

Timebox 2 – Personal information

Oversigt

OpgaveNavn	Implementering af "personal information" siden		
Implementering af krav	Delvis implementering af kravet WEB-4		
Udført af	Marc	Dato	29-09-2021
Timebox	2	Område	Website

Contents

INTRODUKTION.....	1
ANALYSE OG DESIGN.....	2
IMPLEMENTERING	2
VERIFIKATION	5
TESTRESULTAT.....	5
KONKLUSION	6
REFERENCER	7

Introduktion

Færdiggørelsen af hjemmesidens "personal information" for både "field_user" og "office_user", hvor der implementeres funktionerne som at brugeren skal kunne redigere, brugerens egne information, og ændre eget kodeord.

Analyse og Design

Der videre udvikles fra timebox 1 hjemmesidens sider "office_user_info" og "field_user_info", begge sider skal have vist det samme og have samme funktionalitet. Fra "Launch Phase – Uses Cases" blev der bestemt, at brugerne skal have muligheden for at ændre telefonnummer, Email og kodeord, og samtidigt se ens oplysninger, på nær kodeord.

View og HTML til ændring af kodeord er baseret på en guide [1], som anvender Django templates [2].

Implementering

Funktionen "office_user_info" i view.py i appen "office_user" er som på Figur 1, hvor brugerens information indhentes og sendes til HTML. Der er tilføjet en "if" sætning, der reagerer hvis brugeren laver en "POST request" [3], som så via en form "EditFieldUser", finder brugeren og de informationer, som skal anvendes. Derefter tjekkes om formen er gyldig, hvis den er gemmes den i databasen og brugeren sendes tilbage til brugerens hjem, ellers sendes brugeren til en fejl side.

```
42 @login_required
43 @user_controller
44 def office_user_info(request):
45     """Takes changes from user input and saves it"""
46     if request.method == 'POST':
47         form = EditFieldUser(request.POST, instance=request.user)
48         if form.is_valid():
49             form.save()
50             return redirect('/office_user_home')
51         else:
52             return redirect('/office_user_home/edit_user_error')
53     "Gets user information"
54     name = request.user.name
55     phone = request.user.phoneNumber
56     email = request.user.email
57     company = request.user.company
58     dict = {'name': name, 'email': email, 'phoneNumber': phone, 'company': company}
59     return render(request, "office_user_info.html", dict)
```

Figur 1 Funktion "office_user_info" i vews.py i "office_user" appen

Både "office_user" og "field_user" er bygget på samme måde.

HTML for at vise den personlige information er som på Figur 2 og Figur 3, hvor der ved "phoneNumber" er anvendt HTML's pattern som definere at tal mellem 0-9 kan bruges og at der skal mindst være 6 tal og der kan højst være 20, dette er med til at sikre bruger taster rigtigt. På Figur 2 er der to edit knapper der kan lave "POST request".

```

36 <table>
37 <tr>
38 <td><label for="name">Name:</label></td>
39 <td>{{ name }}</td>
40 </tr>
41 <tr>
42 <td><label for="Company">Company:</label></td>
43 <td>{{ company }}</td>
44 </tr>
45 <tr>
46 <td><label for="phoneNumber">Phone number:</label></td>
47 <td><input type="tel" id="phoneNumber" name="phoneNumber" pattern="[0-9]{6,20}" value="{{ phoneNumber }}"></td>
48 <td><button onClick="" style="...">Edit</button></td>
49 </tr>
50 <tr>
51 <td><label for="email">Email:</label></td>
52 <td><input type="email" id="email" name="email" value="{{ email }}"></td>
53 <td><button style="...">Edit</button></td>
54 </tr>
55 </table>
56

```

Figur 2 office_user_info.html, tabel for visning af personlige information

Kodeordet bliver vist for brugeren med 12 asterisk (*) uanset hvor mange tegn der skulle være i kodeordet. Når brugeren skal ændre sit kodeord, bliver brugeren omstillet til en anden page, se Figur 9. HTML er som på Figur 4. Siden for den personlige information kan se på Figur 7 og Figur 8.

```

60 <tr>
61 <td><label for="password">Password:</label> </td>
62 <td><span style="...">**** ***/td>
63 <td><button onClick="document.location.href = '../office_user_change_password/'" style="...">Edit </button></td>
64 </tr>
65

