# **Login pages design**

# Login page

Main page opened for login.

Component ***login***

|  |  |  |  |
| --- | --- | --- | --- |
| Case | Client request | Server response message | Description |
| Endpoint /users/authorize\_request – Session check and authorization into login form | | | |
| AuthenticationDTO | | | |
| Successful authentication | {  “username”: “value”,  “password”: “value”  } | {  “status”: 200  } | Server message response for successful authorization. User should be redirected to home page. |
| Failed Login | {  “username”: “value”,  “password”: “value”  } | {  “status”: 200,  “message”: "Bad credentials",  “error”: "name\_and\_email\_mismatch",  “error\_description”: "Bad credentials"  } | Server message response for unsuccessful authorization. User should be not redirected. |
| Expired credentials | {  “username”: “value”,  “password”: “value”  } | {  “status”: 200,  “error”: "account\_expired",  “error\_description”: "User credentials have expired"  } | User should be redirected to page ***new-password.*** Message “Credentials are expired. Create new password” should be displayed. |
| User is locked and he can’t login | {  “username”: “value”,  “password”: “value”  } | {  “status”: 200,  “error”: "account\_locked",  “error\_description”: "Account is locked"  } | Message in page login should be displayed “Account is locked”. User should be not redirected. |

# Reset password page

Page displayed when button **can’t login** from login page is displayed.

Component ***reset-password***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Case | | Client request | Server response message | Description |
| Redirected from login page | | | |
| Endpoint /users/reset\_request – Send reset password reset | | | |
| AuthenticationDTO (used again) | | | |
| Send successful email for reset | {  “username”: “value”,  “email”: “value”  } | {  “status”: 200  } | E-mail is send to user. User is redirected to login page. |
| Email is not found | {  “username”: “value”,  “email”: “value”  } | {  “status”: 200,  “error”: “user\_email\_not\_found",  “error\_description”: " User or email not found "  } | User or e-mail are not found into database. User should be not redirected. |
| Called when link from reset e-mail is opened | | | | |
| Endpoint /users/reset\_token | | | | |
| ResetPasswordTokenDTO | | | | |
| Open e-mail link and token is checked successfully | | {  “resetToken”: “value”  } | {  “status”: 200  } | Form to enter password and confirmation password is displayed. |
| Open e-mail link and token is not found | | {  “resetToken”: “value”  } | {  “status”: 200,  “error”: “not\_found",  “error\_description”: " Not found"  } | Message “Not found” should be displayed. User should be redirected to login page. |
| Called when server sends status 200 for token confirmation | | | | |
| Endpoint /users/reset\_password | | | | |
| ResetPasswordDTO | | | | |
| Username and password are valid | | {  “password”: “value”,  “confirmationPassword”: “value”  } | {  “status”: 200  } | Server message response for successful authorization. Message 'Password successfully changed' should be displayed at the top and user should be redirected to main login page |
| When user submits new password which is used before | | {  “username”: “value”,  “password”: “value”  } | {  “status”: 200,  “error”: “confirmation\_password\_mismatch",  “error\_description”: " Confirmation password mismatch"  } | Message “Confirmation password mismatch” should be displayed. |

# Activate password

Page opened from e-mail.

Component ***activate-password***

|  |  |  |  |
| --- | --- | --- | --- |
| Case | Client request | Server response message | Description |
| Called when link from reset e-mail is opened | | | |
| Endpoint /users/confirm\_token | | | |
| ActivatePasswordTokenDTO | | | |
| Successful token authorization | {  “confirmationToken”: “value  } | {  “status”: 200  } |  |
| Activation link’s token is invalid | {  “confirmationToken”: “value  } | {  “status”: 200,  “error”: “not\_found",  “error\_description”: " Not found"  } | User should be redirected to main login page (Component **login**). Message “Invalid token” should be displayed. |
| Called when server sends status 200 for token confirmation | | | |
| Endpoint /users/reset\_token | | | |
| ActivatePasswordDTO | | | |
| Successful form submit and password confirmation | {  “password”: “value”,  “confirmPassword”: “value”  } | {  “status”: 200  } | When user enters valid new password and he submits successfully the form he is redirected to main login page(Component **login**) |
| When user submits new password but confirmation password don’t match | {  “password”: “value”,  “confirmPassword”: “value”  } | {  “status”: 200,  “error”: “confirmation\_password\_mismatch",  “error\_description”: " Confirmation password mismatch"  } | Message “Confirmation password mismatch” should be displayed. User should not be redirected. |

# New Password

This page is displayed when user is forced to reset password right after user login.

