

Table des matières

1	Configuration GitHub et outils externes.....	2
2	Setup local et premier push	4

1 Configuration GitHub et outils externes

1. 1.1 Créer le repo distant

Sur github.com, créer un nouveau repository :

- **Nom** : nextjs-template
- **Description** : (optionnel) "Production-ready Next.js template"
- **Visibilité** : Public
- **✗** Ne pas ajouter de README
- **✗** Ne pas ajouter de .gitignore
- **✗** Ne pas ajouter de licence

Cliquer sur **Create repository**.

2. 1.2 Marquer comme template

1. Aller dans **Settings** du repo
 2. Section **General**
 3. Cocher **Template repository**
-

3. 1.3 Activer Dependabot

1. Aller dans **Settings** → **Advanced security**
 2. Activer :
 - Dependency graph
 - Dependabot alerts
-

4. 1.4 Activer Secrets Scanning

1. Aller dans **Settings** → **Advanced security**
2. Section **Secret Protection** :

- Cliquer sur **Enable**
 - Activer aussi **Push protection**
-

5. 1.5 Configurer les Pull Requests

1. Aller dans **Settings → General**
 2. Section **Pull Requests** :
 - Allow auto-merge
 - Allow merge commits
 - Allow rebase merging
 - Allow squash merging → sélectionner "**Pull request title and description**"
 - Always suggest updating pull request branches
 - Automatically delete head branches
-

6. 1.6 Installer les GitHub Apps

Faut aller dans les settings du compte GitHub et dans la section Applications il y a déjà les app installées, faudra juste cliquer sur configurer pour chacune. Voici les spécificités pour chacune d'elles :

3. Renovate (mises à jour automatiques des dépendances)

Choisir Renovate Only.

Ensuite choisir Scan and Alert.

4. Codecov (rapports de code coverage)

Rien à faire de spécial, faut juste ajouter le repo dans les Github Settings et cliquer sur save.

5. DeepSource (analyse statique du code)

Rien à faire de spécial, faut juste ajouter le repo dans les Github Settings et cliquer sur save.

⚠ NE PAS CONFIGURER LE BRANCH RULESET MAINTENANT

On le fera à la Partie 8, après le premier push.

2 Setup local et premier push

- **2.1 Créer le dossier du projet**

```
mkdir ~/projects/nextjs-template
```

```
cd ~/projects/nextjs-template
```

```
code .
```

- **2.2 Initialiser le projet Next.js**

Dans le terminal VS Code :

```
pnpm create next-app@latest .
```

Options à sélectionner :

Question	Réponse
Would you like to use the recommended Next.js defaults?	No, customize settings
Would you like to use TypeScript?	Yes
Which linter would you like to use?	ESLint

Question	Réponse
Would you like to use React Compiler?	Yes
Would you like to use Tailwind CSS?	Yes
Would you like your code inside a 'src/' directory?	Yes
Would you like to use App Router?	Yes
Would you like to customize the import alias?	No

- **2.3 Premier commit**

git init

git add .

git commit -m "chore: initial Next.js setup"

- **2.4 Lier au repo distant et push**

git remote add origin git@github.com:thewebworkshop-ch/nextjs-template.git

git push -u origin main

- **Partie 3 : Copier les Fichiers de Configuration**

Depuis ton repo existant (the-grey-room), copier les fichiers suivants dans nextjs-template :

- **Fichiers à la racine**

.gitattributes

.gitignore

.prettierrc

.prettierignore

.gitleaks.toml
.nvmrc
codecov.yml
commitlint.config.mjs
components.json
docker-compose.test.yml
Dockerfile
.dockerignore
eslint.config.mjs
next.config.ts
package.json
playwright.config.ts
pnpm-workspace.yaml
postcss.config.mjs
prisma.config.ts
renovate.json
tsconfig.json
vitest.config.ts
vitest.setup.ts
vitest.shims.d.ts
.deepsource.toml

- **Dossiers entiers à copier**

.github/
.husky/
.storybook/
.vscode/

```
prisma/  
e2e/  
src/  
public/  
• ⚠ Ne PAS copier  
node_modules/  
.next/  
pnpm-lock.yaml  
README.md  
coverage/  
playwright-report/  
.tsbuildinfo  
test-results/  
.infisical.json
```

Ensuite fais ceci :

```
pnpm install  
chmod +x .husky/_/*  
sudo apt install gitleaks  
npx playwright install --with-deps
```

• Commit

```
git add .  
git commit -m "chore: add template configuration files"  
git push
```

- **Partie 5 : Configurer les Outils Externes**

Maintenant que le repo contient du code et que la CI tourne.

