FlyAway Airlines – Architecture Redesign Specification

Full specification document including all diagrams, deployment insights, domain analysis, and platform recommendations.

1. Background

FlyAway Airlines encountered major issues with their recently deployed microservices-based architecture, including:

- Poor scalability and response time under high load
- Frequent service outages and lack of fault tolerance
- Data inconsistency across services
- Limited observability and operational insight
- Weak support for GDPR and PCI-DSS compliance

This document presents a comprehensive redesign to address these limitations using cloud-native, domain-driven, and scalable infrastructure principles.

2. Functional & Non-Functional Requirements

Must-Have

- JWT + OAuth2 based secure authentication
- User profile management and role-based access control
- Flight search (list/map-based) and booking with class/seat availability
- Real-time payment integration with invoice generation
- GDPR consent tracking and data export/erasure support
- Mobile-first design with high-performance APIs

Should-Have

- Admin dashboards, multi-language, and analytics features

Could-Have

- Loyalty programs, AI flight recommendations, staff offline mode

3. System Architecture (with integrated diagram analysis)

Overview

The system follows a hybrid architecture: core features like booking, payments, and search are implemented as microservices, while airline staff tools use a modular monolith. The system is deployed on Azure with Kubernetes clusters for container orchestration.

Deployment Stack

- Azure Kubernetes Service (AKS)
- PostgreSQL (managed via Azure Database)
- Redis for flight search caching
- Storage Account for documents (passports, invoices)
- Azure Front Door and Load Balancer for routing

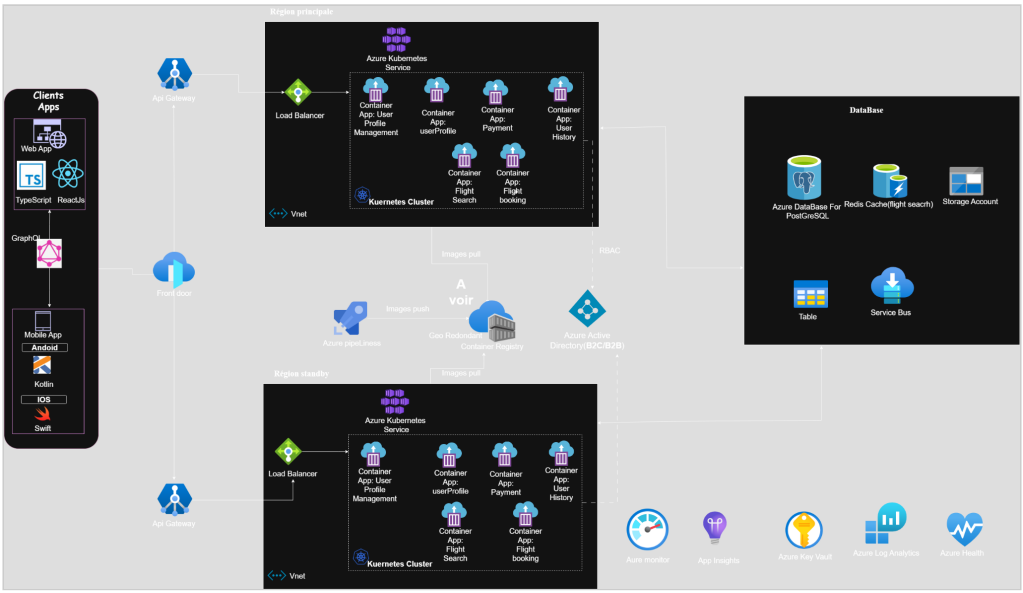
Domains and Responsibilities

- **User Domain:** Management, Profiles, History, and Auth
- **Flight Domain:** Flight Search (Node.js) and Booking (Spring Boot)
- **Payment Domain:** Payment gateway integration, transaction audit

