This document outlines the various functions of UI elements found on each Admin page

HairDressing project

ADMIN FUNCTIONS

Table of Contents

[1. Sign up 2](#_Toc37097312)

[2. Sign in 4](#_Toc37097313)

[3. Forgot Password 6](#_Toc37097314)

[4. New Password 8](#_Toc37097315)

[5. Dashboard 10](#_Toc37097316)

[6. Databases 12](#_Toc37097317)

[6.1. Add Entry Modal 14](#_Toc37097318)

[6.2. Edit Entry Modal 16](#_Toc37097319)

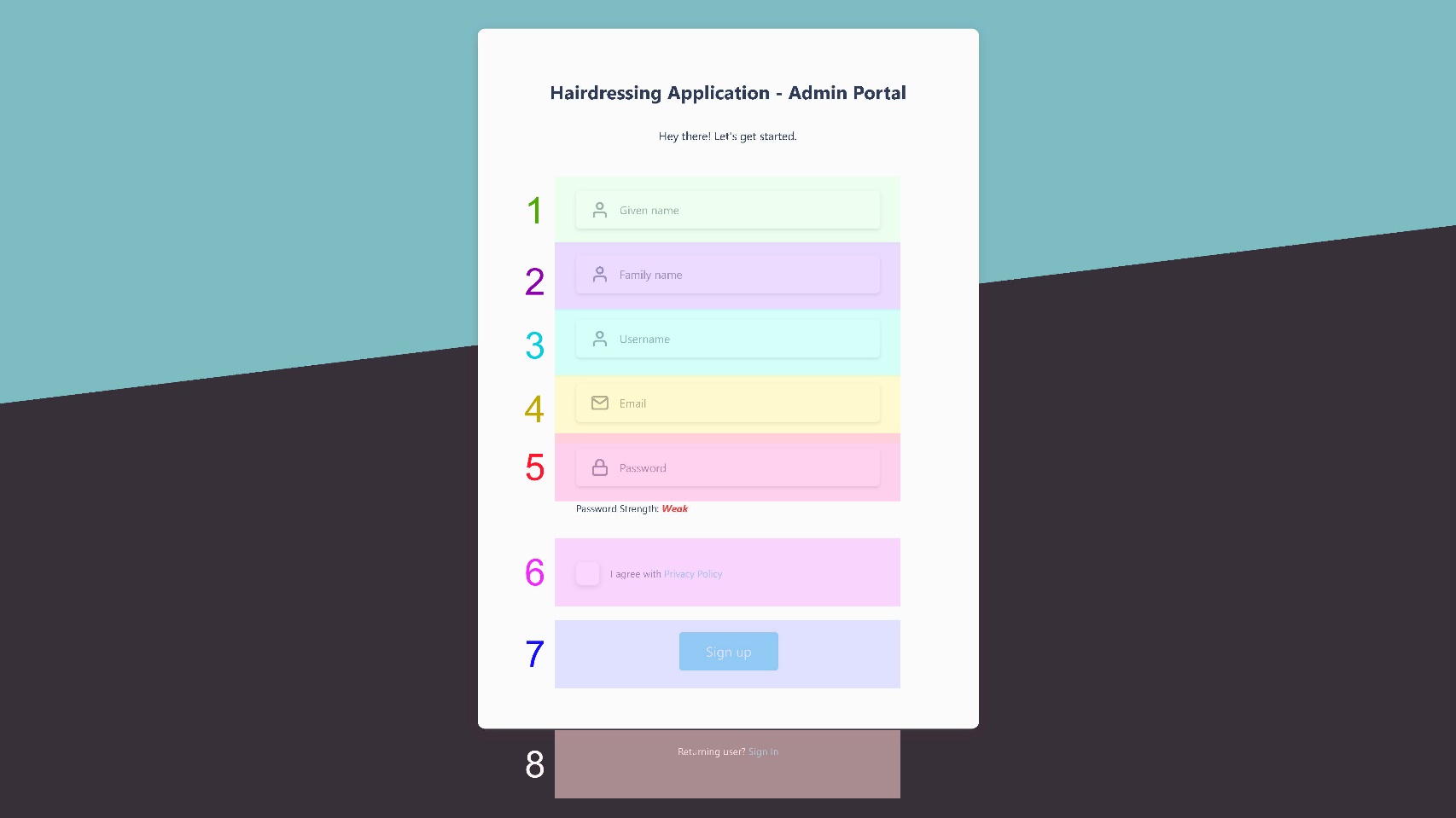
[6.3. Delete Entry Modal 18](#_Toc37097320)

[7. Traffic 20](#_Toc37097321)

[8. Permissions 22](#_Toc37097322)

[9. Pictures 24](#_Toc37097323)

# Sign up



|  |  |  |
| --- | --- | --- |
| Function | Details | Sections/Fields |
| Sign up form | Every new user must enter the details outlined above to register in the admin portal | (1) Given name input (**required**)  (2) Family name input  (3) Username input (**required**)  (4) Email input (**required**)  (5) Password input (**required**)  (6) Privacy policy checkbox (**required**)  (7) Sign up button (submit form)  (8) Sign in link |
| Basic password validation will make accounts slightly more secure |
| A privacy policy might be written for the application itself rather than the admin portal |
| The “Sign up” button should be greyed out if a required field is invalid |
| The “Sign In” link conveniently redirects existing users to the sign in page |

