Helpdesk App

Documentation Technique

Présenté par :
LUTHOMO IBELE BLESSING
NKURA KIKAKALA WINNER
NGANDU KASHINDA FRANCK
NAWEZI TUBALA EULOGIA
WASSO KISEMBE VICTORINA

Version : 1.0 (Finale)
Date : Février 2026

Description du Projet

Une solution de gestion de tickets de support technique multi-plateforme pour l'ONT

Conçue pour optimiser l'assistance utilisateur et la résolution d'incidents en temps réel via une interface moderne et sécurisée.

Cette application implémente un système complet de gestion des incidents technique à l'ONT (Office National du Tourisme). Elle permet aux employés par département de soumettre des tickets d'assistance et offre aux administrateurs ainsi qu'aux techniciens des outils de suivi performants pour garantir la continuité du service et une gestion efficace des flux de travail.

Table des matières

1 Structure du Projet

L'organisation du projet suit une architecture modulaire et maintenable :

Dossier/Fichier	Description
android/	Configuration native Android
assets/	Ressources visuelles et images
lib/	Code source de l'application
controllers/	Logique métier et gestion d'état
models/	Modèles de données
services/	Services Firebase et notifications
utils/	Thèmes, couleurs et constantes
views/	Interfaces utilisateur (Écrans)
admin/	Dashboard et outils administrateur
tech/	Interface dédiée aux techniciens
widgets/	Composants UI réutilisables
firebase_options.dart	Configuration Firebase
main.dart	Point d'entrée de l'application
firebase.json	Configuration Firebase
pubspec.yaml	Dépendances et métadonnées
README.md	Documentation du projet
Documentation_technique.pdf	Ce document (version PDF)

1.1 Organisation des vues

- **views/admin/** : Dashboard administrateur, gestion utilisateurs, statistiques
- **views/tech/** : Interface technicien, tickets assignés
- **Écrans principaux** :
  - add_ticket_screen.dart
  - forgot_password_screen.dart
  - help_center_view.dart
  - home_screen.dart
  - login_screen.dart
  - notifications_screen.dart
  - profile_screen.dart
  - register_screen.dart

2 Analyse des Besoins

2.1 Besoins Fonctionnels

L'application répond aux besoins spécifiques de l'ONT à travers trois profils distincts :

- ✔ **Administrateur :** Gérer les employés et techniciens, assigner les tickets, débloquer les situations critiques, consulter les statistiques et exporter les rapports.
- ✔ **Employé :** Créer des tickets d'assistance (par département et nature), suivre l'évolution en temps réel et recevoir des notifications.
- ✔ **Technicien :** Consulter les tickets assignés, mettre à jour le statut (En cours, Résolu) ou signaler un blocage.

2.2 Besoins Non Fonctionnels

Sécurité : Accès restreint selon les rôles via Firebase Security Rules.

Disponibilité : Système accessible 24h/24 grâce à l'infrastructure Cloud.

Réactivité : Mise à jour instantanée des tickets via les flux Firestore.

3 Architecture Globale

L'application adopte une architecture technique à trois couches (Client, Backend Services et Firebase Cloud). Cette structure permet une synchronisation bidirectionnelle entre les interfaces Flutter et les services Cloud.

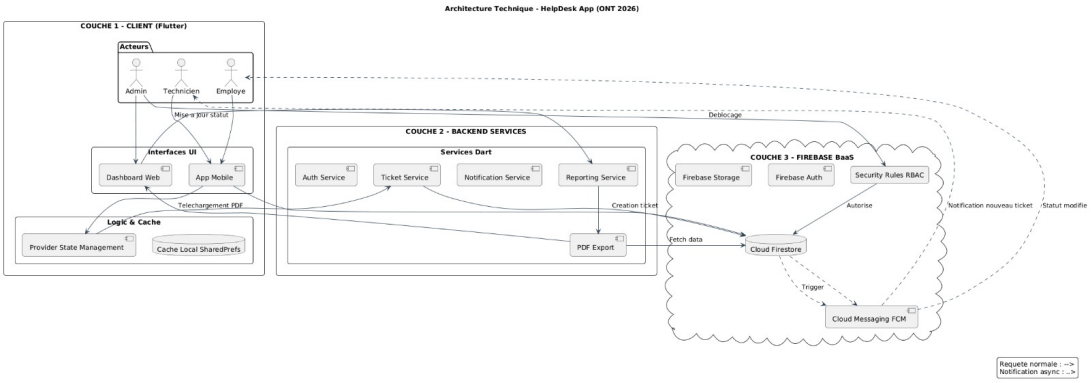

FIGURE 1 – Architecture technique à 3 couches (Flutter / Firebase)

