Beautify User List

now we're gonna focus on beautifying these

```
{% extends 'dashboard/dashboardAdmin.html.twig' %}

{% block title %}User List{% endblock %}

{% block body %}
    <a href="{{ path("dashboard_path") }}"><button class="btn btn-primary">Go back</button></a>

    <h1>All Users</h1>

    <table class="table table-bordered">
        <thead>
        <tr>
            <th>#</th>
            <th>Full Name</th>
            <th>Email</th>
            <th>Roles</th>
            <th>Actions</th>
        </tr>
        </thead>
        <tbody>
        {% for user in users %}
            <tr>
                <td>{{ loop.index }}</td>
                <td>{{ user.firstName }} {{ user.lastName }}</td>
                <td>{{ user.email }}</td>
                <td>{{ user.roles|join(', ') }}</td>
                <td>
                    {{ include('partials/_delete_user_form.html.twig', { user: user }) }}
                </td>
            </tr>
        {% else %}
            <tr>
                <td colspan="5">No users found.</td>
            </tr>
        {% endfor %}
        </tbody>
    </table>

{% endblock %}
```

To **beautify** your Symfony Twig template, we can improve both the **styling** and **structure** of the HTML. Here's an enhanced version that uses Bootstrap classes effectively, adds icons for visual appeal, and ensures proper layout with spacing and card-like presentation:

✅ Improved Twig Template

```twig
{% extends 'dashboard/dashboardAdmin.html.twig' %}

{% block title %}User List{% endblock %}

{% block body %}
<div class="container mt-4">
    <div class="d-flex justify-content-between align-items-center mb-3">
        <h1 class="h3">👥 All Users</h1>
        <a href="{{ path('dashboard_path') }}" class="btn btn-secondary">
            <i class="bi bi-arrow-left"></i> Go back
        </a>
    </div>

    <div class="card shadow-sm border-0">
        <div class="card-body">
            <table class="table table-hover align-middle">
                <thead class="table-light">
                    <tr>
                        <th>#</th>
                        <th>Full Name</th>
                        <th>Email</th>
                        <th>Roles</th>
                        <th>Actions</th>
                    </tr>
                </thead>
                <tbody>
                {% for user in users %}
                    <tr>
                        <td>{{ loop.index }}</td>
                        <td>{{ user.firstName }} {{ user.lastName }}</td>
                        <td>{{ user.email }}</td>
                        <td>
                            {% for role in user.roles %}
                                <span class="badge bg-primary">{{ role|replace({'ROLE_': ''})|lower
                            {% endfor %}
                        </td>
                        <td>
                            {{ include('partials/_delete_user_form.html.twig', { user: user }) }}
                        </td>
                    </tr>
                {% else %}
                    <tr>
                        <td colspan="5" class="text-center text-muted">No users found.</td>
                    </tr>
                {% endfor %}
                </tbody>
            </table>
        </div>
    </div>
</div>
{% endblock %}
```

✅ Notes on Improvements

Aspect	What Changed
Layout	Wrapped in `.container`, `.card`, and `.card-body` for better Bootstrap layout
Spacing	Used `mt-4`, `mb-3`, and `d-flex` for spacing and alignment
Styling	Added Bootstrap's `table-hover`, `table-light`, and `badge` for visual feedback
Icons	Added Bootstrap Icons (optional—see below) for the "Go back" button

Aspect	What Changed
Role display	Cleaned up role names and showed each as a styled badge

---

✅ Optional: Include Bootstrap Icons

If you want to use the icons like `<i class="bi bi-arrow-left"></i>`, include Bootstrap Icons in your base layout (`dashboardAdmin.html.twig`):

```html
<link href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.css" rel="styl
```

Would you like help beautifying the `_delete_user_form.html.twig` too?

```
{% extends 'dashboard/dashboardAdmin.html.twig' %}

{% block title %}Add New User{% endblock %}

{% block body %}
    <h1>Add a New User</h1>

    {{ form_start(form) }}
    {{ form_row(form.firstName) }}
    {{ form_row(form.lastName) }}
    {{ form_row(form.email) }}

    {{ form_end(form) }}

    <a href="{{ path("dashboard_path") }}"><button class="btn btn-primary">Go back</button></a>



{% endblock %}
```

Here's a **beautified** version of your **"Add New User"** form using Bootstrap best practices for layout, spacing, and design consistency.

✅ Improved Twig Template for User Creation

```twig
{% extends 'dashboard/dashboardAdmin.html.twig' %}

{% block title %}Add New User{% endblock %}

{% block body %}
<div class="container mt-4">
```

```
    <div class="d-flex justify-content-between align-items-center mb-3">
        <h1 class="h3">➕ Add a New User</h1>
        <a href="{{ path('dashboard_path') }}" class="btn btn-secondary">
            <i class="bi bi-arrow-left"></i> Go back
        </a>
    </div>

    <div class="card shadow-sm border-0">
        <div class="card-body">
            {{ form_start(form, {'attr': {'class': 'needs-validation'}}) }}

            <div class="mb-3">
                {{ form_row(form.firstName) }}
            </div>
            <div class="mb-3">
                {{ form_row(form.lastName) }}
            </div>
            <div class="mb-3">
                {{ form_row(form.email) }}
            </div>

            <div class="d-flex justify-content-end">
                <button type="submit" class="btn btn-success">
                    <i class="bi bi-person-plus"></i> Add User
                </button>
            </div>

            {{ form_end(form) }}
        </div>
    </div>
</div>
{% endblock %}
```

✅ What's Improved

Area	Update
Layout	Form wrapped in a Bootstrap `card` for clean presentation
Spacing	Added `mb-3` for form rows and `mt-4`/`mb-3` for spacing around sections
Submit Button	Styled with a green `btn-success` and an icon
Go Back Button	Styled as `btn-secondary` with left arrow icon

✅ Include Bootstrap Icons

To make the `<i class="bi bi-arrow-left">` and `<i class="bi bi-person-plus">` icons work, ensure this is in your base layout (`dashboardAdmin.html.twig`):

```
html

<link href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.css" rel="styl
```

Would you also like to:

- Add **password input** to the form?
- Choose **roles** during user creation?
  Let me know and I'll help update the form accordingly.

why for my friend when he enters 127.0.0.1 it don' twork for him is the reason he didn't use localhost the home works but /login don't it don't even show symfony error

The issue your friend is experiencing—**127.0.0.1 not working for `/login`, while `/` (home) works**—can be caused by a few different things. Here's a breakdown of **possible reasons** and how to fix them:

🔍 1. Accessing `127.0.0.1` accesses their own machine

Key Point:
If your Symfony app is running on *your* machine, but your friend types `127.0.0.1` in *their* browser, **they are trying to reach their own computer**, not yours.

✅ Solution:

Your friend should use **your IP address**, not `127.0.0.1`.
You can find it by running:

- On **Linux/macOS**: `ip a` or `ifconfig`
- On **Windows**: `ipconfig`

Example:
If your IP is `192.168.1.100`, your friend should visit:

```cpp
http://192.168.1.100:8000/
```

> Replace `8000` with your Symfony port if different.

🔍 2. Routing or Access Control Issue on `/login`

If `/` works but `/login` doesn't, it could be:

✅ (a) Firewall or Security Settings

Check if your firewall is blocking access to `/login`. Also verify:

```yaml
# config/packages/security.yaml
firewall:
    main:
        ...
        form_login:
            login_path: login
```

✅ (b) Access Control Rules

Symfony might be denying access to `/login` without any error showing up, depending on your `security.yaml`.

Double-check:

```yaml
access_control:
    - { path: ^/login, roles: IS_AUTHENTICATED_ANONYMOUSLY }
```

This line is needed to allow guests to access `/login`.

🔍 3. Symfony Not Showing Errors (No Debug)

If your friend sees nothing—not even a Symfony error page—it could be due to **production mode** or no debug info.

✅ Solution:

Ensure Symfony is running in **dev mode** with error reporting:

```bash
php bin/console server:run --env=dev
```

Or using Symfony CLI:

```bash
symfony serve -d
```

Also, check your `.env` file:

```env
APP_ENV=dev
APP_DEBUG=1
```

✅ Recap: What To Do

Problem	Solution
Friend using `127.0.0.1`	Use your actual local IP (e.g., `192.168.x.x`)
`/login` page doesn't work	Check `security.yaml` and `access_control`
No errors showing	Run Symfony in `dev` mode and enable `APP_DEBUG`
Route not found or misconfigured	Check `routes.yaml` or annotations for the `/login` path

---

If you send me your `security.yaml` and how your `login` route is defined (e.g., controller or routing file), I can debug the issue more precisely.

i mean when he tested the code i put on git

Something went wrong, please try reloading the conversation.

i mean when he tested the code i put on git
he can access to home but not to login and it don't show anything

Ah, thank you for the clarification — now it's clear:

> 🔁 **Your friend cloned your Symfony project from GitHub, runs it locally, accesses `` `http://127.0.0.1:8000/` ``, and home (`` `/` ``) works, but `` `/login` `` shows a blank page or nothing at all.**

✅ Here's What to Check and Fix

1. ✅ Missing `` `.env.local` `` or Database Config

Symfony uses `` `.env.local` `` or `` `.env` `` for environment variables. If your login page depends on database (like loading users), and your friend **didn't configure `` `.env.local` ``**, they may get a blank page.

Check:

```
dotenv

# .env or .env.local
DATABASE_URL="mysql://user:pass@127.0.0.1:3306/dbname"
```

Ask your friend to create the correct `` `.env.local` `` file and set their own database info.

---

2. ✅ Run Migrations / Fixtures

If there's no `` `user` `` in the DB, login might crash silently. Tell them to run:

```
bash

php bin/console doctrine:database:create
php bin/console doctrine:migrations:migrate
php bin/console doctrine:fixtures:load
```

> Or just insert a user manually via `` `doctrine:query:sql` `` or PHPMyAdmin.

---

3. ✅ Check Symfony Logs

Tell your friend to check the Symfony logs:

```bash
tail -f var/log/dev.log
```

Or open manually:

```lua
var/log/dev.log
```

If login page crashes, an exception might be logged here.

---

4. ✅ Clear the Cache

Cache issues can cause strange behavior.

```bash
php bin/console cache:clear
```

Or remove manually:

```bash
rm -rf var/cache/*
```

---

5. ✅ Access Control Rules in `security.yaml`

Ensure `/login` is **public** in `config/packages/security.yaml`:

```yaml
access_control:
    - { path: ^/login, roles: IS_AUTHENTICATED_ANONYMOUSLY }
    - { path: ^/, roles: ROLE_USER }
```

---

6. ✅ Debug Routing

Make sure the route exists:

```bash
php bin/console debug:router
```

You should see a line like:

```pgsql
login               GET     /login
```

---

7. ✅ Twig Template Not Pushed?

Check if the login template is present (e.g., `login.html.twig`). If not, Symfony might fail silently depending on config.

✅ Summary: Ask Your Friend to Do This

1. Set `.env.local` properly (especially DB info)
2. Run DB migrations and add a test user
3. Check `var/log/dev.log` for errors
4. Clear cache: `php bin/console cache:clear`
5. Make sure `/login` route exists and is public
6. Ensure login template file exists

If you'd like, paste your `security.yaml`, routing method (annotations or YAML), and login controller + template name, and I'll review everything quickly.

let's come back why not showing to the admin a list of user paginated and with the highest role they've got
+ with search options

Great — you're aiming to:

1. Show **a paginated list of users**.
2. Show **each user's highest role** clearly (e.g., show `ROLE_ADMIN` over `ROLE_USER`).
3. Add a **search bar** to filter users (by name/email/etc).

Let's build that step-by-step in your Symfony + Twig project.