# 2. Sign in

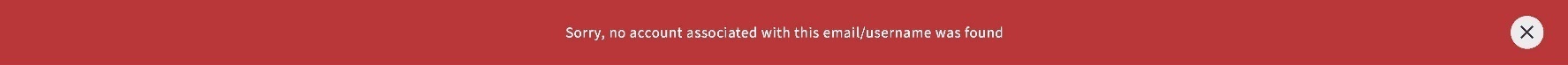


|  |  |  |
| --- | --- | --- |
| Function | Details | Sections/Fields |
| Sign in form | Users have the option to enter either their username or email to sign in | (1) Username or email input (**required**)  (2) Password input (**required**)  (3) Remember me checkbox  (4) Forgot password link  (5) Sign in button (submit form)  (6) Sign up link |
| A pop-up message should be displayed if either field is incorrect |
| Checking the “remember me” checkbox stores a token in the user’s browser that identifies them next time they access the admin portal |
| The “forgot password” link should redirect users to the forgot password page |
| The “forgot password” link should redirect new users to the sign up page |

# 3. Forgot Password







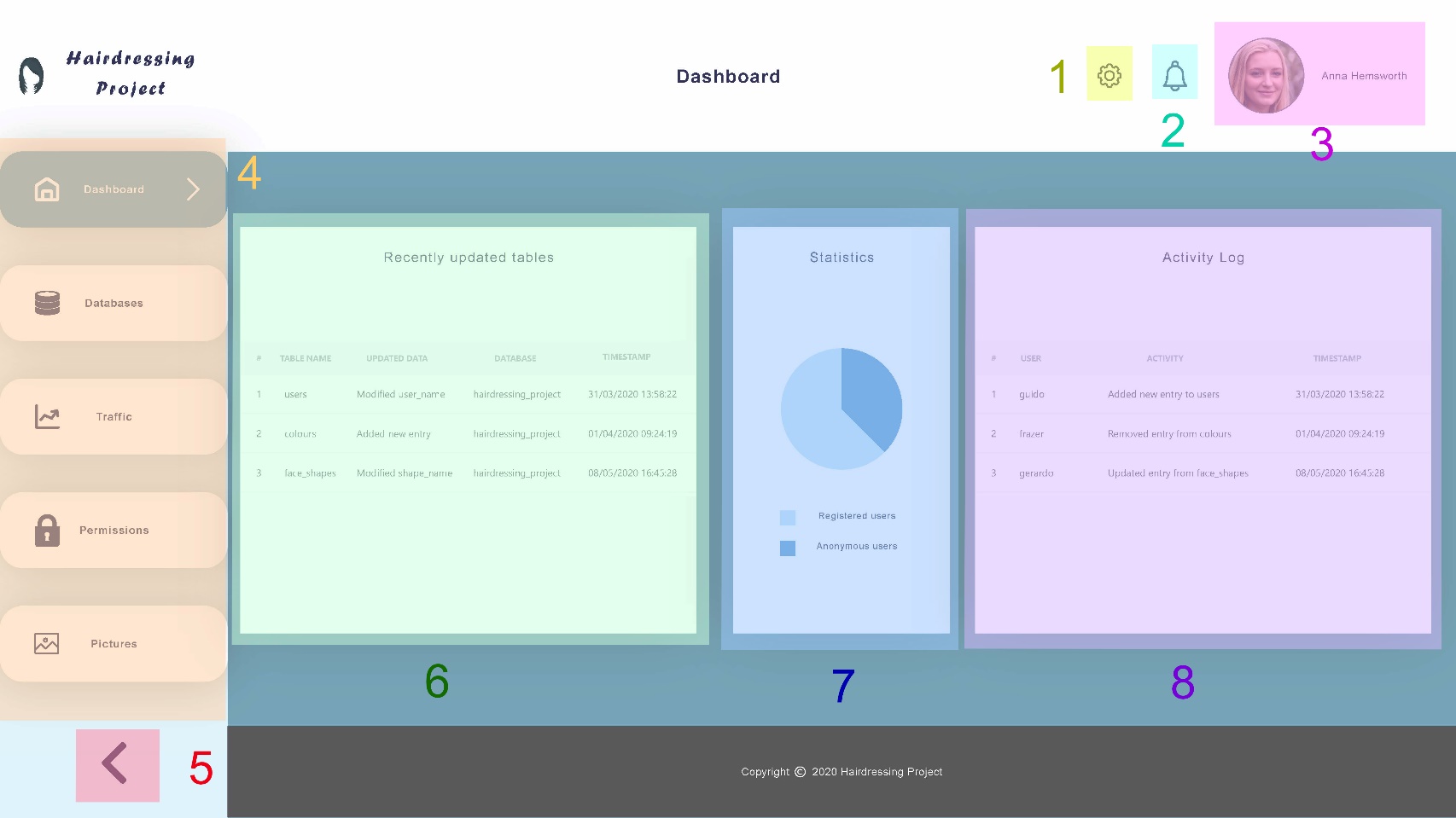
|  |  |  |
| --- | --- | --- |
| Function | Details | Sections/Fields |
| Forgot password form | Users can enter either username or email to recover their account | (1) Username or email input (**required**)  (2) Recover password button (submit form)  (3) Sign in link  (4) Sign up link |
| If an account associated with the information entered is found, an email will be sent to the user with instructions on how to create a new password. Additionally, a success message should pop up at the top of the window to let the user know. |
| If no account is found, an error message should pop up at the top of the window to inform the user. |
| If the username/email input field is invalid or empty, the “recover password” button should be inactive. |
| The sign in/ sign up links below the form should redirect users to the corresponding pages. |

