

Guide de Migration - Amélioration Gestion des Images

Changements apportés

1. Schéma de base de données (Prisma)

Ajout de deux nouveaux champs au modèle Creator : - expertise (String, optionnel) : Domaine d'expertise du créateur - bannerImage (String, optionnel) : URL de l'image de bannière

2. Nouvelle API d'upload d'images

- **Route** : /api/upload/image
- **Méthode** : POST
- **Fonctionnalités** :
 - Upload d'images vers AWS S3
 - Support des formats : JPG, PNG, WEBP
 - Taille maximale : 5MB
 - Validation des fichiers
 - Génération d'URL publique

3. Page Settings améliorée

- Champs d'images avec double option :
 - Saisie manuelle d'URL
 - Upload direct vers S3
- Preview en temps réel :
 - Photo de profil (circulaire)
 - Bannière (rectangulaire)
- Synchronisation correcte avec la base de données
- Gestion des erreurs d'images

Instructions de déploiement

Étape 1 : Mettre à jour les dépendances

```
npm install
```

Étape 2 : Configurer les variables d'environnement AWS

Créez un fichier `.env` à la racine du projet (si pas déjà existant) et ajoutez :

```
# AWS S3 Configuration
AWS_REGION="eu-west-1"
AWS_BUCKET_NAME="votre-bucket-name"
AWS_FOLDER_PREFIX="callastar"

# Option 1 : Utiliser un profil AWS (recommandé en développement local)
AWS_PROFILE="default"

# Option 2 : Utiliser des clés d'accès AWS (pour production/déploiement)
# AWS_ACCESS_KEY_ID="votre-access-key-id"
# AWS_SECRET_ACCESS_KEY="votre-secret-access-key"
```

Étape 3 : Configuration AWS S3

Assurez-vous que votre bucket S3 :

1. Est créé dans la région spécifiée
2. A les permissions publiques en lecture pour les fichiers uploadés
3. A une politique de bucket appropriée (voir ci-dessous)

Exemple de politique de bucket S3 :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicReadGetObject",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::votre-bucket-name/*"
    }
  ]
}
```

Configuration CORS du bucket :

```
[
  {
    "AllowedHeaders": ["*"],
    "AllowedMethods": ["GET", "PUT", "POST"],
    "AllowedOrigins": ["*"],
    "ExposeHeaders": []
  }
]
```

Étape 4 : Appliquer les migrations de base de données

Générer la migration Prisma

```
npx prisma migrate dev --name add_creator_images
```

Ou en production

```
npx prisma migrate deploy
```

Étape 5 : Générer le client Prisma

```
npx prisma generate
```

Étape 6 : Tester en local

```
npm run dev
```

Accédez à la page Settings du créateur et testez : 1. L'affichage des images existantes 2. La saisie manuelle d'URL 3. L'upload d'images vers S3 4. Les previews en temps réel

🔍 Structure des fichiers uploadés

Les images sont organisées dans S3 selon cette structure :

```
s3://BUCKET_NAME/FOLDER_PREFIX/creators/{creatorId}/
├─ profile.jpg (ou .png, .webp)
└─ banner.jpg (ou .png, .webp)
```

✓ Fonctionnalités implémentées

Synchronisation input ↔ base de données

- ✓ Les champs input affichent toujours les valeurs actuelles de la BDD
- ✓ Rechargement automatique après sauvegarde
- ✓ Pas de perte de données lors du rechargement de page

Double option URL/Upload

- ✓ **Option A** : Saisie manuelle d'URL (conservée)
- ✓ **Option B** : Upload vers S3 avec injection automatique de l'URL

Preview immédiate

- ✓ Photo de profil : preview circulaire

- ✓ Bannière : preview rectangulaire
- ✓ Mise à jour en temps réel sans sauvegarde
- ✓ Gestion des erreurs de chargement d'image

Validation et gestion des erreurs

- ✓ Types de fichiers acceptés : JPG, PNG, WEBP
- ✓ Taille maximale : 5MB
- ✓ Messages d'erreur clairs
- ✓ Feedback visuel pendant l'upload

✳ Dépannage

Erreur "Access Denied" lors de l'upload

- Vérifiez les permissions de votre bucket S3
- Vérifiez que les credentials AWS sont correctement configurés
- Vérifiez que l'utilisateur AWS a les permissions `s3:PutObject`

Les images ne s'affichent pas

- Vérifiez que le bucket est public en lecture
- Vérifiez la politique de bucket
- Vérifiez la configuration CORS

Erreur de migration Prisma

- Assurez-vous que la base de données est accessible
- Vérifiez que `DATABASE_URL` est correctement configuré
- Exécutez `npx prisma db pull` pour synchroniser avec la BDD

📄 Notes importantes

1. **Sécurité** : Les uploads sont réservés aux utilisateurs authentifiés avec le rôle `CREATOR`
2. **Performance** : Les images sont stockées sur S3 pour une meilleure performance
3. **URLs publiques** : Les URLs générées sont publiques et accessibles sans authentification
4. **Écrasement** : Les uploads successifs du même type (profile/banner) écrasent l'image précédente

🔗 Support

Pour toute question ou problème, consultez : - La documentation
Prisma : <https://www.prisma.io/docs> - La documentation AWS S3 :
<https://docs.aws.amazon.com/s3/> - La documentation Next.js API
Routes : [https://nextjs.org/docs/app/building-your-](https://nextjs.org/docs/app/building-your-application/routing/route-handlers)
[application/routing/route-handlers](https://nextjs.org/docs/app/building-your-application/routing/route-handlers)