3.1 Intégration de l'API HubSpot pour l'auto-assistance

En complément de la base de connaissances spécifique d'Huptspot, l'application tire parti de l'API HubSpot pour accéder à une base de données élargie de cas d'incidents courants. L'objectif est de fournir des solutions immédiates à l'utilisateur lors de la création de son ticket, réduisant ainsi le besoin de déclarer un nouvel incident pour des problèmes déjà documentés.

3.1.1 Configuration de l'Application Privée HubSpot

Pour utiliser l'API, il est nécessaire de créer une **application privée** dans votre compte HubSpot. Celle-ci fournira un jeton d'accès unique pour authentifier les requêtes.

1. Se connecter à son compte HubSpot.
2. Aller dans `Réglages` > `Intégrations` > `Applications privées`.
3. Créer une nouvelle application privée.
4. Attribuer les **scopes** (permissions) nécessaires, par exemple pour lire les tickets et les articles de la base de connaissances.
5. Générer et copier le **jeton d'accès** (à stocker de manière sécurisée, par exemple dans Firebase Remote Config).

3.1.2 Fonctionnalités clés via l'API

- **Recherche dans la base de connaissances** : Interroger l'API pour trouver des articles d'aide correspondant aux mots-clés de la description de l'incident.
- **Suggestion de solutions** : Présenter à l'utilisateur, avant même la création du ticket, une liste de solutions potentielles issues des articles trouvés.

- **Création automatique de ticket** : Si aucune solution pertinente n'est trouvée, l'API peut être utilisée pour créer automatiquement un ticket d'assistance dans le CRM HubSpot, synchronisant ainsi les demandes.

3.1.3 Exemple de flux utilisateur amélioré par HubSpot

1. L'utilisateur commence à taper une description de son problème.
2. En arrière-plan, une requête est envoyée à l'API HubSpot avec les mots-clés saisis.
3. L'API retourne les articles d'aide les plus pertinents.
4. L'application affiche ces articles à l'utilisateur.
5. **Si l'utilisateur trouve une solution**, il peut résoudre son problème sans créer de ticket.
6. **Sinon**, il continue la création du ticket, qui sera alors créé à la fois dans Firestore et dans HubSpot.

Note : Cette intégration transforme l'application d'un simple outil de ticketing en une véritable plateforme d'auto-assistance, réduisant la charge des techniciens et améliorant la satisfaction des utilisateurs.

4 Modélisation du Système

4.1 Diagramme de Cas d'Utilisation

Ce diagramme détaille les interactions entre les acteurs et les fonctionnalités du système.