# 4. New Password



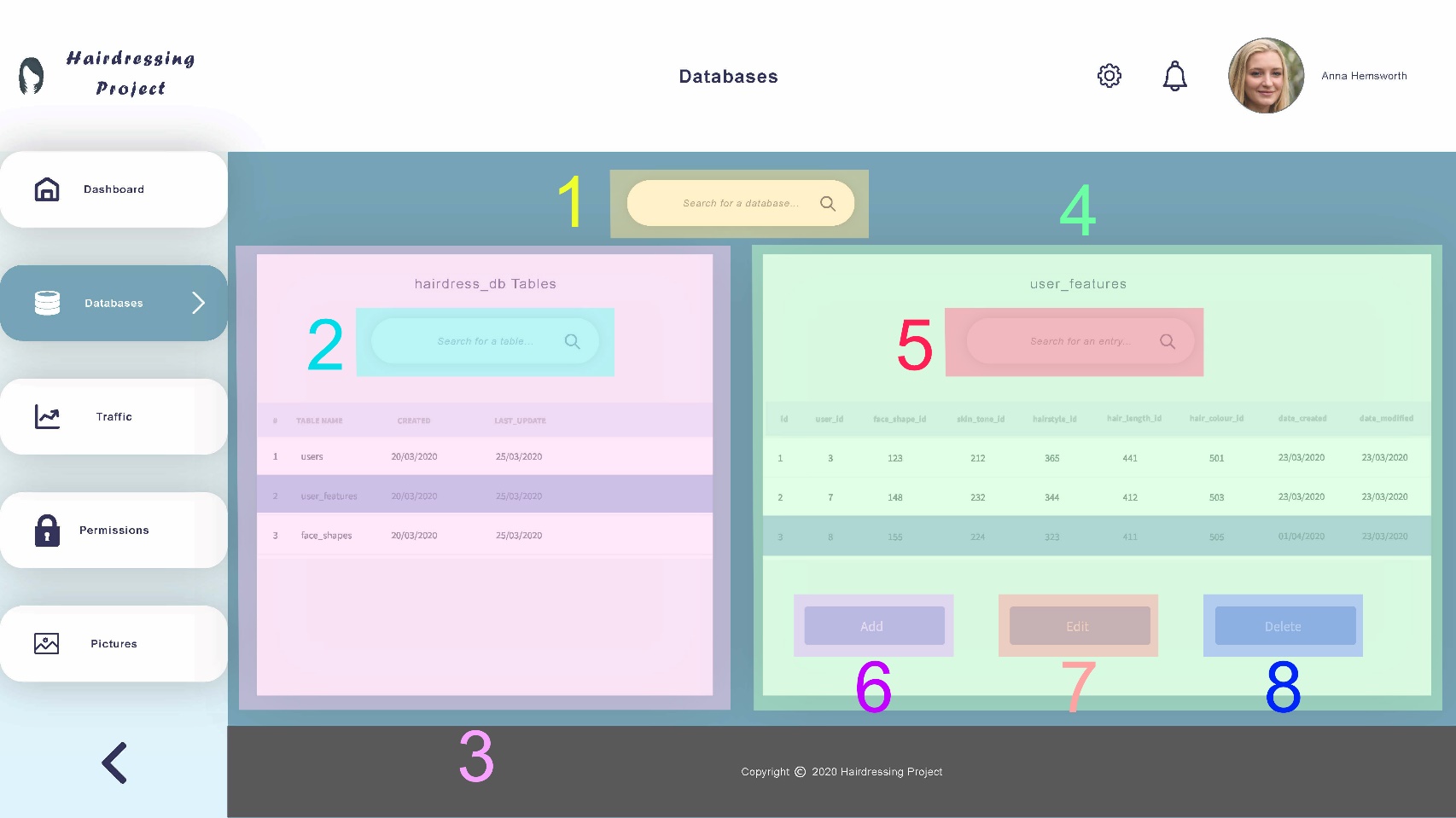
|  |  |  |
| --- | --- | --- |
| Function | Details | Sections/Fields |
| New password form | The “email” field is read-only and lets users be aware of which email is associated with the account that they wish to recover. | (1) Email (read-only)  (2) New password input (**required**)  (3) Confirm new password input (**required**)  (4) Change password button (submit form)  (5) Sign in link  (6) Sign up link |
| Both the “new password” and “confirm new password” input fields have basic validation and must match. |
| If any of the fields is invalid, empty or the passwords do not match, the “change password” button should be inactive. |
| If all inputs are valid, clicking on the “change password” button should redirect users to the dashboard page, with a success message at the top of the window informing them that their password has changed (possibly along with an email). |
| The sign in/ sign up links below the form should redirect users to the corresponding pages. |