✅ 1. Pagination with KnpPaginatorBundle

First, install KnpPaginator if you haven't already:

```bash
composer require knplabs/knp-paginator-bundle
```

In your controller:

```
php

use Knp\Component\Pager\PaginatorInterface;
use Symfony\Component\HttpFoundation\Request;

#[Route('/admin/users', name: 'admin_user_list')]
public function userList(UserRepository $userRepo, Request $request, PaginatorInterface $paginator)
{
    $search = $request->query->get('q', '');

    $queryBuilder = $userRepo->createQueryBuilder('u');

    if ($search) {
        $queryBuilder->where('u.firstName LIKE :search OR u.lastName LIKE :search OR u.email LIKE :
                     ->setParameter('search', '%' . $search . '%');
    }

    $pagination = $paginator->paginate(
        $queryBuilder,
        $request->query->getInt('page', 1),
        10 // items per page
    );

    return $this->render('admin/user_list.html.twig', [
        'pagination' => $pagination,
        'search' => $search,
    ]);
}
```

✅ 2. Update Twig Template

Update your `user_list.html.twig`:

```
twig

{% extends 'dashboard/dashboardAdmin.html.twig' %}

{% block title %}User List{% endblock %}

{% block body %}
    <h1 class="mb-4">All Users</h1>

    <form method="get" class="mb-4 d-flex">
        <input type="text" name="q" value="{{ search }}" placeholder="Search..." class="form-contro
        <button class="btn btn-outline-primary">Search</button>
    </form>

    <table class="table table-bordered table-hover">
        <thead class="table-light">
        <tr>
            <th>#</th>
            <th>Full Name</th>
            <th>Email</th>
            <th>Highest Role</th>
            <th>Actions</th>
        </tr>
        </thead>
        <tbody>
        {% for user in pagination %}
            <tr>
                <td>{{ loop.index + (pagination.currentPageNumber - 1) * pagination.itemNumberPerPa
                <td>{{ user.firstName }} {{ user.lastName }}</td>
                <td>{{ user.email }}</td>
                <td>
                    {% set roles = user.roles %}
                    {% set sortedRoles = ['ROLE_SUPER_ADMIN', 'ROLE_ADMIN', 'ROLE_MANAGER', 'ROLE_U
                    {% for role in sortedRoles if role in roles %}
                        {{ role }}
                        {% break %}
                    {% endfor %}
                </td>
                <td>
                    {{ include('partials/_delete_user_form.html.twig', { user: user }) }}
                </td>
            </tr>
        {% else %}
            <tr><td colspan="5">No users found.</td></tr>
        {% endfor %}
        </tbody>
    </table>

    {{ knp_pagination_render(pagination) }}

    <a href="{{ path("dashboard_path") }}" class="btn btn-primary mt-3">Go back</a>
{% endblock %}
```

✅ 3. Notes on Styling and Experience

- You can use Bootstrap 5 (already used here).
- Consider adding sortable columns if needed.
- You can display badges for roles (like `` `badge bg-success` `` for admin, etc.).

✅ Example Result

#	Name	Email	Highest Role	Actions
1	Jane Smith	jane@example.com	ROLE_ADMIN	Delete
2	John Doe	john@example.com	ROLE_USER	Delete

🔍 Search bar at top.
📄 Pagination at bottom.

Would you like to:

- Add **role badge colors** (e.g., red for admin, blue for user)?
- Add **edit user** buttons?
- Or allow sorting (by name, email, etc)?

Let me know and we'll enhance it!

can we just get the max without any bundles

Yes, absolutely! You **don't need any extra bundle** to:

1. Paginate manually.
2. Get the **highest role** per user.
3. Add a **search bar**.

Here's how to do everything **without KnpPaginatorBundle**.

✅ Step 1: Controller – Manual Pagination + Search

```php
use Symfony\Component\HttpFoundation\Request;
use App\Repository\UserRepository;
use Symfony\Component\Routing\Annotation\Route;

#[Route('/admin/users', name: 'admin_user_list')]
public function index(UserRepository $userRepo, Request $request)
{
    $search = $request->query->get('q', '');
    $page = max(1, (int)$request->query->get('page', 1));
    $limit = 10;
    $offset = ($page - 1) * $limit;

    $qb = $userRepo->createQueryBuilder('u');

    if ($search) {
        $qb->where('u.firstName LIKE :search OR u.lastName LIKE :search OR u.email LIKE :search')
           ->setParameter('search', '%' . $search . '%');
    }

    $qb->setFirstResult($offset)
       ->setMaxResults($limit);

    $users = $qb->getQuery()->getResult();

    // Count total results
    $total = (clone $qb)->select
```

without pagination nor anything just show the highest and if the user is verified or not with different collors

Perfect! Here's how to:

- ✅ Show all users.
- ✅ Show the **highest role** from their `roles` array.
- ✅ Show whether the user is **verified or not** (`user.isVerified`).
- ✅ Add **color styling** (e.g., role badges, verification status badges).

✅ Controller (no changes needed if you're passing `users`)

Assuming you're already passing users to your Twig like this:

```php
#[Route('/admin/users', name: 'admin_user_list')]
public function index(UserRepository $userRepo)
{
    $users = $userRepo->findAll();

    return $this->render('admin/user_list.html.twig', [
        'users' => $users,
    ]);
}
```

✅ Twig Template (fully updated)

```
twig

{% extends 'dashboard/dashboardAdmin.html.twig' %}

{% block title %}User List{% endblock %}

{% block body %}
    <h1 class="mb-4">All Users</h1>

    <table class="table table-bordered table-hover align-middle">
        <thead class="table-light">
            <tr>
                <th>#</th>
                <th>Full Name</th>
                <th>Email</th>
                <th>Highest Role</th>
                <th>Verified</th>
                <th>Actions</th>
            </tr>
        </thead>
        <tbody>
        {% for user in users %}
            <tr>
                <td>{{ loop.index }}</td>
                <td>{{ user.firstName }} {{ user.lastName }}</td>
                <td>{{ user.email }}</td>

                <td>
                    {% set roles = user.roles %}
                    {% set priority = ['ROLE_SUPER_ADMIN', 'ROLE_ADMIN', 'ROLE_MANAGER', 'ROLE_USER
                    {% for role in priority if role in roles %}
                        {% set highest = role %}
                        {% break %}
                    {% endfor %}

                    {% if highest == 'ROLE_SUPER_ADMIN' %}
                        <span class="badge bg-danger">Super Admin</span>
                    {% elseif highest == 'ROLE_ADMIN' %}
                        <span class="badge bg-warning text-dark">Admin</span>
                    {% elseif highest == 'ROLE_MANAGER' %}
                        <span class="badge bg-info text-dark">Manager</span>
                    {% else %}
                        <span class="badge bg-secondary">User</span>
                    {% endif %}
                </td>

                <td>
                    {% if user.isVerified %}
                        <span class="badge bg-success">Verified</span>
                    {% else %}
                        <span class="badge bg-danger">Unverified</span>
                    {% endif %}
                </td>

                <td>
                    {{ include('partials/_delete_user_form.html.twig', { user: user }) }}
                </td>
            </tr>
        {% else %}
            <tr><td colspan="6" class="text-center">No users found.</td></tr>
        {% endfor %}
        </tbody>
    </table>
    <a href="{{ path("dashboard_path") }}" class="btn btn-primary mt-3">Go back</a>
{% endblock %}
```

🔥 Output Example:

#	Name	Email	Role Badge	Verified
1	John Smith	john@x.com	🟥 Super Admin	✅ Green
2	Alice Jones	alice@x.com	🟨 Admin	❌ Red

Want to:

- Add filters for roles?
- Add icons next to roles or verification?
- Add tooltips?

Just ask!

{% for role in priority if role in roles %}

The line:

```
twig

{% for role in priority if role in roles %}
```

is valid Twig **only in recent versions** (Twig ≥ 2.12). It filters the loop inline. However, if your Symfony project uses an **older Twig version**, this syntax will **not work** and throw an error.

✅ Safe Alternative (compatible with all Twig versions):

Use a classic loop and an `` `if` `` inside:

```
twig

{% set highest = null %}
{% for role in priority %}
    {% if role in roles %}
        {% set highest = role %}
        {% break %}
    {% endif %}
{% endfor %}
```

This:

- Loops through `` `priority` ``
- Checks if the role exists in the user's `` `roles` ``
- Stops at the first match and sets `` `highest` ``

💡 Reminder

Make sure `priority` is defined before this block:

```
twig

{% set roles = user.roles %}
{% set priority = ['ROLE_SUPER_ADMIN', 'ROLE_ADMIN', 'ROLE_MANAGER', 'ROLE_USER'] %}
```

Let me know if you'd like the roles mapped to user-friendly names in a separate function or macro!

i have admin formateur and user

Got it! Your roles are:

- `ROLE_ADMIN`
- `ROLE_FORMATEUR`
- `ROLE_USER`

Here’s how to prioritize and display **highest role** accordingly in Twig:

Updated priority array:

```
twig

{% set priority = ['ROLE_ADMIN', 'ROLE_FORMATEUR', 'ROLE_USER'] %}
```

Twig snippet to get highest role safely:

```
twig

{% set highest = null %}
{% for role in priority %}
    {% if role in user.roles %}
        {% set highest = role %}
        {% break %}
    {% endif %}
{% endfor %}
```

Display role badges with colors:

```
twig

<td>
    {% if highest == 'ROLE_ADMIN' %}
        <span class="badge bg-danger">Admin</span>
    {% elseif highest == 'ROLE_FORMATEUR' %}
        <span class="badge bg-info text-dark">Formateur</span>
    {% else %}
        <span class="badge bg-secondary">User</span>
    {% endif %}
</td>
```

Full example in your table row:

```
twig

<tr>
    <td>{{ loop.index }}</td>
    <td>{{ user.firstName }} {{ user.lastName }}</td>
    <td>{{ user.email }}</td>

    {% set priority = ['ROLE_ADMIN', 'ROLE_FORMATEUR', 'ROLE_USER'] %}
    {% set highest = null %}
    {% for role in priority %}
        {% if role in user.roles %}
            {% set highest = role %}
            {% break %}
        {% endif %}
    {% endfor %}

    <td>
        {% if highest == 'ROLE_ADMIN' %}
            <span class="badge bg-danger">Admin</span>
        {% elseif highest == 'ROLE_FORMATEUR' %}
            <span class="badge bg-info text-dark">Formateur</span>
        {% else %}
            <span class="badge bg-secondary">User</span>
        {% endif %}
    </td>

    <td>
        {% if user.isVerified %}
            <span class="badge bg-success">Verified</span>
        {% else %}
            <span class="badge bg-danger">Unverified</span>
        {% endif %}
    </td>

    <td>
        {{ include('partials/_delete_user_form.html.twig', { user: user }) }}
    </td>
</tr>
```

If you want, I can help you write a **Twig macro** for the role badges so you avoid repeating code — just ask!

the verified not verified part

```
{% extends 'dashboard/dashboardAdmin.html.twig' %}

{% block title %}User List{% endblock %}

{% block body %}
  <div class="container mt-4">
    <div class="d-flex justify-content-between align-items-center mb-3">
      <h1 class="h3">👥 All Users</h1>
      <a href="{{ path('dashboard_path') }}" class="btn btn-secondary">
        <i class="bi bi-arrow-left"></i> Go back
```

```
            </a>
        </div>

        <div class="card shadow-sm border-0">
            <div class="card-body">
                <table class="table table-hover align-middle">
                    <thead class="table-light">
                    <tr>
                        <th>#</th>
                        <th>Full Name</th>
                        <th>Email</th>
                        <th>Roles</th>
                        <th>State</th>
                        <th>Actions</th>
                    </tr>
                    </thead>
                    <tbody>
                    {% for user in users %}
                        <tr>
                            <td>{{ loop.index }}</td>
                            <td>{{ user.firstName }} {{ user.lastName }}</td>
                            <td>{{ user.email }}</td>
                            <td>
                                {% for role in user.roles %}
                                    <span class="badge bg-primary">{{ role|replace({'ROLE_': ''})|lower|capitalize }}
</span>
                                {% endfor %}
                            </td>
                            <td>
                                {% if user.verifed %}
                                    <p>Verified</p>
                                    {% else %}
                                    <p>Not verified</p>
                                {% endif %}


                            </td>

                            <td>
                                {{ include('partials/_delete_user_form.html.twig', { user: user }) }}
                            </td>
                        </tr>
                    {% else %}
                        <tr>
                            <td colspan="5" class="text-center text-muted">No users found.</td>
                        </tr>
                    {% endfor %}
                    </tbody>
                </table>
            </div>
        </div>
```

```
    </div>
{% endblock %}
```

Looks great overall! Just one thing — there's a typo in your verification check here:

```twig
{% if user.verifed %}
```

It should be:

```twig
{% if user.verified %}
```

(or whatever your actual property name is; often it's `isVerified`)

Here's a corrected & polished version with badges for verified/unverified:

```twig
<td>
    {% if user.verified %}
        <span class="badge bg-success">Verified</span>
    {% else %}
        <span class="badge bg-danger">Not verified</span>
    {% endif %}
</td>
```

Full snippet integrated in your table row for the verification column:

```twig
<td>
    {% if user.verified %}
        <span class="badge bg-success">Verified</span>
    {% else %}
        <span class="badge bg-danger">Not verified</span>
    {% endif %}
</td>
```

If your property is named differently (like `isVerified`), adjust accordingly.

If you want it super compact and consistent with your role badges, you can also do:

```twig
<td>
    <span class="badge {{ user.verified ? 'bg-success' : 'bg-danger' }}">
        {{ user.verified ? 'Verified' : 'Not verified' }}
    </span>
</td>
```

Would you like me to help with any other UI improvements?

choose the highest role
admin>formateur>user

```
{% for role in user.roles %}

                                   <span class="badge bg-primary">{{ role|replace({'ROLE_': ''})|lower|capitalize }}
</span>
                               {% endfor %}
```

To show **only the highest role** among the user's roles (`admin > formateur > user`), you can:

1. Define a priority list.
2. Loop through it to find the first role that the user has.
3. Display only that role as a badge.

