. For at kunne referere til propertyen fra pagen skal der også tilføjes en binding *[BindProperty]*:

```
[BindProperty]
public UserStory UserStory { get; set; }
```

For at kunne benytte de nye metoder fra Servicen, skal Pagen skal have en reference (et instans felt) til UserStoryService:

```
private UserStoryService userStoryService;
```

og denne skal injiceres via konstruktøren:

```
public DeleteUserStoryModel(UserStoryService userStoryService)
{
this.userStoryService = userStoryService;
}
```

OnGet(int id) er metoden der kaldes når der trykkes på knappen med “trash”-ikonet - den skal hente den UserStory der skal slettes vha Servicen:

```
public void OnGet(int id)
{
        UserStory = userStoryService.GetUserStory(id);
}
```

DeleteUserStory.cshtml

Det er god kutyme ikke at slette et objekt uden bekræftelse - denne feature kan implementeres med følgende form-tag:

```
@page "{id:int}" //Da vi skal sende id med som parameter via routen: DeleteUserStory/id
@model UserStories.Pages.DeleteUserStoryModel
@{
}
<h1>Delete Confirmation</h1>
<div class="alert alert-danger">
   <h5>Are you sure you want to delete the UserStory - @Model.UserStory.Title</h5>
   <form method="post">
      <button type="submit" class="btn btn-danger">Yes</button>
      <a class="btn btn-primary" asp-page="Index">No</a>
   </form>
</div>
```

Det ser således ud i browseren:

DeleteUserStory.cshtml.cs

Når bruger trykker på ”Yes-knappen” sendes et ”http-post request” til serveren og metoden OnPost() kaldes:

```
public IActionResult OnPost(int id)
{
        UserStory deletedUserStory = userStoryService.DeleteUserStory(id);
        return RedirectToPage("UserStories");
}
```

OnPost() kalder DeleteUserStory-metoden, der sletter den valgte UserStory og returnere til siden UserStories.

8. Afprøv