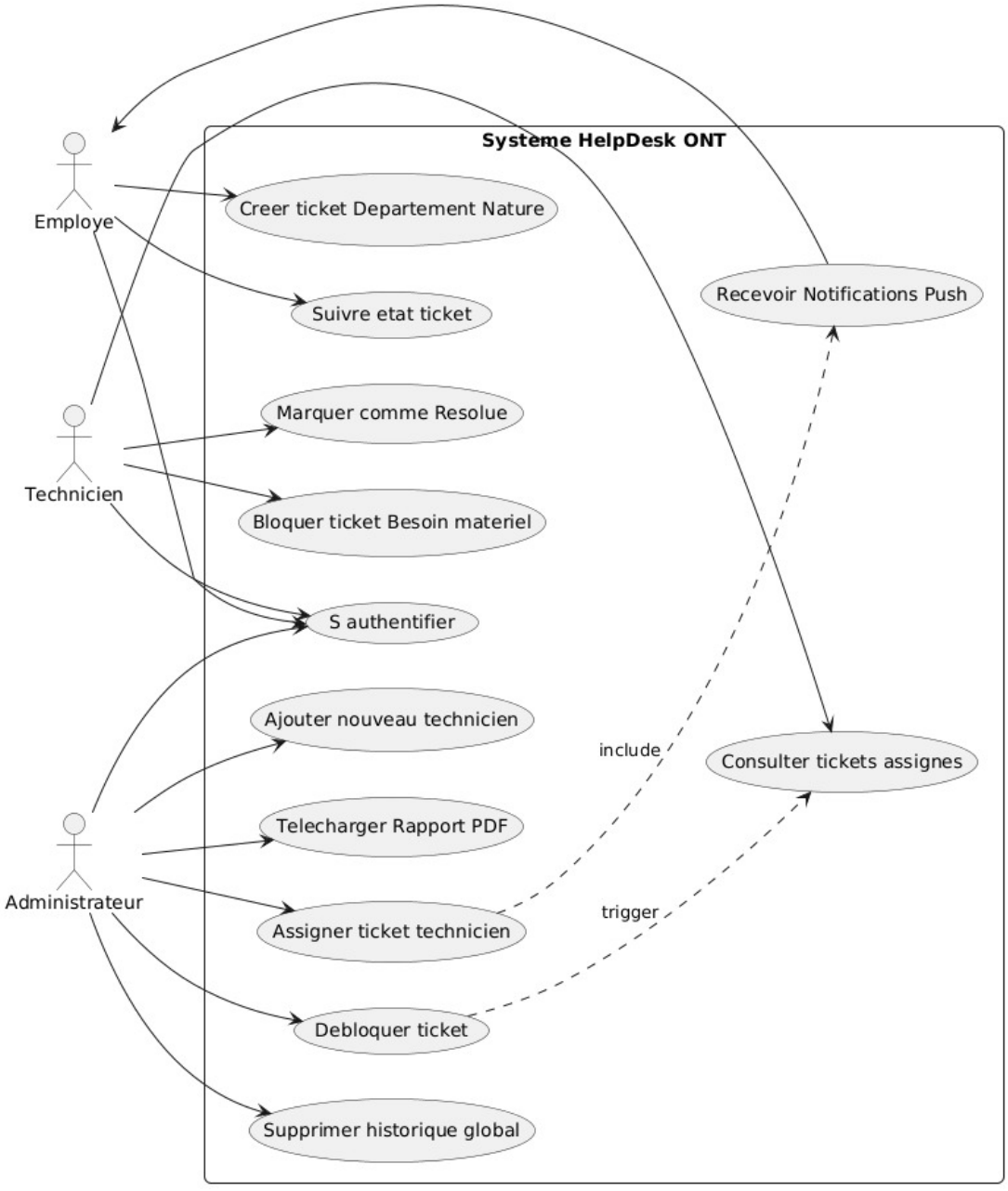

FIGURE 2 – Diagramme de Cas d'Utilisation

4.2 Diagramme de Classes

Représentation de la structure des données (Tickets, Utilisateurs, Notifications) stockées dans Firestore.