# 5. Dashboard



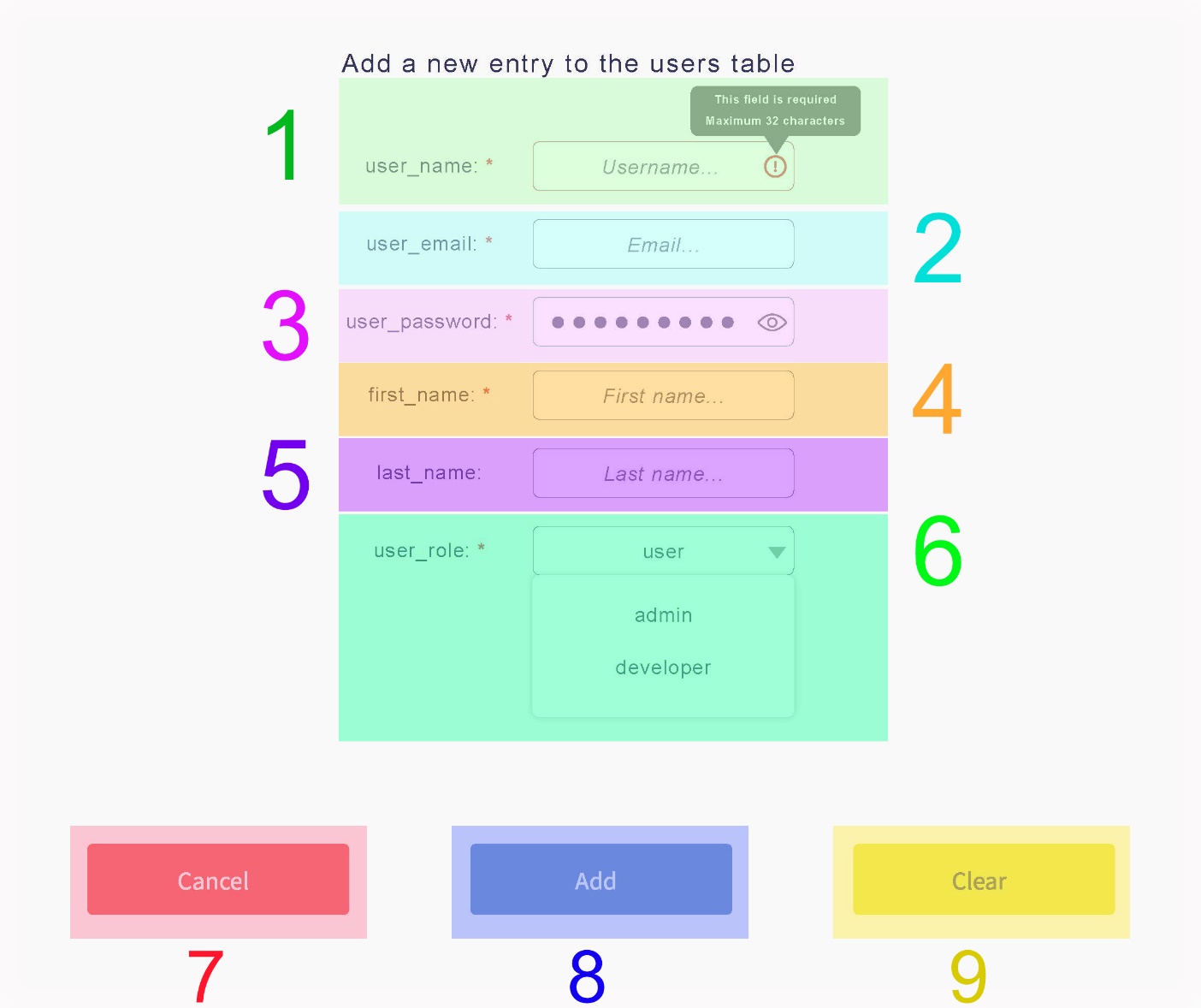
|  |  |  |
| --- | --- | --- |
| Function | Details | Sections/Fields |
| Dashboard (home) page | The dashboard is the home page of the admin portal. It is the first page that users see after signing in. | (1) Settings button  (2) Notifications button  (3) Profile picture and name  (4) Sidebar with links  (5) Collapse sidebar button  (6) Recently updated tables  (7) Statistics pie chart  (8) Activity log table |
| The settings button exposes a menu containing links for account information, logout and more. |
| Clicking on the notifications button should show a small panel with the latest (important) changes in the databases or in the current user’s account, such as password or permissions. It should also display a switch button with the option to disable such notifications. |
| The sidebar is the main navigation element of the admin portal. Clicking on each button should redirect users to the respective page. |
| Recently updated tables should show which column/row has been modified, with the name of the respective table and database and a timestamp. |
| The statistics section should display a simple pie chart or graph with useful data about users. |
| Activity log refers to the latest changes specifically made by registered users of the application, with a timestamp. |

