

Projet D’architecture Cloud

SUPINFO |

Membres du groupe

Achraf Elharfi && abdoul waris konate

Sommaire ou table des matières

Table des matières

[**Contexte du projet** 2](#_Toc196133366)

[**Schéma d’architecture micro services** 2](#_Toc196133367)

[**1.** **User Domain** 2](#_Toc196133368)

[**2.** **Flight Domain** 3](#_Toc196133369)

[**Vu sur le cloud Azure** 0](#_Toc196133370)

[**Service Azure** 0](#_Toc196133371)

[**System Architecture** 0](#_Toc196133372)

[**Data Flow Highlights** 0](#_Toc196133373)

[**Monitoring et sécurité** 1](#_Toc196133374)

[**Gestion de la haute disponibilité** 1](#_Toc196133375)

[**Conformité RGPD** 2](#_Toc196133376)

# **Contexte du projet**

Dans le cadre d’une adoption du cloud afin de manager son système FlyAwayAirlines nous a contacté pour une refonte de son système de réservation de billet d’avion et une adoption du cloud afin de desservir ses millions d’utilisateurs le tout pour assurer une haute disponibilité une plateforme toujours opérationnelle. En effet, FlyAway Airlines a rencontré de graves problèmes avec leur architecture basée sur des microservices récemment déployée, notamment :

* Une **faible scalabilité** et des **temps de réponse élevés** en cas de forte charge
* **Des pannes fréquentes des services** et un **manque de tolérance aux pannes**
* Une **incohérence des données** entre les différents services
* Une **observabilité limitée** et un **manque de visibilité opérationnelle**
* Un **soutien insuffisant à la conformité** avec le **RGPD** (Règlement Général sur la Protection des Données) et la **norme PCI-DSS**

## **Schéma d’architecture micro services**

Le schéma ci-dessous représente l’architecture micro services de la solution ils ont été regroupés par domaine pour faciliter la lecture :

