

Améliorations UX, Téléchargement et Suivi du Budget

Résumé des Améliorations

Ce document détaille les trois principales améliorations apportées à l'application Instagram Content Generator :

1. **Visualisation améliorée des vidéos** avec player intégré
 2. **Téléchargement de vidéos** disponible dans toutes les interfaces
 3. **Suivi du budget en temps réel** basé sur la consommation réelle
-

1. Amélioration du Player Vidéo

Changements dans `video-preview.tsx`

Avant

- Player vidéo basique avec contrôles simples
- Boutons de contrôle visibles uniquement au survol
- Design simple sans indicateurs visuels

Après

- **Player vidéo interactif professionnel** avec :
 - ☒ Bouton play central de grande taille quand la vidéo est en pause
 - ☒ Contrôles en bas de la vidéo avec fond dégradé
 - ☒ Badge "Instagram Ready" en haut à droite
 - ☒ Clic sur la vidéo pour play/pause
 - ☒ Design moderne avec ring et ombre portée
 - ☒ Support de `playsInline` pour mobile
 - ☒ Informations de format améliorées (1080 × 1920, 9:16, MP4)

Fonctionnalités du Player

```
// Contrôles disponibles :  
- Play/Pause (bouton central + bouton en bas)  
- Redémarrage (RotateCcw)  
- Mute/Unmute (Volume2/VolumeX)  
- Lecture en boucle automatique
```

Design

- Texte traduit en français
 - Bouton de téléchargement vert avec ombre lumineuse
 - Transitions fluides sur les contrôles
 - Interface responsive
-

2. Téléchargement de Vidéos Partout

Changements dans `job-history.tsx`

Fonctionnalités Ajoutées

1. Fonction `handleDownload`

- Téléchargement asynchrone via l'API `/api/download/${jobId}`
- Conversion blob → URL temporaire
- Création dynamique d'un lien de téléchargement
- Nom de fichier personnalisé : `instagram-reel-{jobId}-{timestamp}.mp4`
- Gestion d'état pour afficher le spinner pendant le téléchargement

2. Bouton de Téléchargement Amélioré

- Icône Download avec état de chargement (Loader2)
- Désactivé pendant le téléchargement
- Tooltip "Télécharger la vidéo"
- Style vert avec hover effect

3. Toasts de Notification

- Notification de succès : "Téléchargement démarré"
- Notification d'erreur : "Échec du téléchargement"
- Messages en français

Code Exemple

```
<Button
  size="sm"
  variant="ghost"
  onClick={(e) => handleDownload(job.id, e)}
  disabled={downloadingJobId === job.id}
  className="text-green-400 hover:text-green-300 hover:bg-green-500/10"
  title="Télécharger la vidéo"
>
  {downloadingJobId === job.id ? (
    <Loader2 className="w-4 h-4 animate-spin" />
  ) : (
    <Download className="w-4 h-4" />
  )}
</Button>
```

Disponibilité du Téléchargement

- ☒ Page de création (section Preview & Download)
- ☒ Historique des jobs (chaque job complété)
- ☒ Indicateur visuel pendant le téléchargement

💰 3. Suivi du Budget en Temps Réel

Modifications du Schéma de Base de Données

Ajout du Champ `cost` dans `ContentJob`

```
model ContentJob {
  // ... autres champs ...
  cost          Float      @default(0.0) // Cost in USD for this job
  // ...
}
```

Migration appliquée avec `prisma db push`

Calcul du Coût Réel

Dans `app/api/upload/route.ts`

Coûts par opération :

- Image transformation (Flux Dev) : **€0.025**
- Video generation (Luma) : **€0.05**
- **Total par job : €0.075**

Nouvelle fonction `updateJobWithCost` :

```
const imageCost = 0.025; // Flux Dev
const videoCost = 0.05;  // Luma Dream Machine
const totalCost = imageCost + videoCost;

await updateJobWithCost(jobId, "COMPLETED", 100, "COMPLETED", {
  finalVideoUrl,
}, totalCost);
```

Le coût est enregistré dans la base de données à la fin de chaque job.

