

Contents

Azure Vercel : Stratégie de Complémentarité pour Bénéteau	1
Table des matières	1
Positionnement stratégique	1
Vision : Complémentarité, pas remplacement	1
Cas d'usage idéaux	2
Différences techniques détaillées	2
1. Ce que Vercel NE PEUT PAS faire (Azure only)	2
Flux de données et architecture	3
Architecture hybride Azure + Vercel (Recommandé Bénéteau)	3

Azure Vercel : Stratégie de Complémentarité pour Bénéteau

Objectif : Utiliser le Manufacturing Portal comme cas d'usage pour démontrer comment Vercel peut compléter Azure chez Bénéteau, particulièrement pour les POCs, petites apps et déploiements rapides de test.

Table des matières

1. [Positionnement stratégique](#)
 2. [Différences techniques détaillées](#)
 3. [Flux de données et architecture](#)
 4. [Propriété et sécurité des données](#)
 5. [Stratégie de déploiement hybride](#)
 6. [Nouveautés Vercel 2024-2025](#)
 7. [Cas d'usage Manufacturing Portal](#)
-

Positionnement stratégique

Vision : Complémentarité, pas remplacement

ÉCOSYSTÈME BÉNÉTEAU

AZURE (Production)

- Apps critiques
- Infrastructure complexe
- Compliance stricte
- Long-running tasks
- VNet privé

VERCEL (Innovation)

- POCs rapides
- Prototypes
- Apps de test
- Démonstrations clients
- Expérimentations
- Dev/Preview URLs

- Managed Identity

Setup: 1-2 jours

- Itérations rapides

Setup: 30 min

Mêmes ressources :

- Azure APIM (IFS)
- Azure SQL DB
- Microsoft Entra ID
- Azure Storage

Cas d'usage idéaux

Critère	Azure	Vercel
App en production (>1000 users)	Recommandé	Possible
POC / Prototype (<50 users)	Overkill	Parfait
Démo client (temporaire)	Trop lent à setup	URL en 5 min
Tests A/B rapides	Complexe	Preview URLs
App critique 24/7	Recommandé	Cold starts
Long-running jobs (>60s)	Oui	Non (timeout)
VNet privé requis	Oui	Non
Dev rapide	CI/CD à configurer	Push = deploy

Différences techniques détaillées

1. Ce que Vercel NE PEUT PAS faire (Azure only)

A. Managed Identity - Authentification automatique **Azure Managed Identity** : Azure reconnaît automatiquement votre application et lui donne accès aux services (base de données, API) sans avoir besoin de mots de passe.

En pratique : - **Azure** : Pas de mots de passe à stocker, Azure gère l'authentification automatiquement - **Vercel** : Doit obligatoirement stocker les mots de passe et identifiants dans la configuration - **Sécurité Vercel** : Les mots de passe sont chiffrés, mais ils existent et peuvent être volés

Conséquence pour Bénéteau : - Sur Azure : Si un hacker accède à votre code, il ne trouve aucun mot de passe - Sur Vercel : Les mots de passe sont dans la configuration, risque supérieur

B. Synthèse des différences Azure vs Vercel

Fonctionnalité	Azure	Vercel	Impact Bénéteau
Réseau privé (VNet)	Isolé, communication interne sans Internet	Tout passe par Internet public	Azure plus sécurisé, Vercel nécessite firewall
Authentification base de données	Managed Identity (pas de mots de passe)	Mots de passe obligatoires dans config	Azure élimine le risque de vol de credentials
Connexion Azure SQL	Directe via VNet	Via firewall (liste IPs Vercel à maintenir)	Solution : API proxy sur Azure
Surveillance/Monitoring	Azure Monitor (tout centralisé)	Dispersé (logs 24h, outils externes)	Azure facilite le debugging
Authentification users	Easy Auth (config visuelle, 5 min)	NextAuth.js (~100 lignes code, 1-2h)	Azure plus rapide, Vercel plus flexible
Performance	Always-On (50ms constant)	Serverless (1ère visite 800-1500ms puis 50ms)	Vercel adapté aux POCs intermittents
Timeout API	Illimité	60 secondes maximum	Jobs longs doivent passer par Azure
Taille fichiers	Illimité	50 MB maximum	Upload direct vers Azure Blob Storage

Conclusion pour Manufacturing Portal : - Opérations < 5 secondes : Aucun problème sur Vercel - POCs avec trafic intermittent : Démarrage lent acceptable (1-2s première fois) - Données sensibles : API proxy Azure recommandée pour isoler la base de données

Flux de données et architecture

Architecture hybride Azure + Vercel (Recommandé Bénéteau)

graph TB

```

    subgraph DEV[" Développement"]
        DEVELOPER[Developer]
    end

```

```

    subgraph SOURCE[" Code Source - UN SEUL REPO"]
        REPO[GitHub Repository<br/>morepudding/manufacturingPortal]
        MAIN[Branch: main<br/>Production]
        FEATURE[Branch: feature/*<br/>poc/* develop]
    end