# 6. Databases



|  |  |  |
| --- | --- | --- |
| Function | Details | Sections/Fields |
| Databases page | The databases page should show all tables contained in each database (highlighted by (3)), with timestamps for created at and last updated dates. | (1) Search for a database input  (2) Search for a table input  (3) Tables section  (4) Entries section  (5) Search for an entry input  (6) Add new entry button  (7) Edit selected entry button  (8) Delete selected entry button |
| When users select a row in the tables section, a new section should pop up at the right (4), showing all entries of the table selected. If no row in the latter is selected, the “edit” and “delete” buttons should remain inactive. |
| The search input fields (1, 2, 5) let users look for a specific database, table or entry, respectively. As they type, the respective table should automatically display only relevant results. Clearing a field should restore the respective table to show all records. |
| The “add” button (6) allows users to create new entries in the selected table and the “edit” (7) and “delete” (8) buttons enable modification of the selected entry. See 6.1., 6.2. and 6.3. for more details. |

## 6.1. Add Entry Modal



|  |  |  |
| --- | --- | --- |
| Function | Details | Sections/Fields |
| Add entry modal | The add entry modal should pop up when users click on the “add” button from the databases page. | (1) user\_name input field (**required**)  (2) user\_email input field (**required**)  (3) user\_password input field (**required**)  (4) first\_name input field (**required**)  (5) last\_name input field  (6) user\_role dropdown (**required**)  (7) Cancel add entry button  (8) Add entry button  (9) Clear all fields button |
| When users click on the “cancel” button (7) or when they click outside of this modal, it should close. |
| When users click on the “clear” button (9), all input fields should be cleared. |
| Fields marked with (**\***) are required. A validation tooltip should appear when an input field is invalid. |
| The “add” button (8) should be disabled as long as at least one field is invalid or empty. If all fields are valid, it should turn active. Clicking on it should add a new entry to the selected table, close the modal and display a feedback message at the top of the window. |
| **TODO**: Add error handling. |

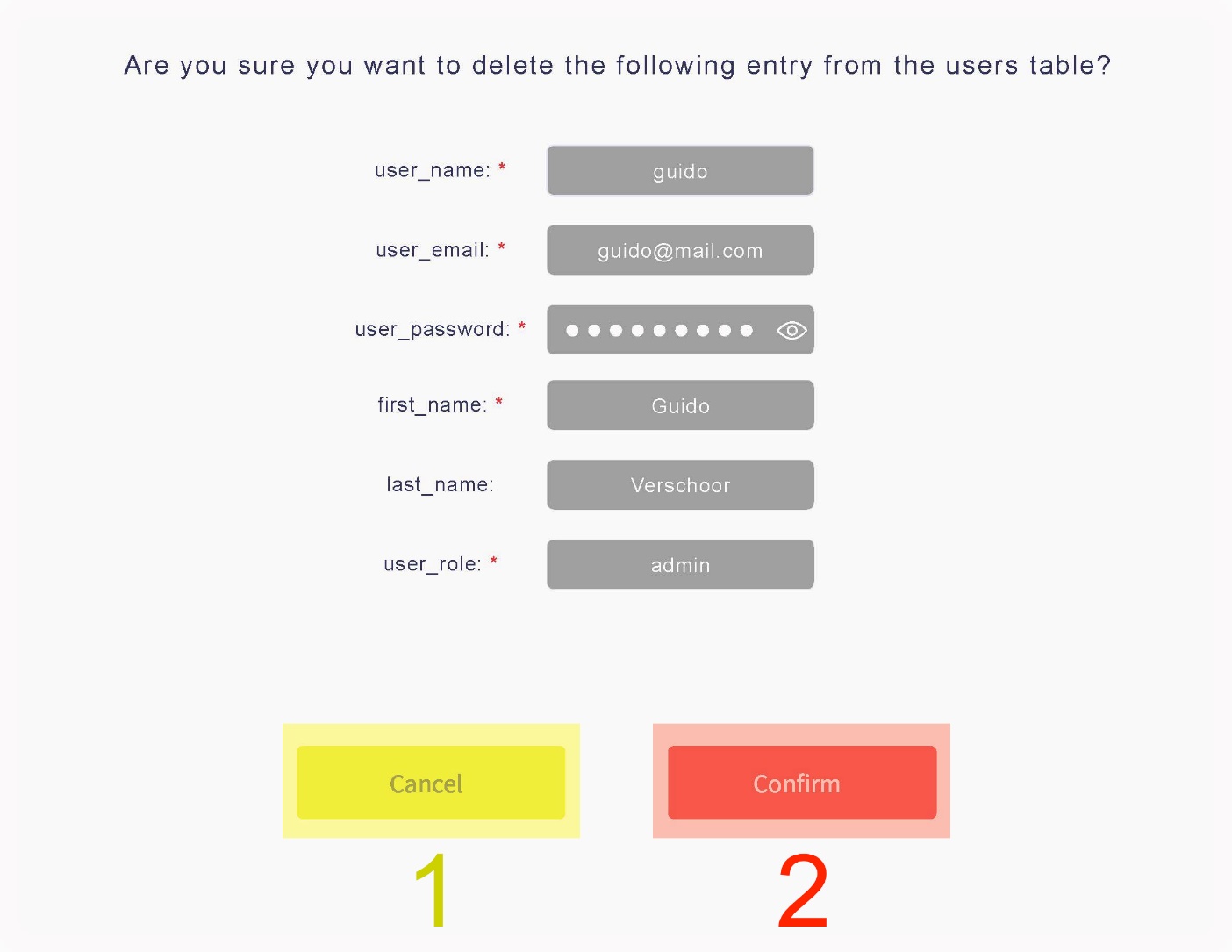
**NOTE:** The users table was only shown as an example here. The same concepts apply to all other tables.

