[Final project] - Checklist

Login page:

1. Verify access to the login page by navigating to the host React App. - pass
2. Test login as an admin (email: admin@gmail.com, password: admin). - pass
3. Test login as a manager (using a previously created user profile). - pass
4. Verify “Email” field with the following requirements:
    a. '@' is recuired; - pass
    b. Period is recuired; - pass
    c. Letter (a-z) is recuired; - pass
    d. Numbers (0-9) are allowed; - pass
    e. Underscores are allowed (must be followed by one or more letter or number); - pass
    f. Dashes are allowed (must be followed by one or more letter or number); - pass
    g. The last portion of the domain must be at least two characters, for example: .com, .org, .cc. - pass
5. Check the UI elements on the login page. - fail M2Q-360: [Login page] - Unnecessary "Cancel" button To Do M2Q-361: [Login page] - Registration window displays at an incorrect size To Do

Applications page:

1. Verify the applications list includes fields:
    a. id; - pass
    b. name; - pass
    c. surname; - pass
    d. email; - pass
    e. phone; - pass
    f. age; - pass
    g. course; - pass
    h. course_format; - pass
    i. course_type; - pass
    j. status; - pass
    k. sum; - pass

    l. alreadyPaid; - pass
    m. group; - pass
    n. created_at; - pass
    o. manager. - pass
2. Check pagination with 25 items per page. - pass
3. Ensure applications are ordered from latest to earliest by default. - pass
4. Test pagination behavior on:
    a. the first page; - fail M2Q-363: [Applications page] - Pagination does not work correctly at the first page To Do
    b. a page in the first half of all pages but not the first one; - pass
    c. a page in the second half of all pages but not the last one. - pass
    d. the last page. - fail M2Q-364: [Applications page] - Pagination does not work correctly at the last page To Do
5. Verify if the current page number appears in QueryParams. - pass
6. Verify that clicking on each column name enables ordering of data in - pass
    a. ascending (ASC) order; - pass
    b. descending (DESC) order. - fail M2Q-418: [Applications page] - Second click on the column name disables DESC sorting To Do
7. Ensure sorting updates QueryParams accordingly. - pass
8. Check header elements:
    a. logo; - pass
    b. user name; - pass
    c. logout button; - pass
    d. admin panel button (for admin only). - fail M2Q-390: [Applications page] - Clicking the "Admin" button leads to a blank page To Do
9. Check the UI of the applications page. - fail M2Q-406: [Applications page] - The "Surname" field is in Ukrainian To Do

Filtering

1. Confirm that the following text field filters search for matches across the entire field:
    a. Name; - pass

b. Surname; - pass

c. Email; - pass

d. Phone; - pass

e. Age. - fail M2Q-411: [Filtering] - The "Age" filter fails with a single digit To Do

2. Check that filtering triggers when typing stops:

a. Name; - pass

b. Surname; - pass

c. Email; - pass

d. Phone; - pass

e. Age. - fail M2Q-411: [Filtering] - The "Age" filter fails with a single digit To Do

3. Check the following dropdowns:

a. all courses; - fail M2Q-413: [Filtering] - The "FE" course is listed twice To Do

b. all formats; - pass

c. all types; - pass

d. all statuses; - fail M2Q-414: [Filtering] - Selecting a status, the page loads continuously To Do M2Q-415: [Filtering] - Status names are in Ukrainian To Do

e. all groups. - pass

4. Check the following data filters:

a. Start date; - pass

b. End date. - pass

5. Verify the checkbox “My” to filter only the current user’s applications. - fail M2Q-412: [Filtering] - The "My" checkbox does not display applications To Do

6. Verify all filter settings are stored in QueryParams. - pass

7. Check the page returns to its original state after clearing the filter. - fail M2Q-417: [Filtering] - The page does not reset correctly after clearing filters To Do

8. Verify the reset button clears all filters. - pass

9. Verify the functionality of a button generates an Excel file based on the applied filters. - pass

Comment section

1. Confirm that clicking an application expands it to show details. - pass
2. Ensure "Message" and "UTM" fields display data from the database. - pass
3. Check the input field for adding comments. - pass
4. Verify the logged-in user's last name appears in the "manager" column after submitting a comment. - pass
5. Ensure "In Work" status appears in the "status" column if the previous status was "null" or "New". - pass
6. Confirm the comment displays with the author's name and submission date. - pass
7. Check the UI of the comment section. - pass

