Chương 6: Cấu hình Hệ thống (System Configuration)

Để hệ thống hoạt động chính xác với hạ tầng của đối tác/khách hàng, Signing Service Tân Cảng cung cấp các file cấu hình tách biệt hoàn toàn khỏi mã nguồn. Chương này giải thích các file cấu hình quan trọng nhất trên hệ thống.

6.1. Cấu hình Web Service (application.yml)

Mọi thông số kết nối trọng yếu của API Server được đặt trong thư mục `src/main/resources/application.yml` (khi lập trình) hoặc ghi đè bằng **Environment Variables** (khi chạy Docker).

1. Kết nối Cơ sở dữ liệu (MSSQL Server)

Hệ thống sử dụng SQL Server để lưu thông tin Audit Log (lịch sử ký) và quản lý User/Role phân quyền truy cập.

```
spring:
  datasource:
    url: jdbc:sqlserver://192.168.0.92:1433;databaseName=signing_db;encrypt=true;trustServerCertificate=true
    username: lifetex
    password: LTLT@2025
    driverClassName: com.microsoft.sqlserver.jdbc.SQLServerDriver
  jpa:
    hibernate:
      ddl-auto: update # (Chỉ nên dùng update ở môi trường Dev, Production nên dùng validate/none)
```

2. Cấu hình Redis (EJBCA Cache)

Hệ thống EJBCA CA có tần suất truy vấn siêu lớn ("Kiểm tra chứng thư này còn tồn tại không?"). Để tránh sập tầng mạng CA, ta cấu hình Redis để lưu tạm (Cache) dữ liệu này.

```
spring:
  data:
    redis:
      host: kyso-redis # Trỏ tới container redis nếu dùng chung docker-network
      port: 6379
      # Có thể thiết lập thêm 'password' nếu cụm Redis Server yêu cầu auth.
```

3. Cấu hình Hệ thống Email & SMS (Dùng cho Notification Diagnostics)

Các thông báo cảnh báo hệ thống (như khi không gọi được CA Server) sẽ được đẩy qua giao thức SMTP.

```
spring:
  mail:
    host: smtp.gmail.com
    port: 587
    username: admin@domain.com
    password: app-password
    properties:
      mail.smtp.auth: true
      mail.smtp.starttls.enable: true
```

6.2. Cấu hình Máy trạm Desktop (token-config.json)

Do USB Token (SmartCard) đến từ rất nhiều định dạng nhà cung cấp khác nhau (Viettel, VNPT, FPT, BKAV, HILO), mỗi hãng lại dùng một file Driver `.dll` (trên Windows) riêng biệt.

Ứng dụng `signing-desktop` quản lý cấu hình các Token này thông qua file **token-config.json** nằm tại thư mục gốc của file thực thi.

Định dạng JSON mẫu:

```
[
  {
    "id": "95893e6d-ef56-46ff-981f-5a73f81982b6",
    "name": "USB_HILO_CA",
    "libraryPath": "C:\\Windows\\System32\\hiloca_csp11_v1.dll",
    "slotIndex": 0
  },
  {
    "id": "e5d...-...",
    "name": "VIETTEL_CA",
    "libraryPath": "C:\\Windows\\System32\\viettel-ca_v6.dll",
    "slotIndex": 0
  }
]
```

Giải thích thông số:

- name: Tên định danh của dòng USB Token (để hiển thị trên giao diện `TokenProfileDialog` cho người dùng chọn).
- `libraryPath`: Đường dẫn tuyệt đối trỏ tới file thư viện **PKCS#11** (File `.dll` / `.so` / `.dylib`) của hãng đang cài trong hệ điều hành.
- `slotIndex`: Thứ tự khe cắm thiết bị lưu trữ trên Driver (mặc định đa số là `0`).

> TIP
>
> **Mẹo chuẩn đoán:** Nếu hệ thống chạy Desktop App bị treo và báo "*Module PKCS11 không khả dụng*", hãy mở file **token-config.json** và kiểm tra xem biến `libraryPath` có đúng với đường dẫn cài của hãng cung cấp hay không.

6.3. Cấu hình giao tiếp EJBCA Server

Các cấu hình giao tiếp mã hóa (Keystore / Truststore) để kết nối an toàn với máy chủ cấp phát EJBCA thông qua Rest/SOAP API (nếu có yêu cầu Mutual-TLS) thường nằm trong các tham số Custom Properties tự định nghĩa.

```
ejbca:
  client:
    base-url: "https://ca-server.tancang.com.vn:8443/ejbca/ejbca-rest-api"
    keystore-path: "classpath:certs/ejbca_client.p12"
    keystore-password: "changeit"
    truststore-path: "classpath:certs/truststore.jks"
    truststore-password: "changeit"
```

Khi mã nguồn khởi tạo (tại module `signing-core`), lớp Service sẽ nạp cấu hình SSL/TLS này vào Context để duy trì liên kết an toàn tuyệt đối với máy chủ bảo mật của phía bên trong hạ tầng (Internal Network).