```

Figur 3 office_user_info.html, tabel med pseudokodeord og knap med link

Funktionen "office_user_change_password" bruger Django's API [2] for at ændre kodeorden med hashing (linje 264), når kodeordet er ændret, logges brugeren ud og skal logge ind igen. Implementeringen er som på Figur 4.

```

255 "office user password change"
256 @login_required
257 @user_controller
258 def office_user_change_password(request):
259     """ Uses Django API """
260     if request.method == 'POST':
261         form = PasswordChangeForm(request.user, request.POST)
262         if form.is_valid():
263             user = form.save()
264             update_session_auth_hash(request, user) # Important!
265             messages.success(request, 'Your password was successfully updated!')
266             logout(request)
267             return redirect('/accounts/login/')
268         else:
269             messages.error(request, 'Please correct the error below.')
270     else:
271         form = PasswordChangeForm(request.user)
272     return render(request, "office_user_change_password.html", {'form': form})

```

Figur 4 Funktionen "office_user_change_password" i vews.py i "office_user" appen

HTML'en for at ændre kodeord er som på Figur 5 og resultatet er på Figur 9. På samme måde er ændring af "field user" kodeord.

```
28 <fieldset>
29     <legend>Change password</legend>
30     <form method="post">
31         {% csrf_token %}
32         {{ form }}
33         <button type="submit">Save changes</button>
34     </form>
35     <br><br>
36     <button onClick="document.location.href = '../office_user_info/'" style="background-color: #f0f0f0;">back</button>
37
38 </fieldset>
```

Figur 5 office_user_change_password.html

I appen skal man huske at tilføje path som på Figur 6, for begge brugertyper.

```
24 path('edit_user_error/', views.edit_user_error, name="office_user_edit_error"),
25 path('office_user_change_password/', views.office_user_change_password, name="office_user_change_password"),
```

Figur 6 Ændringer i urls.py i appen "office_user"

Verifikation

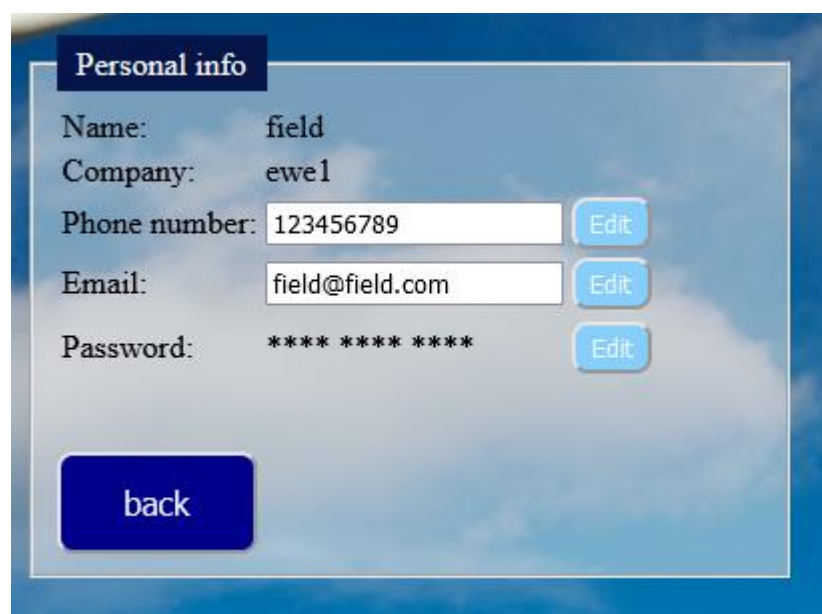
For at teste om implementering virker er testen udført som skrevet i Tabel 1.