Example Twig code:

```twig
{% set priority = ['ROLE_ADMIN', 'ROLE_FORMATEUR', 'ROLE_USER'] %}
{% set highestRole = null %}

{% for role in priority %}
    {% if role in user.roles %}
        {% set highestRole = role %}
        {% break %}
    {% endif %}
{% endfor %}

{% if highestRole %}
    <span class="badge
        {% if highestRole == 'ROLE_ADMIN' %}bg-danger
        {% elseif highestRole == 'ROLE_FORMATEUR' %}bg-info text-dark
        {% else %}bg-secondary
        {% endif %}
    ">
        {{ highestRole|replace({'ROLE_': ''})|lower|capitalize }}
    </span>
{% endif %}
```

How it works:

- The priority list sets the order of importance.
- The loop finds the highest role the user has.
- The badge color matches the role.

If you want me to integrate it into your full table row or add verified status too, just ask!

format the spacing

```
{% extends 'dashboard/dashboardAdmin.html.twig' %}

{% block title %}User List{% endblock %}

{% block body %}
  <div class="container mt-4">
    <div class="d-flex justify-content-between align-items-center mb-3">
      <h1 class="h3">👥 All Users</h1>
      <a href="{{ path('dashboard_path') }}" class="btn btn-secondary">
        <i class="bi bi-arrow-left"></i> Go back
      </a>
    </div>

    <div class="card shadow-sm border-0">
      <div class="card-body">
        <table class="table table-hover align-middle">
          <thead class="table-light">
          <tr>
            <th>#</th>
            <th>Full Name</th>
            <th>Email</th>
            <th>Roles</th>
            <th>State</th>
            <th>Actions</th>
          </tr>
          </thead>
          <tbody>
          {% for user in users %}
            <tr>
              <td>{{ loop.index }}</td>
              <td>{{ user.firstName }} {{ user.lastName }}</td>
              <td>{{ user.email }}</td>
              <td>
                {% set priority = ['ROLE_ADMIN', 'ROLE_FORMATEUR', 'ROLE_USER'] %}
                {% set highestRole = null %}

                {% for role in priority %}
                  {% if role in user.roles %}
                    {% set highestRole = role %}
                    {% break %}
                  {% endif %}
                {% endfor %}

                {% if highestRole %}
                  <span class="badge
    {% if highestRole == 'ROLE_ADMIN' %}bg-danger
    {% elseif highestRole == 'ROLE_FORMATEUR' %}bg-info text-dark
    {% else %}bg-secondary
    {% endif %}
  ">
    {{ highestRole|replace({'ROLE_': ''})|lower|capitalize }}
  </span>
                {% endif %}

              </td>

              <td>
                {% if user.verified %}
                  <span class="badge bg-success">Verified</span>
                {% else %}
                  <span class="badge bg-danger">Not verified</span>
                {% endif %}
              </td>



              </td>

              <td>
                {{ include('partials/_delete_user_form.html.twig', { user: user }) }}
              </td>
            </tr>
          {% else %}
            <tr>
              <td colspan="5" class="text-center text-muted">No users found.</td>
            </tr>
          {% endfor %}
          </tbody>
        </table>
      </div>
    </div>
  </div>
{% endblock %}
```

Sure! Here's your Twig template with consistent indentation and nicely formatted spacing for better readability:

```twig
{% extends 'dashboard/dashboardAdmin.html.twig' %}

{% block title %}User List{% endblock %}

{% block body %}
    <div class="container mt-4">
        <div class="d-flex justify-content-between align-items-center mb-3">
            <h1 class="h3">👥 All Users</h1>
            <a href="{{ path('dashboard_path') }}" class="btn btn-secondary">
                <i class="bi bi-arrow-left"></i> Go back
            </a>
        </div>

        <div class="card shadow-sm border-0">
            <div class="card-body">
                <table class="table table-hover align-middle">
```

```
                    <thead class="table-light">
                        <tr>
                            <th>#</th>
                            <th>Full Name</th>
                            <th>Email</th>
                            <th>Roles</th>
                            <th>State</th>
                            <th>Actions</th>
                        </tr>
                    </thead>
                    <tbody>
                        {% for user in users %}
                            <tr>
                                <td>{{ loop.index }}</td>
                                <td>{{ user.firstName }} {{ user.lastName }}</td>
                                <td>{{ user.email }}</td>
                                <td>
                                    {% set priority = ['ROLE_ADMIN', 'ROLE_FORMATEUR', 'ROLE_USER']
                                    {% set highestRole = null %}

                                    {% for role in priority %}
                                        {% if role in user.roles %}
                                            {% set highestRole = role %}
                                            {% break %}
                                        {% endif %}
                                    {% endfor %}

                                    {% if highestRole %}
                                        <span class="badge
                                            {% if highestRole == 'ROLE_ADMIN' %} bg-danger
                                            {% elseif highestRole == 'ROLE_FORMATEUR' %} bg-info te
                                            {% else %} bg-secondary
                                            {% endif %}
                                        ">
                                            {{ highestRole|replace({'ROLE_': ''})|lower|capitalize
                                        </span>
                                    {% endif %}
                                </td>

                                <td>
                                    {% if user.verified %}
                                        <span class="badge bg-success">Verified</span>
                                    {% else %}
                                        <span class="badge bg-danger">Not verified</span>
                                    {% endif %}
                                </td>

                                <td>
                                    {{ include('partials/_delete_user_form.html.twig', { user: user
                                </td>
                            </tr>
                        {% else %}
                            <tr>
                                <td colspan="6" class="text-center text-muted">No users found.</td>
                            </tr>
                        {% endfor %}
                    </tbody>
                </table>
            </div>
        </div>
    </div>
{% endblock %}
```

Notes:

- Indented nested blocks and tags uniformly.
- Fixed the `colspan` in the "No users found" row to 6 (because your table has 6 columns).
- Cleaned the badge class lines so they align properly and the closing tags are tidy.

If you want me to help add search options or pagination next, just say!

Unexpected "break" tag (expecting closing tag for the "if" tag defined near line 39) in admin_roles/user_list.html.twig at line 40.

Ah, Twig doesn't support the `break` tag inside loops. To mimic `break` behavior, you need to restructure the code.

Here's a clean workaround using a macro or just a filter-free approach:

Simple fix without `break`:

```twig
{% set priority = ['ROLE_ADMIN', 'ROLE_FORMATEUR', 'ROLE_USER'] %}
{% set highestRole = null %}

{% for role in priority %}
    {% if role in user.roles and highestRole is null %}
        {% set highestRole = role %}
    {% endif %}
{% endfor %}
```

Twig will run the whole loop but only set `highestRole` once because of the `highestRole is null` condition.

Full corrected snippet for your role detection:

```twig
{% set priority = ['ROLE_ADMIN', 'ROLE_FORMATEUR', 'ROLE_USER'] %}
{% set highestRole = null %}

{% for role in priority %}
    {% if role in user.roles and highestRole is null %}
        {% set highestRole = role %}
    {% endif %}
{% endfor %}

{% if highestRole %}
    <span class="badge
        {% if highestRole == 'ROLE_ADMIN' %} bg-danger
        {% elseif highestRole == 'ROLE_FORMATEUR' %} bg-info text-dark
        {% else %} bg-secondary
        {% endif %}
    ">
        {{ highestRole|replace({'ROLE_': ''})|lower|capitalize }}
    </span>
{% endif %}
```

This avoids `break` but gets the same logic done in pure Twig.

Would you like me to patch this into your whole template?

beautfy this

```
{# templates/admin/formateur_requests.html.twig #}

{% extends 'dashboard/dashboardAdmin.html.twig' %}

{% block title %}Gestion des demandes formateurs{% endblock %}

{% block body %}

    <h1>Demandes pour devenir formateur</h1>

    {% for message in app.flashes('success') %}
        <div class="alert alert-success">{{ message }}</div>
    {% endfor %}
    {% for message in app.flashes('info') %}
        <div class="alert alert-info">{{ message }}</div>
    {% endfor %}
    {% for message in app.flashes('warning') %}
        <div class="alert alert-warning">{{ message }}</div>
    {% endfor %}

    <table class="table table-bordered">
        <thead>
        <tr>
            <th>Utilisateur</th>
            <th>Email</th>
            <th>Motivation</th>
            <th>experience</th>
            <th>Date de demande</th>
            <th>cvPath</th>
        </tr>
        </thead>
        <tbody>
        {% for request in requests %}
            <tr>
                <td>{{ request.user.firstname }} {{ request.user.lastname }}</td>
                <td>{{ request.user.email }}</td>
                <td>{{ request.motivation }}</td>
                <td>{{ request.experience }}</td>
                <td>{{ request.requestedAt ? request.requestedAt|date('d/m/Y H:i') : 'N/A' }}</td>
                <td>
                    {% if request.pdfFilename %}
                        <a href="{{ asset('uploads/pdfs/' ~ request.pdfFilename) }}" target="_blank">Voir PDF</a>
                    {% else %}
                        N/A
                    {% endif %}
                </td>
                <td>
                    {% if not request.treated %}
```

```
                        <span class="badge bg-warning text-dark">En attente</span>
                    {% elseif request.accepted %}
                        <span class="badge bg-success">Acceptée</span>
                    {% else %}
                        <span class="badge bg-danger">Refusée</span>
                    {% endif %}
                </td>
                <td>
                    {% if not request.treated %}
                        <form method="post" action="{{ path('admin_formateur_accept', {'id': request.id}) }}"
style="display:inline-block;">
                            <button class="btn btn-success btn-sm" type="submit" onclick="return
confirm('Accepter cette demande ?')">Accepter</button>
                        </form>
                        <form method="post" action="{{ path('admin_formateur_refuse', {'id': request.id}) }}"
style="display:inline-block;">
                            <button class="btn btn-danger btn-sm" type="submit" onclick="return
confirm('Refuser cette demande ?')">Refuser</button>
                        </form>
                    {% else %}
                        <em>Aucune action disponible</em>
                    {% endif %}
                </td>
            </tr>
        {% else %}
            <tr><td colspan="6">Aucune demande trouvée.</td></tr>
        {% endfor %}
        </tbody>
    </table>

{% endblock %}
```

Here's a cleaner, better formatted, and visually improved version of your Twig template:

```twig
{# templates/admin/formateur_requests.html.twig #}

{% extends 'dashboard/dashboardAdmin.html.twig' %}

{% block title %}Gestion des demandes formateurs{% endblock %}

{% block body %}
    <div class="container mt-4">
        <h1 class="mb-4">Demandes pour devenir formateur</h1>

        {# Flash messages #}
        {% for type in ['success', 'info', 'warning'] %}
            {% for message in app.flashes(type) %}
                <div class="alert alert-{{ type }} alert-dismissible fade show" role="alert">
                    {{ message }}
                    <button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Clo
                </div>
            {% endfor %}
        {% endfor %}

        <div class="card shadow-sm border-0">
            <div class="card-body p-0">
```

```
                <table class="table table-hover align-middle mb-0">
                    <thead class="table-light">
                        <tr>
                            <th>Utilisateur</th>
                            <th>Email</th>
                            <th>Motivation</th>
                            <th>Expérience</th>
                            <th>Date de demande</th>
                            <th>CV</th>
                            <th>Statut</th>
                            <th>Actions</th>
                        </tr>
                    </thead>
                    <tbody>
                        {% for request in requests %}
                            <tr>
                                <td>{{ request.user.firstname }} {{ request.user.lastname }}</td>
                                <td>{{ request.user.email }}</td>
                                <td>{{ request.motivation }}</td>
                                <td>{{ request.experience }}</td>
                                <td>{{ request.requestedAt ? request.requestedAt|date('d/m/Y H:i')
                                <td>
                                    {% if request.pdfFilename %}
                                        <a href="{{ asset('uploads/pdfs/' ~ request.pdfFilename) }}
                                            Voir PDF
                                        </a>
                                    {% else %}
                                        <span class="text-muted">N/A</span>
                                    {% endif %}
                                </td>
                                <td>
                                    {% if not request.treated %}
                                        <span class="badge bg-warning text-dark">En attente</span>
                                    {% elseif request.accepted %}
                                        <span class="badge bg-success">Acceptée</span>
                                    {% else %}
                                        <span class="badge bg-danger">Refusée</span>
                                    {% endif %}
                                </td>
                                <td>
                                    {% if not request.treated %}
                                        <form method="post" action="{{ path('admin_formateur_accept
                                            <button
                                                class="btn btn-success btn-sm"
                                                type="submit"
                                                onclick="return confirm('Accepter cette demande ?')
                                            >
                                                Accepter
                                            </button>
                                        </form>
                                        <form method="post" action="{{ path('admin_formateur_refuse
                                            <button
                                                class="btn btn-danger btn-sm"
                                                type="submit"
                                                onclick="return confirm('Refuser cette demande ?')"
                                            >
                                                Refuser
                                            </button>
                                        </form>
                                    {% else %}
                                        <em class="text-muted">Aucune action disponible</em>
                                    {% endif %}
                                </td>
                            </tr>
                        {% else %}
                            <tr>
                                <td colspan="8" class="text-center text-muted">Aucune demande trouv
                            </tr>
                        {% endfor %}
                    </tbody>
                </table>
            </div>
        </div>
```

```
    </div>
{% endblock %}
```

Improvements:

- Wrapped content in a `.container` with spacing classes for consistent padding/margin.
- Used Bootstrap's alert-dismissible style for flash messages with close buttons.
- Added `table-hover` and `align-middle` for better UI.
- Fixed inconsistent column spans (added 8 cols to `colspan` in "no requests" row).
- Used Bootstrap spacing utilities (`d-inline`, `ms-2`) to space action buttons nicely.
- Added semantic `text-muted` for "N/A" and "no actions" texts.
- Cleaned indentation for better readability.
- Added `text-decoration-none` on PDF link for nicer styling.