Editing an application

1. Verify only applications without a manager or assigned to the logged-in user are editable. - pass
2. Check if the EDIT button opens a modal window with a form. - pass
3. Verify all form fields can be left blank if needed. - pass
4. Validate the following dropdowns:
    a. Status:
        i. In Work - pass
        ii. New - pass
        iii. Agree - pass
        iv. Disagree - pass
        v. Dubbing - pass
    b. Course:
        i. FS - pass
        ii. QACX - pass
        iii. JCX - pass
        iv. JSCX - pass
        v. FE - pass
        vi. PCX- pass
    c. Course type:
        i. pro - pass

ii. minimal - pass
iii. premium - pass
iv. incubator - pass
v. vip - pass

d. Course format:
i. static - pass
ii. online - pass

5. Check functionality for adding a unique group to the Group field. - fail M2Q-439: [Editing an application] - Unable to create a new group To Do
6. Verify input fields for:
a. Name
i. Latinic letters pass
1. Uppercase letters - pass
2. Lowercase letters - pass
ii. Cyrillic letters pass
1. Uppercase letters - pass
2. Lowercase letters - pass
b. Surname
i. Name
1. Latinic letters pass
a. Uppercase letters - pass
b. Lowercase letters - pass
2. Cyrillic letters pass
a. Uppercase letters - pass
b. Lowercase letters - pass
c. Email
i. '@' is recuired - pass
ii. Period is recuired - pass
iii. Letter (a-z) is recuired - pass
iv. Numbers (0-9) are allowed - pass
v. Underscores are allowed (must be followed by one or more letter or number) - pass

vi. Dashes are allowed (must be followed by one or more letter or number) - pass

vii. The last portion of the domain must be at least two characters, for example: .com, .org, .cc. - pass

d. Phone

i. 38xxxxxxxxxx (start with“38”+10 numbers) - pass

e. Age

i. from 16 to 90 - pass

f. Sum

i. from 1 to 2147483647 - pass

g. Already Paid

i. from 1 to 2147483647 - pass

7. Check the UI of the Edit form. - fail M2Q-440: [Editing an application] - Status field options are in Ukrainian To Do

Admin panel - blocked by M2Q-390: [Applications page] - Clicking the "Admin" button leads to a blank page To Do

1. Verify the button to create new managers.
2. Ensure application statistics by status are available.
3. Confirm the "Create" button opens a form in a modal window.
4. Verify that a new manager appears in the list after form submission.
5. Check "Activate" button generates an activation link with 30-minute validity.
6. Ensure the activation link copies to the clipboard.
7. Test that the activation link directs the manager to a password creation page.
8. Verify “Password” field with the following requirements:
    a. at least one uppercase letter
    b. at least one lowercase letter
    c. at least one non-alpha numeric symbol
    d. at least one number (0-9)
    e. 8-50 characters without space
9. Verify the following buttons for each user:
    a. Activate button for activating unactivated users.
    b. Recovery Password (following the activation process).

c. Ban button to block users, preventing them from logging in.

d. Unban button to unblock previously banned users.

10. Ensure that individual statistics are available for each manager, showing data on their managed applications.

11. Verify pagination on the manager list, displaying newest managers first.

API:

1. **Auth**
    a. **POST /login**
        i. Verify login with next credentials:
            1. Valid data:
                a. Verify the repsonse body - pass
                b. Verify status code - 200 - pass
                c. Verify response headers- - pass
                d. Verify response time - fail M2Q-441: [Auth] - POST/login - Response time is not within an acceptable range To Do
            2. Invalid data:
                a. Wrong login:
                    i. Verify the repsonse body; - pass
                    ii. Verify status code - 401; - pass
                    iii. Verify response headers; - pass
                    iv. Verify response time; - fail M2Q-441: [Auth] - POST/login - Response time is not within an acceptable range To Do
                b. Wrong password:
                    i. Verify the repsonse body; - pass
                    ii. Verify status code - 401; - pass
                    iii. Verify response headers; - pass
                    iv. Verify response time; - fail M2Q-441: [Auth] - POST/login - Response time is not within an acceptable range To Do

3. Absent data:
    a. Verify the repsonse body; - pass
    b. Verify status code - 400; - pass
    c. Verify response headers; - pass
    d. Verify response time; - pass

b. **POST /refresh**
    i. Valid token:
        1. Verify the repsonse body; - pass
        2. Verify status code; - pass
        3. Verify response headers; - pass
        4. Verify response time; - pass
    ii. Expired or invalid token:
        1. Verify the repsonse body; - pass
        2. Verify status code; - pass
        3. Verify response headers; - pass
        4. Verify response time; - pass

c. **POST /set_password**
    i. Valid re-token + valid password:
        1. Verify the repsonse body; - pass
        2. Verify status code; - pass
        3. Verify response headers; - pass
        4. Verify response time; - pass
    ii. Valid re-token + invalid password:
        1. Verify the repsonse body; - pass
        2. Verify status code; - pass
        3. Verify response headers; - pass
        4. Verify response time; - pass
    iii. Expired or invalid re-token + valid password:
        1. Verify the repsonse body; - pass
        2. Verify status code; - pass
        3. Verify response headers; - pass

4. Verify response time; - pass

iv. Expired or invalid re-token + invalid password:

1. Verify the repsonse body; - pass
2. Verify status code; - pass
3. Verify response headers; - pass
4. Verify response time; - pass

2. **Orders**

a. **GET /orders_list**

i. Check the number of orders in the response body; - pass

ii. Verify response headers; - pass

iii. Verify response time; - pass

iv. Verify status code; - pass

b. **GET /Order by id**

i. Test with valid order ID:

1. Verify the repsonse body; - pass
2. Verify status code; - pass
3. Verify response headers; - pass
4. Verify response time; - pass

ii. Test with invalid or non-existent order ID:

1. Verify the repsonse body; - pass
2. Verify status code; - pass
3. Verify response headers; - pass
4. Verify response time; - pass

c. **PATCH /Update Order**

i. Check the response body in Postman; - pass

ii. Check if changes are saved in the database; - pass

iii. Check if changes are saved on the website; - pass

iv. Verify status code; - pass

v. Verify response headers; - pass

vi. Verify response time; - pass

d. **GET /Excel**

i. Verify status code; - pass

ii. Verify response headers; - pass

iii. Verify response time; - pass

3. **Comments**

   a. **POST/ create comment**

      i. Edit or add a new comment:

         1. Check the response body; - pass
         2. Check if changes are saved in the db; - pass
         3. Check if changes are saved on the website; - pass
         4. Verify status code; - pass
         5. Verify response headers; - pass
         6. Verify response time; - pass

      ii. Leave the comment field empty.

         1. Check the response body; - pass
         2. Check if changes are saved in the db; - pass
         3. Check if changes are saved on the website; - pass
         4. Verify status code; - pass
         5. Verify response headers; - pass
         6. Verify response time; - pass

   b. **GET/ get comments**

      i. Verify if the data in the response body:

         1. matches with data in the db; - pass
         2. matches with data on the website; - pass

      ii. Status code; - pass

      iii. Response headers; - pass

      iv. Response time; - pass

4. **Users**

   a. **GET /My**

      i. Verify the response body; - pass

      ii. Verify status code; - pass

      iii. Verify response headers; - pass

iv. Verify response time; - pass

5. **Groups**
    a. **GET /group list**
        i. Verify if the data in the response body:
            1. matches with data in the db; - pass
            2. matches with data on the website; - pass
        ii. Status code; - pass
        iii. Response headers; - pass
        iv. Response time; - pass
    b. **POST /create new group**
        i. Create a new group:
            1. check the response body;- pass
            2. check the db; - pass
            3. check the site; - pass
            4. status code; - pass
            5. response headers; - pass
            6. response time; - pass
        ii. Leave the field with name empty:
            1. check the response body; - pass
            2. status code; - pass
            3. response headers; - pass
            4. response time; - pass
6. **Admin**
    a. **GET /users list**
        i. Verify if the data in the response body:
            1. matches with data in the db; - pass
            2. matches with data on the website; - pass
        ii. Status code; - pass
        iii. Response headers; - pass
        iv. Response time; - pass
    b. **POST /create user**

i. Verify with valid data (email+name+surname):
    1. Verify the data in the response body; - pass
    2. Verify if the data is saved in the db; - pass
    3. Verify if the data is displayed on the website; - pass
    4. Status code; - pass
    5. Response headers; - pass
    6. Response time. - pass

ii. Verify with Invalid data (email+name+surname):
    1. Verify the response body; - pass
    2. Status code; - pass
    3. Response headers; - pass
    4. Response time. - pass

7. **GET /Re_Token**
    a. Verify with valid user ID:
        i. The response body: - pass
        ii. Status code; - pass
        iii. Response headers; - pass
        iv. Response time; - pass
    b. Verify with invalid or non-existent user ID:
        i. The response body; - pass
        ii. Status code; - pass
        iii. Response headers; - pass
        iv. Response time; - pass
8. **PATCH /ban**
    a. Verify with valid user ID:
        i. The response body; - pass
        ii. Data in the db; - pass
        iii. Status code; - pass
        iv. Response headers; - pass
        v. Response time; - pass
        vi. Verify the account functionality by doing the following:

1. Log in on the website; - pass
2. Check if the user has its access rights:
    a. edit orders; - pass
    b. create comments; - pass

b. Verify with invalid or non-existent user ID:
    i. The response body; - pass
    ii. Status code; - pass
    iii. Response headers; - pass
    iv. Response time; - pass

9. **PATCH /unban**
    a. Verify the response body; - pass
    b. Data in the db; - pass
    c. Status code; - pass
    d. Response headers; - pass
    e. Response time; - pass
    f. Verify the account functionality by doing the following:
        i. Log in on the website; - pass
        ii. Check if the user has its access rights:
            1. edit orders; - pass
            2. create comments; - pass
10. **GET /users statistic**
    a. Verify with valid user ID:
        i. The response body; - pass
        ii. Status code; - pass
        iii. Response headers; - pass
        iv. Response time; - pass
    b. Verify with invalid or non-existent user ID:
        i. The response body; - pass
        ii. Status code; - pass
        iii. Response headers; - pass
        iv. Response time; - pass

11. **GET /orders statistic**
    a. Verify the response body; - pass
    b. Status code; - pass
    c. Response headers; - pass
    d. Response time; - pass