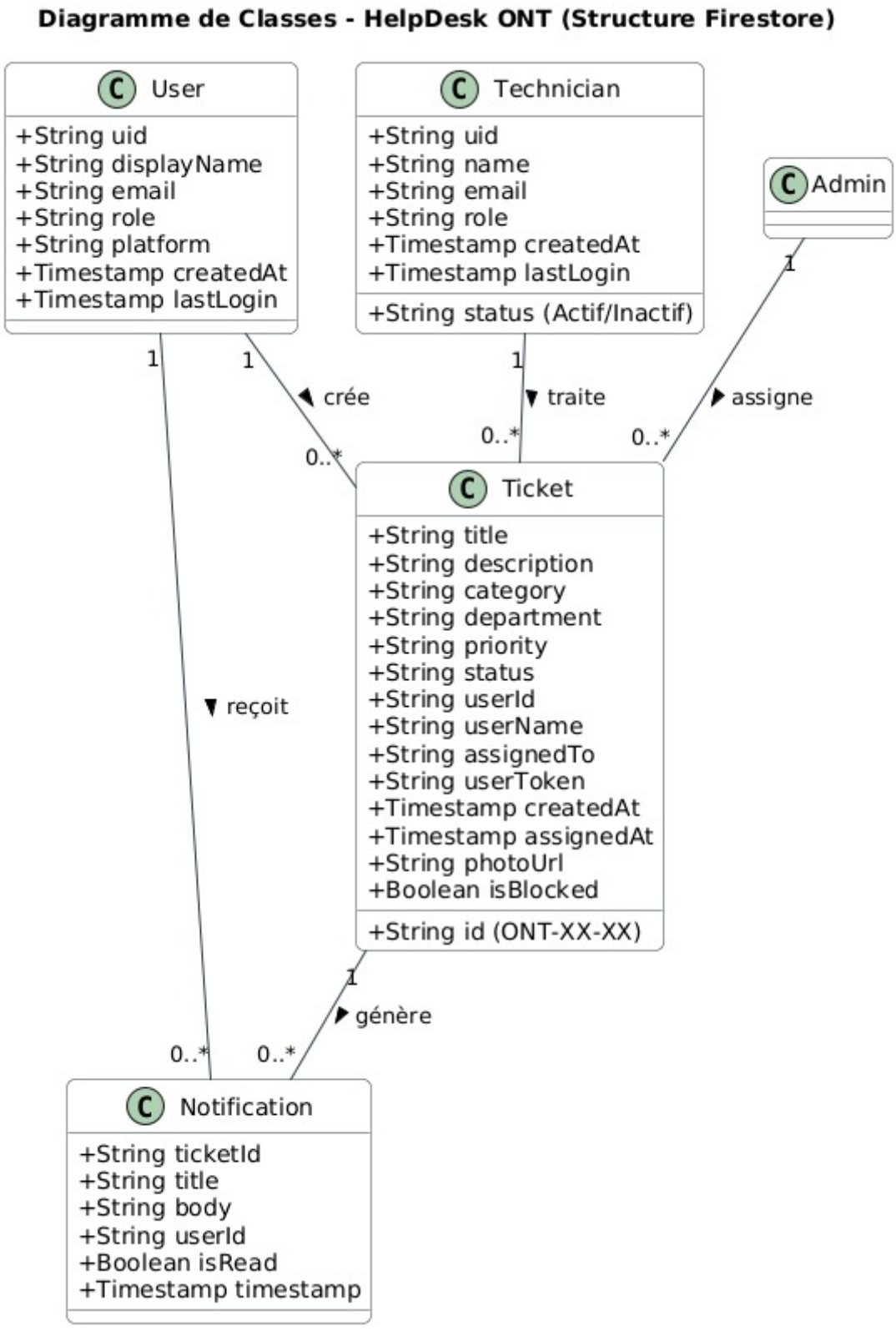

FIGURE 3 – Diagramme de Classes

4.3 Diagramme de Séquence

Illustration du flux de travail chronologique (Création → Assignation → Résolution).