If you want, I can also help you add search/filter or pagination later!

if user.email== currentuser.email
u can't delete yourself

Got it! You want to prevent the current logged-in user from deleting their own account in the user list.

Here's how you can do it in Twig, assuming you have access to the current user in your template as `app.user`:

Replace your delete form part with a condition that disables or hides the delete button if the `user.email` equals the current logged-in user's email:

```twig
<td>
    {% if user.email != app.user.email %}
        {{ include('partials/_delete_user_form.html.twig', { user: user }) }}
    {% else %}
        <span class="text-muted" title="Vous ne pouvez pas supprimer votre propre compte">—</span>
    {% endif %}
</td>
```

Explanation:

- If the user in the loop is **not** the current user, show the delete form.
- Otherwise, show a dash or any placeholder with a tooltip explaining that you can't delete yourself.

If you want the button visible but disabled, you can also do:

```twig
<td>
    {% if user.email == app.user.email %}
        <button class="btn btn-danger btn-sm" disabled title="Vous ne pouvez pas supprimer votre pr
    {% else %}
        {{ include('partials/_delete_user_form.html.twig', { user: user }) }}
    {% endif %}
</td>
```

Would you like me to generate the full block or any additional UI tweak?

```
{% extends 'base.html.twig' %}

{% block title %}Espace Admin{% endblock %}
{% block nav %}
    <div class="collapse navbar-collapse justify-content-end" id="navLinks">
        <ul class="navbar-nav">
            <li class="nav-item"><a class="nav-link" href="{{ path('admin_formateur_requests') }}">See Requests</a></li>
            <li class="nav-item"><a class="nav-link" href="{{ path('admin_user_add') }}">Add a user</a></li>
            <li class="nav-item"><a class="nav-link" href="{{ path('admin_user_list') }}">Users list</a></li>
            <li class="nav-item"><a class="nav-link" href="{{ path('dashboard_path') }}">Dashboard</a></li>
            <li class="nav-item"><a class="nav-link" href="{{ path('app_logout') }}"> log out</a></li>
        </ul>
    </div>
{% endblock %}
{% block body %}
    <div class="container mt-5">
        <h1 class="mb-4 text-primary">Espace Admin</h1>

        <div class="row g-4">
            <div class="col-md-4">
                <div class="card border-primary h-100 text-center">
                    <div class="card-body d-flex flex-column justify-content-center">
                        <h5 class="card-title">Modifier les utilisateurs</h5>
                        <p class="card-text">Voir, modifier ou supprimer les utilisateurs existants.</p>
                        <a href="{{ path('admin_user_list') }}" class="btn btn-primary mt-auto">Gérer les utilisateurs</a>
                    </div>
                </div>
            </div>

            <div class="col-md-4">
                <div class="card border-success h-100 text-center">
                    <div class="card-body d-flex flex-column justify-content-center">
                        <h5 class="card-title">Ajouter un utilisateur</h5>
                        <p class="card-text">Créer un nouvel utilisateur dans le système.</p>
                        <a href="{{ path('admin_user_add') }}" class="btn btn-success mt-auto">Ajouter un utilisateur</a>
                    </div>
                </div>
            </div>

            <div class="col-md-4">
                <div class="card border-warning h-100 text-center">
                    <div class="card-body d-flex flex-column justify-content-center">
                        <h5 class="card-title">Gérer les demandes formateur</h5>
                        <p class="card-text">Accepter ou refuser les demandes de formateur.</p>
                        <a href="{{ path('admin_formateur_requests') }}" class="btn btn-warning mt-auto">Voir les demandes</a>
                    </div>
                </div>
            </div>
        </div>
    </div>
{% endblock %}
```

Your Twig template looks well-structured and clean already! Here's a slightly "beautified" version with consistent indentation and some minor formatting tweaks to keep things clear and neat:

```twig
{% extends 'base.html.twig' %}

{% block title %}Espace Admin{% endblock %}

{% block nav %}
    <div class="collapse navbar-collapse justify-content-end" id="navLinks">
        <ul class="navbar-nav">
            <li class="nav-item">
                <a class="nav-link" href="{{ path('admin_formateur_requests') }}">See Requests</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="{{ path('admin_user_add') }}">Add a user</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="{{ path('admin_user_list') }}">Users list</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="{{ path('dashboard_path') }}">Dashboard</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="{{ path('app_logout') }}">Log out</a>
            </li>
        </ul>
    </div>
{% endblock %}

{% block body %}
    <div class="container mt-5">
        <h1 class="mb-4 text-primary">Espace Admin</h1>

        <div class="row g-4">
            <div class="col-md-4">
                <div class="card border-primary h-100 text-center">
                    <div class="card-body d-flex flex-column justify-content-center">
                        <h5 class="card-title">Modifier les utilisateurs</h5>
                        <p class="card-text">Voir, modifier ou supprimer les utilisateurs existants
                        <a href="{{ path('admin_user_list') }}" class="btn btn-primary mt-auto">Gér
                    </div>
                </div>
            </div>

            <div class="col-md-4">
                <div class="card border-success h-100 text-center">
```

```
                    <div class="card-body d-flex flex-column justify-content-center">
                        <h5 class="card-title">Ajouter un utilisateur</h5>
                        <p class="card-text">Créer un nouvel utilisateur dans le système.</p>
                        <a href="{{ path('admin_user_add') }}" class="btn btn-success mt-auto">Ajou
                    </div>
                </div>
            </div>

            <div class="col-md-4">
                <div class="card border-warning h-100 text-center">
                    <div class="card-body d-flex flex-column justify-content-center">
                        <h5 class="card-title">Gérer les demandes formateur</h5>
                        <p class="card-text">Accepter ou refuser les demandes de formateur.</p>
                        <a href="{{ path('admin_formateur_requests') }}" class="btn btn-warning mt-
                    </div>
                </div>
            </div>
        </div>
    </div>
{% endblock %}
```

What I did:

- Added consistent indentation for readability.
- Each `` `<li>` `` and `` `<a>` `` are on their own lines.
- Capitalized "Log out" in nav for consistency.
- Kept the Bootstrap classes and structure intact.

If you want, I can also help with making the nav responsive or add active link highlighting! Just ask.

how to uncommit change branch without my code getting deleted in editor

If you've committed changes **on the wrong branch** but want to undo that commit **without losing your code changes in the editor** (i.e., keep them as uncommitted changes in your working directory), here's what you can do with Git:

Undo last commit but keep changes unstaged (in your editor)

```bash
git reset --soft HEAD~1
```

- This **undoes the last commit**, but keeps **all changes staged** (ready to commit).
- If you want the changes **unstaged** (like before you ran `` `git add` ``), use:

```bash
git reset HEAD~1
```

- This **undoes the commit** and **unstages** the changes, but leaves them in your working directory so your editor still shows your changes.

Then, switch to the correct branch and recommit

1. Switch to the correct branch:

```bash
git checkout correct-branch-name
```

2. Commit your changes there:

```bash
git commit -m "Your commit message"
```

Summary

- **`` `git reset HEAD~1` ``** — Undo last commit, keep changes in your working directory (unstaged).
- Your files remain unchanged in the editor.
- You can now switch branches safely and recommit.

If you want me to help with commands to handle multiple commits or other cases, just ask!

beautify this and make it proper new and moder it's an e_commerce website

```
{% extends 'base.html.twig' %}

{% block title %}Mon Dashboard{% endblock %}

{% block nav %}
    <div class="collapse navbar-collapse justify-content-end" id="navLinks">
        <ul class="navbar-nav">
            <li class="nav-item"><a class="nav-link" href="{{ path('dashboard_path') }}">Dashboard</a></li>
            <li class="nav-item"><a class="nav-link" href="{{ path('user_demand') }}">Devient a formateur</a></li>
            <li class="nav-item"><a class="nav-link" href="{{ path('user_show_profile') }}">Profil</a></li>
            <li class="nav-item"><a href="{{ path('app_panier') }}" class="btn btn-outline-secondary position-relative me-2">
                    🛒
                    {# Si tu veux afficher un nombre de produits dans le panier plus tard : #}
                    {# <span class="position-absolute top-0 start-100 translate-middle badge rounded-pill bg-danger">3</span> #}
                </a></li>
            <li class="nav-item"><a class="nav-link" href="{{ path('app_logout') }}"> log out</a></li>

        </ul>
    </div>
{% endblock %}

{% block body %}
    <style>
        .block-wrapper {
            position: relative;
            display: inline-block;
            width: 100%;
            height: 100%;
        }

        /* The red diagonal "band" */
        .block-wrapper.blocked::before {
            content: "UNVERIFIED";
            position: absolute;
            top: 20px;
            right: -50px;
            width: 200px;
            padding: 5px 0;
            background: rgba(255, 0, 0, 0.8);
            color: white;
            font-weight: bold;
            font-size: 14px;
            text-align: center;
            transform: rotate(45deg);
            pointer-events: none;
            z-index: 10;
            box-shadow: 0 0 10px red;
        }

        /* Disable clicking and interaction */
        .block-wrapper.blocked {
            pointer-events: none;
            opacity: 0.6;
        }

        /* Optional: allow buttons inside block-content to be clickable if you want */
        .block-wrapper.blocked .block-content {
            pointer-events: auto;
        }
    </style>

    <div class="container mt-5">
        <h2 class="mb-4">Bienvenue, {{ app.user.firstname }}</h2>

        {% for label, messages in app.flashes %}
            {% for message in messages %}
                <div class="alert alert-{{ label }}">
                    {{ message }}
                </div>
```

```
      {% endfor %}
    {% endfor %}

    <div class="row row-cols-1 row-cols-md-2 g-4">

      <div class="col">
        <div class="card h-100 shadow-sm">
          <div class="card-body text-center">
            <h5 class="card-title">📚 Voir les catalogues</h5>
            <p class="card-text">Explorez les formations disponibles sur la plateforme.</p>
            <a href="{{ path('dashboard_path') }}" class="btn btn-primary">Consulter</a>
          </div>
        </div>
      </div>

      <div class="col">
        <div class="card h-100 shadow-sm">
          <div class="card-body text-center">
            <h5 class="card-title">👤 Mon compte</h5>
            <p class="card-text">Voir ou modifier vos informations personnelles.</p>
            <a href="{{ path('user_show_profile') }}" class="btn btn-primary">Profil</a>
          </div>
        </div>
      </div>

      {% if 'ROLE_VERIFIED' in app.user.roles %}
        <div class="col">
          <div class="card h-100 shadow-sm">
            <div class="card-body text-center">
              <h5 class="card-title">📘 Mes cours</h5>
              <p class="card-text">Accéder à tous les cours disponibles pour vous.</p>
              <a href="{{ path('dashboard_path') }}" class="btn btn-success">Voir mes cours</a>
            </div>
          </div>
        </div>
      {% endif %}


      {% if 'ROLE_FORMATEUR' in app.user.roles %}
        <div class="col">
          <div class="card h-100 shadow-sm">
            <div class="card-body text-center">
              <h5 class="card-title">🎓 Gérer mes formations</h5>
              <p class="card-text">Ajoutez ou modifiez vos propres formations.</p>
              <a href="{{ path('dashboard_path') }}" class="btn btn-warning">Mes formations</a>
            </div>
          </div>
        </div>
      {% else %}
        <div class="col">
          <div class="block-wrapper h-100">
```

```
                        <div class="card h-100 shadow-sm block-content ">
                            <div class="card-body text-center">
                                <h5 class="card-title">📝 Devenir formateur</h5>
                                <p class="card-text">Soumettez une demande pour rejoindre l'équipe des formateurs.</p>
                                <a href="{{ path('user_demand') }}" class="btn btn-outline-secondary ">Faire une demande</a>
                            </div>
                        </div>
                    </div>
                </div>
            {% endif %}


        </div>
    </div>
{% endblock %}
```