```

```

    subgraph AZURE_PIPELINE[" CI/CD Azure"]
        AZURE_CI[GitHub Actions<br/>Workflow: azure-deploy.yml<br/>Trigger: push on main<br/>Build: azure-pipelines.yml]
        AZURE_BUILD[Build & Tests<br/>npm install<br/>npm run build<br/>npm test]
    end

```

```

    subgraph VERCEL_PIPELINE[" CI/CD Vercel"]

```

```

    VERCEL_CI[Vercel Auto-Deploy<br/>Trigger: push on feature/*<br/>Build: 2 min]
end

subgraph PRODUCTION[" PRODUCTION"]
    AZURE_APP[ Azure App Service<br/>manufacturing-portal<br/>.azurewebsites.net<br/><br/>]
end

subgraph INNOVATION[" POC / TESTS"]
    VERCEL_APP[ Vercel Deployment<br/>manufacturing-portal-poc<br/>.vercel.app<br/><br/>&l
end

subgraph SHARED[" RESSOURCES PARTAGÉES"]
    subgraph AUTH_SHARED[" Authentification"]
        AAD_SHARED[ Microsoft Entra ID<br/>App Registration unique<br/>Redirect URIs:<br/>]
    end

    subgraph DATA_SHARED[" Base de Données"]
        SQL_SHARED[( Azure SQL Database<br/>Firewall Rules:<br/>Azure Services<br/>Vercel
    end

    subgraph GATEWAY_SHARED[" API Gateway"]
        APIM_SHARED[ Azure APIM<br/>Public endpoint<br/>Subscription keys<br/>Partagé Azur
    end
end

subgraph EXTERNAL[" Externe"]
    IFS[ IFS Cloud<br/>API OData]
end

subgraph MONITORING[" Observabilité"]
    AZURE_MON[ Azure Monitor<br/>Prod metrics<br/>SQL queries<br/>APIM logs]
    VERCEL_MON[ Vercel Analytics<br/>POC metrics<br/>Web Vitals<br/>Logs 24h]
end

DEVELOPER -->|git push origin main| REPO
DEVELOPER -->|git push origin feature/xyz| REPO

REPO --> MAIN
REPO --> FEATURE

MAIN -->|Webhook trigger| AZURE_CI
AZURE_CI --> AZURE_BUILD
AZURE_BUILD -->|deploy| AZURE_APP

FEATURE -->|Auto-detect| VERCEL_CI
VERCEL_CI -->|deploy| VERCEL_APP

AZURE_APP -->|authentifie| AAD_SHARED

```

```
VERCEL_APP -->|authentifie| AAD_SHARED
```

```
AZURE_APP -->|requêtes SQL| SQL_SHARED
```

```
VERCEL_APP -->|requêtes SQL| SQL_SHARED
```

```
AZURE_APP -->|appels API| APIM_SHARED
```

```
VERCEL_APP -->|appels API| APIM_SHARED
```

```
APIM_SHARED -->|OAuth2| IFS
```

```
AZURE_APP -.->|logs| AZURE_MON
```

```
VERCEL_APP -.->|analytics| VERCEL_MON
```

```
SQL_SHARED -.->|metrics| AZURE_MON
```

```
APIM_SHARED -.->|logs| AZURE_MON
```

```
classDef azureStyle fill:#0078D4,stroke:#003366,stroke-width:3px,color:#fff
classDef vercelStyle fill:#000000,stroke:#333,stroke-width:3px,color:#fff
classDef sharedStyle fill:#10B981,stroke:#065F46,stroke-width:3px,color:#fff
classDef externalStyle fill:#FF6B6B,stroke:#C92A2A,stroke-width:2px,color:#fff
classDef monitorStyle fill:#8B5CF6,stroke:#5B21B6,stroke-width:2px,color:#fff
classDef sourceStyle fill:#F59E0B,stroke:#D97706,stroke-width:2px,color:#fff
```

```
class AZURE_APP,AZURE_CI,AZURE_BUILD,AZURE_PIPELINE azureStyle
```

```
class VERCEL_APP,VERCEL_CI,VERCEL_PIPELINE vercelStyle
```

```
class AAD_SHARED,SQL_SHARED,APIM_SHARED sharedStyle
```

```
class IFS externalStyle
```

```
class AZURE_MON,VERCEL_MON monitorStyle
```

```
class REPO,MAIN,FEATURE sourceStyle
```

Avantages de cette architecture : - **UN SEUL REPO** : Code partagé, pas de duplication - **Déploiement automatique** : Pas d'intervention manuelle selon la branche - **Ressources partagées** : Azure AD, SQL, APIM utilisés par les deux - **Preview URLs automatiques** : Chaque branche feature = URL de test unique - **CI/CD Azure robuste** : Tests automatiques avant production - **Flexibilité** : Production stable (Azure) + Innovation rapide (Vercel)

Je continue avec la partie suivante sur la propriété des données et la sécurité ?