

Implémentation : Système Complet de Gestion des Bookings et Appels Vidéo

Résumé

Cette fonctionnalité complète transforme l'expérience utilisateur de Callastar avec un système de gestion des appels vidéo de niveau professionnel, offrant une expérience proche de Zoom mais intégrée directement dans la plateforme.

Branche: feature/bookings-calls-experience

Commit: 59f07ee

Status:  Prêt pour review et merge

Fonctionnalités Implémentées

1. Dashboard Utilisateur Enrichi

Le dashboard principal (`/dashboard/user`) a été amélioré avec des cartes cliquables pour une navigation rapide vers :

- **Appels à venir** (`/dashboard/user/calls`)
- **Historique des appels** (`/dashboard/user/history`)
- **Demandes envoyées** (`/dashboard/user/requests`)
- **Notifications** (`/dashboard/user/notifications`)

2. Pages Dédiées

`/dashboard/user/calls` - Appels à Venir

- Liste tous les appels confirmés et à venir
- Bouton “Rejoindre” activé 15 minutes avant l’heure prévue
- Téléchargement de fichier calendrier (.ics)
- Informations détaillées : date, heure, durée, créateur

`/dashboard/user/history` - Historique

- Liste tous les appels passés et terminés
- Possibilité de laisser un avis après l’appel
- Statut de chaque appel (Terminé, Annulé, etc.)
- Badge indiquant si un avis a été laissé

`/dashboard/user/requests` - Demandes

- Vue organisée par statut (En attente, Acceptées, Rejetées)
- Statistiques en haut de page
- Détails complets de chaque demande
- Message envoyé au créateur

`/dashboard/user/notifications` - Notifications

- Notifications non lues en premier

- Badge “Nouveau” pour les notifications non lues
- Bouton “Marquer comme lu” individuel
- Bouton “Tout marquer comme lu”
- Types de notifications colorés
- Liens vers les ressources concernées

3. Système d'Appel Vidéo Complet

Phase 1 : Attente (Plus de 15 min avant l'appel)

- **Écran de compte à rebours** avec timer précis
- Affichage du temps restant avant accès
- Date et heure de l'appel prévu
- Conseils de préparation
- Auto-refresh vers phase pré-appel quand le temps est atteint

Phase 2 : Pré-Appel (15 min avant jusqu'au début)

- **Test des équipements**
- Aperçu vidéo en direct de la caméra
- Test du microphone
- Détection automatique des périphériques
- Messages d'erreur clairs si problème d'accès
- **Informations de l'appel**
- Titre, créateur, durée prévue
- Conseils de préparation
- **Bouton “Rejoindre l'appel”** pour démarrer

Phase 3 : En Appel

- **Interface Daily.co intégrée**
- Vidéo plein écran
- Interface propre et professionnelle
- **Contrôles en bas de l'écran**
- Toggle caméra (on/off)
- Toggle micro (on/off)
- Bouton raccrocher (rouge)
- Tous les contrôles avec feedback visuel
- **Timer en haut de l'écran**
- Temps écoulé depuis le début
- Temps restant jusqu'à la fin prévue
- Overlay semi-transparent, non intrusif
- **Fin automatique**
- L'appel se termine automatiquement quand le temps prévu est écoulé
- Sortie propre de la salle Daily.co

Phase 4 : Post-Appel

- Redirection automatique vers la page de résumé
- Message de transition élégant

4. Page de Résumé Post-Appel (/call/:bookingId/summary)

Affiche un résumé détaillé de l'appel terminé :

Statut de l'appel

- Badge coloré : Terminé normalement / Interrrompu / Absent

Participants

- Créateur et fan avec leurs noms

Détails temporels

- Date et heure prévues
- Durée prévue
- **Heure de début réelle**
- **Heure de fin réelle**
- **Durée effective** (en minutes et secondes)

Chronologie de l'appel

- Liste des événements horodatés
- Entrée pré-appel, début, fin, toggles caméra/micro, etc.
- Utile pour le support/debug

Actions

- Retour au dashboard
- Voir l'historique



API Routes Crées

POST /api/call-logs

Enregistre les événements d'appel côté backend.

