**Упражнение 4: HTTP Clients**

**Task 1 – (Crud Users)**

[**https://reqres.in/**](https://reqres.in/)

Using an HTTP client (e.g., HttpClient in Java), create a class that interacts with the ReqRes API. Implement the following tasks using appropriate HTTP methods (GET, POST, PUT, and DELETE):

1. **List Users**

Endpoint: https://reqres.in/api/users?page=2

Objective: Retrieve a list of users from the second page.

Task: Use the HTTP client to make a request to fetch users and print the response in a formatted way (using POJO object).

2. **Get Single User**

* Endpoint: https://reqres.in/api/users/2
* Objective: Retrieve a specific user by their ID.
* Task: Create a method that accepts a user ID as a parameter and retrieves the user’s details. Print the response to the console.

**3. Handle "User Not Found"**

* Endpoint: https://reqres.in/api/users/23
* Objective: Handle scenarios where the requested user is not found.
* Task: Create a method that tries to retrieve a user by a non-existent ID (like 23) and handles the error gracefully by printing an appropriate message when the user is not found.

**4. Create a New User**

* Endpoint: https://reqres.in/api/users
* Objective: Send a request to create a new user with a name and a job.
* Task: Write a method that accepts two parameters: name and job, and creates a new user by sending a POST request. Print the newly created user’s details from the response.

**5. Update User**

* Endpoint:
  + https://reqres.in/api/users/2
* Objective: Update the details of an existing user using
* Task 1: Write a method that accepts a user ID, name, and job, and updates the user’s. Print the updated details from the response.
* Task 2:Write a method that updates the job of user.Print the result of updated object

**6. Delete User**

Endpoint: https://reqres.in/api/users/2

* Objective: Send a request to remove a user by their ID.
* Task: Write a method that accepts a user ID and deletes the user. Print a success message once the user is deleted or if the user does not exist.

Others tesing api’s(which we used in the lection):

https://jsonplaceholder.typicode.com/

Task 2: **URL Validation and Encoding**

In this task, you will create a Java program that:

1. **Validates** if a given string is a correctly formatted URL.
2. **Encodes** the URL by converting special characters to a format that can be safely transmitted over the internet.

**Steps:**

1. **Prompt the User for Input**:
   * Ask the user to input a URL.
2. **Validate the URL**:
   * Use Java’s URL class to check if the input is a valid URL.
   * A valid URL should have a proper format (e.g., starting with http:// or https://).
3. **Encode the URL**:
   * If the URL is valid, encode it using URLEncoder with UTF-8 encoding.
   * This encoding should replace spaces and special characters with their encoded form (e.g., space is replaced by %20).
4. Decode the Url

If you have encoded url decode it using URLDecoder:

URLDecoder.decode(encodedUrl, "UTF-8");

1. **Display Results**:
   * If the URL is valid, display the encoded URL.
   * If you have encoded URL,decode it
   * If the URL is invalid, display a message saying it’s an invalid URL.

Examples :

Encode:

String baseUrl = "https://www.example.com/search";

String query = "hello world & java encoding";

Encoded URL: https://www.example.com/search?query=hello+world+%26+java+encoding

String invalidUrl = "https://example.com/search?query=dog&cat";

(Special Characters like &, ?, =, #)

https://example.com/search?query=бира