### **User Domain**

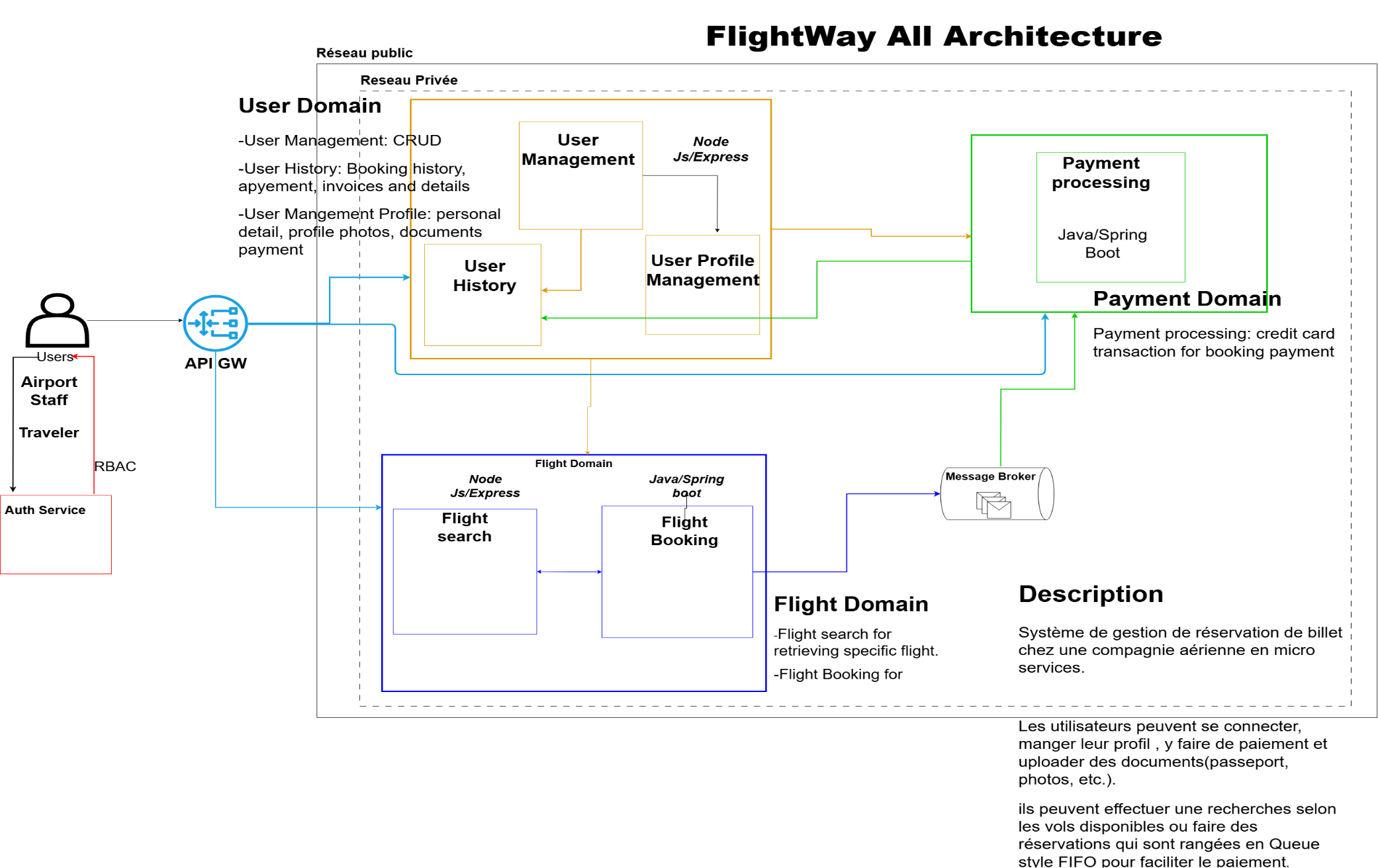
* **User Management** : le micro service chargée de tout ce qui est gestion d’utilisateurs exemple du CRUD ou des opérations sur les utilisateurs (affectation de rôle, blocage, etc.)
* **User Management Profile**: micro service chargé de gérer tout ce qui est en rapport avec la personnalisation du profile des utilisateurs (changement de photo de profil, updating du profil)
* **User History** : c’est le service en charge de retracer tout le parcours utilisateur (historique des vols, des billets, des paiements, des recherches, etc.)

### **Flight Domain**

* **Flight Search** : Ce service est en charge de gérer toutes les opérations de recherche sur les vols, les billets et même des services d’embarquement
* **Flight booking** : Il est le service en charge des réservations des billets d’avions, du statut de la réservation, et de la disponibilité des billets pour un vol donnée. Le message broker permet de gérer plusieurs réservations dans un panier suivant l’ordre FIFO gardant un ordre de réservation facilitant le paiement.

1. **Paiement Service**

Micro service servant à gérer les paiements soit par carte bancaire (Visa, MasterCard) ou l’application des réductions. La figure ci-dessous montre l’enchaînement entre les micro services.



**Figure 1** : Diagramme d'architecture micro services

# **Vu sur le cloud Azure**

Azure est l’un des trois (03) principaux cloud provider avec 27 % de part de part de marché ainsi notre choix s’est porté sur ce dernier. Pour une scalabilité et une haute disponibilité de notre solution nous avons choisi quelques-uns des services azure que nous expliciterons dans les prochaines lignes.

