Passage sur Azure DevOps – panorama complet

Vous allez désormais héberger le projet dans l'organisation `https://dev.azure.com/Septeo-SECIB/OnboardingClient` – autrement dit, l'écosystème **Azure DevOps** devient votre centre névralgique pour le code, la livraison, la gestion de projet et la documentation.

Ci-dessous :

1. Impact immédiat sur votre projet (ce qui change).
2. Catalogue exhaustif des briques Azure DevOps et de leurs atouts.
3. Bonnes pratiques & incontournables à mettre en place dès le départ.
4. Pistes d'évolution / cas d'usage avancés.

1. Ce qui change tout de suite

Domaine	Avant (GitHub + Render)	Après (Azure DevOps)	Action à prévoir
Hébergement Git	tlibouban.github.io sur GitHub	`Azure Repos` (repo Git privé)	• Pousser un miroir du dépôt • Mettre à jour origin/remotes
CI / CD	(aucune ou GitHub Actions manuelles)	`Azure Pipelines` (agents hébergés Microsoft)	• Créer un pipeline YAML (ci-test) • Ajouter stage deploy (Render ou Azure)
Gestion de tâches	Informel / README	`Azure Boards` (Work Items, Sprints, Kanban)	• Définir Work Item types (Epic, User Story, Task) • Importer backlog

Domaine	Avant (GitHub + Render)	Après (Azure DevOps)	Action à prévoir
Documentation	Fichiers Markdown dispersés	`Azure Wiki` (Markdown + pages versionnées)	• Migrer / structurer la doc existante
Packages (npm, pg)	npmjs public	`Azure Artifacts` (registry privé npm/nuget/docker)	• Pas obligatoire, mais dispo

2. Panorama des briques Azure DevOps

A. Azure Repos (Contrôle de version Git)

- Repos privés illimités, branche *par dépôt* ou mono-repo.
- Politiques de branche : PR obligatoires, reviewers minimum, validation build, *auto-merge*.
- Hooks & *status checks* (lint, tests).
- **Code Search** plein texte.
- *Pull-request templates*, commentaires résolus, règles d'approbation par dossier.
- Liens automatiques entre commits/PR et Work Items (*Fixes #123*).

B. Azure Boards (ALM / Gestion de projet)

- Backlog Agile, Scrum, Kanban ; épics | features | user stories | tasks | bugs.
- Sprints (capacités, vélocité), *Burndown* & *Cumulative Flow*.
- **Delivery Plans** = road-map multiprojets.
- Règles d'automatisation : déplacer un WI quand la PR est mergée.
- Liens Gmail/Outlook ou Teams pour créer un work item depuis un mail.

C. Azure Pipelines (CI / CD)

- Agents Microsoft hosted (Win/Linux/macOS) gratuits (1 job parallèle) ; auto-scale.
- YAML as code (versionné dans le repo).
- Stages, jobs, steps, matrix, *cache task*.
- Environnements & approbations manuelles (préprod → prod).
- **Templates réutilisables** pour Node, Python, Docker, Terraform…
- Secrets/variables chiffrées dans *Variable Groups* ou *Azure Key Vault*.

- Intégration Render : pipeline → `curl` deploy-hook ou `render.yaml` .

D. Azure Artifacts

- Feeds privés npm, NuGet, Maven, PyPI, Universal packages.
- Retention & immutabilité via *Upstream sources* (miroir npmjs).

E. Test Plans

- Gestion des plans de tests manuels / exploratoires.
- Exécution automatisée (Selenium, Playwright) avec reporting.

F. Dashboard & Analytics

- Tuiles personnalisables : build health, Récap sprint, PR ouvertes.
- **Power BI** connector natif pour extraire WI & builds.

G. Extensions Marketplace

- +1000 extensions (SonarCloud, Snyk, Dependabot, Mermaid, PlantUML, Draw.io, Slack, Teams, Retrospectives…).

3. Incontournables à activer dès le départ

1. **Politique de branche**
   - `main` protégé : PR + 1 review + build passing.
   - `feature/*` → squash merge.
2. **Pipeline de base**

```yaml
trigger:
  branches: [ main, develop ]
jobs:
- job: CI
  pool: 'ubuntu-latest'
  steps:
    - checkout: self
    - uses: actions/setup-node@v4 # tâche équivalente
    - script: npm ci
    - script: npm test
    - script: node run_schema.js  # migrations dry-run
```

3. **Variable Group “prod”**
   - `DATABASE_URL` (secret)
   - `RENDER_API_KEY` (secret)
4. **Boards configuration**
   - Epics = gros lots fonctionnels ; Features = modules (Check-list, API, DB).
   - *Definition of Done* sur chaque WI : tests, PR mergée, doc.
5. **Wiki**
   - Conventions de code, architecture (schéma ERD déjà produit), procédure de déploiement Render.
6. **Dashboards**
   - Widget *Burndown* sprint, Build status, PR à reviewer.

4. Champs du possible / évolutions

Besoin futur	Fonctionnalité Azure DevOps/App Service
Déploiement full Azure	Pipelines → Stage “Azure App Service” ou “Static Web Apps”.
Infra as Code	Tâches Terraform/Bicep dans Pipeline ; backend state dans Azure RM.
Qualité / Sécu	Extensions SonarCloud, Snyk, WhiteSource ; gating sur Quality Gate.
Release Train multi-produits	Delivery Plans + multi-repo Stages ; environment-checks manuelles.

Besoin futur	Fonctionnalité Azure DevOps/App Service
Tests exploratoires	Azure Test Plans (Chrome extension), capture vidéo, bug auto-logging.
Monitoring post-prod	Pipeline → step “az monitor app-service” ou intégration Application Insights (logs + alertes).
ChatOps	Extension Azure DevOps ⇄ Teams : notification build/PR/WorkItem, approbations via Teams.
Packages internes Node	Publier un *scope* `@septeo-secib/*` dans Azure Artifacts (npm).
Automatisation Backlog	Azure DevOps CLI (`az boards work-item create`) ou REST API ; scriptable depuis Render hooks.
Single-Sign-On	AAD groups → rôles DevOps (Admins, Contributors, Readers).

5. Étapes prioritaires

1. **Créer le repo** `OnboardingClient` → importer depuis GitHub.
2. **Configurer Boards** : workflow Agile & backlog initial.
3. **Ajouter le pipeline CI** minimal (lint + tests).
4. **Stocker secrets DB/Render** dans un Variable Group lié à la bibliothèque de sécurités Azure.
5. **Migrer la documentation** (vos `.md` , schémas) dans Azure Wiki.
6. **Former l’équipe** : 30 min d’introduction Boards + PR policies.

6. Ressources utiles

- Documentation officielle Azure DevOps : https://learn.microsoft.com/azure/devops
- Bonnes pratiques pipelines (YAML) : https://learn.microsoft.com/azure/devops/pipelines/best-practices
- Extension Marketplace : https://marketplace.visualstudio.com/azuredevops

Une fois ces fondations posées, vous disposerez d’un **hub unifié** couvrant 100 % du cycle de vie logiciel : planification, code, build, test, release, suivi des incidents, documentation – le tout

interconnecté, traçable et sécurisé.