## 6.2. Edit Entry Modal



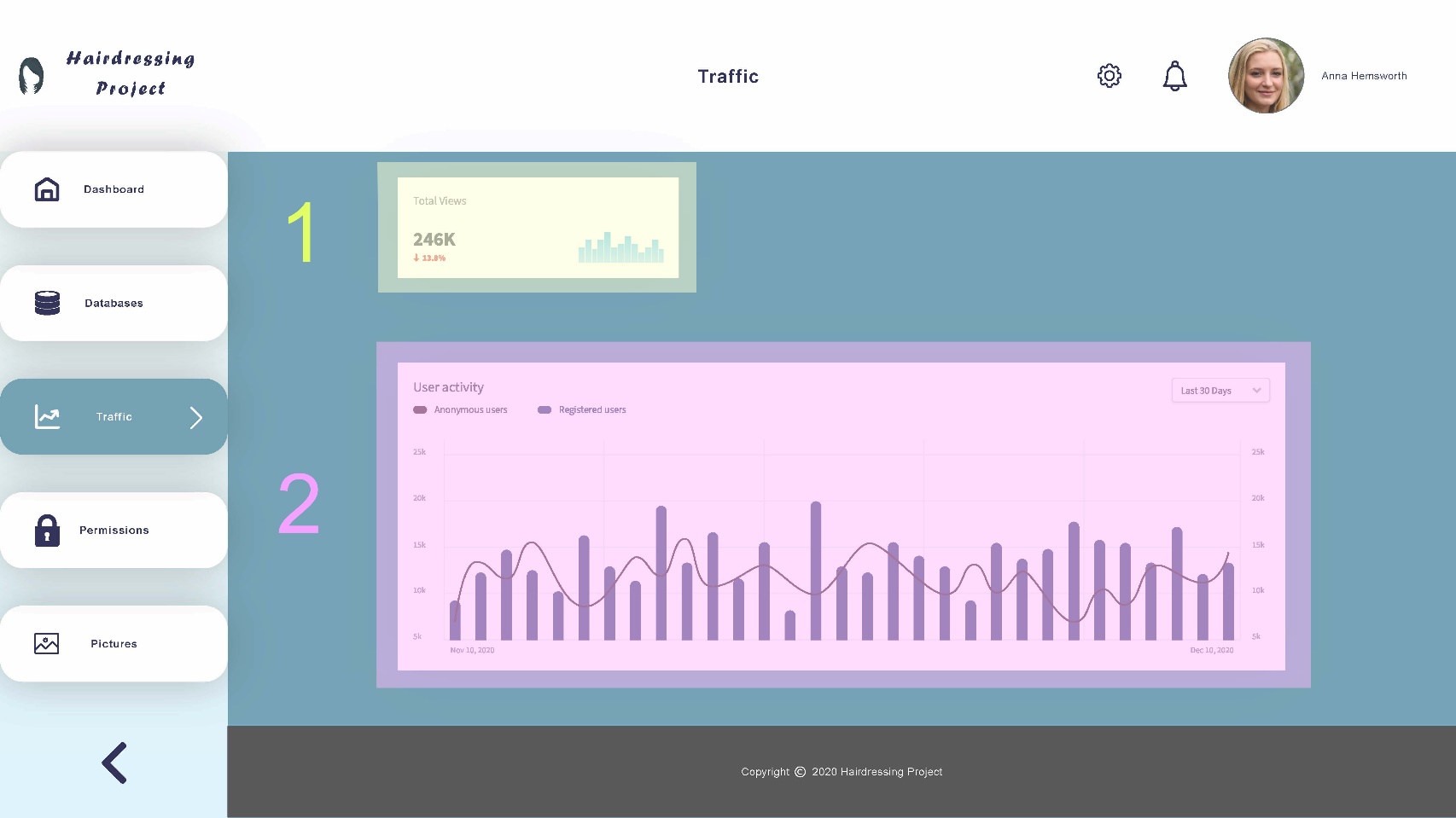
|  |  |  |
| --- | --- | --- |
| Function | Details | Sections/Fields |
| Edit entry modal | The edit entry modal should pop up when users click on the “edit” button from the databases page. | (1) Cancel edit entry button  (2) Edit entry button  (3) Restore all fields button |
| When users click on the “restore” button (3), all input fields should be overwritten with their respective initial values. |
| The “edit” button (2) should be disabled as long as at least one field is invalid or empty. If all fields are valid, it should turn active. Clicking on it should edit the selected entry of the current table, close the modal and display a feedback message at the top of the window. |
| Every other UI element not mentioned here works in the way as 6.1. |

