👥 Module de Gestion des Clients - Yoozak

Vue d'ensemble

Le module de gestion des clients permet de visualiser, rechercher et gérer tous les clients de la plateforme Yoozak avec leurs commandes associées.

🚀 Fonctionnalités

Interface Web

- **Liste des clients** avec recherche avancée et filtres
- **Page de détail client** avec historique complet des commandes
- **Statistiques en temps réel** (nombre de commandes, montant total, etc.)
- **Activation/désactivation** des comptes clients
- **Pagination** pour une navigation fluide

Administration Django

- **Interface admin améliorée** avec statistiques intégrées
- **Actions en lot** pour activer/désactiver plusieurs clients
- **Liens directs** vers les commandes d'un client
- **Recherche multi-critères** avancée

Commandes de Gestion

1. Liaison automatique des commandes

```
# Simulation (sans modification)
python manage.py link_orders_to_clients --dry-run

# Exécution réelle
python manage.py link_orders_to_clients

# Forcer la mise à jour de toutes les commandes
python manage.py link_orders_to_clients --force
```

2. Création de clients de test

```
# Créer 10 clients de test
python manage.py create_test_clients

# Créer 50 clients de test
python manage.py create_test_clients --count 50
```

```
# Supprimer les clients existants et en créer de nouveaux
python manage.py create_test_clients --clear --count 20
```

📊 Statistiques Disponibles

Vue Liste

- Total des clients
- Clients actifs/inactifs
- Nouveaux clients (aujourd'hui/7 jours)
- Nombre de commandes par client
- Montant total dépensé par client

Vue Détail Client

- Total des commandes
- Commandes confirmées/en attente/annulées
- Montant total dépensé
- Valeur panier moyen
- Date de dernière commande

🔗 Intégration avec les Commandes

Le système crée automatiquement des liens entre les commandes et les clients basés sur :

1. **Email** (priorité la plus haute)
2. **Numéro de téléphone** (correspondance exacte ou partielle)
3. **Nom complet** (correspondance exacte ou similaire)

Algorithme de Correspondance

```
# 1. Recherche par email exact
client = Client.objects.filter(user__email__iexact=order.email).first()

# 2. Recherche par téléphone nettoyé
clean_phone = clean_phone_number(order.phone)
client = Client.objects.filter(phone__icontains=clean_phone).first()

# 3. Recherche par nom avec similarité
if names_are_similar(order.client_name, client.full_name, threshold=0.7):
    return client
```

🎯 URLs et Navigation

URLs Principales

- `/client/` - Liste des clients

- `/client/{id}/` - Détail d'un client
- `/client/api/stats/` - API statistiques AJAX
- `/client/api/{id}/orders/` - API commandes client AJAX

Navigation

Le menu "Gestion des clients" est accessible dans la sidebar pour les administrateurs.

📱 Interface Responsive

L'interface est entièrement responsive et optimisée pour :

- **Desktop** - Tableaux complets avec toutes les colonnes
- **Tablet** - Colonnes adaptées avec information essentielle
- **Mobile** - Cartes compactes avec navigation tactile

🔍 Recherche et Filtres

Critères de Recherche

- Nom complet
- Email
- Nom d'utilisateur
- Numéro de téléphone
- Adresse
- Nom d'entreprise

Filtres Disponibles

- **Statut** : Tous, Actif, Inactif
- **Période** : Toutes, Aujourd'hui, 7 jours, 30 jours
- **Tri** : Date création, Nom, etc.

🛠️ Modèles de Données

Client

```
class Client(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE)
    phone = models.CharField(max_length=15, blank=True)
    address = models.TextField(blank=True)
    company_name = models.CharField(max_length=100, blank=True)
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)
    is_active = models.BooleanField(default=True)
```

Relation avec Order

```python
class Order(models.Model):
    # ... autres champs ...
    client = models.ForeignKey(
        'client.Client',
        on_delete=models.SET_NULL,
        null=True,
        blank=True,
        related_name='client_orders'
    )
```

Sécurité

- **Authentification requise** pour toutes les vues
- **Vérification des droits** administrateur
- **Protection CSRF** sur toutes les actions
- **Validation des données** côté serveur et client

Performance

Optimisations Implémentées

- **Select Related** pour éviter les requêtes N+1
- **Annotations** pour calculer les statistiques en base
- **Pagination** pour limiter les résultats
- **Index de base de données** sur les champs de recherche

Requêtes Optimisées

```python
# Exemple de requête optimisée
clients = Client.objects.select_related('user').annotate(
    total_orders=Count('client_orders'),
    total_spent=Sum('client_orders__price'),
    avg_order_value=Avg('client_orders__price')
)
```

API AJAX

Endpoints Disponibles

- `GET /client/api/stats/` - Statistiques générales
- `GET /client/api/{id}/orders/` - Commandes d'un client
- `POST /client/api/{id}/toggle-status/` - Changer le statut

Exemple d'utilisation

```
// Récupérer les statistiques
fetch('/client/api/stats/')
    .then(response => response.json())
    .then(data => console.log(data));

// Changer le statut d'un client
fetch(`/client/api/${clientId}/toggle-status/`, {
    method: 'POST',
    headers: {
        'X-CSRFToken': csrfToken,
        'Content-Type': 'application/json',
    },
})
.then(response => response.json())
.then(data => {
    if (data.success) {
        location.reload();
    }
});
```

🔧 Maintenance

Tâches Régulières

1. **Liaison des nouvelles commandes** avec `link_orders_to_clients`
2. **Nettoyage des données** clients inactifs
3. **Mise à jour des statistiques** si nécessaire

Surveillance

- Surveiller les logs de liaison automatique
- Vérifier la cohérence des données client-commande
- Contrôler les performances des requêtes

🆕 Améliorations Futures

- **Export des données** clients en CSV/Excel
- **Segmentation avancée** des clients (VIP, occasionnels, etc.)
- **Notifications** pour nouveaux clients
- **Intégration avec le CRM** externe
- **Historique des modifications** client
- **API REST complète** pour intégrations externes

Documentation mise à jour le {{ date.today }}