- **5.1 Codecov**

1. Aller sur <https://app.codecov.io>
 2. Se connecter avec GitHub
 3. Cliquer sur **Add repository** → sélectionner nextjs-template
 4. Copier le **CODECOV_TOKEN** affiché
 5. Sur GitHub, aller dans **Settings** → **Secrets and variables** → **Actions**
 6. Cliquer sur **New repository secret**
 - **Name** : CODECOV_TOKEN
 - **Secret** : coller le token
 7. Cliquer sur **Add secret**
-

- **5.2 DeepSource**

1. Aller sur <https://app.deepsource.com>
 2. Se connecter avec GitHub
 3. Cliquer sur **Activate repository** → sélectionner nextjs-template
 4. La configuration est déjà présente dans .deepsource.toml
-

- **5.3 Snyk**

1. Aller sur <https://app.snyk.io>
2. Se connecter avec GitHub
3. Cliquer sur **Add project**
4. Sélectionner nextjs-template

-
- 5. Snyk scannera automatiquement chaque PR (pas besoin de token)

- **5.4 Infisical**

1. Aller sur <https://app.infisical.com>
2. Se connecter
3. Créer un nouveau projet ou utiliser un projet existant
4. Ajouter les variables d'environnement :
 - DATABASE_URL
 - AUTH_SECRET
 - RESEND_API_KEY

Optionnel - Pour injecter les secrets dans la CI :

1. Dans Infisical : **Project Settings** → **Service Tokens** → Créer un token
 2. Sur GitHub : **Settings** → **Secrets and variables** → **Actions**
 3. Ajouter le secret INFISICAL_TOKEN
-

- **Partie 6 : Vérification**

Exécuter tous les checks pour vérifier que tout fonctionne :

pnpm format:check

pnpm lint

pnpm test

pnpm build

Si tout passe, continuer. Sinon, corriger les erreurs.

- **Partie 7 : README**

Créer le fichier README.md à la racine.

- **Commit**

```
git add README.md
```

```
git commit -m "docs: update README with tech stack overview"
```

```
git push
```

- **Partie 8 : Activer le Branch Ruleset**

Maintenant que la CI a tourné au moins une fois, on peut activer le ruleset.

1. Aller dans **Settings** → **Rules** → **Rulesets**
2. Cliquer sur **New ruleset** → **New branch ruleset**

- **Configuration du ruleset**

Ruleset name : Main branch protection

Enforcement status : Active

Target branches : Cliquer sur **Add target** → **Include default branch**

Ensuite coche ceci :

- Restrict deletions
- Require linear history
- Require signed commits
- Require a pull request before merging et mets ceci dans les additionnal settings :
 - Required approvals : 0
 - Allowed merge methods : uniquement squash
- Require status checks to pass et mets ceci dans les additionnal settings :
 - Coche Require branches to be up to date before merging
 - Dans les checks ajoute tous les checks de ta ci. Commence par taper le nom du check (le nom du check c'est ce qui est écrit après le « name : » dans ton fichier ci.yml) et quand tu commenceras à écrire le début du

nom, le check apparaitra automatiquement comme suggestion, faudra juste cliquer dessus pour l'ajouter. Les checks à ajouter sont les suivants :

- Check formatting
 - Type Check
 - Lint Code
 - Run Tests
 - Secret Scanning
 - codecov/patch
 - Deepsource : Javascript
 - Docker Build & E2E Tests
 - security/snyk
- Block force pushes

- **Partie 9 : Workflow Futur**

À partir de maintenant, **plus de push direct sur main**. Pour chaque modification :

```
# Créer une nouvelle branche
```

```
git checkout -b feat/ma-feature
```

```
# Faire les changements...
```

```
# ...
```

```
# Commit
```

```
git add .
```

```
git commit -m "feat: description du changement"
```

```
# Push de la branche
```

```
git push -u origin feat/ma-feature
```

```
# Sur GitHub : créer une Pull Request
```

```
# Attendre que la CI passe
```

```
# Merger via Squash and merge
```

Résumé des commits

#	Message	Push ?
1	chore: initial Next.js setup	<input checked="" type="checkbox"/>
2	chore: add template configuration files	<input checked="" type="checkbox"/>
3	chore: install dependencies	<input checked="" type="checkbox"/>
4	docs: add README	<input checked="" type="checkbox"/>

Après activation du ruleset → Tout passe par des PR.

Tableau récapitulatif des outils

Catégorie	Outil
Framework	Next.js 16, React 19, TypeScript, React Compiler
UI	Tailwind CSS 4, shadcn/ui, CVA, Lucide
Database	Prisma + PostgreSQL
Auth	NextAuth.js v4
Forms	React Hook Form + Zod
Emails	Resend
Env vars	t3-oss/env + Infisical
Tests unitaires	Vitest + React Testing Library
Tests E2E	Playwright
Mocking API	MSW
Documentation	Storybook
Formatting	Prettier
Linting	ESLint
Git hooks	Husky + lint-staged + commitlint
CI/CD	GitHub Actions
Dépendances	Renovate + Dependabot
Sécurité code	Gitleaks, Snyk
Qualité code	DeepSource
Coverage	Codecov
Sécurité web	Security headers
Monitoring	Health check endpoint
Déploiement	Docker multi-stage Debian
DX	VS Code settings + extensions