## 6.3. Delete Entry Modal



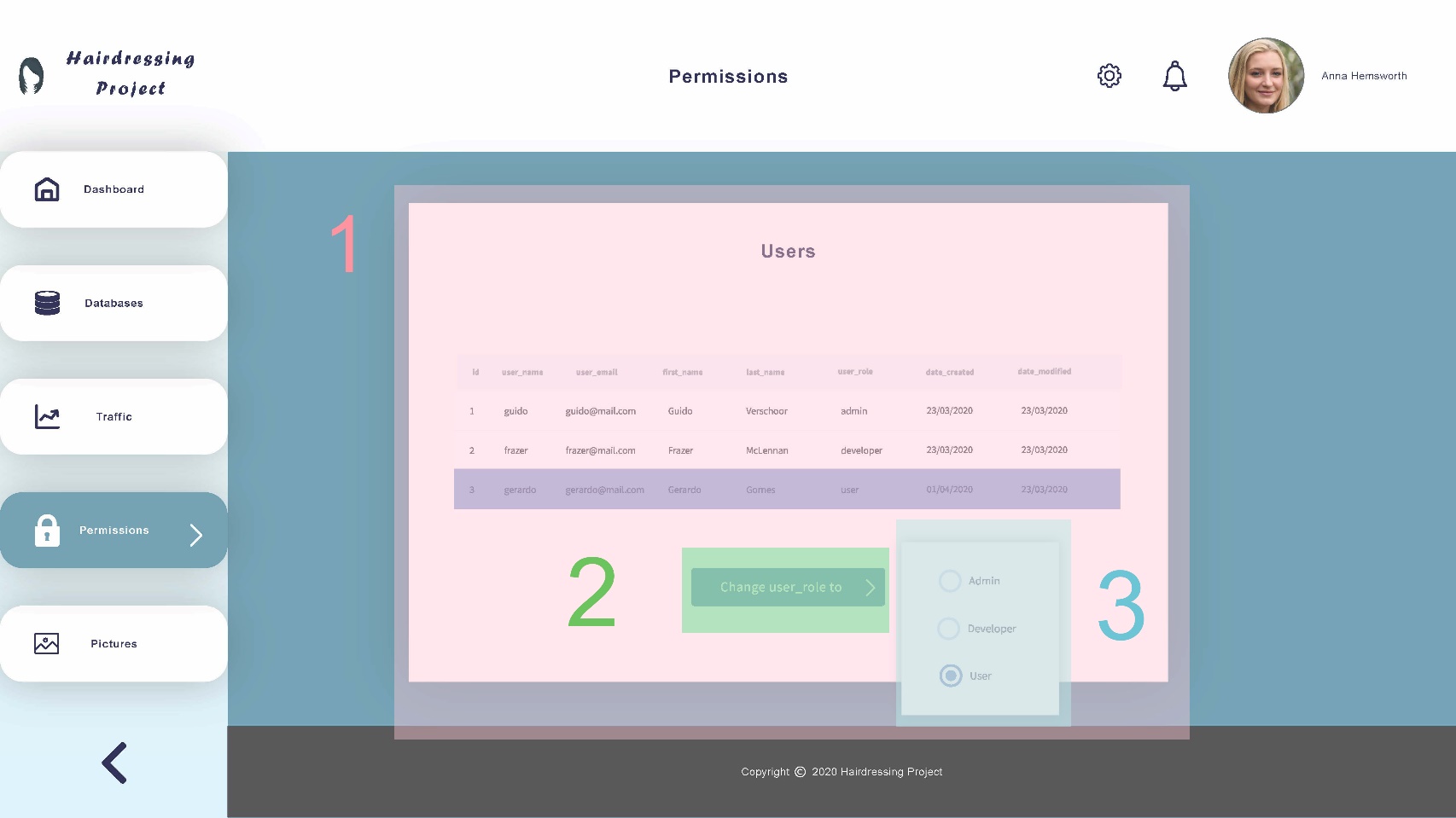
|  |  |  |
| --- | --- | --- |
| Function | Details | Sections/Fields |
| Delete entry modal | The delete entry modal should pop up when users click on the “delete” button from the databases page. | (1) Cancel delete entry button  (2) Confirm delete entry button |
| The “cancel” button (1) should close the modal and the “confirm” (2) button should delete the selected entry from the table, close the modal and then display a feedback message at the top of the window. |
| In this modal all fields should be read-only, since there is no need to modify any of them. |