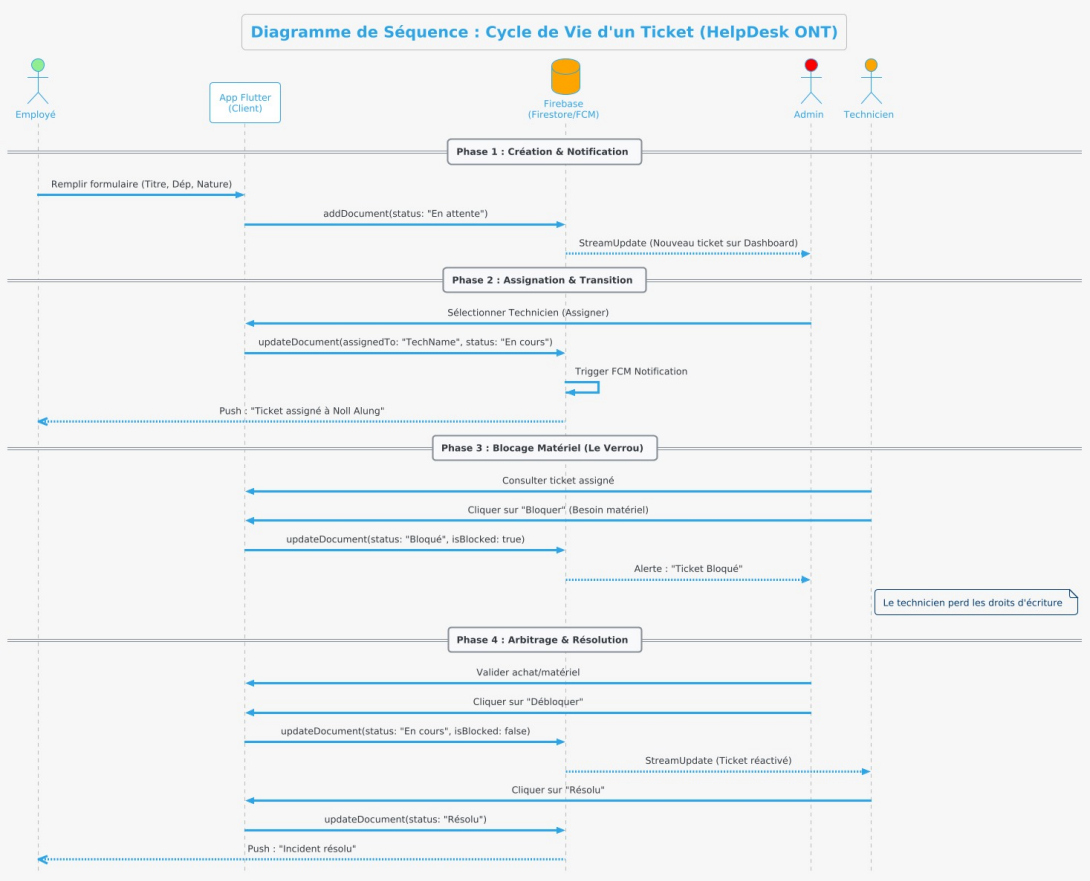

FIGURE 4 – Diagramme de Séquence

5 Technologies Utilisées

Le projet s'appuie sur un stack technologique moderne et robuste pour garantir performance et scalabilité :

Composant	Technologie	Utilisation
Framework	Flutter	Développement multi-plateforme (iOS, Android, Web, Desktop)
Langage	Dart	Programmation orientée objet, typage fort
Backend	Firebase	Solution backend complète
Authentification	Firebase Authentication	Gestion sécurisée des comptes (Email, Google, GitHub)
Base de données	Cloud Firestore	Base de données NoSQL en temps réel
Notifications	Firebase Cloud Messaging	Notifications push
Stockage	Firebase Storage	Images et pièces jointes (Upload via caméra/galerie)
Gestion d'état	Provider	Gestion réactive des données
Stockage local	Shared Preferences	Persistance des sessions utilisateur
Design	Material 3	Interface moderne et adaptative

Note : Cette combinaison technologique permet un développement rapide tout en garantissant une excellente expérience utilisateur sur toutes les plateformes.

6 Installation et Configuration

6.1 Prérequis

- Flutter SDK (Version stable)
- Dart SDK
- Un compte Firebase actif
- Git

6.2 Étapes d'installation

6.2.1 1. Cloner le repository

```
git clone https://github.com/votre-pseudo/helpdesk_app.git
cd helpdesk_app
```

6.2.2 2. Installer les dépendances

```
flutter pub get
```

6.2.3 3. Configuration Firebase

a. Créer un projet : Rendez-vous sur la Console Firebase et créez un nouveau projet.
b. Activer l'Authentification : Dans la section Authentication, activez les fournisseurs suivants :

— Email/Mot de passe
— Google
— GitHub

c. Base de données : Initialisez une base de données Cloud Firestore.
d. Configuration locale : Utilisez la CLI Firebase pour générer automatiquement le fichier de configuration :

```
flutterfire configure
```

6.3 Lancement de l'application

```
# Sur un appareil mobile
flutter run

# Sur navigateur Web
flutter run -d chrome

# Sur Desktop
flutter run -d windows
```

7 Architecture Firebase

7.1 Services Firebase utilisés

Service	Utilisation
Firebase Authentication	Gestion des comptes utilisateurs (Email, Google, GitHub)
Cloud Firestore	Stockage des tickets, commentaires, utilisateurs et notifications
Firebase Cloud Messaging	Envoi de notifications push en temps réel
Firebase Storage	Stockage des images et pièces jointes (Upload via caméra/galerie)

7.2 Structure des collections Firestore

7.2.1 Collection : users

Champ	Type	Description
createdAt	timestamp	Date de création du compte (ex: 16 février 2026 à 13:15:32)
displayName	string	Nom affiché de l'utilisateur
email	string	Adresse email unique
lastLogin	timestamp	Dernière connexion (ex: 16 février 2026 à 13:16:14)
platform	string	Plateforme utilisée (mobile, web, desktop)
role	string	Rôle (user, tech, admin)
uid	string	Identifiant unique Firebase Auth
status	string	Statut du compte (Actif, Inactif)

