Test Case 1: Freelancer Data Fetch from /users Endpoint

Objective: Ensure that the application correctly fetches freelancer data from the /users endpoint and displays it in the dashboard.

Steps:

1. Navigate to the **dashboard** page.
2. Send a GET request to the **/users** endpoint:
    - Sample Response:

```
json
[
  {
    "id": 1,
    "name": "Omer Ornek",
    "email": "odine@test.com",
    "phone": "+905383340484",
    "photo": "https://randomuser.me/api/portraits/men/1.jpg",
    "finishedJobCount": 3,
    "city": "Izmir"
  },
  {
    "id": 2,
    "name": "Odine Labs",
    "email": "labs@test.com",
    "phone": "+902123470363",
    "photo": "https://randomuser.me/api/portraits/men/2.jpg",
    "finishedJobCount": 5,
    "city": "Izmir"
  }
]
```

3. Ensure that the following attributes are correctly displayed for each freelancer:
    - **Name**
    - **Email**
    - **Phone**
    - **Photo** (mock photo)
    - **Finished Job Count** (calculated as the number of posts for the freelancer)
    - **City**

Expected Result:

- Freelancer details should be displayed correctly on the dashboard, with appropriate fields such as name, email, phone number, photo, finished job count, and city.
- Photo should be a valid image URL and displayed correctly.

Test Case 2: Job Data Fetch from /posts Endpoint

Objective: Ensure that the application correctly fetches past job data from the /posts endpoint for each freelancer.

Steps:

1. Navigate to the **portfolio page** of a freelancer.
2. Send a GET request to the **/posts** endpoint:
    - Sample Response:

```
json
[
  {
    "userId": 1,
    "id": 1,
    "title": "Test post 1",
    "body": "Performance Tester"
  },
  {
    "userId": 1,
    "id": 2,
    "title": "Test post 2",
    "body": "Automation Tester"
  }
]
```

3. Ensure that each job has the following details displayed:
    - **Title**
    - **Job Description** (Body)

Expected Result:

- Past job titles and descriptions should be displayed correctly in the portfolio page for the freelancer.

Test Case 3: Fetch Comments Data from /comments Endpoint

Objective: Ensure that the application correctly fetches and displays job-related comments from the /comments endpoint.

Steps:

1. Navigate to the **portfolio page** of a freelancer.
2. Click on the **Show Comments** button for a specific job.
3. Send a GET request to the **/comments** endpoint to fetch job-related comments:
    - Sample Response:

```
json
[
  {
    "postId": 1,
    "id": 1,
    "name": "Omer",
    "email": "ornek@example.com",
    "body": "Good work!"
  },
```

```
  {
    "postId": 1,
    "id": 2,
    "name": "Odine",
    "email": "odine@example.com",
    "body": "Nice job!"
  }
]
```


4. Ensure that the following comment attributes are displayed correctly:
    - **Commenter Name**
    - **Commenter Email**
    - **Comment Body**

Expected Result:

- Comments for the selected job should be displayed correctly under the job title with commenter details such as name, email, and body.

Ömer Örnek