Great idea — [Nextras Migrations](https://github.com/nextras/migrations) is a solid, Nette-native tool for managing migrations without needing Doctrine. It integrates cleanly with the Nette DI container and works with **Nette Database Explorer**, so it's a perfect choice for your setup.

✅ **How to Use Nextras Migrations in a Nette Project**

🔧 **1. Install via Composer**

bash

CopyEdit

composer require nextras/migrations

🏗️ **2. Register the Extension in config.neon**

neon

CopyEdit

extensions:

migrations: Nextras\Migrations\Bridges\NetteDI\MigrationsExtension

migrations:

dir: %appDir%/../migrations

You can change the dir path as needed.

📁 **3. Create the Migration Directory**

Create the directory where your SQL or PHP migration files will live:

pgsql

CopyEdit

your-project/

├─ app/

├─ migrations/

│ └─ 20250708100000\_init.sql

The filename must begin with a timestamp: YYYYMMDDHHMMSS\_description.sql

✍️ **4. Write a Migration File**

Example: migrations/20250708100000\_init.sql

sql

CopyEdit

CREATE TABLE users (

id INT AUTO\_INCREMENT PRIMARY KEY,

username VARCHAR(255) NOT NULL,

password VARCHAR(255) NOT NULL

);

You can also use PHP-based migrations if needed.

🚀 **5. Run the Migration**

Use the command-line tool to run migrations:

bash

CopyEdit

php www/index.php migrations:continue

Or if you're using bin/console, or a custom CLI script in app/bootstrap.php, just make sure the CLI runner is available (see below).

🧰 **6. Setup a CLI Tool (Optional but Recommended)**

Create a console.php:

php

CopyEdit

#!/usr/bin/env php

<?php

require \_\_DIR\_\_ . '/vendor/autoload.php';

$configurator = new Nette\Configurator;

$configurator->setTempDirectory(\_\_DIR\_\_ . '/temp');

$configurator->addConfig(\_\_DIR\_\_ . '/app/config/common.neon'); // adjust as needed

$container = $configurator->createContainer();

/\*\* @var Nextras\Migrations\Engine $migrations \*/

$migrations = $container->getByType(Nextras\Migrations\Engine::class);

$migrations->run(); // runs all pending migrations

Make it executable and run it:

bash

CopyEdit

php console.php

✅ **Advantages of Nextras Migrations**

* Simple integration with **Nette**
* Supports both **SQL and PHP** migrations
* Keeps a \_\_migrations table to track what's been applied
* Easy to maintain in team environments

Would you like me to generate a working example migration setup (e.g., config.neon, example migration file, and script)?

**You said:**

yes

**ChatGPT said:**

Here’s a **working example** of how to set up **Nextras Migrations** in a typical Nette 3.1+ project with:

* config.neon registration
* example SQL migration
* a CLI runner (console.php)

✅ **1. composer.json (Make sure Nextras is installed)**

Run in terminal:

bash

CopyEdit

composer require nextras/migrations

✅ **2. app/config/common.neon**

Add this to your configuration:

neon

CopyEdit

extensions:

migrations: Nextras\Migrations\Bridges\NetteDI\MigrationsExtension

migrations:

dir: %appDir%/../migrations

💡 Make sure common.neon or your main config file is loaded in your bootstrap.php.

✅ **3. Create the Migrations Directory and Example SQL Migration**

**Directory: migrations/**

Create the folder (if not already):

pgsql

CopyEdit

your-project/

├── app/

├── migrations/

│ └── 20250708100000\_init.sql

**File: migrations/20250708100000\_init.sql**

sql

CopyEdit

-- creates the users table

CREATE TABLE users (

id INT AUTO\_INCREMENT PRIMARY KEY,

username VARCHAR(255) NOT NULL,

password VARCHAR(255) NOT NULL

);

📅 Name format must begin with YYYYMMDDHHMMSS\_Description.sql.

✅ **4. Create console.php in your project root**

Make a file in the root (same level as vendor/):

**console.php**

php

CopyEdit

#!/usr/bin/env php

<?php

declare(strict\_types=1);

require \_\_DIR\_\_ . '/vendor/autoload.php';

$configurator = new Nette\Configurator;

$configurator->setTempDirectory(\_\_DIR\_\_ . '/temp');

$configurator->addConfig(\_\_DIR\_\_ . '/app/config/common.neon'); // adjust if needed

$container = $configurator->createContainer();

/\*\* @var Nextras\Migrations\Engine $migrations \*/

$migrations = $container->getByType(Nextras\Migrations\Engine::class);

$migrations->run(); // runs all pending migrations

Make it executable:

bash

CopyEdit

chmod +x console.php

Then run:

bash

CopyEdit

php console.php

✅ **5. Check the Result**

* It will create a table \_\_migrations in your DB.
* Your users table will be created.
* If you rerun it, it won’t apply the same migration again.

🔁 **Adding More Migrations Later**

1. Create a new file: migrations/20250709123000\_add\_notes\_table.sql
2. Add SQL:  
   sql  
   CopyEdit  
     
     
   CREATE TABLE notes (
3. id INT AUTO\_INCREMENT PRIMARY KEY,
4. user\_id INT NOT NULL,
5. content TEXT NOT NULL,
6. created\_at DATETIME NOT NULL
7. );
9. Run again:  
   bash  
   CopyEdit  
     
     
   php console.php

Would you like me to zip up a full skeleton project using this setup, or generate additional PHP-style migrations too?