7.2.2 Collection : tickets

Champ	Type	Description
category	string	Catégorie du problème (ex: Téléphonie)
createdAt	timestamp	Date de création (ex: 18 février 2026 à 20:45:42)
department	string	Département concerné (ex: Autre)
description	string	Description détaillée du problème
photoUrl	string/null	URL des photos jointes stockées dans Firebase Storage (null si aucune)
priority	string	Niveau de priorité
status	string	Statut (Ouvert, En cours, Résolu, Bloqué)
title	string	Titre du ticket
ticketId	string	Identifiant unique formaté (ex: ONT-26-75)
userId	string	ID de l'utilisateur créateur
userName	string	Nom de l'utilisateur créateur
userToken	string/null	Token FCM pour notifications

7.2.3 Collection : notifications

Champ	Type	Description
body	string	Corps du message (ex: "Votre ticket 'test test test' a été assigné au technicien Vedie.")
isRead	boolean	État de lecture (true = lu, false = non lu)
ticketId	string	Identifiant du ticket concerné (ex: ONT-26-75)
timestamp	timestamp	Date et heure d'envoi
title	string	Titre de la notification (ex: Ticket Assigné)
userId	string	ID de l'utilisateur destinataire

7.3 Rôles utilisateurs

Rôle	**Interface**	**Permissions**
user	Home, Mes tickets	Créer ticket, consulter ses tickets, commenter
tech	Tech Dashboard	Voir tickets assignés, changer statut, commenter
admin	Admin Dashboard	Gérer utilisateurs, statistiques, tous les tickets

8 Fonctionnalités Principales

8.1 Système d'Authentification

- Connexion par Email/Mot de passe
- Authentification Google et GitHub
- Gestion des rôles (user, tech, admin)
- Récupération de mot de passe

8.2 Gestion des Tickets

Création de tickets avec catégorie et priorité

Upload de photos : Capture via caméra ou sélection depuis la galerie

Suivi en temps réel des statuts

Notifications push instantanées

8.3 Notifications

- Alertes pour assignation de ticket
- Notifications de changement de statut
- Marquage lecture/non lecture

9 Présentation de l'Interface

Cette section présente les principaux écrans de l'application Helpdesk App, illustrant l'expérience utilisateur pour chaque profil.