Here's a cleaner, modernized, and polished version of your Twig template for your e-commerce dashboard page, with improved structure, Bootstrap 5 classes, and minor UI/UX tweaks:

```twig
{% extends 'base.html.twig' %}

{% block title %}Mon Dashboard{% endblock %}

{% block nav %}
    <div class="collapse navbar-collapse justify-content-end" id="navLinks">
        <ul class="navbar-nav align-items-center">
            <li class="nav-item"><a class="nav-link" href="{{ path('dashboard_path') }}">Dashboard<
            <li class="nav-item"><a class="nav-link" href="{{ path('user_demand') }}">Devenir forma
            <li class="nav-item"><a class="nav-link" href="{{ path('user_show_profile') }}">Profil<
            <li class="nav-item">
                <a href="{{ path('app_panier') }}" class="btn btn-outline-secondary position-relati
                    🛒
                    {# Uncomment below to show product count badge in future #}
                    {# <span class="position-absolute top-0 start-100 translate-middle badge rounde
                </a>
            </li>
            <li class="nav-item"><a class="nav-link" href="{{ path('app_logout') }}">Déconnexion</a
        </ul>
    </div>
{% endblock %}

{% block body %}
<style>
    /* Diagonal "UNVERIFIED" ribbon */
    .block-wrapper {
        position: relative;
        width: 100%;
        height: 100%;
        display: inline-block;
    }
    .block-wrapper.blocked::before {
        content: "UNVERIFIED";
        position: absolute;
        top: 20px;
        right: -50px;
        width: 220px;
        padding: 5px 0;
        background: rgba(220, 53, 69, 0.85); /* Bootstrap danger red with opacity */
        color: #fff;
        font-weight: 700;
        font-size: 0.85rem;
        text-align: center;
        transform: rotate(45deg);
        pointer-events: none;
        z-index: 10;
        box-shadow: 0 0 10px rgba(220, 53, 69, 0.7);
        user-select: none;
    }
    .block-wrapper.blocked {
        pointer-events: none;
        opacity: 0.65;
    }
    /* Allow interactions inside content even if blocked */
    .block-wrapper.blocked .block-content {
        pointer-events: auto;
    }
</style>

<div class="container mt-5">
    <h2 class="mb-4 fw-bold">Bienvenue, {{ app.user.firstname }} !</h2>

    {% for label, messages in app.flashes %}
        {% for message in messages %}
            <div class="alert alert-{{ label }} alert-dismissible fade show" role="alert">
                {{ message }}
                <button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close">
            </div>
        {% endfor %}
    {% endfor %}

    <div class="row row-cols-1 row-cols-md-2 row-cols-lg-3 g-4">

        <div class="col">
            <div class="card h-100 shadow-sm text-center">
                <div class="card-body d-flex flex-column justify-content-center">
                    <h5 class="card-title">📚 Voir les catalogues</h5>
                    <p class="card-text">Explorez les formations disponibles sur la plateforme.</p>
                    <a href="{{ path('dashboard_path') }}" class="btn btn-primary mt-auto">Consulte
                </div>
            </div>
        </div>

        <div class="col">
            <div class="card h-100 shadow-sm text-center">
                <div class="card-body d-flex flex-column justify-content-center">
                    <h5 class="card-title">👤 Mon compte</h5>
                    <p class="card-text">Voir ou modifier vos informations personnelles.</p>
                    <a href="{{ path('user_show_profile') }}" class="btn btn-primary mt-auto">Profi
                </div>
            </div>
        </div>

        {% if 'ROLE_VERIFIED' in app.user.roles %}
            <div class="col">
                <div class="card h-100 shadow-sm text-center border-success">
                    <div class="card-body d-flex flex-column justify-content-center">
                        <h5 class="card-title">📘 Mes cours</h5>
                        <p class="card-text">Accédez à tous les cours disponibles pour vous.</p>
                        <a href="{{ path('dashboard_path') }}" class="btn btn-success mt-auto">Voir
                    </div>
                </div>
            </div>
        {% endif %}

        {% if 'ROLE_FORMATEUR' in app.user.roles %}
            <div class="col">
                <div class="card h-100 shadow-sm text-center border-warning">
                    <div class="card-body d-flex flex-column justify-content-center">
                        <h5 class="card-title">🎓 Gérer mes formations</h5>
                        <p class="card-text">Ajoutez ou modifiez vos propres formations.</p>
                        <a href="{{ path('dashboard_path') }}" class="btn btn-warning mt-auto">Mes
                    </div>
                </div>
            </div>
        {% else %}
            <div class="col">
                <div class="block-wrapper h-100">
                    <div class="card h-100 shadow-sm block-content text-center">
                        <div class="card-body d-flex flex-column justify-content-center">
                            <h5 class="card-title">📝 Devenir formateur</h5>
                            <p class="card-text">Soumettez une demande pour rejoindre l'équipe des
                            <a href="{{ path('user_demand') }}" class="btn btn-outline-secondary mt
                        </div>
                    </div>
                </div>
            </div>
        {% endif %}

    </div>
</div>
{% endblock %}
```

What I improved:

- Added consistent Bootstrap 5 spacing, flexbox for vertical alignment inside cards.
- Improved alert styles with dismissible alerts.
- Made nav items better aligned and fixed the shopping cart button.
- Polished the diagonal ribbon styling with a Bootstrap danger color and better positioning.
- Used semantic French text fixes (e.g., "Devenir formateur" instead of "Devient a formateur").
- Added `mt-auto` on buttons so they stick to bottom in cards with flexible height.
- Responsive grid with 1 column on mobile, 2 on tablet, 3 on desktop.
- Clear comments and clean indentation.

If you want, I can help add more features or responsive improvements!

```
{% extends 'dashboard/UserDashboard.html.twig' %}

{% block body %}
    <h1>Mon Profil</h1>

    <ul>
        <li>Email : {{ user.email }}</li>
        <li>Prénom : {{ user.firstName }}</li>
        <li>Nom : {{ user.lastName }}</li>
    </ul>

    <a href="{{ path('user_profile') }}" class="btn btn-primary">Modifier mon profil</a>

    <form method="post" action="{{ path('app_delete_user') }}" onsubmit="return confirm('Es-tu sûr ?');">
        <input type="hidden" name="_method" value="DELETE">
        <button class="btn btn-danger mt-2">Supprimer mon compte</button>
    </form>
{% endblock %}
```

Here's a modern, clean, and user-friendly version of your profile page Twig template with Bootstrap styling and improved semantics:

```twig
{% extends 'dashboard/UserDashboard.html.twig' %}

{% block body %}
<div class="container py-4">
    <h1 class="mb-4">Mon Profil</h1>

    <div class="card shadow-sm mb-4">
        <div class="card-body">
            <ul class="list-group list-group-flush">
                <li class="list-group-item"><strong>Email :</strong> {{ user.email }}</li>
                <li class="list-group-item"><strong>Prénom :</strong> {{ user.firstName }}</li>
                <li class="list-group-item"><strong>Nom :</strong> {{ user.lastName }}</li>
            </ul>
        </div>
    </div>

    <div class="d-flex gap-2">
        <a href="{{ path('user_profile') }}" class="btn btn-primary">Modifier mon profil</a>

        <form method="post" action="{{ path('app_delete_user') }}" onsubmit="return confirm('Es-tu
            <input type="hidden" name="_method" value="DELETE">
            <button type="submit" class="btn btn-danger">Supprimer mon compte</button>
        </form>
    </div>
</div>
{% endblock %}
```

Improvements:

- Wrapped content in a Bootstrap container with padding for spacing.
- Used a Bootstrap card with `list-group` for better profile info display.
- Added `strong` tags for label emphasis.
- Grouped the buttons side by side with `d-flex gap-2` for nice spacing.
- Improved the confirmation message for clarity.
- Added `type="submit"` on the delete button for semantic correctness.

If you want, I can help you add flash messages, or make this page responsive with better mobile UX!

```
{% extends 'dashboard/UserDashboard.html.twig' %}

{% block body %}
    <h1>Modifier mon profil</h1>

    {{ form_start(form) }}
    {{ form_row(form.email) }}
    {{ form_row(form.firstName) }}
    {{ form_row(form.lastName) }}
    <button class="btn btn-success">Enregistrer</button>
    {{ form_end(form) }}

    <a href="{{ path('user_show_profile') }}" class="btn btn-secondary mt-3">Retour au profil</a>
{% endblock %}
```

Here's a clean, modern, and nicely styled version of your profile edit Twig template with Bootstrap 5 styling and improved layout:

```twig
{% extends 'dashboard/UserDashboard.html.twig' %}

{% block body %}
<div class="container py-4" style="max-width: 600px;">
    <h1 class="mb-4">Modifier mon profil</h1>

    {{ form_start(form) }}

    <div class="mb-3">
        {{ form_label(form.email) }}
        {{ form_widget(form.email, {'attr': {'class': 'form-control'}}) }}
        {{ form_errors(form.email) }}
    </div>

    <div class="mb-3">
        {{ form_label(form.firstName) }}
        {{ form_widget(form.firstName, {'attr': {'class': 'form-control'}}) }}
        {{ form_errors(form.firstName) }}
    </div>

    <div class="mb-3">
        {{ form_label(form.lastName) }}
        {{ form_widget(form.lastName, {'attr': {'class': 'form-control'}}) }}
        {{ form_errors(form.lastName) }}
    </div>

    <button type="submit" class="btn btn-success">Enregistrer</button>

    {{ form_end(form) }}

    <a href="{{ path('user_show_profile') }}" class="btn btn-secondary mt-3">Retour au profil</a>
</div>
{% endblock %}
```

What I improved:

- Wrapped the form in a container with max width for better readability.
- Explicitly added Bootstrap `form-control` class to form widgets.
- Displayed errors under each input.
- Added margin-bottom spacing between fields.
- Added a submit button with `type="submit"` for semantic clarity.
- Kept the "Retour au profil" button below the form with spacing.

Would you like me to help add client-side validation or flash messages as well?

ADD A Modifier mon password which redirect to change my password

Got it! Here's your updated Twig template with a new button **"Modifier mon mot de passe"** that redirects to the password change page (assuming your route is named `user_change_password`):

```twig
{% extends 'dashboard/UserDashboard.html.twig' %}

{% block body %}
<div class="container py-4" style="max-width: 600px;">
    <h1 class="mb-4">Modifier mon profil</h1>

    {{ form_start(form) }}

    <div class="mb-3">
        {{ form_label(form.email) }}
        {{ form_widget(form.email, {'attr': {'class': 'form-control'}}) }}
        {{ form_errors(form.email) }}
    </div>

    <div class="mb-3">
        {{ form_label(form.firstName) }}
        {{ form_widget(form.firstName, {'attr': {'class': 'form-control'}}) }}
        {{ form_errors(form.firstName) }}
    </div>

    <div class="mb-3">
        {{ form_label(form.lastName) }}
        {{ form_widget(form.lastName, {'attr': {'class': 'form-control'}}) }}
        {{ form_errors(form.lastName) }}
    </div>

    <button type="submit" class="btn btn-success">Enregistrer</button>

    {{ form_end(form) }}

    <div class="mt-4">
        <a href="{{ path('user_change_password') }}" class="btn btn-warning me-2">Modifier mon mot
        <a href="{{ path('user_show_profile') }}" class="btn btn-secondary">Retour au profil</a>
    </div>
</div>
{% endblock %}
```

Notes:

- The password change button uses a **warning** Bootstrap color to distinguish it.
- Buttons are side by side with a small margin between (`me-2`).
- Adjust the route name `user_change_password` if your actual route is different.

If you want, I can help you create the password change page too!

Fatal error: Allowed memory size of 134217728 bytes exhausted (tried to allocate 528384 bytes) in C:\Users\mohan\e-commerce-small-udemy-e-book-store-\second try\e-learning\vendor\twig\twig\src\Extension\ProfilerExtension.php on line 32

Fatal error: Allowed memory size of 134217728 bytes exhausted (tried to allocate 20480 bytes) in

C:\Users\mohan\e-commerce-small-udemy-e-book-store-\second try\e-learning\vendor\symfony\error-handler\ErrorRenderer\HtmlErrorRenderer.php on line 122

This error means your PHP script ran out of allowed memory (128MB in your case) while rendering Twig templates — usually caused by an infinite loop or too heavy processing.

How to fix:

1. Increase PHP memory limit temporarily

Edit your `php.ini` (find its location with `phpinfo()` or your PHP config), and increase the memory limit:

```ini
memory_limit = 256M
```

Then restart your server (Apache, Nginx, PHP-FPM, etc).

Or, for quick testing, add at the top of your PHP entry script (e.g. `public/index.php` or `app.php`):

```php
ini_set('memory_limit', '256M');
```

2. Check for infinite loops or recursive includes

Look in your Twig templates and PHP controllers for:

- Recursive Twig includes or extends causing infinite rendering.
- Infinite loops in data preparation that get passed to Twig.
- Huge data structures passed to Twig by mistake.

3. Disable the Twig Profiler (if enabled)

Your error mentions `ProfilerExtension.php` — Twig's profiler can consume a lot of memory.

If you enabled Twig Profiler for debugging, disable it in your `services.yaml` or wherever you configured it.