Body:

```
{
  "bookingId": "clxxx...",
  "event": "CALL_STARTED",
  "metadata": {
    "timestamp": "2024-12-28T14:30:00.000Z",
    "participants": 2
  },
  "message": "Call started"
}
```

Événements supportés:

- PRE_CALL_ENTERED
- CALL_JOINED
- CALL_STARTED
- CALL_ENDED
- CALL_ERROR
- PARTICIPANT_JOINED
- PARTICIPANT_LEFT

- CAMERA_TOGGLED
- MIC_TOGGLED

Réponse:

```
{
  "success": true,
  "log": { ... }
}
```

GET /api/call-logs?bookingId=xxx

Récupère tous les logs d'un appel spécifique.

Réponse:

```
{
  "logs": [
    {
      "id": "...",
      "type": "CALL_CALL_STARTED",
      "actor": "USER",
      "actorId": "...",
      "message": "Call event: CALL_STARTED",
      "metadata": { ... },
      "createdAt": "..."
    }
  ]
}
```

GET /api/call-summary/:bookingId

Retourne un résumé complet de l'appel avec calculs automatiques.

Réponse:

```
{
  "summary": {
    "booking": { ... },
    "callOffer": {
      "title": "...",
      "scheduledDateTime": "...",
      "scheduledDuration": 30
    },
    "participants": {
      "creator": { "name": "...", "id": "..." },
      "user": { "name": "...", "id": "..." }
    },
    "callDetails": {
      "actualStartTime": "2024-12-28T14:30:00.000Z",
      "actualEndTime": "2024-12-28T15:00:00.000Z",
      "actualDuration": 1800,
      "scheduledDuration": 1800,
      "status": "completed"
    },
    "logs": [ ... ]
  }
}
```



Logs Backend

Système de Logging Complet

Tous les événements d'appel sont enregistrés dans la table `Log` avec :

- **Niveau de log** (INFO, ERROR)
- **Type d'événement** (CALL_)
- Acteur (USER, CREATOR)
- Message descriptif
- Métadonnées structurées (JSON)
- Timestamp automatique*

Événements Enregistrés

1. **Entrée pré-appel** : Quand l'utilisateur accède à l'écran de préparation
2. **Rejoindre** : Quand l'utilisateur clique sur "Rejoindre l'appel"
3. **Début effectif** : Quand la connexion Daily.co est établie
4. **Participant rejoint/quitté** : Suivi des entrées/sorties
5. **Toggles média** : Caméra et micro activés/désactivés
6. **Fin** : Quand l'appel se termine (normal ou interrompu)
7. **Erreurs** : Toute erreur de connexion ou technique

Debug Frontend

Un flag `DEBUG_MODE` est disponible dans `/app/call/[bookingId]/page.tsx` :

- Mettre à `true` pour afficher les logs dans la console
- Mettre à `false` en production (défaut)
- Les logs backend sont toujours enregistrés



Expérience Utilisateur

Parité Fan/Créateur

L'expérience d'appel est **identique** pour le fan et le créateur :

- Même écran pré-appel avec test caméra/micro
- Même interface pendant l'appel
- Mêmes contrôles (caméra, micro, raccrocher)
- Même timer et informations
- Même page de résumé post-appel
- Logs pour les deux parties

Points Forts

1. **Compte à rebours précis** : Timer en temps réel jusqu'à l'accès
2. **Test avant l'appel** : Évite les problèmes techniques une fois connecté
3. **Interface professionnelle** : Comparable à Zoom, Teams, etc.
4. **Timer visible** : Temps écoulé et restant toujours affichés
5. **Fin automatique** : Respect strict de la durée prévue
6. **Résumé détaillé** : Transparence totale sur la durée effective