9.1 Interface Employé

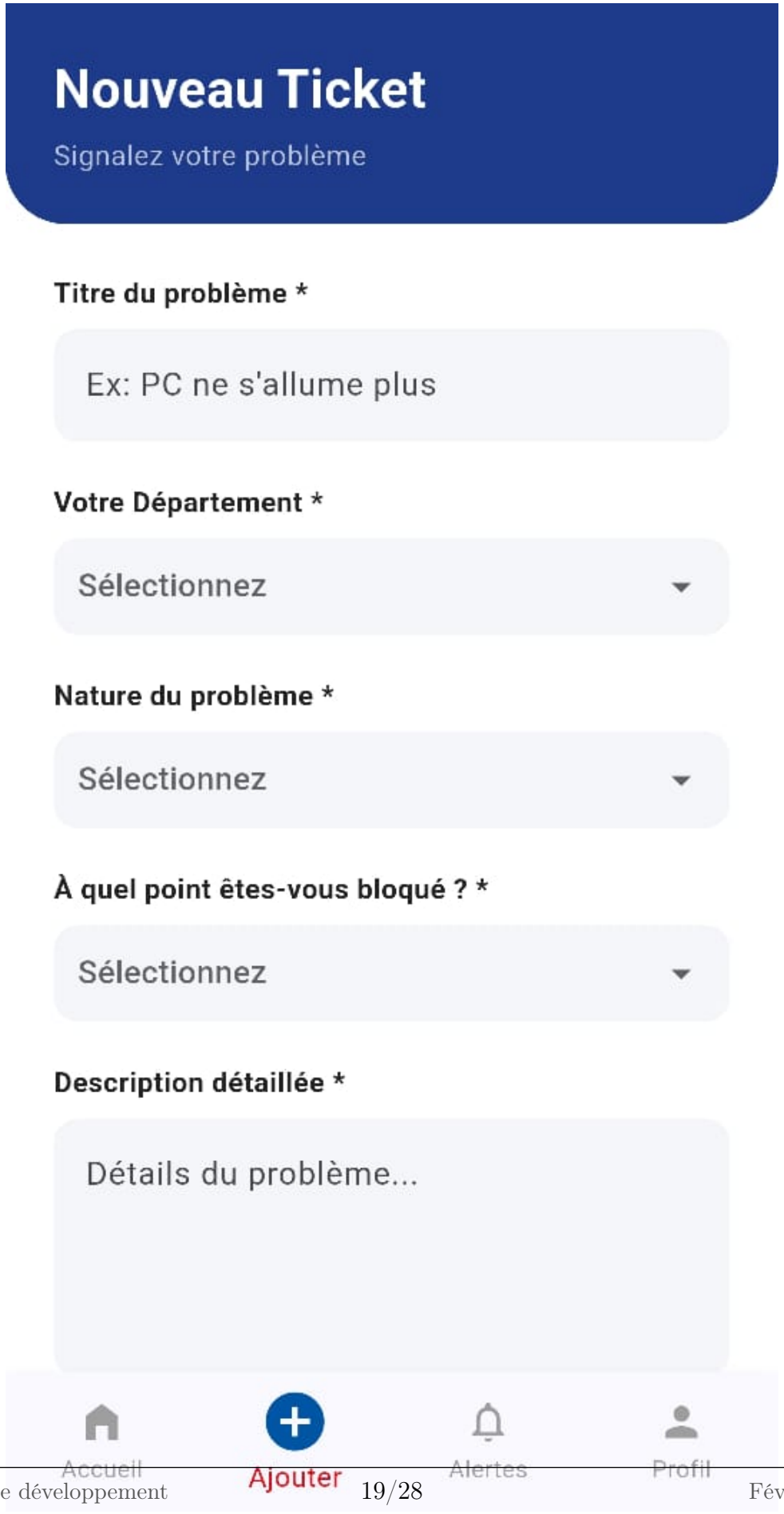

FIGURE 6 – Page de création de ticket - Formulaire de soumission d'incident avec options d'upload de photos

9.2 Interface Technicien

9.3 Interface Administrateur

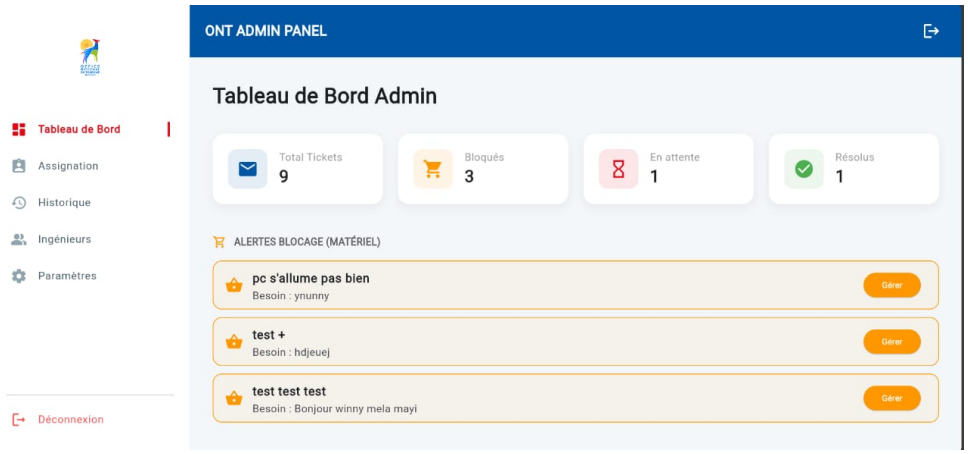

FIGURE 8 – Dashboard administrateur - Vue d'ensemble et statistiques globales

Note : Ces captures d'écran illustrent l'interface responsive de l'application, accessible aussi bien sur mobile que sur desktop.

10 Contraintes Techniques : Firebase Storage

10.1 Fonctionnalité d'upload de photos

Lors de la création d'un ticket, l'employé a la possibilité d'ajouter des photos pour illustrer le problème rencontré. Cette fonctionnalité a été entièrement développée dans l'interface utilisateur :

- **Prise de photo directe :** Accès à l'appareil photo du mobile
- **Sélection depuis la galerie :** Choix de photos existantes
- **Aperçu avant envoi :** Visualisation des photos sélectionnées

10.2 Limitation actuelle

⚠ **Attention :** La fonctionnalité d'upload de photos est développée dans l'interface utilisateur mais n'est pas fonctionnelle en backend car Firebase Storage nécessite un passage au forfait payant pour être activé sur ce projet.