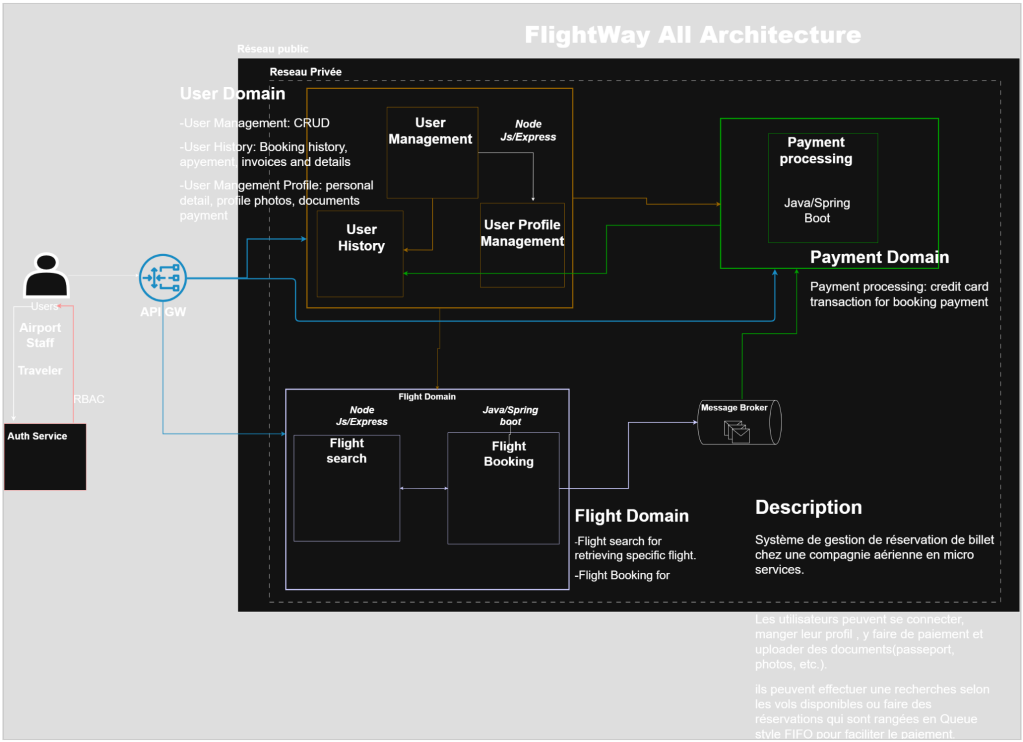
Security & Observability

- RBAC with Azure AD B2C/B2B
- Secrets managed in Azure Key Vault
- Monitoring with Azure Monitor, App Insights, Log Analytics