Tabel 1: Tests til verifikation af opgave

Test	Test Steps	Pre-requisites	Pass-betingelser	Resultat
Edit information	<ol style="list-style-type: none">1. Login på brugeren "field"2. Klik på "Personal info"3. Indtast et nyt "Phone number" og klik på "Edit"4. Klik på "Personal info"5. Bekræft at ændringen er der.6. Indtast et nyt Email og klik på "Edit"7. Klik på "Personal info"8. Bekræft at ændringen er der.9. Klik på "Log out"10. Login på brugeren "office"11. Gentag punkt 2-9.	<ol style="list-style-type: none">1. Have adgang til to bruger field og office	<ol style="list-style-type: none">1. De ændringer man laver, er som ved bekræftelsen.	Bestået
Edit password	<ol style="list-style-type: none">1. Login på brugeren "field"2. Klik på "Personal info"3. Klik på "Edit" ud fra "password"4. Udfyld "Old password", "New password" og "New password confirmation" og klik på "Save changes"5. Login på brugeren "field" med det nye kodeord.6. Klik på "Log out"7. Login på brugeren "office"8. Gentag punkt 2-6.	<ol style="list-style-type: none">1. Have adgang til to bruger field og office	<ol style="list-style-type: none">1. kan logge ind med det nye kodeord	Bestået

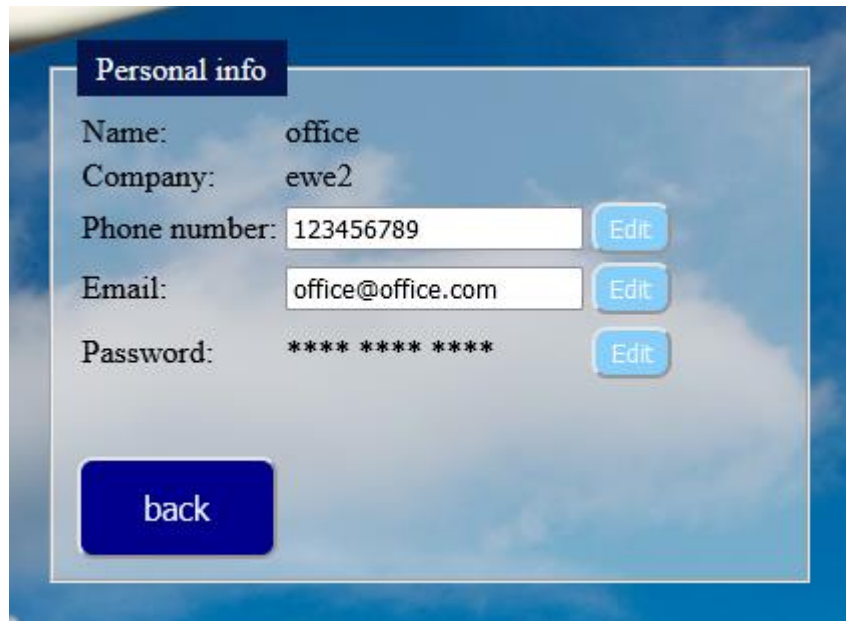
Testresultat

Personlige side for en "field user", se Figur 7.



Figur 7 personlige side for en "field user" ved navn field

Personlige side for en "office user", se Figur 8.

The screenshot shows a web interface titled "Personal info" in a dark blue header. Below the header, there are five rows of user information: "Name: office", "Company: ewe2", "Phone number: 123456789", "Email: office@office.com", and "Password: **** *". Each row has a light blue "Edit" button to its right. At the bottom left of the form area is a dark blue "back" button. The background of the page is a blue sky with white clouds.

Figur 8 personlige side for en "office user" ved navn office

Siden for at ændre kodeord, se Figur 9.

The screenshot shows a web interface titled "Change password" in a dark blue header. Below the header, there are two input fields: "Old password:" and "New password:". Below these fields is a bulleted list of password requirements: "Your password can't be too similar to your other personal information.", "Your password must contain at least 8 characters.", "Your password can't be a commonly used password.", and "Your password can't be entirely numeric." Below the list is a "New password confirmation:" input field and a "Save changes" button. At the bottom left is a dark blue "back" button. The background is a blue sky with white clouds.

Figur 9 Siden for at ændre kodeord

Konklusion

Den personlige side for både "office_user" og "field_user" er færdig udviklet og testet, hvor det er muligt at ændre i egne oplysninger og kodeord.

Referencer

[1]	Guide til kodeords ændring https://simpleisbetterthancomplex.com/tips/2016/08/04/django-tip-9-password-change-form.html (sidst besøgt 29-09-2021)
[2]	Django dokumentation for authentication https://docs.djangoproject.com/en/3.2/topics/auth/default/ (sidst besøgt 29-09-2021)
[3]	Respons og request https://docs.djangoproject.com/en/3.2/ref/request-response/ (sidst besøgt 29-09-2021)