10.3 Preuve de l'implémentation

Les captures d'écran ci-dessous montrent l'interface fonctionnelle de gestion des photos, démontrant que le développement de cette fonctionnalité a bien été réalisé :

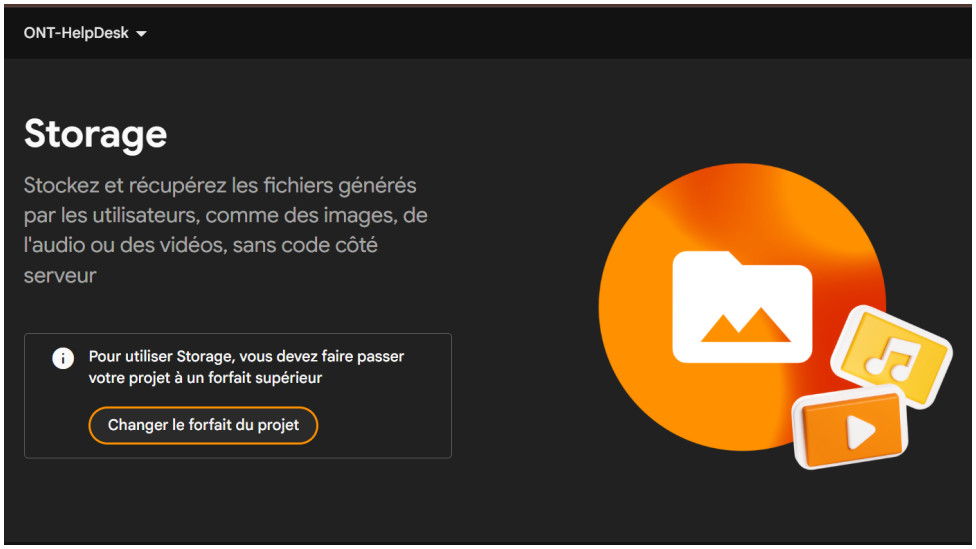

FIGURE 9 – Preuve Storage

FIGURE 10 – Preuve Storage

10.4 Structure de données préparée

Le champ `photoUrl` est déjà présent dans la collection `tickets` de Firestore pour accueillir les URLs des photos une fois le service Storage activé :

```
photoUrl: string/null  // URL des photos stockées dans Firebase Storage
```

11 Sécurité

11.1 Mesures de sécurité

Authentification Firebase sécurisée

Règles Firestore basées sur les rôles

Communications chiffrées (HTTPS)

Sessions sécurisées

12 Tests

12.1 Stratégie de test

— Tests unitaires des contrôleurs
— Tests d'intégration des flux critiques
— Tests manuels de recette

12.2 Scénarios testés

— Création de ticket avec sélection de photos (interface uniquement)
— Réception de notifications push
— Changement de statut en temps réel
— Gestion des rôles et permissions

13 Déploiement

13.1 Environnements

- Développement : Émulateurs et appareils physiques
- Production : Stores (Google Play, App Store) et Web

13.2 Commandes de build

```
# Android
flutter build apk --release

# iOS
flutter build ios --release

# Web
flutter build web --release
```

14 Conclusion

La réalisation de l'application Helpdesk App pour l'Office National du Tourisme (ONT) constitue une avancée significative dans la modernisation de la gestion des incidents techniques au sein de l'institution.

Ce projet a permis de développer une solution complète et adaptée aux besoins spécifiques de l'ONT, offrant :

- Une plateforme unifiée accessible sur mobile, web et desktop
- Un système de ticketing en temps réel avec notifications push
- Une interface de création de tickets avec upload de photos (interface développée, en attente d'activation du service Storage)
- Une gestion efficace des rôles et des permissions
- Une architecture robuste basée sur Firebase garantissant disponibilité et scalabilité

L'application répond pleinement aux objectifs fixés : optimiser le flux de traitement des incidents, améliorer la communication entre les employés et les techniciens, et assurer un suivi transparent de chaque demande d'assistance.