For example, in `config/services.yaml`:

```yaml
# Comment out or remove the profiler extension
twig:
    # ...
    profiler: false
```

Or in your `twig.yaml`:

```yaml
twig:
    profiler: false
```

4. Clear cache

Symfony cache can cause weird issues sometimes:

```bash
php bin/console cache:clear
php bin/console cache:warmup
```

5. Check for large data dumps

If you dump very large variables (`dump()`, `var_dump()`) inside Twig or PHP, it can exhaust memory.

If you want, share your recent changes or relevant template code, I can help spot what might cause the problem. But start with increasing memory and disabling profiler first.

```
{% extends 'base.html.twig' %}

{% block title %}Password Reset Email Sent{% endblock %}

{% block body %}
    <p>
        If an account matching your email exists, then an email was just sent that contains a link that you can use to reset your password.
        This link will expire in {{ resetToken.expirationMessageKey|trans(resetToken.expirationMessageData, 'ResetPasswordBundle') }}.
    </p>
    <p>If you don't receive an email please check your spam folder or <a href="{{ path('app_forgot_password_request') }}">try again</a>.</p>
{% endblock %}
```

Your Twig template looks fine and simple — it just shows a message after a password reset email is sent.

If you're seeing the **memory exhausted** error related to `ProfilerExtension` or your password reset flow, here are a few focused checks and tips related to this template and the reset password process:

What might cause memory issues here?

1. **Large or problematic `resetToken` object**
   If the `resetToken` passed to the template contains very large data or has some cyclic references, Twig might consume a lot of memory trying to process it.

    - Try to debug what exactly `resetToken` contains before rendering.
    - Make sure you only pass minimal data needed for the template.
2. **Translation issues**
   The `trans` filter uses the `ResetPasswordBundle` translation domain.
   If translations are missing or cause recursive lookups, it might cause issues.
3. **Profiler enabled**
   If Twig Profiler is enabled, it can easily cause memory exhaustion on complex pages or with certain data.

Recommendations:

- **Debug the `resetToken` content** before passing it to Twig: pass only the expiration message key and data, not the entire token object.

For example, instead of:

```
php

return $this->render('reset_password/check_email.html.twig', [
    'resetToken' => $resetToken,
]);
```

Try:

```
php

return $this->render('reset_password/check_email.html.twig', [
    'expirationMessageKey' => $resetToken->getExpirationMessageKey(),
    'expirationMessageData' => $resetToken->getExpirationMessageData(),
]);
```

And in Twig:

```
twig

{{ expirationMessageKey|trans(expirationMessageData, 'ResetPasswordBundle') }}
```

- **Disable Twig Profiler** if active (see previous answer).
- **Increase memory limit temporarily** if you suspect your app just needs more memory.
- **Clear cache** with `php bin/console cache:clear`.

Bonus: Cleaned up Twig code with the above suggestion

```
twig

{% extends 'base.html.twig' %}

{% block title %}Password Reset Email Sent{% endblock %}

{% block body %}
    <p>
```

```
        If an account matching your email exists, then an email was just sent that contains a link
        This link will expire in {{ expirationMessageKey|trans(expirationMessageData, 'ResetPasswor
    </p>
    <p>If you don't receive an email please check your spam folder or <a href="{{ path('app_forgot_
{% endblock %}
```

If you want

beautify it

Sure! Here's a clean, modern, and nicely formatted version of your password reset confirmation Twig template with better HTML structure and some Bootstrap styling for a polished look:

```
twig

{% extends 'base.html.twig' %}

{% block title %}Password Reset Email Sent{% endblock %}

{% block body %}
<div class="container mt-5">
    <div class="row justify-content-center">
        <div class="col-md-8 col-lg-6">
            <div class="card shadow-sm border-0">
                <div class="card-body text-center p-4">
                    <h2 class="mb-4">Password Reset Email Sent</h2>
                    <p class="lead">
                        If an account matching your email exists, an email has been sent containing
                    </p>
                    <p class="mb-4 text-muted">
                        This link will expire in <strong>{{ expirationMessageKey|trans(expirationMe
                    </p>
                    <p>
                        If you don't receive an email, please check your spam folder or
                        <a href="{{ path('app_forgot_password_request') }}" class="btn btn-link p-0
                            try again
                        </a>.
                    </p>
                </div>
            </div>
        </div>
    </div>
</div>
{% endblock %}
```

What I improved:

- Wrapped content in a Bootstrap `` `.container` `` with responsive `` `.row` `` and `` `.col` `` for nice centered layout.
- Used a card component with subtle shadow for a modern box effect.
- Added headings and paragraph spacing for readability.
- Added Bootstrap classes like `` `lead` ``, `` `text-muted` `` and buttons for better UX.
- Highlighted expiration time with `` `<strong>` `` for emphasis.
- Kept it semantic and clean.

If you want me to customize colors or add icons, just say so!

this is the one sent by email should we modify it
<h1>Hi!</h1>

<p>To reset your password, please visit the following link</p>

<a href="{{ url('app_reset_password', {token: resetToken.token}) }}">{{ url('app_reset_password', {token: resetToken.token}) }}</a>

<p>This link will expire in {{ resetToken.expirationMessageKey|trans(resetToken.expirationMessageData, 'ResetPasswordBundle') }}.</p>

<p>Cheers!</p>

Yes, since this email is sent to users, it’s a great idea to make it look more professional and user-friendly while keeping it simple and compatible with email clients. Here’s a polished, clean, and mobile-friendly HTML email template version for your password reset email:

```twig
<h1 style="font-family: Arial, sans-serif; color: #333;">Hello!</h1>

<p style="font-family: Arial, sans-serif; font-size: 16px; color: #555;">
    To reset your password, please click the link below:
</p>

<p>
    <a href="{{ url('app_reset_password', {token: resetToken.token}) }}"
       style="display: inline-block; padding: 10px 20px; font-family: Arial, sans-serif; font-size:
        Reset My Password
    </a>
</p>

<p style="font-family: Arial, sans-serif; font-size: 14px; color: #999;">
    This link will expire in <strong>{{ resetToken.expirationMessageKey|trans(resetToken.expirationM
</p>

<p style="font-family: Arial, sans-serif; font-size: 16px; color: #555;">
    Cheers,<br>
    The Team
</p>
```

Why this is better for emails:

- **Inline styles** for best compatibility across all email clients.
- A **clear call-to-action button** styled as a blue clickable link.
- Simple, clean typography.
- Avoids complex CSS or external dependencies which can break in many email clients.
- Friendly, professional tone.

If you want, I can help you generate a full HTML email with a header/footer and responsive design too!

```
{% extends 'base.html.twig' %}

{% block title %}Reset your password{% endblock %}

{% block body %}
  {% for flash_error in app.flashes('reset_password_error') %}
    <div class="alert alert-danger" role="alert">{{ flash_error }}</div>
  {% endfor %}
  <h1>Reset your password</h1>

  {{ form_start(requestForm) }}
    {{ form_row(requestForm.email) }}
    <div>
      <small>
        Enter your email address, and we will send you a
        link to reset your password.
      </small>
    </div>

    <button class="btn btn-primary">Send password reset email</button>
  {{ form_end(requestForm) }}
{% endblock %}
```

Here's a modern, clean, and nicely formatted version of your Twig template for the password reset request page, using Bootstrap classes and improved spacing for better user experience:

```twig
{% extends 'base.html.twig' %}

{% block title %}Reset your password{% endblock %}

{% block body %}
    <div class="container mt-5" style="max-width: 480px;">
        {% for flash_error in app.flashes('reset_password_error') %}
            <div class="alert alert-danger" role="alert">
                {{ flash_error }}
            </div>
        {% endfor %}

        <h1 class="mb-4">Reset your password</h1>

        {{ form_start(requestForm) }}
            <div class="mb-3">
                {{ form_label(requestForm.email) }}
                {{ form_widget(requestForm.email, {'attr': {'class': 'form-control'}}) }}
                {{ form_errors(requestForm.email) }}
            </div>

            <div class="mb-3">
                <small class="text-muted">
                    Enter your email address, and we will send you a link to reset your password.
                </small>
            </div>
            <button type="submit" class="btn btn-primary w-100">Send password reset email</button>
        {{ form_end(requestForm) }}
    </div>
{% endblock %}
```

Improvements:

- Wrapped content inside a centered container with max width for better readability.
- Added Bootstrap spacing (`mb-3`, `mb-4`, `mt-5`) for clean vertical rhythm.
- Added Bootstrap form classes for consistent styling.
- Button styled full width for easier clicking on mobile devices.
- Flash error messages are inside alert boxes for clear user feedback.

If you want, I can help you add client-side validation or other UX improvements!

```
{% extends 'base.html.twig' %}

{% block title %}Reset your password{% endblock %}

{% block body %}
    <h1>Reset your password</h1>

    {{ form_start(resetForm) }}
        {{ form_row(resetForm.plainPassword) }}
        <button class="btn btn-primary">Reset password</button>
    {{ form_end(resetForm) }}
{% endblock %}
```

Here’s a clean, modern, and nicely spaced version of your password reset form template with Bootstrap styling:

```
twig

{% extends 'base.html.twig' %}

{% block title %}Reset your password{% endblock %}

{% block body %}
    <div class="container mt-5" style="max-width: 480px;">
        <h1 class="mb-4">Reset your password</h1>

        {{ form_start(resetForm) }}
            <div class="mb-3">
                {{ form_label(resetForm.plainPassword) }}
                {{ form_widget(resetForm.plainPassword, {'attr': {'class': 'form-control'}}) }}
                {{ form_errors(resetForm.plainPassword) }}
            </div>

            <button type="submit" class="btn btn-primary w-100">Reset password</button>
        {{ form_end(resetForm) }}
    </div>
{% endblock %}
```

Notes:

- Added container with max-width and margin-top for centering and spacing.
- Bootstrap classes for input and button for consistent UI.
- Display errors if any under the input.

Want me to help with a success message or redirect after reset?

```
{% extends 'dashboard/UserDashboard.html.twig' %}

{% block title %}Demande pour devenir formateur{% endblock %}

{% block body %}
    <h1>Demande pour devenir formateur</h1>

    {{ form_start(form) }}
    {{ form_row(form.motivation) }}
    {{ form_row(form.experience) }}
    {{ form_end(form) }}
    <a href="{{ path("dashboard_path") }}"><button class="btn btn-primary">Go back</button></a>

{% endblock %}
```

Here's a polished and modern version of your "Demande pour devenir formateur" template with better structure and Bootstrap styling:

```twig
{% extends 'dashboard/UserDashboard.html.twig' %}

{% block title %}Demande pour devenir formateur{% endblock %}

{% block body %}
    <div class="container mt-4" style="max-width: 600px;">
        <h1 class="mb-4">Demande pour devenir formateur</h1>

        {{ form_start(form) }}
            <div class="mb-3">
                {{ form_label(form.motivation) }}
                {{ form_widget(form.motivation, {'attr': {'class': 'form-control', 'rows': 4}}) }}
                {{ form_errors(form.motivation) }}
            </div>

            <div class="mb-3">
                {{ form_label(form.experience) }}
                {{ form_widget(form.experience, {'attr': {'class': 'form-control', 'rows': 4}}) }}
                {{ form_errors(form.experience) }}
            </div>

            <button type="submit" class="btn btn-success">Envoyer la demande</button>
            <a href="{{ path('dashboard_path') }}" class="btn btn-secondary ms-2">Retour au tableau
        {{ form_end(form) }}
    </div>
{% endblock %}
```

Changes:

- Wrapped the form in a container with margin for spacing.
- Added Bootstrap classes for form controls.
- Changed buttons to be side by side with spacing (`ms-2` for margin start).
- Used `rows` attribute for textareas (assuming motivation and experience are textarea fields).

Would you like me to help you handle flash messages or confirmation after submitting?

```
<h1>Hi! Please confirm your email!</h1>

<p>
    Please confirm your email address by clicking the following link: <br><br>
    <a href="{{ signedUrl|raw }}">Confirm my Email</a>.
    This link will expire in {{ expiresAtMessageKey|trans(expiresAtMessageData, 'VerifyEmailBundle') }}.
</p>

<p>
    Cheers!
</p>
```

Here's a **beautified and professional version** of your email confirmation template using HTML with a more polished structure and styling that looks better in most email clients:

```html
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Email Confirmation</title>
</head>
<body style="font-family: Arial, sans-serif; line-height: 1.6; background-color: #f9f9f9; padding:
    <div style="max-width: 600px; margin: auto; background: white; padding: 30px; border-radius: 8p
        <h1 style="color: #2c3e50;">Hi 👋, please confirm your email!</h1>

        <p>
            Thank you for signing up! To complete your registration, please confirm your email addr
        </p>

        <p style="text-align: center; margin: 30px 0;">
            <a href="{{ signedUrl|raw }}"
               style="background-color: #28a745; color: white; padding: 12px 24px; text-decoration:
                ✅ Confirm my Email
            </a>
        </p>

        <p>
            This link will expire in <strong>{{ expiresAtMessageKey|trans(expiresAtMessageData, 'Ve
        </p>

        <p style="margin-top: 40px;">
            Cheers!<br>
            <strong>The Team</strong>
        </p>
    </div>
```

```
</body>
</html>
```

Why this is better:

- Clean layout using modern fonts and spacing.
- Includes a **call-to-action button** instead of just a plain link.
- Mobile-friendly design with inline styles (email clients prefer inline styles).
- Gracefully handles the expiration message.

