Danipa Platform Stack Runbook

This document provides a comprehensive **operations guide** for running and maintaining the **Danipa Fintech Platform**.

It complements environment-specific docs and service READMEs.

■ Platform Overview

The Danipa Platform is a microservices-based fintech solution with support for:

• **MTN MoMo API integration** (Remittance, Collection, Disbursement)

• **Stripe** and **PayPal** integrations (future support)

• **Centralized configuration management** via Spring Cloud Config + Vault + Git

• **Event-driven messaging** via Kafka & Spring Cloud Bus

• **Relational persistence** with PostgreSQL (multi-tenant ready)

• **Secrets management** via HashiCorp Vault

■ Core Infrastructure Services

1. Vault

• **Purpose:** Central secrets manager for credentials and API keys.

• **Ports:** 18300 (host → container:8300)

• **Secrets stored:**

• Config Server (`CONFIG_USER`, `CONFIG_PASS`, `SPRING_PROFILES_ACTIVE`)

• Actuator users

• Postgres users per environment

• MTN MoMo API credentials

[See detailed Vault runbook](./Danipa_Vault_Runbook.md).

---

2. PostgreSQL

• **Purpose:** Application database (multi-schema per service).

• **Image:** `postgres:17`

• **Ports:** 5433 (host → container:5432)

• **Seeded Users:**

• `danipa_owner_dev` (DB owner)

• `danipa_app_dev` (app service user)

• **RLS (Row-Level Security):**

• Configured per customer/tenant

Validation:

```bash
docker exec -it danipa-postgres-dev psql -U danipa_app_dev -d danipa_dev
```

---

3. Kafka + Zookeeper

• **Purpose:** Event streaming and service communication (Spring Cloud Bus).

• **Ports:**

• Zookeeper: 2181

• Kafka Broker: 9092

• **Topics:**

• `springCloudBus`

• Service-specific topics

Validation:

```bash
docker exec -it danipa-kafka kafka-topics.sh --list --bootstrap-server localhost:9092
```

---

4. Config Server

• **Purpose:** Centralized configuration backed by Vault + Git.

• **Profile:** `composite` (Vault + Git fallback)

• **Ports:** 8088

• **Authentication:** `CONFIG_USER` / `CONFIG_PASS` (from Vault)

• **Sources:**

• Vault backend → `secret//`

• Git backend → [danipa-config-repo](https://github.com/paboagye/danipa-config-repo)

Validation:

```bash
curl -u cfg-user:cfg-pass http://localhost:8088/danipa-fintech-service/default
```

■ Fintech Microservices

1. MoMo Service

• **Purpose:** Integration with MTN MoMo Remittance, Collection, and Disbursement APIs.

• **Secrets from Vault:**

• `MOMO_API_USER_ID`, `MOMO_API_KEY`

• Subscription keys (per API)

• Callback URL

• **Validation:**

`curl http://localhost:/actuator/health`

2. Stripe Service

• **Purpose:** Stripe payments integration (planned).

• **Secrets:** API keys stored in Vault under `secret/stripe/`.

3. PayPal Service

• **Purpose:** PayPal payments integration (planned).

• **Secrets:** API credentials stored in Vault under `secret/paypal/`.

■ Setup Guide

1. Prerequisites

• Docker + Docker Compose

• PowerShell (Windows) or Bash (Linux/macOS)

• `.env.` files in project root

• Vault seeds (`infra/vault/seeds/`)

2. Start Infra

```bash
docker compose -f docker-compose.vault.yml up -d
docker compose -f docker-compose.postgres.yml up -d
docker compose -f docker-compose.kafka.yml up -d
```

3. Seed Vault

```powershell
$env:VAULT_ADDR = "http://127.0.0.1:18300"
$env:VAULT_TOKEN = ""
.\scripts
ault-seed.ps1 -EnvName dev -FilePath .\seeds\dev.json
```

4. Start Config Server

```bash
docker compose -f docker-compose.config.yml up -d
```

5. Start Microservices

```bash
docker compose -f docker-compose.services.yml up -d
```

■ Validation

• **Vault:** `curl $env:VAULT_ADDR/v1/sys/health`

• **Postgres:** connect with `psql`

• **Kafka:** list topics

• **Config Server:** fetch config for a service

• **Service health:** `/actuator/health` endpoints

■ Security & Maintenance

• Rotate secrets regularly (update seeds + re-seed Vault).

• Backup `vault/data` volume + Postgres volumes.

• Store **unseal keys** and **root tokens** securely.

• Use **AppRole authentication** for services in staging/prod.

■ Troubleshooting

Vault Port Conflicts

• Error: `listen tcp 0.0.0.0:8200: bind: address already in use`

• Solution: Use remapped port (`18300`)

Vault Seeding "no data provided"

• Ensure `$body = @{ data = $value } | ConvertTo-Json -Depth 10 -Compress`

Postgres Permission Errors

• Verify correct user (`danipa_app_dev`) and password are used from Vault.

Kafka Not Starting

• Check Zookeeper is running before Kafka starts.

■ References

• [HashiCorp Vault](https://developer.hashicorp.com/vault/docs)

• [Spring Cloud Config](https://docs.spring.io/spring-cloud-config/docs/current/reference/html/)

• [Spring Cloud Bus](https://spring.io/projects/spring-cloud-bus)

• [PostgreSQL Docs](https://www.postgresql.org/docs/)

• [Apache Kafka](https://kafka.apache.org/documentation/)