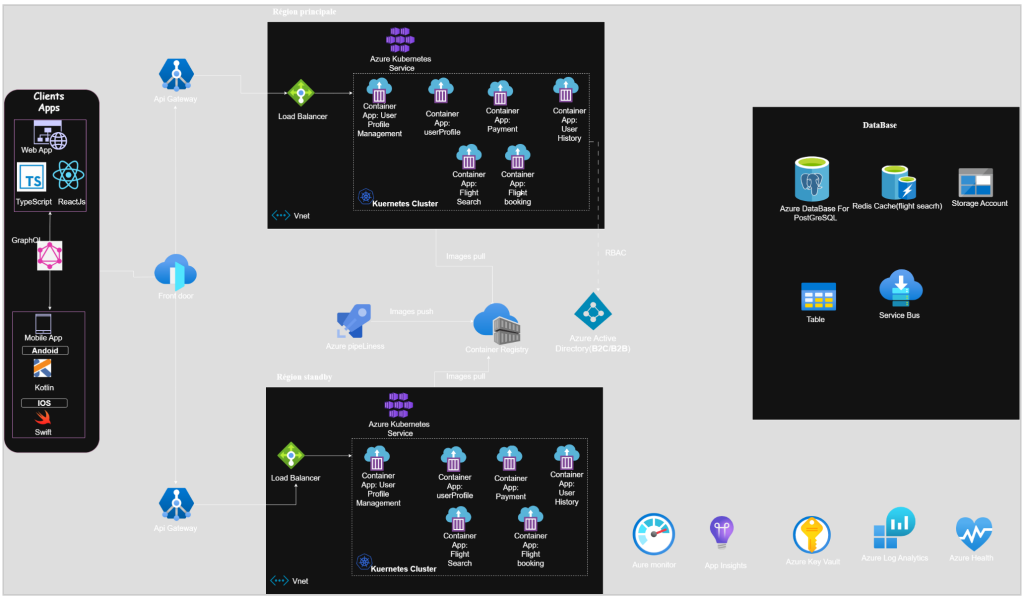
Azure Cloud Deployment Diagram



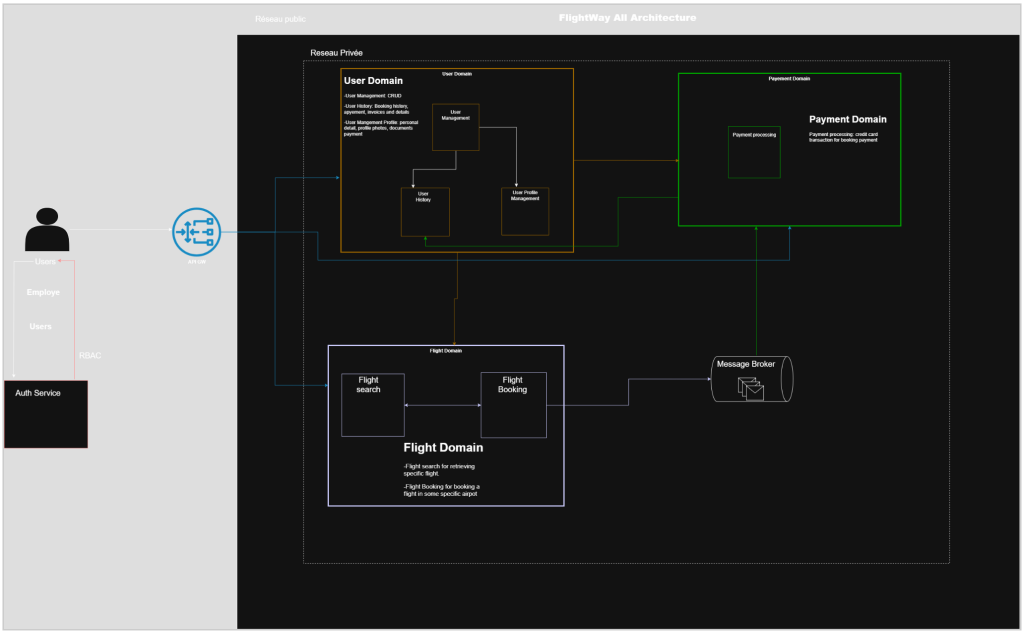
System Handling – Domain Breakdown



FlightWay Architecture – Services (Page 2)



FlightWay Architecture – Domain View (Page 1)



4. Implementation Plan

- **Weeks 1–2:** Infrastructure setup, network, IAM, CI/CD
- **Weeks 3–4:** Auth and User microservices with RBAC
- **Weeks 5–6:** Flight search service and profile management
- **Weeks 7–8:** Booking service with locking mechanism
- **Weeks 9–10:** Payment service, webhook listener, invoices

- **Weeks 11–12:** Admin dashboard (staff monolith)
- **Weeks 13–14:** Frontend/mobile integration and UI polish
- **Weeks 15–16:** Security audit, compliance review, go-live

5. Evaluation & Success Metrics

- **Performance:** P95 API latency < 300ms, < 1.5s page load
- **Availability:** ≥99.95% uptime with autoscaling
- **Data Integrity:** Zero booking duplication or payment errors
- **Compliance:** GDPR user data access, PCI-DSS passed
- **Adoption:** Staff tool adoption, low abandonment rate, positive NPS