Component ***new-password***

|  |  |  |  |
| --- | --- | --- | --- |
| Case | Client request | Server response message | Description |
| Called when user is successfully authenticated | | | |
| Endpoint /users/new\_password | | | |
| NewPasswordDTO | | | |
| User enters his new personal password | {  “password”: “value”,  “confirmPassword”: “value”  } | {  “status”: 200  } | User should be redirected to home page |
| User enters his new personal password but it’s used last 3 times. | {  “password”: “value”,  “confirmPassword”: “value”  } | {  “status”: 200,  “error”: “password\_already\_user",  “error\_description”: " Password already used"  } | When PASSWORD\_ALREADY\_USED response is returned from BE display it as a message "Password is used last 3 times" - user should not be redirected |
| When user submits new password which is used before | {  “password”: “value”,  “confirmPassword”: “value”  } | {  “status”: 200,  “error”: “confirmation\_password\_mismatch",  “error\_description”: " Confirmation password mismatch"  } | Message “Confirmation password mismatch” should be displayed. |

# Workflow

Case create user

1. Create account
2. E-mail is send
3. Activation link is opened
   1. When activation status is Ok then page Activate account (Component ***activate-password***) is loaded.
   2. If activation token is invalid or expired message server should return NOT\_FOUND and user should be redirected to main login page. Message “Invalid token” should be displayed.
      1. After entering valid new password user is redirected to login page (Component **login**)

Case reset user password

1. Open login page (Component **login**) and click button reset password
2. User is redirected to page (Component **reset-password**)

If user or e-mail is not found server response NAME\_AND\_EMAIL\_MISMATCH is returned and message “Username or e-mail are not found”

Case new password (force password reset, used also when password is old more than 45 days)

1. Administrator selects button “force password change”
2. When user authenticates successfully he is redirected to page (Component **new-password**)
3. After entering valid new password user is redirected to login page (Component **login**)

If password is already used server returns PASSWORD\_ALREADY\_USED. "Password is used last 3 times" - user should not be redirected to login page (Component **login**).

1. After entering valid new password user is redirected to login page (Component **login**)

Case expired user

1. User opens login page and enters his credentials
2. If password is older than 45 days server returns ACCOUNT\_EXPIRED
3. User should be redirected to page (Component **new-password**). Message “Credentials are expired. Create new password” should be displayed.

Case locked user

1. User opens login page and enters his credentials
2. After authentication attempt server returns USER\_LOCKED. Message in page login should be displayed “Account is locked”

Important security requirements:

1. Passwords should not be older than 45 days
2. Old passwords should be remembered and last 3 passwords should not be accepted when user try to enter new password
3. JWTs can be stored in **cookies**. JWTs are stored as cookies on many occasions, and cookies are vulnerable/susceptible to CSRF (Cross-site Request Forgery) attacks. One of the many ways to prevent CSRF attacks is to ensure that your cookie is accessible by only your domain. As a developer, ensure that necessary CSRF protections are put in place to avoid these attacks, regardless of the use of JWTs.
4. JWTs and **session ids** can also be exposed to unmitigated replay attacks. It is totally up to the developers to establish what replay-mitigation techniques are appropriate for their systems. One way of solving this problem is ensuring that JWTs rely on short expiration times. Although, this technique doesn’t totally solve the problem. However, other alternatives for solving this challenge are issuing JWTs to specific IP addresses and using browser fingerprinting.
5. Use HTTPS/SSL to ensure that your cookies and JWTs are encrypted by default during client and server transmission. This helps avoid man-in-the-middle attacks!