## **Service Azure**

Pour le déployer la solution nous avons la stack suivante :

### **System Architecture**

* **Pour le front end** : pour la plateforme web nous avons choisi React Js et TypeSrcipt pour son SPA (**S**ingle **P**age **A**pplication) et pour le mobile du Kotlin et Swift afin de profiter des fonctionnalités natives d’Android ou de IOS et du GraphQL pour récupérer ce dont nous avons besoin et gagner ainsi en temps
* **Azure Front Door** : Service azure qui gère l’accès aux services, elle fournit un WAF, un CDN, et un routage intelligent avec les Failover.
* **Azure Api Mangement (Gateway)**: Pour l’accès à la solution déployée.
* **Azure Load Balancer**: Service qui partage les tâches de travail entre les services afin de supporter les grands trafics sur la platform.
* **Azure Kubernetes Services**: Ce service nous permet de déployer nos micro services en container et se charge de l’autoscaling entre les différentes pods en fonction de la demande.
* **Azure pipeline**: service connecter sur notre depôt permettant la création de pipeline CI/CD et build nos micro services.
* **Azure Container Apps && Azure Container Instance**: Nous les utilisons respectivement pour, l’un stocker les containers des micro services et l’autre pour le déploiement dans **AKS.**

### **Data Flow Highlights**

* **Inscription des utilisateurs** :  
  Le service d'authentification génère un **JWT**, les **données de profil** sont stockées dans la **base de données des utilisateurs**, et les **documents** sont enregistrés dans **Blob Storage**.
* **Recherche de vols** :  
  Les recherches interrogent **Redis**, puis sont enrichies avec des **données géographiques d’aéroports**.
* **Réservation** :  
  Les **sièges sont verrouillés** via **Redis** + une transaction sur la table de réservation. Et gestion rangée en queue dans Service Bus
* **Paiement** :  
  Appelle une **passerelle de paiement externe**, met à jour **l’état de la réservation** et crée une instance dans sa table paiement en cas de succès, et sauvegarde la facture **dans Blob Storage**.
* **Audit** (History):  
  Toutes les actions sont **enregistrées dans un Tableau (Azure Table)**, accessible aux **utilisateurs** et aux **administrateurs**.

### **Monitoring et sécurité**

* Contrôle d'accès basé sur les rôles **(RBAC)** avec **Azure AD B2C (**pour les clients de la compagnie) **/B2B (**pour les employés, comme des conditionnal access)
* **Gestion des secrets** via **Azure Key Vault**
* Supervision et journalisation avec **Azure Monitor**, **Application Insights** et **Log Analytics et** configuration de **Logic App** pour l’envoie de mail en cas de pannes
* Supervision de l’état de santé du provider avec **Azure Heath**
* L’utilisation **de Vnet** pour chaque service nous permet de configurer lePeering**(communication)** avec plus de sécurité et de contraintes

### **Gestion de la haute disponibilité**

Pour la haute disponibilité, nous avons répliquer les services de la zone principale dans une autre région azure et avec **Front door** en amont comme Entrée unique mondiale et gestion du basculement en cas de fail. Un backup des bases de données(**dump**) ou le container des documents uploadé par l’utilisateur rangée dans un **Blob Storage** externe qui lui-même est répliqué afin d’éviter toute éventuelle perte. Pour le **container registry** l’option **GRS** à été choisi pour la répliquer automatiquement et assurer moins de latence lors du pull des images. Des scripts **Terraform** et **Ansible** pets pour rapidement mettre à niveau la nouvelle zone.

### **Conformité RGPD**

Nos déploiements sont fait dans une région de L’UE(France central),Le cloud Azure est conforme au [RGPD](https://learn.microsoft.com/fr-fr/purview/compliance-manager) , cependant vu pour nous conformer encore plus aussi nous prévoyons demander de manière explicit(opt-in) l’autorisation des utilisateurs pour le traitement des données, la nomination d’un DPO et documentation de nos utilisations(les traitements de données, les finalités, les sous-traitants utilisés (ex: Stripe, Blob Storage, etc.), garantir le droit à l’oubli, le stockage des données utilisateurs pendant 5ans,garantir que les données ne sortiront pas de l’espace L’UE, le chiffrement des donnée et une protection DDos afin d’éviter les fuites de données.