Interface Utilisateur - Budget

1. Header du Dashboard (`content-generator.tsx`)

Affichage dynamique du budget :

- Budget restant affiché dans le header
- Couleur adaptative :
 - Vert : > €10 restants
 - Jaune : €5-€10 restants
 - Rouge : < €5 restants
- Format : `€XX.XX / €20.00`
- Rafraîchissement automatique toutes les 10 secondes

2. Page Profil (`user-profile.tsx`)

Carte Budget Dédiée avec :

- Budget initial : €20.00
- Budget restant (avec couleur adaptative)
- Barre de progression colorée :
 - < 50% consommé
 - 50-80% consommé

- 🟡 > 80% consommé
- Coût moyen par vidéo (affiché si jobs > 0)
- Design avec dégradé vert/émeraude

Statistiques Améliorées :

- 📺 Vidéos Créées (total)
- 🏆 Complétées (succès)
- 📊 Taux de Succès (pourcentage)

3. Historique des Jobs (`job-history.tsx`)

Affichage du coût par job :

- Coût affiché à côté de la date de complétion
- Format : `€0.075` (3 décimales)
- Couleur verte pour indiquer un coût
- Visible uniquement pour les jobs complétés avec coût > 0



Récapitulatif des Fichiers Modifiés

Fichiers Modifiés

1. ✓ `prisma/schema.prisma` - Ajout du champ `cost`
2. ✓ `app/api/upload/route.ts` - Calcul et enregistrement du coût
3. ✓ `app/dashboard/_components/video-preview.tsx` - Player amélioré
4. ✓ `app/dashboard/_components/job-history.tsx` - Téléchargement + affichage coût
5. ✓ `app/dashboard/_components/content-generator.tsx` - Budget dynamique dans le header
6. ✓ `app/dashboard/_components/user-profile.tsx` - Statistiques et budget détaillés

Dépendances Ajoutées

- Aucune nouvelle dépendance (utilisation des packages existants)



Fonctionnalités Clés







Player Vidéo

- ✓ Interface moderne et intuitive
- ✓ Contrôles accessibles (clavier + souris)
- ✓ Design responsive
- ✓ Badge Instagram Ready
- ✓ Informations techniques visibles

Téléchargement

- ✓ Disponible dans tous les contextes
- ✓ Feedback visuel (spinner)
- ✓ Notifications toast
- ✓ Noms de fichiers descriptifs
- ✓ Gestion d'erreurs

Budget

-  Suivi en temps réel
 -  Coût réel par opération
 -  Statistiques détaillées
 -  Visualisation claire (barre de progression)
 -  Alertes visuelles (couleurs)
 -  Historique par job
-



Expérience Utilisateur Améliorée

Avant

- Player vidéo basique
- Téléchargement limité
- Budget statique (€19.87)
- Pas de suivi des coûts réels

Après

- 🎬 **Player professionnel** avec contrôles complets
 - 💾 **Téléchargement universel** avec feedback
 - 💰 **Budget dynamique** basé sur la consommation réelle
 - 📊 **Statistiques détaillées** par utilisateur
 - 🎯 **Transparence totale** sur les coûts
-








Prochaines Étapes Possibles



Améliorations Futures

1. **Prévisualisation en temps réel** pendant la génération
 2. **Historique des coûts** avec graphiques
 3. **Alertes de budget** (emails)
 4. **Système de crédits** prépayés
 5. **Partage direct** vers Instagram
 6. **Galerie publique** des créations
 7. **Export en différentes résolutions**
-



Tests Effectués

-  Compilation TypeScript sans erreurs
-  Build Next.js réussi
-  Migration Prisma appliquée
-  Interface utilisateur responsive
-  Téléchargement fonctionnel

-  Calcul des coûts correct
-  Affichage du budget en temps réel



Notes de Développement

Budget Initial

Le budget initial est fixé à **€20.00** dans :

- `content-generator.tsx` : `const INITIAL_BUDGET = 20.0`
- `user-profile.tsx` : `const INITIAL_BUDGET = 20.0`

Pour modifier le budget initial, changer cette valeur dans les deux fichiers.

Coûts FAL.ai

Les coûts sont basés sur les tarifs approximatifs de FAL.ai :

- Flux Dev (image-to-image) : ~\$0.025 par image
- Luma Dream Machine (video) : ~\$0.05 par vidéo

Ces valeurs peuvent être ajustées dans `app/api/upload/route.ts`.

Base de Données

Le champ `cost` est de type `Float` pour permettre des décimales précises.

Tous les coûts sont stockés en USD (\$) puis convertis en EUR (€) pour l'affichage.

Date de mise à jour : 26 octobre 2025

Version : 2.0

Status :  Déployé et fonctionnel