Let me know if you want to add your site logo or change the language to French.

```
{% extends 'base.html.twig' %}

{% block title %}Access Denied{% endblock %}

{% block body %}
    <div class="container mt-5 text-center">
        <h1 class="text-danger">Access Denied</h1>
        <p>You do not have permission to access this page.</p>
        <a href="{{ path('app_login') }}" class="btn btn-primary">Login</a>
    </div>
{% endblock %}
```

Here’s a **beautified and more user-friendly version** of your `Access Denied` page with improved styling and layout using Bootstrap:

```twig
{% extends 'base.html.twig' %}

{% block title %}Access Denied{% endblock %}

{% block body %}
    <div class="container mt-5 d-flex flex-column align-items-center justify-content-center" style=
        <div class="card p-4 shadow" style="max-width: 500px; width: 100%;">
            <h1 class="text-danger text-center mb-3">🚫 Access Denied</h1>
            <p class="text-center">You do not have permission to access this page.</p>
            <div class="text-center mt-4">
                <a href="{{ path('app_login') }}" class="btn btn-primary">🔐 Login</a>
            </div>
        </div>
    </div>
{% endblock %}
```

Highlights of the changes:

- Added a Bootstrap card for better visual structure.
- Used `d-flex` and `justify-content-center` to vertically center the content.
- Emoji for a touch of friendliness and clarity.

- Clearer padding, spacing, and container width control for responsiveness.

Let me know if you want a "Go Back" button or redirect to a homepage/dashboard instead.

feature/design

Access Denied Page

✅ I've enhanced the **Access Denied** page with a more polished and centered card layout using Bootstrap utilities, an emoji for emphasis, and better spacing. Let me know if you'd like to add an illustration or redirect logic!

```
{% extends 'base.html.twig' %}
{% block nav %}
{% endblock %}
{% block title %}Please Verify Your Email{% endblock %}

{% block body %}
    <div class="container mt-5 d-flex justify-content-center align-items-center" style="min-height: 70vh;">
        <div class="card shadow p-4" style="max-width: 400px; width: 100%;">
            <h3 class="card-title mb-3 text-center">Please Verify Your Email</h3>
            <p class="card-text text-center mb-4">
                You must verify your email address to access all features.<br>
                Check your inbox for the verification link.
            </p>
            {% for label, messages in app.flashes %}
                {% for message in messages %}
                    <div class="alert alert-{{ label }}" role="alert">
                        {{ message }}
                    </div>
                {% endfor %}
            {% endfor %}

            <form method="post" action="{{ path('send_verification_email') }}">
                <button type="submit" class="btn btn-primary w-100">
                    Resend Verification Email
                </button>
            </form>
        </div>
    </div>
{% endblock %}
```

Your "Please Verify Your Email" template is already quite solid, but here’s a refined version with minor design and structure enhancements for a better user experience:

```twig
twig


{% extends 'base.html.twig' %}

{% block nav %}{% endblock %}
{% block title %}Please Verify Your Email{% endblock %}

{% block body %}
<div class="container mt-5 d-flex justify-content-center align-items-center" style="min-height: 70v
    <div class="card shadow-sm p-4 rounded" style="max-width: 420px; width: 100%;">
        <h2 class="text-center text-primary mb-3">📧 Verify Your Email</h2>
        <p class="text-center text-muted mb-4">
            To continue, please verify your email address.<br>
            A verification link has been sent to your inbox.
        </p>

        {% for label, messages in app.flashes %}
            {% for message in messages %}
                <div class="alert alert-{{ label }} alert-dismissible fade show" role="alert">
                    {{ message }}
                    <button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Clo
                </div>
            {% endfor %}
        {% endfor %}

        <form method="post" action="{{ path('send_verification_email') }}">
            <button type="submit" class="btn btn-outline-primary w-100">
                Resend Verification Email
            </button>
        </form>
    </div>
</div>
{% endblock %}
```

Changes Made:

- ✅ Added an email emoji for friendliness.
- ✅ Used Bootstrap's `` `alert-dismissible` `` and `` `btn-close` `` for flash messages.
- ✅ Adjusted headings and button styles for visual appeal and clarity.

Let me know if you'd like to include:

- An automatic resend timer.
- A link to support.
- A dark mode version.

generate fake data for formations with multiple formateur_id and generate multiple formateur to manualy insert into mysql
and link each formateur of a user id from 1 to 6 and 8 to 11
<?php

namespace App\Entity;

use App\Repository\FormationRepository;
use Doctrine\Common\Collections\ArrayCollection;
use Doctrine\Common\Collections\Collection;

```
use Doctrine\ORM\Mapping as ORM;

#[ORM\Entity(repositoryClass: FormationRepository::class)]
class Formation
{
    #[ORM\Id]
    #[ORM\GeneratedValue]
    #[ORM\Column]
    private ?int $id = null;

    #[ORM\Column(length: 255)]
    private ?string $title = null;

    #[ORM\Column(length: 255, nullable: true)]
    private ?string $description = null;

    #[ORM\Column]
    private ?int $price = null;

    #[ORM\ManyToOne(inversedBy: 'formation_created')]
    private ?Formateur $formateur = null;

    #[ORM\Column]
    private ?bool $published = null;
    /**
     * @ORM\Column(type="string", length=255, nullable=true)
     */
    private $pdfFilename;

    /**
     * @var Collection<int, Formateur>
     */
   /* #[ORM\ManyToMany(targetEntity: Formateur::class, mappedBy: 'formation_created')]
    private Collection $formateurs;*/

    public function __construct()
    {
        //$this->formateurs = new ArrayCollection();
    }

    public function getPdfFilename(): ?string
    {
        return $this->pdfFilename;
    }

    public function setPdfFilename(?string $pdfFilename): self
    {
        $this->pdfFilename = $pdfFilename;

        return $this;
    }
    public function getId(): ?int
    {
        return $this->id;
    }

    public function getTitle(): ?string
    {
        return $this->title;
    }

    public function setTitle(string $title): static
    {
        $this->title = $title;

        return $this;
    }

    public function getDescription(): ?string
    {
        return $this->description;
    }

    public function setDescription(?string $description): static
    {
        $this->description = $description;

        return $this;
    }

    public function getPrice(): ?int
    {
        return $this->price;
    }

    public function setPrice(int $price): static
    {
        $this->price = $price;

        return $this;
    }

    public function getFormateur(): ?Formateur
    {
        return $this->formateur;
    }

    public function setFormateur(?Formateur $formateur): static
    {
        $this->formateur = $formateur;

        return $this;
    }

    public function isPublished(): ?bool
    {
        return $this->published;
    }

    public function setPublished(bool $published): static
    {
        $this->published = $published;

        return $this;
    }




}
```

Based on your entity structure, here's how you can manually generate fake SQL `INSERT` statements for:

- **Formateurs** linked to user IDs (1 to 6 and 8 to 11),
- **Formations** with multiple `formateur_id` values.

✅ Step 1: Generate `formateur` Table Inserts

Assume your `formateur` table has at least `id`, `user_id`, `motivation`, `experience`.

```sql
-- FORMATEURS (linked to users 1–6 and 8–11)
INSERT INTO formateur (id, user_id, motivation, experience) VALUES
(1, 1, 'I love teaching coding', '5 years as a software engineer'),
(2, 2, 'Passionate about design', '3 years in UI/UX'),
(3, 3, 'Data science is my thing', '4 years in data analysis'),
(4, 4, 'I enjoy sharing my knowledge', '6 years in teaching'),
(5, 5, 'Helping others succeed motivates me', '2 years in project management'),
(6, 6, 'Experienced backend developer', '7 years in PHP and Symfony'),
(7, 8, 'Cybersecurity expert', '5 years in IT security'),
(8, 9, 'Mobile dev and mentor', '3 years in Flutter'),
(9, 10, 'AI & ML instructor', '4 years research experience'),
(10, 11, 'Cloud and DevOps coach', '5 years in AWS & CI/CD');
```

✅ Step 2: Generate `formation` Table Inserts

Assume your `formation` table has:
`id`, `title`, `description`, `price`, `formateur_id`, `published`, `pdf_filename`.

```sql
-- FORMATIONS (each linked to a formateur from above)
INSERT INTO formation (id, title, description, price, formateur_id, published, pdf_filename) VALUES
(1, 'Introduction to Python', 'Learn Python from scratch', 120, 1, true, 'python_intro.pdf'),
(2, 'UI/UX Design Basics', 'Design user-friendly interfaces', 90, 2, true, 'design_basics.pdf'),
(3, 'Data Science Bootcamp', 'From data cleaning to ML', 200, 3, true, 'ds_bootcamp.pdf'),
(4, 'Symfony for Beginners', 'Start building PHP apps', 150, 6, false, 'symfony_beginners.pdf'),
(5, 'Advanced PHP Techniques', 'Level up your PHP skills', 180, 6, true, 'php_advanced.pdf'),
(6, 'Cybersecurity 101', 'Protect systems and data', 170, 7, true, 'cybersecurity.pdf'),
(7, 'Mobile App Development', 'Build apps with Flutter', 160, 8, true, 'flutter_dev.pdf'),
(8, 'AI Essentials', 'Understand Artificial Intelligence', 220, 9, false, 'ai_essentials.pdf'),
(9, 'DevOps Best Practices', 'Automate and deploy efficiently', 200, 10, true, 'devops.pdf'),
(10, 'Teaching Online Effectively', 'How to create great online lessons', 100, 4, true, 'teach_onli
```

💡 Optional: Simulate Many-to-Many (if implemented later)

If you later reintroduce a `formation_formateur` table for a Many-to-Many relationship, you could add something like:

```
sql

-- FORMATION_FORMATEUR (junction table for many-to-many)
INSERT INTO formation_formateur (formation_id, formateur_id) VALUES
(1, 1),
(1, 4),
(2, 2),
(3, 3),
(3, 9),
(4, 6),
(5, 6),
(5, 10),
(6, 7),
(6, 1),
(7, 8),
(8, 9),
(9, 10),
(10, 4),
(10, 2);
```

📌 Notes:

- Adjust the table and column names if different in your schema.
- Use `FALSE/TRUE` or `0/1` depending on DB setup for booleans.
- Make sure you respect foreign key constraints: `formateur_id` must exist in `formateur`.

Would you like a `Faker` PHP script or Symfony `Fixtures` to generate this dynamically?

```
<?php

namespace App\Entity;

use App\Repository\FormateurRepository;
use Doctrine\Common\Collections\ArrayCollection;
use Doctrine\Common\Collections\Collection;
use Doctrine\DBAL\Types\Types;
use Doctrine\ORM\Mapping as ORM;

#[ORM\Entity(repositoryClass: FormateurRepository::class)]
```

```
class Formateur
{
    #[ORM\Id]
    #[ORM\OneToOne(inversedBy: 'formateur', cascade: ['persist', 'remove'])]
    #[ORM\JoinColumn(name: 'id', referencedColumnName: 'id')]
    private ?User $user = null;

    #[ORM\Column(type: Types::TEXT, nullable: true)]
    private ?string $description = null;

    #[ORM\Column(nullable: true)]
    private ?float $wallet = null;

    #[ORM\Column(nullable: true)]
    private ?float $totalRevenue = null;

    /**
     * @var Collection<int, Formation>
     */
    #[ORM\ManyToMany(targetEntity: Formation::class, inversedBy: 'formateur')]
    private Collection $formation_created;

    public function __construct()
    {
        $this->formation_created = new ArrayCollection();

    }
      public function getId(): ?int
    {
        return $this->user?->getId();
    }
    public function getUser(): ?User
    {
        return $this->user;
    }

    public function setUser(User $user): static
    {
        $this->user = $user;

        return $this;
    }

    public function getDescription(): ?string
    {
        return $this->description;
    }

    public function setDescription(?string $description): static
    {
        $this->description = $description;
        return $this;
    }

    public function getWallet(): ?float
    {
        return $this->wallet;
    }

    public function setWallet(?float $wallet): static
    {
        $this->wallet = $wallet;

        return $this;
    }

    public function getTotalRevenue(): ?float
    {
        return $this->totalRevenue;
    }

    public function setTotalRevenue(?float $totalRevenue): static
    {
        $this->totalRevenue = $totalRevenue;

        return $this;
    }

    /**
     * @return Collection<int, Formation>
     */
    public function getFormationCreated(): Collection
    {
        return $this->formation_created;
    }

    public function addFormationCreated(Formation $formationCreated): static
    {
        if (!$this->formation_created->contains($formationCreated)) {
            $this->formation_created->add($formationCreated);
        }

        return $this;
    }

    public function removeFormationCreated(Formation $formationCreated): static
    {
        $this->formation_created->removeElement($formationCreated);

        return $this;
    }
}
```