7. Logs complets : Support et debug facilités

Sécurité et Validation

Contrôles d'Accès

- Vérification de l'authentification à chaque étape
- Validation que l'utilisateur est bien participant (fan ou créateur)
- Respect de la fenêtre d'accès (15 min avant → 24h après)
- Tokens Daily.co uniques et éphémères
- Logs avec identification de l'acteur

Gestion des Erreurs

- Messages d'erreur clairs et en français
 - Toasts pour les actions utilisateur
 - Gestion des erreurs Daily.co
 - Cleanup automatique des ressources (streams, call frame)
 - Fallback sur erreur
-

Testing Recommandé

1. Tests Manuels

Test du Compte à Rebours

1. Créer un appel dans plus de 20 minutes
2. Accéder à `/call/:bookingId`
3. Vérifier l'écran de compte à rebours
4. Vérifier que le timer se met à jour chaque seconde
5. Attendre (ou modifier manuellement la date) jusqu'à 15 min avant
6. Vérifier le passage automatique en mode pré-appel

Test Pré-Appel

1. Créer un appel dans moins de 15 minutes
2. Accéder à `/call/:bookingId`
3. Cliquer sur "Tester caméra et micro"
4. Vérifier l'aperçu vidéo
5. Vérifier les permissions navigateur
6. Cliquer sur "Rejoindre l'appel"

Test En Appel

1. Rejoindre l'appel
2. Vérifier l'affichage du timer
3. Toggle caméra on/off → vérifier le feedback visuel
4. Toggle micro on/off → vérifier le feedback visuel
5. Vérifier que le temps restant décrémente

6. Laisser l'appel se terminer automatiquement ou cliquer sur raccrocher
7. Vérifier la redirection vers le résumé

Test Post-Appel

1. Accéder à `/call/:bookingId/summary`
2. Vérifier toutes les informations
3. Vérifier les durées (effective vs prévue)
4. Vérifier la chronologie des événements
5. Tester les boutons de navigation

Test Dashboard

1. Accéder à `/dashboard/user`
2. Cliquer sur chaque carte (Appels, Historique, Demandes, Notifications)
3. Vérifier que les compteurs sont corrects
4. Vérifier la navigation retour

2. Tests avec Deux Utilisateurs

1. Créateur et fan rejoignent le même appel
2. Vérifier que les deux voient le même timer
3. Vérifier la détection mutuelle dans Daily.co
4. L'un toggle sa caméra → vérifier que l'autre le voit
5. Vérifier que les logs enregistrent les deux participants

3. Tests des Logs

1. Après un appel, accéder à la console admin (si disponible)
 2. Rechercher les logs de type `CALL_*`
 3. Vérifier la présence de tous les événements
 4. Vérifier les métadonnées (timestamps, actorId, etc.)
-



Dépendances

Toutes les dépendances nécessaires sont **déjà installées** :

- `@daily-co/daily-js` : `^0.85.0`
- `date-fns` : `^3.6.0`
- `next` : (version du projet)
- Prisma Client
- Composants UI (shadcn/ui)

Aucune installation supplémentaire requise !



Déploiement

Étapes de Merge

1. **Review du code** sur GitHub
2. **Tests en staging** recommandés

3. **Merge** dans la branche principale
4. **Migration Prisma** (si schema modifié - aucune modification dans cette feature)
5. **Deploy** sur production

Points d'Attention

- ! Vérifier que la variable `DAILY_API_KEY` est configurée
- ! Tester avec de vrais tokens Daily.co
- ! Vérifier les permissions caméra/micro dans les navigateurs
- ! Tester sur mobile et desktop
- ! Vérifier que les webhooks Stripe continuent de fonctionner

Rollback

Si besoin de rollback :

```
git revert 59f07ee
```

Ou simplement merger la branche précédente.