# 7. Traffic



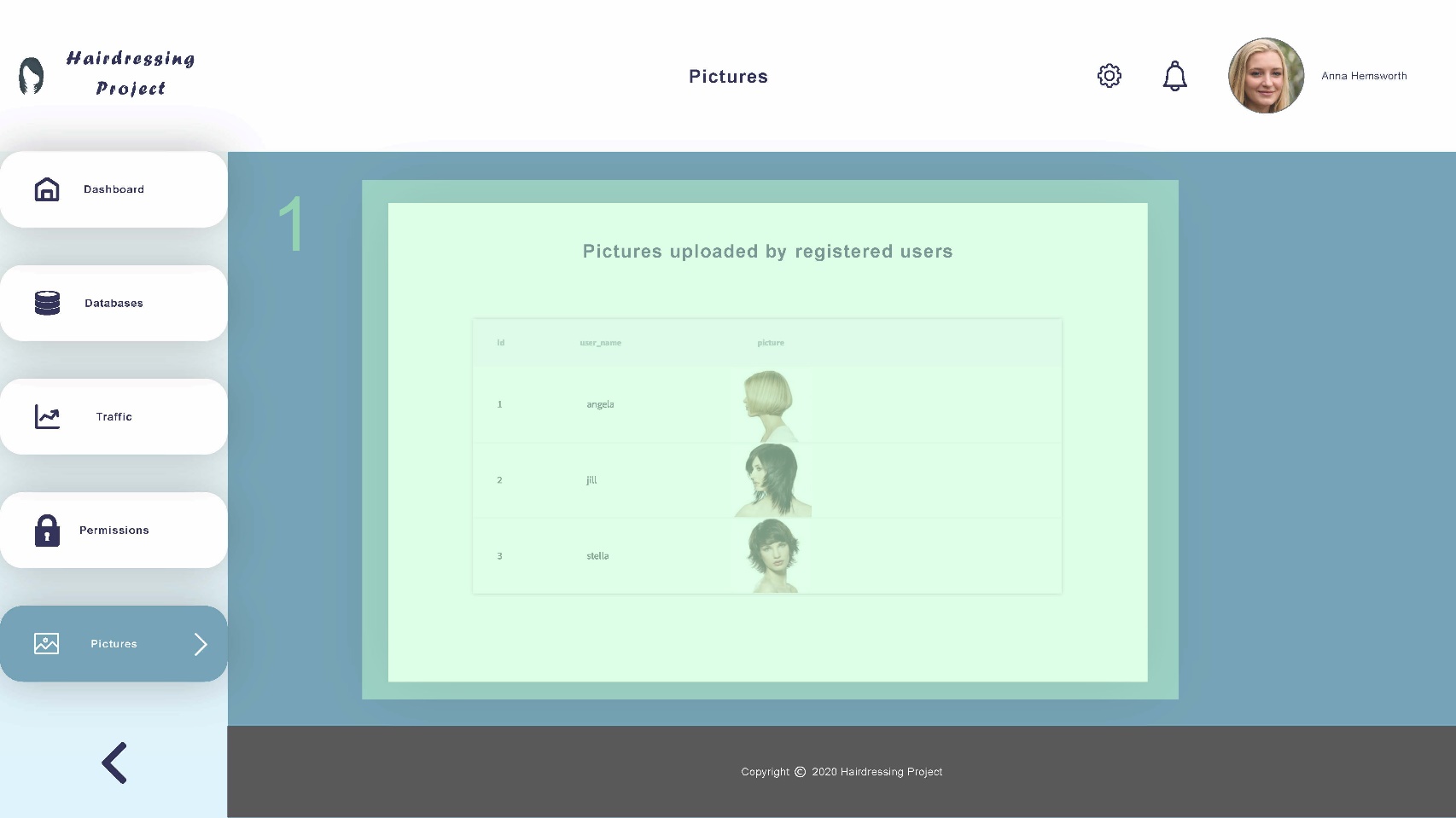
|  |  |  |
| --- | --- | --- |
| Function | Details | Sections/Fields |
| Traffic page | The traffic page should show relevant data regarding usage of the Hairdressing application. | (1) Total views data  (2) User activity data |
| The first section (1) should display the number of times that the application has been browsed so far. The total views could optionally count only unique users. |
| The second section (2) should present a graph that summarises user activity in the Hairdressing application by date, which could distinguish registered users from anonymous ones. |
| Ideally, a well-supported JavaScript library would be used to handle data visualisation for this page, so that users would be able to hover or click on specific points in the graph to see more details about the corresponding dates. |
| All other UI elements not outlined in the previous picture have already been covered. |

# 8. Permissions



|  |  |  |
| --- | --- | --- |
| Function | Details | Sections/Fields |
| Permissions page | The permissions page allows admins to change the role of any registered user. As such, it can only be viewed by users with “admin” as their user\_role | (1) Users table (main section)  (2) Change user\_role to button  (3) User role drop-right |
| The main section of the page (1) shows all entries of the “users” table. If there is no entry currently selected, the “change user\_role to” button (2) should be disabled. |
| Once a user is selected, clicking on the “change user\_role to” button (2) should open a small drop-right with all the available user roles as radio buttons (3). |
| After a user role is chosen, the “users” table should be automatically updated. A feedback message should be displayed at the top of the page. |
| All other UI elements not outlined in the previous picture have already been covered. |

# 9. Pictures



|  |  |  |
| --- | --- | --- |
| Function | Details | Sections/Fields |
| Pictures page | The pictures page should display user-submitted pictures from the Hairdressing application. Those pictures are not present in the databases, they are stored in a separate cloud instance (possibly AWS S3), which is why this page is needed. | (1) Pictures uploaded (main section) |
| The main section of this page (1) should show all pictures available in the cloud storage in a table, sorted by user/user id. |
| Double-clicking on a row should bring users to the URL where pictures submitted by that particular user is stored in the cloud. |
| **Note:** See the project specification document for more info about how pictures are handled in this project. |
| All other UI elements not outlined in the previous picture have already been covered. |