Thanks! Since your `Formateur` entity uses a **OneToOne relation with User** (i.e., the `formateur.id` is the same as `user.id`), and a **ManyToMany relation with Formation**, here's a complete set of **manual SQL insert statements** for MySQL (based on your latest structure) including:

1. **Formateurs** (user_id: 1–6 and 8–11)
2. **Formations**
3. **Formation-Formateur association** (junction table for ManyToMany)

✅ 1. Insert Formateurs

Each formateur has the same ID as their related user (`user_id`):

```sql
-- FORMATEURS (IDs = user IDs)
INSERT INTO formateur (id, description, wallet, total_revenue) VALUES
(1, 'Experienced in teaching Python', 120.50, 1000),
(2, 'UX Designer with passion for teaching', 95.00, 700),
(3, 'Data analyst and trainer', 110.00, 850),
(4, 'Project manager turned instructor', 80.00, 600),
(5, 'Backend PHP developer', 130.00, 1050),
(6, 'Symfony and Laravel coach', 150.00, 1200),
(8, 'Cybersecurity mentor', 90.00, 750),
(9, 'Flutter app dev mentor', 70.00, 550),
(10, 'AI researcher and trainer', 140.00, 1100),
(11, 'DevOps trainer and consultant', 160.00, 1300);
```

✅ 2. Insert Formations

```sql
-- FORMATIONS
INSERT INTO formation (id, title, description, price, published, pdf_filename) VALUES
(1, 'Python Basics', 'Learn Python from scratch', 100, 1, 'python_basics.pdf'),
(2, 'Advanced PHP', 'Level up your PHP skills', 180, 1, 'php_advanced.pdf'),
(3, 'Intro to Data Science', 'Work with real datasets', 220, 1, 'data_science.pdf'),
(4, 'UX/UI Fundamentals', 'Design user-friendly interfaces', 130, 1, 'ux_ui.pdf'),
(5, 'Cybersecurity Essentials', 'Learn how to protect systems', 160, 1, 'cybersecurity.pdf'),
(6, 'Mobile Dev with Flutter', 'Create cross-platform apps', 140, 1, 'flutter.pdf'),
(7, 'AI for Beginners', 'Start your AI journey', 200, 1, 'ai_intro.pdf'),
(8, 'DevOps with Docker', 'Automate and deploy software', 190, 1, 'devops_docker.pdf');
```

✅ 3. Link Many Formateurs to Formations (ManyToMany Table)

Assuming your join table is called `formateur_formation_created` with fields: `formateur_id`, `formation_id`

```sql
-- FORMATEUR <-> FORMATION (ManyToMany relation)
INSERT INTO formateur_formation_created (formateur_id, formation_id) VALUES
```

```
(1, 1),
(3, 1),
(5, 2),
(6, 2),
(3, 3),
(10, 3),
(2, 4),
(4, 4),
(8, 5),
(1, 5),
(9, 6),
(3, 6),
(10, 7),
(11, 7),
(6, 8),
(11, 8);
```

🔄 If You're Using Symfony Fixtures Instead

Let me know and I'll generate a `Fixture` class using `Faker` that:

- Creates 10 formateurs for users 1–6 and 8–11
- Creates 8 formations
- Randomly links 1–3 formateurs per formation.

Would you like that?

now add to admindashboard a statiscs and show statistics about formateur like average salary ... the gainof the website

To add **statistics** to your Symfony `AdminDashboardController`, we'll follow these steps:

✅ 1. Update `AdminDashboardController.php`

You'll compute:

- Total number of formateurs
- Average formateur wallet
- Total revenue made by formateurs
- **Website gain** (e.g., a percentage from each formateur's total revenue, say 20%)

```php
// src/Controller/AdminDashboardController.php
namespace App\Controller;

use App\Repository\FormateurRepository;
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Annotation\Route;

class AdminDashboardController extends AbstractController
{
    #[Route('/admin/dashboard', name: 'admin_dashboard')]
    public function index(FormateurRepository $formateurRepository): Response
    {
        $formateurs = $formateurRepository->findAll();

        $totalFormateurs = count($formateurs);
        $totalWallet = 0;
        $totalRevenue = 0;

        foreach ($formateurs as $formateur) {
            $totalWallet += $formateur->getWallet() ?? 0;
            $totalRevenue += $formateur->getTotalRevenue() ?? 0;
        }

        $averageWallet = $totalFormateurs > 0 ? $totalWallet / $totalFormateurs : 0;

        // Let's say the platform takes 20% cut from each formateur's revenue
        $websiteGain = $totalRevenue * 0.2;

        return $this->render('admin/dashboard.html.twig', [
            'totalFormateurs' => $totalFormateurs,
            'averageWallet' => $averageWallet,
            'totalRevenue' => $totalRevenue,
            'websiteGain' => $websiteGain,
        ]);
    }
}
```

✅ 2. Create the Twig template: `dashboard.html.twig`

Place it at: `templates/admin/dashboard.html.twig`

```
twig

{% extends 'base.html.twig' %}

{% block title %}Admin Dashboard{% endblock %}

{% block body %}
<div class="container mt-5">
    <h1 class="mb-4">📊 Formateur Statistics</h1>

    <div class="row text-center">
        <div class="col-md-3">
            <div class="card shadow-sm">
                <div class="card-body">
                    <h5>Total Formateurs</h5>
                    <p class="display-6">{{ totalFormateurs }}</p>
                </div>
            </div>
        </div>

        <div class="col-md-3">
            <div class="card shadow-sm">
                <div class="card-body">
                    <h5>Average Wallet</h5>
                    <p class="display-6">{{ averageWallet|number_format(2) }} DT</p>
                </div>
            </div>
        </div>

        <div class="col-md-3">
            <div class="card shadow-sm">
                <div class="card-body">
                    <h5>Total Revenue</h5>
                    <p class="display-6">{{ totalRevenue|number_format(2) }} DT</p>
                </div>
            </div>
        </div>
        <div class="col-md-3">
            <div class="card shadow-sm">
                <div class="card-body">
                    <h5>Website Gain (20%)</h5>
                    <p class="display-6 text-success">{{ websiteGain|number_format(2) }} DT</p>
                </div>
            </div>
        </div>
    </div>
</div>
{% endblock %}
```

🔧 3. Optional Enhancements

If you want to:

- Show a graph: use Chart.js
- Include more KPIs: formations count, revenue per formation, etc.

✅ Result

You now have an `/admin/dashboard` page showing:

- Total formateurs
- Average wallet value
- Total revenue of formateurs
- The website's 20% commission income

Let me know if you want to:

- Add charts
- Export to CSV
- Add per-month or per-formateur stats!

make statistics card

Here's how to create **beautiful Bootstrap "statistics cards"** to display formateur statistics in your admin dashboard.

✅ 1. Use This HTML in `dashboard.html.twig`

Place this inside the `{% block body %}` in `templates/admin/dashboard.html.twig`:

```twig

<div class="container mt-5">
    <h1 class="mb-4">📊 Formateur Statistics</h1>
    <div class="row g-4">
        <div class="col-md-3">
            <div class="card text-white bg-primary h-100 shadow">
                <div class="card-body text-center">
                    <h5 class="card-title">Total Formateurs</h5>
                    <p class="display-6">{{ totalFormateurs }}</p>
                </div>
            </div>
        </div>

        <div class="col-md-3">
            <div class="card text-white bg-info h-100 shadow">
                <div class="card-body text-center">
                    <h5 class="card-title">Average Wallet</h5>
                    <p class="display-6">{{ averageWallet|number_format(2) }} DT</p>
                </div>
            </div>
        </div>

        <div class="col-md-3">
            <div class="card text-white bg-success h-100 shadow">
                <div class="card-body text-center">
                    <h5 class="card-title">Total Revenue</h5>
                    <p class="display-6">{{ totalRevenue|number_format(2) }} DT</p>
                </div>
            </div>
        </div>

        <div class="col-md-3">
            <div class="card text-white bg-dark h-100 shadow">
                <div class="card-body text-center">
                    <h5 class="card-title">Website Gain (20%)</h5>
                    <p class="display-6">{{ websiteGain|number_format(2) }} DT</p>
                </div>
            </div>
        </div>
    </div>
</div>
```

✅ 2. Ensure You Have Bootstrap 5

Make sure your `base.html.twig` includes Bootstrap (probably already if you're using Symfony UX or admin templates):

```twig
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="styleshee
```

✅ 3. Result Preview

Each statistic will appear as a **colored card with a shadow**, like this:

💙 Total	💧 Avg Wallet	✅ Revenue	⚫ Gain

Let me know if you want to:

- Add **icons**
- Display **charts**
- Show **statistics per month or per formateur**

with not icons with charts and all

To add a **statistics card** section with charts and data about formateurs (like average wallet, total revenue, and website gain) in your `AdminDashboardController`, follow these steps:

✅ Step 1: Update Your Controller

In `AdminDashboardController.php`:

```php
#[Route('/admin', name: 'admin_dashboard')]
public function index(FormateurRepository $formateurRepository): Response
{
    $formateurs = $formateurRepository->findAll();

    $totalFormateurs = count($formateurs);
    $totalWallet = array_sum(array_map(fn($f) => $f->getWallet() ?? 0, $formateurs));
    $totalRevenue = array_sum(array_map(fn($f) => $f->getTotalRevenue() ?? 0, $formateurs));
    $averageWallet = $totalFormateurs > 0 ? $totalWallet / $totalFormateurs : 0;
    $websiteGain = $totalRevenue * 0.2;

    // Prepare chart data
    $chartLabels = array_map(fn($f) => 'User ' . $f->getUser()->getId(), $formateurs);
    $chartData = array_map(fn($f) => $f->getTotalRevenue() ?? 0, $formateurs);

    return $this->render('admin/dashboard.html.twig', [
        'totalFormateurs' => $totalFormateurs,
        'totalRevenue' => $totalRevenue,
        'averageWallet' => $averageWallet,
        'websiteGain' => $websiteGain,
        'chartLabels' => $chartLabels,
        'chartData' => $chartData,
    ]);
}
```

✅ Step 2: Update the Twig Template

Replace your `dashboard.html.twig` with this full statistics card and chart version (or share it and I can do it for you in Canvas):

```twig
{% extends 'base.html.twig' %}

{% block title %}Admin Dashboard{% endblock %}

{% block body %}
<div class="container mt-5">
    <h1 class="mb-4">Formateur Statistics</h1>

    <div class="row g-4 mb-5">
        <div class="col-md-3">
            <div class="card bg-primary text-white text-center shadow h-100">
                <div class="card-body">
                    <h5 class="card-title">Total Formateurs</h5>
                    <p class="display-6">{{ totalFormateurs }}</p>
                </div>
            </div>
        </div>

        <div class="col-md-3">
            <div class="card bg-info text-white text-center shadow h-100">
                <div class="card-body">
                    <h5 class="card-title">Average Wallet</h5>
                    <p class="display-6">{{ averageWallet|number_format(2) }} DT</p>
                </div>
            </div>
        </div>

        <div class="col-md-3">
            <div class="card bg-success text-white text-center shadow h-100">
                <div class="card-body">
                    <h5 class="card-title">Total Revenue</h5>
                    <p class="display-6">{{ totalRevenue|number_format(2) }} DT</p>
                </div>
            </div>
        </div>

        <div class="col-md-3">
            <div class="card bg-dark text-white text-center shadow h-100">
                <div class="card-body">
                    <h5 class="card-title">Website Gain (20%)</h5>
                    <p class="display-6">{{ websiteGain|number_format(2) }} DT</p>
                </div>
            </div>
        </div>
    </div>

    <div class="row">
        <div class="col-md-12">
            <div class="card shadow">
                <div class="card-body">
                    <canvas id="revenueChart"></canvas>
                </div>
            </div>
        </div>
    </div>
</div>

<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
<script>
    const ctx = document.getElementById('revenueChart').getContext('2d');
    const chart = new Chart(ctx, {
        type: 'bar',
        data: {
            labels: {{ chartLabels|json_encode|raw }},
            datasets: [{
                label: 'Revenue per Formateur',
                data: {{ chartData|json_encode|raw }},
                backgroundColor: 'rgba(40, 167, 69, 0.7)',
                borderColor: 'rgba(40, 167, 69, 1)',
                borderWidth: 1
            }]
        },
        options: {
            scales: {
                y: {
                    beginAtZero: true,
                    ticks: {
                        callback: function(value) {
                            return value + ' DT';
                        }
                    }
                }
            }
        }
    });
</script>
{% endblock %}
```

If you'd like me to add this into Canvas or update your Twig file directly, just say **"yes, update my Twig with this chart and stats"**.