Métriques et Monitoring

Métriques à Suivre

1. **Taux d'adoption** : % d'utilisateurs utilisant les nouvelles pages
2. **Taux de compléction** : % d'appels terminés normalement
3. **Durée moyenne effective** vs durée prévue
4. **Erreurs d'appel** : Nombre et types d'erreurs dans les logs
5. **Temps moyen en pré-appel** : Combien de temps les utilisateurs testent

Logs à Monitorer

- Erreurs de type `CALL_ERROR`
 - Appels avec statut `interrupted` ou `no-show`
 - Problèmes d'accès caméra/micro
 - Échecs de connexion Daily.co
-

🎓 Guide pour les Développeurs

Architecture

```

app/
  dashboard/user/
    calls/          # Appels à venir
    history/        # Historique
    requests/       # Demandes
    notifications/ # Notifications
    page.tsx        # Dashboard principal (modifié)
    call/[bookingId]/
      page.tsx      # Page d'appel complète (réécrite)
    summary/
      page.tsx      # Résumé post-appel
  api/
    call-logs/
      route.ts      # Endpoint pour les logs
    call-summary/
      [bookingId]/
        route.ts    # Endpoint pour le résumé

```

Cycle de Vie d'un Appel

1. Utilisateur crée un booking
↓
2. Webhook Stripe confirme le paiement
↓
3. Booking.status = CONFIRMED
↓
4. [T-15 min] Utilisateur accède à /call/:bookingId
↓
5. Phase PRE_CALL : Test caméra/micro
↓
6. Utilisateur clique "Rejoindre"
↓
7. API /daily/get-token génère un token
↓
8. Daily.co call frame créé
↓
9. Phase IN_CALL : Appel actif
↓
10. [T+duration] Fin automatique OU utilisateur raccroche
↓
11. Log CALL_ENDED enregistré
↓
12. Redirection vers /call/:bookingId/summary
↓
13. Affichage du résumé avec durée effective

Personnalisation

Changer le délai d'accès (actuellement 15 min)

```

// Dans app/call/[bookingId]/page.tsx
const fifteenMinutesBefore = callTime - 15 * 60 * 1000;
// Modifier en : 10 * 60 * 1000 pour 10 minutes

```

Activer les logs debug en production

```
// Dans app/call/[bookingId]/page.tsx
const DEBUG_MODE = true; // Mettre à true
```

Modifier les couleurs des notifications

```
// Dans app/dashboard/user/notifications/page.tsx
const getNotificationTypeColor = (type: string) => {
  // Personnaliser les couleurs ici
}
```

Checklist Finale

- [x] Branche créée : feature/bookings-calls-experience
- [x] Dashboard utilisateur enrichi avec navigation
- [x] Pages dédiées : Appels, Historique, Demandes, Notifications
- [x] Système de notifications in-app fonctionnel
- [x] Page d'appel complète avec 4 phases
- [x] Compte à rebours avant accès
- [x] Test caméra/micro avant appel
- [x] Intégration Daily.co avec contrôles
- [x] Timer avec temps écoulé et restant
- [x] Fin automatique de l'appel
- [x] Page de résumé post-appel détaillée
- [x] API routes pour les logs
- [x] API route pour le résumé
- [x] Logs backend complets
- [x] Logs debug frontend (avec flag)
- [x] Parité fan/créateur
- [x] Gestion des erreurs
- [x] Cleanup des ressources
- [x] TypeScript type safety
- [x] Code committé avec message descriptif
- [x] Code pushé sur GitHub
- [x] Documentation complète

Conclusion

Cette implémentation offre une **expérience d'appel vidéo complète et professionnelle** pour Cal-lastar, rivalisant avec les solutions établies tout en étant parfaitement intégrée à l'écosystème existant.

Prêt pour review et merge !

Pour toute question ou assistance sur cette feature :

- Consulter ce document
- Vérifier les logs backend
- Activer DEBUG_MODE pour le frontend
- Examiner le code source (bien commenté)

Happy calling! 🎥✨